**编 号：**

**审定成绩：**

|  |  |  |  |
| --- | --- | --- | --- |
| **重庆邮电大学**  **毕业设计（论文）** | | | |
|  | | | |
|  | **中文题目** | **基于.NET的社团之家系统的设计与实现** |  |
|  |  |
| **英文题目** | **Design and implementation of the system** |
|  | **of the home of community based on .NET** |
| **学院名称** | **软件工程学院** |
| **学生姓名** | **王经纬** |
| **专 业** | **软件工程** |
| **班 级** | **1301416** |
| **学 号** | **2014214087** |
| **指导教师** | **姓名 陈霖 职称 讲师** |
| **答 辩 组**  **负 责 人** | **姓名 王江涛 职称 高级实验师** |
|  | |

**二零一八 年 六 月**

**重庆邮电大学教务处制**

学院本科毕业设计(论文)诚信承诺书

本人郑重承诺：

我向学院呈交的论文《 》，是本人在指导教师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明并致谢。本人完全意识到本声明的法律结果由本人承担。

年级

专业

班级

承诺人签名

年 月 日

学位论文版权使用授权书

本人完全了解重庆邮电大学有权保留、使用学位论文纸质版和电子版的规定，即学校有权向国家有关部门或机构送交论文，允许论文被查阅和借阅等。本人授权重庆邮电大学可以公布本学位论文的全部或部分内容，可编入有关数据库或信息系统进行检索、分析或评价，可以采用影印、缩印、扫描或拷贝等复制手段保存、汇编本学位论文。

（注：保密的学位论文在解密后适用本授权书。）

|  |  |
| --- | --- |
| 学生签名： | 指导老师签名： |
| 日期： 年 月 日 | 日期： 年 月 日 |

摘要

高校社团活动作为高校学生组织活动中的重要组成部分，在同学们的课外生活中占据很大的比重，具有受众广泛，种类多样，管理难度大等特点。尤其是现在同学们参与社团活动的积极性越来越高，对于高效便捷地管理社团事务的需求也越来越迫切。有鉴于此，我特地开发了这款社团管理综合服务系统，旨在能够更好得对社团事务进行管理和参与。

该款软件为采用B/S架构的社团综合管理系统，以ASP.NET MVC5为主干技术实现，主要面向在校大学生，由社团成员，社团创始人和管理员组成。其内在功能涵盖了对于社团的管理，活动的发布与审核以及新闻公告的发布等方面，大体上可分为前台操作界面以及后台管理界面两大部分，即客户端和管理端，分别面向普通社员和相应管理人员。其中管理员可对该系统所有信息进行审核和修订。

该款软件在开发过程中经历了需求分析，设计，实现，测试等多个阶段，其中在进行数据库设计时，充分考虑软件功能，合理分配字段容量，在界面设计上，力求页面的简洁大方，保留主干成分，给予用户良好的体验。

在采用技术上，以ASP.NET MVC5为核心，通过采用Entity Formwork对象关系映像(ORM)框架，简化数据库连接操作，提升开发效率。并适当引用Fineui等第三方控件库，提升软件开发质量，增强软件可用性。

**关键词：**社团之家，ASP.NET MVC，B/S，社团管理，界面设计

**Abstract**

As an important part of college student organization activities, college community activities occupy a large proportion in students' extracurricular life. They have the characteristics of wide audience, variety and difficulty in management. The demand for efficient and convenient management of community affairs is becoming more and more urgent. In view of this, I developed this software specially for the purpose of better management and participation in community affairs.

The software is a club integrated management system with B/S architecture. It is implemented with ASP.NET MVC5 as the backbone technology. It is mainly for college students. It is composed of community members, association founders and super administrators. Its internal function covers the management of the community, the release and audit of the activities, and the publication of news bulletin. It can be divided into two parts, the front desk and the backstage management interface, which are faced for the ordinary members and the corresponding managers. The super administrator can check and revise all the information of the system.

In the development process, the software has experienced many stages, such as demand analysis, design, implementation and testing. In the design of the database, the function of the software is fully considered, the field capacity is allocated reasonably. In the design of the Interface, I make efforts to make it simple and generous to give users nice experience.

As for technology, the ASP.NET MVC5 is the core and the Entity Formwork object relation model (ORM) framework is adopted to simplify the database connection operation and improve the development efficiency.

**Keywords**: The home of the community, ASP.NET MVC, B/S, Community management, Interface design

目录

[第1章 引言 1](#_Toc515707841)

[1.1系统开发背景 1](#_Toc515707842)

[1.1.1 国内开发背景介绍 1](#_Toc515707843)

[1.1.2 国内研究现状 1](#_Toc515707844)

[1.1.3 系统需要解决的主要问题 2](#_Toc515707845)

[1.2本文内容简介 2](#_Toc515707846)

[1.3本章总结 3](#_Toc515707847)

[第2章 可行性分析及相关知识点 5](#_Toc515707848)

[2.1技术可行性分析 5](#_Toc515707849)

[2.1.1技术风险分析 5](#_Toc515707850)

[2.1.2资源分析 5](#_Toc515707851)

[2.2所用主要技术简介 5](#_Toc515707852)

[2.2.1 ASP.NET MVC 5](#_Toc515707853)

[2.2.2 Entity Formwork (EF) 6](#_Toc515707854)

[2.2.3 Bootstrap 6](#_Toc515707855)

[2.3主要工具简介 6](#_Toc515707856)

[2.3.1 Visual Studio Community 2015 6](#_Toc515707857)

[2.3.2 Visio 2010 7](#_Toc515707858)

[2.3.3 Sql Server 2012 7](#_Toc515707859)

[2.4本章总结 7](#_Toc515707860)

[第3章 系统需求分析 9](#_Toc515707861)

[3.1 社团之家系统需求概述 9](#_Toc515707862)

[3.1.1 社团之家项目总述 9](#_Toc515707863)

[3.1.2 社团之家软件功能结构 9](#_Toc515707864)

[3.2 社团之家系统需求分组描述 10](#_Toc515707865)

[3.2.1用户模块 10](#_Toc515707866)

[3.2.2社团模块 11](#_Toc515707867)

[3.2.3活动模块 12](#_Toc515707868)

[3.2.4公告模块 12](#_Toc515707869)

[3.2.5评比模块 13](#_Toc515707870)

[3.2.6评论模块 14](#_Toc515707871)

[3.2.7交友模块 14](#_Toc515707872)

[3.3 本章总结 15](#_Toc515707873)

[第4章 系统概要设计 17](#_Toc515707874)

[4.1 社团之家设计描述 17](#_Toc515707875)

[4.1.1社团之家第0层设计描述 17](#_Toc515707876)

[4.2 社团之家系统用例图 18](#_Toc515707877)

[4.2.1社团之家用例说明 18](#_Toc515707878)

[4.2.2社团之家用例图 19](#_Toc515707879)

[4.3 社团之家数据流图 22](#_Toc515707880)

[4.3.1社团之家顶层数据流图 22](#_Toc515707881)

[4.3.2社团之家第一层数据流图 22](#_Toc515707882)

[4.4 社团之家数据结构/数据库设计 24](#_Toc515707883)

[4.4.1数据库概念结构设计 24](#_Toc515707884)

[4.4.2 社团之家数据库逻辑结构设计 26](#_Toc515707885)

[4.5 本章总结 30](#_Toc515707886)

[第5章 系统详细设计 31](#_Toc515707887)

[5.1 系统模块详细设计 31](#_Toc515707888)

[5.1.1 用户模块分解描述 31](#_Toc515707889)

[5.1.2 社团模块分解描述 34](#_Toc515707890)

[5.1.3 活动模块分解描述 37](#_Toc515707891)

[5.1.4 评论模块分解描述 40](#_Toc515707892)

[5.1.5 公告模块分解描述 43](#_Toc515707893)

[5.1.6 好友模块分解描述 45](#_Toc515707894)

[5.2社团之家接口描述 48](#_Toc515707895)

[5.2.1用户接口 48](#_Toc515707896)

[5.2.2软件接口 48](#_Toc515707897)

[5.2.3硬件接口 48](#_Toc515707898)

[5.2.4通讯接口 48](#_Toc515707899)

[5.3社团之家总体设计约束 48](#_Toc515707900)

[5.3.1标准符合性 48](#_Toc515707901)

[5.3.2硬件约束 48](#_Toc515707902)

[5.3.3技术限制 49](#_Toc515707903)

[5.4社团之家系统类图 49](#_Toc515707904)

[5.5本章总结 49](#_Toc515707905)

[第6章 系统实现 51](#_Toc515707906)

[6.1管理端页面实现 51](#_Toc515707907)

[6.1.1登录页面实现 51](#_Toc515707908)

[6.1.2管理页面实现 52](#_Toc515707909)

[6.2用户端页面实现 53](#_Toc515707910)

[6.2.1登录/注册页面实现 53](#_Toc515707911)

[6.2.2社团信息页面实现 54](#_Toc515707912)

[6.2.3活动信息页面实现 55](#_Toc515707913)

[6.2.4公告信息页面实现 55](#_Toc515707914)

[6.3本章总结 56](#_Toc515707915)

[第7章 系统测试 57](#_Toc515707916)

[7.1测试计划 57](#_Toc515707917)

[7.2系统测试 57](#_Toc515707918)

[7.2.1用户模块测试 57](#_Toc515707919)

[7.2.2社团模块测试 58](#_Toc515707920)

[7.2.3活动模块测试 59](#_Toc515707921)

[7.3本章总结 60](#_Toc515707922)

[第8章 总结展望 61](#_Toc515707923)

[8.1相关难点及创新点 61](#_Toc515707924)

[8.2问题以及改进措施 61](#_Toc515707925)

[参考文献 62](#_Toc515707926)

[致谢 64](#_Toc515707927)

[附录A 部分源代码 65](#_Toc515707928)

[附录B 英文翻译 68](#_Toc515707929)

# 第1章 引言

## 1.1系统开发背景

### 1.1.1 国内开发背景介绍

高校学生社团是由高校学生以共同追求为基础而建立的学生组织,是具有明确发展目标和规范、符合所在学校社团管理条例的非正式群体[1],随着信息的丰富，如今高校社团数量与日俱增，我国高校社团的地位和作用也日益显现[2]。据统计，仅就浙江省而言，每所高校都有几十个甚至上百个社团，平均每个高校有三十多个社团，社团成员已占到高校学生总数的60%[3]。所以说，目前社团数量的剧增加大了社团的管理难度。

就社团本身而言，其自身也早已摆脱了最初片面化，单一化的特征，变得种类繁多，包含广泛，各种新奇社团也层出不穷。由于我国高校社团的研究还处于起步阶段[4]，而传统人力手工管理社团的方式所暴露出来的种种弊端，诸如耗时久，难以统计等逐渐显现。但在如今，针对高校的社团管理也越来越得到人们的重视，好的社团管理，对于促进高校和谐校园的构建、大学生成长成才机制的完善都具有十分重要的作用[5]，再加上大学生正在努力寻求能力培养途径和新的团队归属, 这样社团的作用就日益凸显、地位不断提升[6]，因此，依据最新软件和网络技术，开发一款高效，便捷，方便管理和参与的社团系统是很有必要的。为此，我专门开发了这款名为社团之家的服务系统，为大家参与，管理社团事务提供一个方便，快捷的平台。

### 1.1.2 国内研究现状

采用信息化的方式管理社团，可以极大地提高社团管理效率，有利于学生群体和老师快速开展工作。但在当今的大部分高校信息化建设中, 通常只针对学生档案、教务考试、办公人事等信息功能着重规划和建设, 往往忽视了学生社团管理的信息化需求和应用[7]，这就导致了在如今的社团管理中，传统手工登记方式管理社团仍占据主导地位，部分高校已经尝试推行信息化方式管理社团事务，虽然在推行实践中简化了部分事务，但仍然显露出一些待解决的问题，诸如社团管理与社团参与相互杂糅，难以区分；社团创建门槛过低，导致大量垃圾社团出现；一些不活跃的社团相互累加堆积导致出现大量僵尸社团且没能得到及时处理等问题。本系统将重点针对这些问题进行解决。

### 1.1.3 系统需要解决的主要问题

为了实现对社团事务高校便捷地管理，同时针对上述描述的社团管理中出现的若干问题，该系统要重点解决的问题如下：

1) 普通社团事务管理效率低下，审批登记时间过长，难以有效开展社团管理或参与工作。

2) 社团数量庞杂，导致大量僵尸社团出现，占用学校资源却无法发挥其应有的作用。

3) 社团传播影响力较差，无法在第一时间获取相关社团成立信息，学生无法及时选择自己喜爱社团。

## 1.2本文内容简介

本文将根据软件的开发具体流程对本系统进行介绍，全文共分为8章，对全文各章的内容大致介绍如下：

第一章为引言，主要引出本系统开发背景以及相关研究意义。诸如系统开发缘由，过去管理方式的不足和与新系统的相关对比，同时简要陈述每一章内容。

第二章为可行性分析及相关知识点，主要从技术可行性说明并介绍系统所用相关技术，诸如ASP.NET MVC以及相应编译工具等介绍。

第三章为系统的需求分析，主要解决系统是做什么的这一问题，具体可以描述在系统功能模块的划分以及具体需求的描述。

第四章为系统概要设计，描述系统总的业务流程以及相关数据流和数据库设计相关介绍。并对相应用例进行介绍。

第五章为系统详细设计，重点讲述系统各个模块执行的详细流程，并且给出接口描述。

第六章为系统实现，重点讲述系统功能实现过程以及相应界面截图。

第七章为系统测试，主要通过黑盒测试等相关方法设计测试用例，从功能对程序进行测试。

第八章最后对系统进行总结，指出系统相应创新点，同时也对进一步改进做出了说明。

## 1.3本章总结

本章主要引出社团之家服务系统开发背景，介绍了其开发的背景和意义，同时针对国内类似的系统情况做了一定比较，在本章结尾又对本论文的大致内容给与了简要介绍。

# 第2章 可行性分析及相关知识点

## 2.1技术可行性分析

可行性研究从经济，技术，法律，用户操作等方面分析所给出的解决方案是否可行[8]。

### 2.1.1技术风险分析

该项目为基于B/S架构的社团之家综合服务系统，采用的主要技术为ASP.NET MVC5,作为一种构建WEB应用程序的框架，自2002年ASP.NET 1.0发布以来，历经十余年的发展，本身已经相当成熟且拥有庞大数量的开发者和应用场景。由于它本身具有开发Web应用程序所需要的一切解决方案，天生就对WEB应用程序具有良好的支持。可以在较短时间内开发出高效，方便，安全性高的应用程序。

本系统采用VS2015 community以及SqlServer2012作为开发工具，同样作为非常成熟的开发工具，两者都对.NET开发提供了比较强大的支持，其中VS2015 Community 自带MVC5模板，即可自动创建ASP.NET MVC5空项目。后者同样作为微软开发的数据库管理工具，有助于该系统的快速开发。

### 2.1.2资源分析

该系统作为个人毕业设计项目，由个人独自完成分析，设计和开发，由于本人曾经完成相关ASP.NET程序开发，并且由于ASP.NET易上手，快速开发的特性，开发周期控制会比较合理，部署同样采用微软IIS10.0,易于部署。硬件环境方面，由于该WEB应用程序对需求配置不高，所以现有硬件环境完全可以胜任开发任务。

## 2.2所用主要技术简介

### 2.2.1 ASP.NET MVC

.NET平台是由Microsoft 推出的全新的应用程序开发平台[9]，其中ASP.NET MVC 是微软官方提供的以MVC模式为基础的ASP.NET Web应用程序框架[10]，它是免费的,开源的，基于测试驱动的Web应用程序编程模型[11]。它以ASP.NET技术为基础，结合最新的MVC设计模式，可以方便，高效地帮助我们开发WEB相关应用程序[12]。其中ASP.NET为新一代基于.NET Formwork的开发平台，以C#为主要开发语言，早期C#可看作2001年Java语言升级版[13],后来逐渐加入自己的特色。MVC模式则体现了“关注点分离”这一基本的方针[14]。

### 2.2.2 Entity Formwork (EF)

EF是一个开源的基于ADO.NET对象映射关系模型，它将数据库表转换为一个模型（Model）,从而使得我们可以像操作类去使用它[15]。它提供了SQL类似的LINQ语法，使得我们方便地去操作数据库，在具体执行时，EF会将LINQ转换为SQL语句然后执行，EF提供了三种执行流程，分别为Code First, Data First和Model First,该系统在客户端和管理端分别采用了前两种执行模式，Code First采用先创立相关模型类，并给出数据注解，EF会自动帮助我们创立数据库，而Data First则是根据已经建立好的数据库，创立相关模型类，之后还可通过数据迁移来实现数据字段的改变。

### 2.2.3 Bootstrap

Bootstrap为来自于Twitter的前端开发框架，它开源且完全免费，其自身集成了HTML，CSS，JAVASCRIPT，它灵活简便，旨在使Web开发更为便捷，其自身具备响应式，自动交互等功能，一经推出便大受好评，目前在国内有大批开发者将其运用到自己的项目中，截止目前，其版本已更新到Bootstrap3。

## 2.3主要工具简介

### 2.3.1 Visual Studio Community 2015

VS2015 Community是由微软公司所开发的功能完备且可扩展的集成开发环境(IDE),完全免费。使用它，可使开发任务更简单[16]。它主要可用于开发C#相关，基于ASP.NET的WEB应用程序。其具备强大的编码提示功能，拥有高级调试等多种功能，且具有跨平台的能力。

### 2.3.2 Visio 2010

Visio 2010是由微软公司开发的Office办公套件中的一款，是一种方便，快捷，功能强大的软件[17]，旨在帮助IT服务人员更好地进行可视化分析以及交流相关复杂数据，该款软件通过创建于与数据相关的Visio图表来显示数据，这些图表易于显示内容信息且能显著提高生产力，简单外加数据驱动的形状以及与 Web 共享相结合，使得 Visio 2010 成为查看和了解信息的功能最强大的工具之一。

### 2.3.3 Sql Server 2012

数据库的实质为数据的集合,数据库是根据数据结构来组织、存储和管理数据的仓库[18]。SQL Server是当前较为常用的数据库管理系统之一,具有速度快、功能强、易使用等特点,它主要解决数据的存储与管理问题[19]。Sql Server 2012是微软发布新一代数据平台产品，为包含强大功能数据库管理工具，全面支持云技术等相关平台。具有使用方便可伸缩性好以及相关软件集成程度高等优点，且具有很高的安全性和可用性，由于本系统以ASP.NET为主干技术实现，外加SQL Server自身对VS2015开发的良好支持，其自然就成为开发所需工具的首要选择。

## 2.4本章总结

本章对系统的可行性进行了分析研究，重点介绍了本系统技术可行性，在本章后半部分着重介绍了本系统所用技术和开发工具。

# 第3章 系统需求分析

## 3.1 社团之家系统需求概述

学生已成为我国网民中规模最大的群体，约占比重为30.2%[20]。现代大学生越来越倾向于通过电脑上网来满足于生活的一些基本需求，例如网上购物，网上聊天等。基于大学校园内庞大的网络用户数量，构造一个网络社团信息服务平台，以此来方便大家对于社团事务的管理和参与，是很有意义和价值的。同样，这样一款服务系统，在大学生群体中也包含有一定的市场。

### 3.1.1 社团之家项目总述

该系统为一在线社团事务服务平台，主要供社团成员，社团负责人以及管理员进行社团事务的参与和管理，其主要功能模块包括对社团事务的管理，对活动事务的管理，对社员的管理，对新闻公告的管理等等，力求打造一个可操作性强，界面美观，舒适的综合社团服务网站。

### 3.1.2 社团之家软件功能结构

系统组织结构图如图3.1所示:



图3.1 社团之家系统组织结构图

1. **用户模块**：普通游客通过注册来获取用户权限，并通过登陆成为会员，获得会员权限之后，即可获取权限浏览相应社团，活动具体页面。
2. **社团模块**：普通用户登录成功之后，获取权限即可提交社团申请，待管理员同意之后，完成社团的创建，其他用户可申请加入社团或退出社团，管理员和本社团负责人均可以直接解散本社团。
3. **活动模块**：社团负责人可以申请由本社团举办活动，待管理员审核通过后，活动即可在客户端中出现，其他会员可以申请加入活动，待日期超过后，活动即自动结束，管理员也可以强制结束活动。
4. **公告模块**：管理员可以在管理端发布公告，删除公告。普通会员可以浏览相应公告。
5. **交友模块**：普通会员可以关注其他会员，了解他们的动态及最新消息，同时也可以取消对他们的关注。
6. **评比模块**：会员和社团均采用积分制，在创建社团成功或者创办活动成功之后，对于社员或者社团均有积分加成，每周会根据此来评比优秀社团和社员。另外，管理员会依据积分以及创办时间来搜寻垫底社团，即僵尸社团，每隔固定时间清理这些社团。
7. **评论模块**：会员和社团负责人都可以针对社团或者活动发表评论，管理员可以删除相应的评论。

## 3.2 社团之家系统需求分组描述

### 3.2.1用户模块

**1) 模块功能简介**

普通游客通过该功能能够获取设置自己的用户名和密码，以后游客即可通过自己的账户名和密码登录成为会员并获取相应权限，输入要求账户名和密码必须在6-18位之间，此外，会员依据自己的密保问题修改密码从而实现找回密码。

**2) 输入**

输入来源：注册或登录时，输入者为普通游客，可输入的字符包含数字，英文字符，下划线等，长度符合要求。修改头像时，输入要求为符合格式的图片。

输入长度:输入字符长度要求在6-18位之间

度量单位：度量单位为输入的字符长度。

时间要求：无

3) 输出

输出位置：数据输入至数据库且文件上传至指定定点，结果返回页面显示。

成功输出：系统提示注册或登录或修改头像成功，并同时跳转至系统主页面。

失败输入：系统提示注册或登录或修改头像失败，要求用户再次输入。

### 3.2.2社团模块

**1) 模块功能简介**

普通会员用户可以提交社团申请，待管理员同意之后即可成立社团并自动成为社团负责人，其他用户可申请加入社团，由社团负责人或管理员决定是否同意该人加入，加入之后社员可以选择退出，管理员根据活跃度和积分定时清理一批僵尸社团。

**2) 输入**

输入来源：提交申请者为登录之后的普通会员，要求在申请社团或选择加入社团时选择社团的名称，社团的类别，具体的申请理由等信息,完善之后由管理员在后台界面选择同意或拒绝申请，同意则社团创建成功，拒绝则社团创建失败。社团负责人或管理员针对其他用户加入申请选择同意或拒绝。

输入长度：社团名称要求为6-18字符之间，申请理由不少于20字符。

度量单位：单位为可输入的字符长度。

时间要求：无

**3) 输出**

输出位置：若成功提交社团申请且管理员同意社团申请，则自动创建社团界面返回社团创建结果且社团数据插入数据库中，若拒绝社团申请则删除申请数据且界面返回申请拒绝结果。加入社团处理与创建社团处理一致。

成功输出：自动创立社团并成为社团负责人，同时社团数据插入数据库中。若为加入社团，则系统中插入相应会员信息。退出则删除该信息。

失败输出：删除社团申请数据或社团加入数据，界面返回失败提示。

### 3.2.3活动模块

**1) 模块功能简介**

社团负责人可以通过本社团发起活动，提交活动申请，并由管理员同意后举办活动，其他会员可以申请加入该活动，此外，管理员可以随时终止该活动。举办活动成功对该社团和社团负责人均有积分加成。有助于明星社团或者优秀社员评比。

**2) 输入**

输入来源：社团负责人进入活动申请页面，输入要求输入的活动名称，活动种类以及活动相关介绍并最终确认提交，其他社员选择活动执行加入活动操作，管理员在后台依据申请选择同意或是拒绝活动的创建以及活动的加入。

输入长度：活动名称要求6-18字符之间，活动内容或加入理由要求不少于20字符，创建活动时活动种类要求选择其中一种。

度量要求：单位为可输入的字符长度。

时间要求：无。

**3) 输出**

输出位置：若社团负责人成功提交活动申请或普通社员提交加入活动申请，则数据库中插入申请信息，管理员或社团负责人登录后台管理页面后执行操作，选择同意或拒绝，若同意，则完成活动创建或加入活动申请，数据库插入活动信息，否则拒绝活动创建或加入活动申请，数据库删除相关信息。

成功输出：系统提示活动创建或申请成功，数据库插入活动信息或加入信息。

失败输出：系统提示活动创建失败或加入失败，保留在原页面。

### 3.2.4公告模块

**1) 模块功能简介**

管理员可以在后台发布公告信息，同时对已发布的公告信息执行删除操作，普通会员或社团负责人可以在前台查看已发布的公告信息，同时根据公告类别选择自己要查看的公告。

**2) 输入**

输入来源：管理员进入后台公告发布页面，填写公告标题，输入发布内容，选择发布种类，确认发表之后完成公告的发表。删除时进入浏览公告页面，选择相应公告执行删除操作。

输入长度：要求公告标题在6-18字符之间，且为大小写字母，符号或数字。公告内容不少于20字符。

度量单位：可输入的字符长度。

时间要求：无

**3) 输出**

输出位置：若公告发布成功，则数据库中新增新闻发表信息，同时显示于前台供普通用户浏览，后台公告浏览页面也会出现公告信息。若公告发布失败，则保留在公告发布页面，系统提示发布失败。若公告删除成功，数据库删除相应信息。

成功输出：系统提示公告发布成功，数据库新增公告信息，并回到公告发布页面。删除时成功则提示成功并返回浏览主页面。

失败输出：系统提示公告发布失败，保留原页面。若删除失败，则保留在原有位置。

### 3.2.5评比模块

**1) 模块功能简介**

社团之间通过积分进行评比，积分来源于社团活动数以及社团加入人数，角逐出积分数较高的明星社团和积分数较低的垫底社团，以供会员在加入社团时参考。

**2) 输入**

输入来源：普通用户或社团管理员进入社团或用户浏览首页，查看社团排行榜或社员排行榜，查看明星社团或社员。

输入长度：无

度量单位：无

时间要求：无

**3) 输出**

输出位置：页面返回社团/社员排行，同时显示明星社团或社员以供普通用户加入或关注。

成功输出：显示明星社团或明星社员信息。

失败输出：无

### 3.2.6评论模块

**1) 模块功能简介**

普通用户针对特定的社团或活动可以发表评论，同时也可查看其它用户发表的类似评论，管理员有权限对所有的评论执行删除操作。

**2) 输入**

输入来源：普通用户浏览社团或活动详细信息时，选择发表评论，输入相应评论内容并确认发表即可完成评论发表，管理员后台浏览评论信息，选择特定评论执行删除。

输入长度：评论内容要求输入长度不少于20字符。

度量单位：为可输入的字符长度。

时间要求：无。

**3) 输出**

输出位置：若用户评论发布成功，系统提示评论发布成功，数据库中新增评论信息，评论出现于下方评论栏，若评论发表失败，则系统提示评论失败，保留在当前评论发表页面。若系统删除评论成功，则提示成功同时数据库删除相应信息。

成功输出：系统提示发表成功或删除成功，返回该社团，活动或评论浏览页面。

失败输出：系统提示发表失败或删除失败，并保留在原页面。

### 3.2.7交友模块

**1) 模块功能简介**

普通用户在用户浏览界面可以关注特定用户，关注之后该用户出现在其个人界面的用户关注列表中，普通用户可以随时了解关注用户的最新动态，包括创建社团，参与社团等情况，普通用户也可以取消关注已关注的用户。

**2) 输入**

输入来源：普通用户浏览用户列表，选择特定用户并执行关注操作。关注之后进入个人主页，选择已关注的用户可执行取消关注操作。

输入长度：无。

度量单位：无

时间要求：无

**3) 输出**

输出位置：若用户成功完成关注操作，则数据库中新增关注信息，系统提示关注成功，该特定用户出现在用户的关注列表中。若取消关注，该关注用户从页面消失同时数据库删除相应关注信息。

成功输出：系统提示关注用户成功或取消关注成功，同时返回用户列表。

失败输出：系统提示关注用户失败或取消关注失败，并保留在原来位置。

## 3.3 本章总结

本章重点对本系统需求进行分析，结合图文对包括本项目的描述以及对各个不同模块的功能进行了描述。重点解决本系统是做什么的这一问题，并对接下来的系统概要设计和详细设计打好基础。

# 第4章 系统概要设计

## 4.1 社团之家设计描述

### 4.1.1社团之家第0层设计描述

**1) 上下文定义**

社团之家上下文定义，如图4.1所示



图4.1 社团之家上下文定义图

**2) 设计思路**

社团之家设计思路，如图4.2所示



图4.2 社团之家包图

**3) 业务流程图**

社团之家业务流程图，如图4.3所示



图4.3 社团之家业务流程图

在该业务流程图中，主要角色包含会员（包括社团负责人）以及管理员，普通游客通过注册功能成为会员，根据系统即可实现申请社团，申请加入社团或申请活动加入活动等操作，管理员审核该申请，依次选择同意或拒绝社团，活动的创建加入请求，并对公告评论执行类似操作。

## 4.2 社团之家系统用例图

### 4.2.1社团之家用例说明

本系统主要包含以下三类用户

1. 普通会员以及普通社员，这一类用户可以浏览社团,活动，新闻公告及所有相关信息，也可以选择加入，退出社团以及关注好友等。
2. 普通会员在申请社团成功之后自动成为社团负责人，之后便可申请活动，审核社员加入请求等。
3. 管理员为权限最大的一类用户，包含对本系统所有信息的操作，包含对社团的清理，用户的封禁等。

### 4.2.2社团之家用例图

系统总用例图如图4.4，图4.5，图4.6所示。



图4.4 普通会员用例图



图4.5 社团负责人用例图



图4.6 管理员用例图

各模块分解用例图如下所示。

**1) 用户模块子用例图**

用户模块包括用户登录，注册，修改头像等功能，具体用例图如图4.7所示



图4.7 用户模块子用例图

**2) 社团模块子用例图**

社团模块包含普通会员对社团的申请创建或者加入，以及管理员对社团的审核和管理，具体见图4.8所示



图4.8 社团模块子用例图

**3) 活动模块子用例图**

活动模块包含普通会员对活动申请加入以及管理员对活动的审核，其具体见图4.9所示



图4.9 活动模块子用例图

## 4.3 社团之家数据流图

### 4.3.1社团之家顶层数据流图

系统顶层数据流图如图4.10所示



图4.10 顶层数据流

### 4.3.2社团之家第一层数据流图

系统第一层数据流图如下所示，其中一层数据流图如图4.11所示，申请社团数据流图如图4.12所示，申请活动数据流图如图4.13所示。



图4.11 一层数据流图

图4.12申请社团数据流图

图4.13 申请活动数据流图

## 4.4 社团之家数据结构/数据库设计

基于B/S架构的社团之家综合服务系统是以数据库为核心，在Sql Server的支持下，完成数据信息的搜索，处理，检索，储存和加工等。数据库是本系统的核心和基础部分，数据库的好坏直接关系到本系统的正常运行效率。然而，由于数据库的复杂性以及它与环境的复杂联系，导致数据库的设计并不是一件容易的事情。

该系统的数据库设计，针对ASP.NET复杂的应用环境，期望提供一个最优的数据模型和处理模型相对应的逻辑结构，尽可能提升系统的运行效率。

### 4.4.1数据库概念结构设计

将需求分析得到的用户需求抽象为信息结构即概念模型的过程就是概念结构设计，它是数据库设计的关键，是其它各种模型的共同基础，它比数据模型更独立于机器，更加抽象和稳定，描述概念模型最强有力的方法即是ER模型，该系统的ER模型如图4.14所示



图4.14 社团之家ER图

从上方ER图中可以看出，普通会员可以参加一个或多个社团或活动，同时也能提交一个或多个活动或社团创立申请，同样，管理员可以对一个或多个活动或社团申请进行审核，并能对多个社团或活动进行管理，用户可以发表相关评论，管理员针对此类评论执行删除操作，管理员还可以对多个公告进行管理，包括发布或删除公告。

会员ER图如图4.15所示:



图4.15 会员ER图

社团ER图如图4.16所示:



图4.16 社团ER图

活动ER图如图4.17所示:



图4.17 活动ER图

评论ER图如图4.18所示:



图4.18 评论ER图

公告ER图如图4.19所示:



图4.19 公告ER图

管理员ER图如图4.20所示:



图4.20 管理员ER图

### 4.4.2 社团之家数据库逻辑结构设计

逻辑结构设计的主要任务即是把上述ER图转变为与所选用数据库所支持逻辑模型相符合的逻辑结构。由系统ER图，我们可以设计出10张表，分别为Members表、SuperUsers表、CreateActivities表、Comments表、Clubs表、ApplyFirends表、ApplyClubs表、ApplyClubMembers表、ApplyActivities表、Activities表。

**1) Members表**

Members表用于存储所有与用户相关的信息，包括用户名密码等。其结构如表4.1所示。

表4.1 数据库Members表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 可为空 | 描述 |
| MemberId | Guid | NO | 主键 自增长 |
| MemberName | VarChar(200) | NO | 用户名称 |
| MemberPassword | VarChar(200) | NO | 用户密码 |
| MemberQues | Int32 | NO | 密保问题 |
| MemberAnswer | Varchar(200) | NO | 密保答案 |
| Grade | Int32 | NO | 积分 |
| MemberTime | DateTime | NO | 注册时间 |
| Email | Varchar | NO | 邮箱 |
| Phone | Varchar | NO | 用户手机 |
| Pic | Varchar | NO | 头像地址 |
| IsValid | Int32 | NO | 是否封禁 |

**2) Clubs表**

该表用于存储与社团相关信息，包括社团ID，社团名称，信息等内容，其结构如表4.2所示。

表4.2 数据库Clubs表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 可为空 | 描述 |
| ClubId | Guid | NO | 主键 自增长 |
| ClubName | VarChar(200) | NO | 社团名称 |
| ClubIntroduce | VarChar(2000) | NO | 社团简介 |
| BeginTime | DateTime | NO | 创办时间 |
| Grade | Int32 | NO | 社团积分 |
| kind | VarChar(200) | NO | 社团种类 |
| ClubActivitiesNum | Int32 | NO | 社团活动数量 |
| ClubMemberNum | Int32 | NO | 社团会员数量 |
| Member\_MemberId | Guid | NO | 外键 |
| url | VarChar(200) | NO | 社团图片地址 |

**3) Activities表**

该表用于存储与活动相关信息，包括活动ID，活动名称，活动信息等内容，其结构如表4.3所示。

表4.3 数据库Activities表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 可为空 | 描述 |
| ActivityId | Guid | NO | 主键 自增长 |
| ActivityName | VarChar(200) | NO | 活动名称 |
| ActivityIntro | VarChar(2000) | NO | 活动简介 |
| BeginTime | DateTime | NO | 开始时间 |
| EndTime | DateTime | NO | 结束时间 |
| IsEnd | Bool | NO | 是否结束 |
| ActivityNum | Int32 | NO | 参加人数 |
| ActivityKind | VarChar | NO | 活动种类 |
| ActivityUrl | VarChar | NO | 活动图像 |
| Club\_ClubId | Guid | NO | 外键 |

**4) Comments表**

该表用于存储与评论相关信息，包括评论ID，评论内容，评论人等内容，其结构如表4.4所示。

表4.4 数据库Comments表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 可为空 | 描述 |
| CommentId | Guid | NO | 主键 自增长 |
| CommentKind | Int32 | NO | 评论种类 |
| CommentWords | VarChar | NO | 评论内容 |
| CommentTime | DateTime | NO | 评论时间 |
| Member\_MemberId | Guid | NO | 外键 |
| Activity\_ActivityId | Guid | Yes | 外键 |
| Club\_ClubId | Guid | Yes | 外键 |

**5) CreateActivities表**

该表用于存储与创建活动相关信息，包括创建ID，申请理由，申请人等内容，其结构如表4.5所示。

表4.5数据库CreateActivities表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 可为空 | 描述 |
| CreateId | Guid | NO | 主键 自增长 |
| ActivityName | VarChar(200) | NO | 活动名称 |
| ApplyReason | VarChar(2000) | NO | 申请理由 |
| ApplyTime | DateTime | NO | 申请时间 |
| IsAgree | Bool | NO | 是否同意 |
| BeginTime | DateTime | NO | 开始时间 |
| EndTime | DateTime | NO | 结束时间 |
| ApproveTime | DateTime | Yes | 审批时间 |
| Club\_ClubId | Guid | NO | 外键 |

**6) ApplyActivities表**

该表用于存储与加入活动相关信息，包括加入ID，申请理由，申请人等内容，其结构如表4.6所示。

表4.6数据库ApplyActivities表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 可为空 | 描述 |
| ApplyActivityId | Guid | NO | 主键 自增长 |
| ApplyTime | DateTime | NO | 申请时间 |
| IsAgree | Bool | NO | 是否同意 |
| Activity\_ActivityId | Guid | NO | 外键 |
| Member\_MemberId | Guid | NO | 外键 |

**7) ApplyClubs表**

该表用于存储与创建社团相关信息，包括创建ID，申请理由，申请人等内容，其结构如表4.7所示。

表4.7数据库ApplyClubs表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 可为空 | 描述 |
| ApplyClubId | Guid | NO | 主键 自增长 |
| ClubName | VarChar(200) | NO | 社团名称 |
| ApplyClubReason | VarChar(200) | NO | 加入理由 |
| ApplyClubTime | DateTime | NO | 申请时间 |
| IsAgree | Bool | NO | 是否同意 |
| ApprovalTime | DateTime | NO | 审批时间 |
| Member\_MemberId | Guid | NO | 外键 |
| ApplyKind | VarChar(200) | NO | 申请种类 |

**8) ApplyClubMembers表**

该表用于存储与加入社团相关信息，包括申请ID，申请人等内容，其结构如表4.8所示。

表4.8数据库ApplyClubMembers表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 可为空 | 描述 |
| ApplyId | Guid | NO | 主键 自增长 |
| ApplyTime | DateTime | NO | 申请时间 |
| IsAgree | Bool | NO | 是否同意 |
| ApproveTime | DateTime | NO | 审批时间 |
| Club\_ClubId | Guid | NO | 外键 |
| Member\_MemberId | Guid | NO | 外键 |

**9) ApplyFirends表**

该表用于存储与好友相关信息，包括关注ID，关注人ID,被关注人ID等内容，其结构如表4.9所示。

表4.9数据库ApplyFirends表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 可为空 | 描述 |
| ApplyId | Guid | NO | 主键 自增长 |
| ApplyTime | DateTime | NO | 审批时间 |
| Member\_MemberId | Guid | NO | 外键 |
| MainUser | Guid | NO | 外键 |

**10) SuperUsers表**

该表用于存储与管理员相关信息，包括管理员ID，用户名,密码等内容，其结构如表4.10所示。

表4.10数据库SuperUsers表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 可为空 | 描述 |
| GuserId | Guid | NO | 主键 自增长 |
| GuserName | VarChar(200) | NO | 管理员名称 |
| GuserPass | VarChar(200) | NO | 管理员密码 |

## 4.5 本章总结

本章重点讲述对系统的概要设计，包括系统总的执行流程，系统0层和1层的数据流图，ER图以及数据库表设计等，使得该系统呈现出大致清晰的轮廓。

# 第5章 系统详细设计

依据社团之家需求分析及相关概要设计，下面主要对各个功能进行相应详细设计。设计包含两个部分，即各个功能模块执行详细流程和相关类图，首先重点对各个功能模块进行介绍。

## 5.1 系统模块详细设计

### 5.1.1 用户模块分解描述

**1) 简介**

该功能模块实现了注册登录，密码找回，管理个人头像信息等功能。

**2) 功能列表**

注册登录：普通游客可在在本系统上通过注册获得会员资格，并登录该注册账号从而获得会员权限。

找回密码：若用户忘记个人密码，可以在密码找回页面通过密保问题修改密码。

修改个人头像：用户可在个人管理页面通过上传图片实现头像信息的修改。

**3) 功能列表**

(1) 注册功能

用户可以通过此功能注册从而获得会员账号，用户必须完成表单上的必选项同时要求输入数据符合相应格式方能完成注册。该功能要求用户输入邮箱地址，手机号码，密码（两次输入密码必须一致），昵称等之后完成注册。

执行流程如图5.1所示:



图5.1 注册功能流程图

(2) 登录功能

用户进入登录页面，输入正确的用户名和密码，系统连接数据库自动比对用户名以及密码，若比对成功，返回用户数据并跳转至主页面，若比对失败，提示用户名或密码错误并要求用户重新输入。

执行流程如图5.2所示:



图5.2 登录功能流程图

(3) 修改个人头像

用户登录成功，进入个人管理页面，选择图片上传，上传完成之后完成头像的修改。

执行流程如图5.3所示:



图5.3 修改头像流程图

**4) 类**

(1) Member类

用于存储所有和用户信息相关的数据。

(2) MemberController类

包含所有与用户信息逻辑处理相关代码。

(3) DB类

显示与数据库相连时读取相关的数据库信息。

(4) DataBaseConnect类

数据库连接类，用于与数据库连接操作的类。

(5) Member UI类

显示用户界面操作相关类。

**5) 类之间关系**

Member类，MemberController类等相关类之间其具体关系如图5.4所示



图5.4 用例模块类图

**6) 文件列表**

所有相关类及其存放位置如下表所示

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| Member.cs | .cs | C#Moddel/Member.cs | 存放用户所有相关基本信息的 |
| MemberController | .cs | C#Controller/MemberController.cs | 存放处理用户信息所有逻辑相关代码。 |
| ClubDbContext.cs | .cs | C#DAL/ClubDbContext.cs | 与数据库连接的接口文件。 |

**7) 功能实现**

用户模块功能时序图，包括登录或注册登相关功能如图5.5所示



图5.5 用例模块时序图

### 5.1.2 社团模块分解描述

**1) 简介**

该功能包括创建社团，审核社团，加入社团，退出社团等操作。

**2) 功能列表**

创建审核社团：普通用户提交创建社团申请并交由管理员审核。

加入退出社团：普通用户可以选择加入或退出社团。

清理社团：管理员在后台选择僵尸社团执行清理操作。

**3) 功能详细描述**

(1) 创建审核社团

普通用户进入个人信息管理选择社团创建，要求填写社团名称，选择社团类别，填写社团申请理由，其中社团名称和社团申请理由要求符合规范格式。管理员进入后台审核，选择同意或拒绝社团的创建。

执行流程如图5.6所示:



图5.6 申请/审核社团流程图

(2) 加入退出社团

普通用户选择加入某一社团，填写加入理由并确认加入，之后等待管理员或社团负责人审核，审核通过后自动成为该社社员。社员也可选择退出社团，确认退出之后即可完成退出社团操作。

执行流程图如图5.7所示:

图5.7 申请/审核社员流程图

(3) 清理社团

管理员登录后台管理，根据积分等排名，选择相应僵尸社团并执行清理操作。完成操作之后即可完成对僵尸社团的清理。

执行流程图如图5.8所示:



图5.8 清理社团流程图

**4) 类**

(1) Club类

用于存储所有与社团相关信息

(2) ClubController类

包含所有与社团信息逻辑处理相关代码。

(3) DB类

显示与数据库相连时读取相关的数据库信息。

(4) Database Connect类

数据库连接类，用于与数据库连接操作的类。

(5) Club UI类

显示所有与社团信息相关处理界面。

**5) 类之间关系**

Club类，ClubController类等相关类之间关系如图5.9所示



图5.9 社团模块类图

**6) 文件列表**

社团模块所有类及其存储位置如下表所示

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| Club.cs | .cs | C#Model/Club.cs | 存放社团所有相关基本信息的 |
| ClubController | .cs | C#Controller/ClubController.cs | 存放处理社团信息所有逻辑相关代码。 |
| ClubDbContext.cs | .cs | C#DAL/ClubDbContext.cs | 与数据库连接的接口文件。 |

**7) 功能实现**

社团模块功能时序图如图5.10所示

图5.10 社团模块时序图

### 5.1.3 活动模块分解描述

**1) 简介**

本模块包括创建活动，加入活动，清理活动等功能。

**2) 功能列表**

创建审核活动：申请活动并审核通过即可完成活动的创建

加入退出活动：普通用户针对某一活动完成加入或退出操作。

终止活动：管理员在管理端提前终止相关活动。

**3) 功能详细描述**

(1) 创建审核活动

普通用户在成为社团负责人之后即可进申请活动，填写活动名称，输入活动申请理由即可完成活动申请，其中活动名称和活动内容格式要求符合标准。提交之后，管理员登录后台管理查看相应活动申请，选择同意或是拒绝。同意即可完成活动申请与创建。

执行流程图如图5.11所示:



图5.11 申请/审核活动流程图

(2) 加入退出活动

普通会员选择加入其他活动，填写申请理由并确认加入即可完成加入操作，加入活动之后，直至活动开始之前选择退出活动即可退出相应活动。

执行流程如图5.12所示:

图5.12 加入/退出活动流程图

(3) 终止活动

管理员登录管理端，选择相应活动，确认终止即可完成终止活动操作，终止活动会删除所有活动相关信息。

执行流程图如图5.13所示:

图5.13 终止活动流程图

**4) 类**

(1) Activity类

用于存储所有与活动相关信息。

(2) ActivityController类

包含所有与社团信息逻辑处理相关代码。

(3) CreateActivityController类

存储所有与申请活动有关的逻辑处理代码。

(4) ActivityMember类

存储所有与加入活动的相关信息。

(5) CreateActivity类

存储所有与申请活动相关信息。

(6) Database Connect类

数据库连接类，用于与数据库连接操作的类。

(7) DB类

显示与数据库相连时读取相关的数据库信息。

(8) Activity UI类

显示所有与活动信息相关页面。

**5) 类之间关系**

活动模块类之间关系如图5.14所示



图5.14 活动模块类图

**6) 文件列表**

活动模块所有相关类及类存放位置如下表所示

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| Activity.cs | .cs | C#Model/Activity.cs | 存放活动所有相关基本信息的 |
| ActivityController | .cs | C#Controller/ActivityController.cs | 存放处理活动信息所有逻辑相关代码。 |
| ClubDbContext.cs | .cs | C#DAL/ClubDbContext.cs | 与数据库连接的接口文件。 |
| CreateActivity.cs | .cs | C#Model/CreateActivity.cs | 存放所有活动申请相关信息。 |
| CreateActivityController.cs | .cs | C#Controller/ActivityController.cs | 存放批准或拒绝活动等信息。 |

**7) 功能实现**

活动模块功能时序图如图5.15所示

 图5.15 活动模块时序图

### 5.1.4 评论模块分解描述

**1) 简介**

本模块包含发布评论以及删除评论等功能。

**2) 功能列表**

发表评论：普通用户对社团或活动发布评论。

删除评论：管理员在管理端删除相应评论。

**3) 功能详细描述**

(1) 发表评论

普通用户在浏览社团或活动详细信息时选择发表评论，输入相应评论后，要求输入评论字数格式符合要求。确认发表之后，即可完成评论的发表。

执行流程图如图5.16所示:



图5.16 发布评论流程图

(2) 删除评论

管理员登录后台管理，进入评论管理，选择相应评论并执行删除操作，完成删除之后。相应评论从评论界面中删去。

执行流程如图5.17所示:



图5.17 删除评论流程图

**4) 类**

(1) Comment类

用于存储所有与评论相关信息。

(2) CommentController类

存储所有与评论信息逻辑处理相关代码。

(3) CommentUI类

用于显示评论相关界面。

(4) Database Connect类

数据库连接类，用于与数据库连接操作的类。

(5) DB类

显示与数据库相连时读取相关的数据库信息。

**5) 类之间关系**

评论模块相关类及类之间关系如图5.18所示



图5.18 评论模块类图

**6) 文件列表**

评论模块相关类存放位置及功能如下表所示

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| Comment.cs | .cs | C#Model/Comment.cs | 存放所有评论相关信息 |
| CommentController | .cs | C#Controller/CommentController.cs | 存放所有处理评论相关逻辑代码 |
| ClubDbContext.cs | .cs | C#DAL/ClubDbContext.cs | 与数据库连接的接口文件。 |

**7) 功能实现**

评论模块功能时序图如图5.19所示

图5.19 评论模块时序图

### 5.1.5 公告模块分解描述

**1) 简介**

本模块包括发布公告，删除公告等功能。

**2) 功能列表**

发布公告：管理员在管理端发布相应公告。

删除公告：管理员在管理端对已发布的公告执行删除操作。

**3) 功能详细描述**

(1) 发布公告

管理员进入管理端发布公告页面，依次输入公告标题，选择公告类别，并输入公告内容，其中公告标题，以及公告内容输入要求符合格式，其中标题字符数量在6-18字符之间。

执行流程图如图5.20所示:



图5.20 发布公告流程图

(2) 删除公告

管理员进入管理端浏览公告页面，选择特定公告，确认删除即可完成删除操作。同时也可以对评论执行批量删除操作。

执行流程图如图5.21所示:



图5.21 删除公告流程图

**4) 类**

(1) News类

用于存储所有发布公告相关信息。

(2) NewsController类

包含所有发布删除公告相关逻辑代码。

(3) NewsUI类

包含所有发布新闻，展示新闻的页面信息。

(4) Database Connect类

数据库连接类，用于与数据库连接操作的类。

(5) Database Connect类

显示与数据库相连时读取相关的数据库信息。

**5) 类之间关系**

公告模块类之间关系如图5.22所示



图5.22 公告模块类图

**6) 文件列表**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| News.cs | .cs | C#Model/News.cs | 存放所有公告有关信息 |
| NewsController | .cs | C#Controller/NewsController.cs | 存放所有公告处理相关逻辑代码。 |
| ClubDbContext.cs | .cs | C#DAL/ClubDbContext.cs | 与数据库连接的接口文件。 |

**7) 功能实现**

公告模块功能实现如图5.23所示

图5.23 公告模块时序图

### 5.1.6 好友模块分解描述

**1) 简介**

本模块包括关注好友和取消关注两个功能。

**2) 功能列表**

关注好友：普通用户可以关注其他用户。

取消关注：普通用户可以取消对其他用户的关注。

**3) 功能详细描述**

(1) 关注好友

普通用户进入其他用户主页，选择关注该名用户即可完成关注，关注之后，被关注用户会出现在关注用户的关注列表中，普通用户可以随时查看关注用户的最新动态，包括创建社团，参与活动等。

执行流程如图5.24所示:



图5.24 关注好友流程

(2) 取消关注

普通用户对已关注的用户可以执行取消关注操作，取消关注之后的用户会自动从用户的关注栏中消失。

执行流程如图5.25所示



图5.25 取消关注流程图

**4) 类**

(1)ApplyFirend类

存储所有申请关注相关信息。

(2)ApplyFirendController类

存储所有关注好友相关逻辑处理代码。

(3)Database Connect类

数据库连接类，用于与数据库连接操作的类。

(4)DB类

显示与数据库相连时读取相关的数据库信息。

(5) ApplyFirendUI类

显示好友申请界面。

**5) 类之间关系**

交友模块类之间关系如图5.26所示



图5.26 交友模块类图

**6) 文件列表**

交友模块相关类存放位置及类具体功能说明如下表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| ApplyFirend.cs | .cs | C#Model/ApplyFirend.cs | 存放所有好友相关信息。 |
| ApplyFirend Controller.cs | .cs | C#Controller/ApplyFirendController.cs | 存放添加关注，取消关注相关逻辑代码。 |
| ClubDbContext.cs | .cs | C#DAL/ClubDbContext.cs | 与数据库连接的接口文件。 |

**7) 功能实现**

好友模块功能时序图如图5.27所示



图5.27 交友模块时序图

## 5.2社团之家接口描述

### 5.2.1用户接口

该系统为B/S相关系统，用户通过浏览器实现交互，显示分辨率包含1920\*1080,1366\*768等多种，由于系统前端采用Bootstrap框架，系统因此具有良好的自适应性，能在不同尺寸的屏幕上查看。

### 5.2.2软件接口

与SqlServer2012连接的接口，通过ADO.NET连接数据库。

### 5.2.3硬件接口

服务器端建议采用专用服务器。

### 5.2.4通讯接口

无特殊要求

## 5.3社团之家总体设计约束

### 5.3.1标准符合性

社团之家系统在编写过程中应严格按照ASP.NET编程规范，另外设计模式严格按照MVC设计标准，使得项目层次结构更加合理。

### 5.3.2硬件约束

社团之家系统要求服务器具有较大的存储容量和较快的运行速度，要求服务器内存至少应达到8G，同时具备至少500G的存储容量。

### 5.3.3技术限制

无特定的技术限制

## 5.4社团之家系统类图

社团之家总共分为10张表，分别对应用户，社团，活动等信息，其中包含用户参与社团，活动表示的一对多等关系，该系统类图如图5.28所示

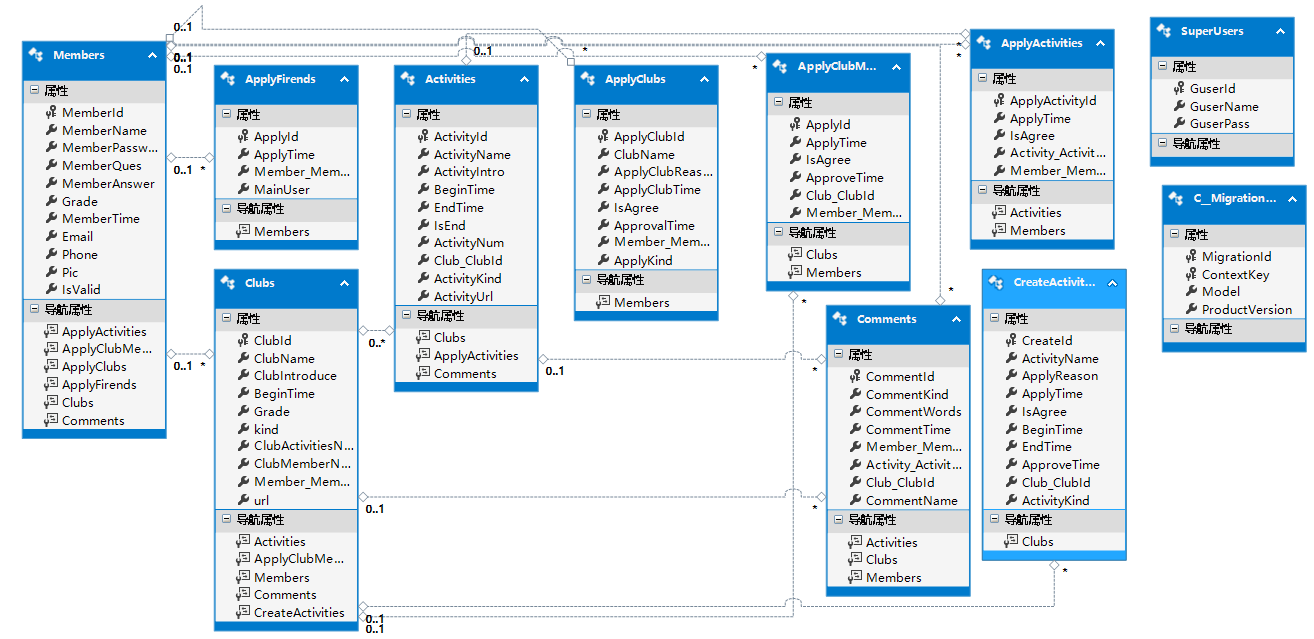


图5.28 社团之家类图

## 5.5本章总结

本章重点对系统详细设计进行描述，包括对系统模块功能的划分以及对各个模块内部详细执行流程，包含类图，流程图，时序图等辅助说明。

# 第6章 系统实现

## 6.1管理端页面实现

### 6.1.1登录页面实现

管理人员登录后台管理系统后，账户名称将存入浏览器的cookie中，并且一直保持该登录状态，直到用户关闭浏览器退出为止。登录成功之后，系统会自动显示相应管理权限和功能。后台管理端登录页面如图6.1所示。



图6.1 后台系统登录

部分核心代码:

Gpassword = Md5(Gpassword); //对密码进行MD5加密

var User = db.SuperUsers.SingleOrDefault(m => m.GuserName == Guser && m.GuserPass == Gpassword);

//与数据库比对账号密码找寻用户

if(User!=null) //如果找到该用户

{

//FormsAuthentication.SetAuthCookie(Guser,false);

FormsAuthenticationTicket ticket = new FormsAuthenticationTicket //阻止匿名访问

(1,

Guser,

DateTime.Now,

DateTime.Now.AddMinutes(10),

true,

"7,1,8",

"/"

); //设置浏览器访问cookie

var Gcookie = new HttpCookie(FormsAuthentication.FormsCookieName, FormsAuthentication.Encrypt(ticket)); //创建新cookie对象

Gcookie.HttpOnly = true; //防止他人篡改

HttpContext.Response.Cookies.Add(Gcookie);

HttpCookie cookie = new HttpCookie("NewUser");

cookie.Values.Add("UserId",User.GuserId.ToString());

cookie.Expires = DateTime.Now.AddMinutes(10);

Response.AppendCookie(cookie); //添加cookie至浏览器中

return RedirectToAction("Index", "Home"); //返回主页

### 6.1.2管理页面实现

管理员登录成功之后进入具体管理页面，其中左边部分为提示栏，管理员选择提示栏中的不同选项，系统会自动跳转至不同的管理页面。包括社团管理，社员管理等。相应数据显示于不同管理页面，其中在数据栏结尾显示一系列操作选项，包括查看具体数据，修改社团信息，删除社团信息等，可以依据管理员的选择执行相应操作。其具体显示页面如图6.2所示。

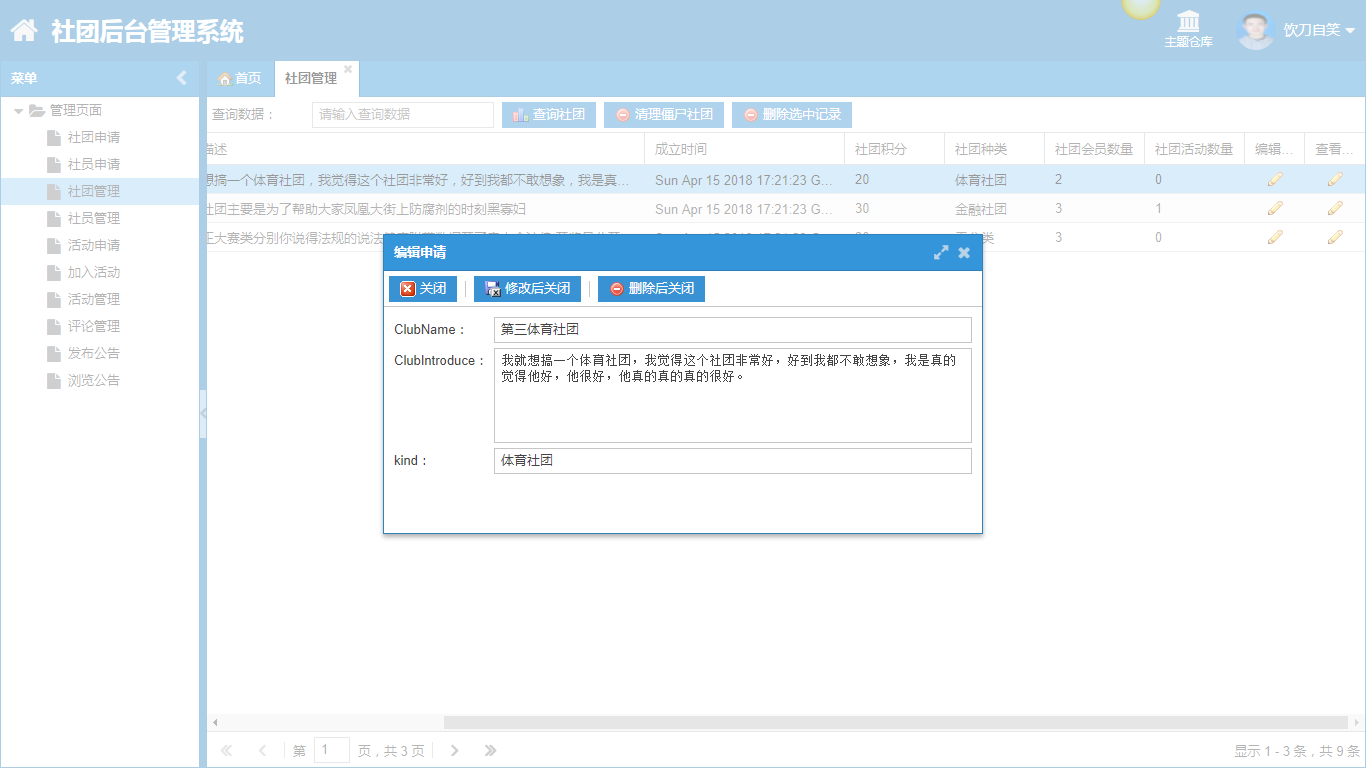


图6.2 后台管理界面

## 6.2用户端页面实现

### 6.2.1登录/注册页面实现

普通游客进入系统主页面，依次选择登录或注册后，系统会跳转至登录或注册页面，其中在注册页面要求输入用户名，密码，密保问题，密保答案等信息，系统自动比对注册信息是否符合格式，若匹配出错则不予注册。登陆要求用户输入用户名密码并比对数据库，若数据吻合完成登录，登录账号名存入浏览器Cookie中，保存时间为1天，若用户选择退出登录，则自动删除cookie信息，界面显示如图6.3所示。



图6.3 注册页面

### 6.2.2社团信息页面实现

用户登录成功后，选择某一具体社团进入，来到社团具体展示页面，此时系统会自动判断该用户是否为社团负责人，若是，页面显示已加入社员和未加入社员，用户可选择批准加入社团或踢出社团，若不是负责人，则页面显示加入社团选项，用户可以选择是否加入该社团。若已加入，则显示已经加入该社团。具体页面显示如图6.4所示。



图6.4 社团详细信息

### 6.2.3活动信息页面实现

用户进入具体活动展示页面，页面显示活动详情，活动内容以及活动介绍，若该名用户不是活动发起人，则可以选择加入该活动，若活动已结束，则页面显示活动已结束。此外也能发表相应评论。选择页面快速发表，输入发表内容发表即可。具体显示页面如图6.5所示。



图6.5 活动页面

### 6.2.4公告信息页面实现

用户进入系统公告浏览页面，系统显示管理员所发送的公告列表，用户可选择相应的公告进行浏览。具体浏览页面如图6.7所示。



图6.7 公告页面

## 6.3本章总结

本章重点对系统实现做出了说明，包括系统的实现界面，实现代码等内容，实现代码中又包含了相应注释，借由本章描述，可以将系统的实现过程以及效果较为清晰地表达出来。

# 第7章 系统测试

## 7.1测试计划

社团之家主要采用黑盒测试方法，用以验证该平台的功能是否满足学生社团管理的功能需求。具体测试安排如下：

|  |  |
| --- | --- |
| 测试目标 | 系统可以完成所有社团处理事务，且系统具有一定的安全性。 |
| 测试内容 | 主要测试包含用户信息，社团事务，活动事务等功能。 |
| 测试方法 | 主要采用黑盒测试，包括边界值分析等。 |
| 测试前提 | 系统已完成编译并能正常运行，同时软硬件环境配置正确。 |
| 完成标准 | 核心功能完全覆盖，基本功能全部实现。 |
| 特殊事项 | 不同用户的权限问题，ASP.NET安全性问题。 |

## 7.2系统测试

### 7.2.1用户模块测试

**1) 登录/注销/注册测试**

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 01 | 测试需求项 | 用户注册信息，登录以及注销 |
| 测试需求标识 | User |
| 测试目标状态及测试数据 | | 达到预期效果 | |
| 序号 | 测试项 | 输入说明(操作) | 输出说明(预期结果) |
| 1 | 登录系统 | 输入正确的用户名和密码 | 登录系统成功，并跳转至个人主页 |
| 输入错误的用户名和密码 | 显示“登录失败” |
| 直接输入空格 | 提示请输入数据 |
| 2 | 退出系统 | 点击退出按钮 | 提示退出成功并回到主页 |
| 3 | 注册信息 | 输入符合要求的所有数据 | 提示注册成功并回到主页 |
| 邮箱或密码等数据不符要求 | 提示输入正确格式的数据 |

**2) 修改头像测试**

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 02 | 测试需求项 | 用户修改头像测试 |
| 测试需求标识 | Pic |
| 测试目标状态及测试数据 | | 达到预期效果 | |
| 序号 | 测试项 | 输入说明(操作) | 输出说明(预期结果) |
| 1 | 修改头像 | 选择正确格式图片并上传 | 头像修改成功，并跳转个人主页 |
| 选择错误格式或大小文件上传 | 系统提示修改失败 |
| 空白上传 | 提示请选择文件 |

### 7.2.2社团模块测试

**1) 创建/加入社团测试**

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 03 | 测试需求项 | 用户创建/加入社团测试 |
| 测试需求标识 | Club |
| 前置条件 | | 用户已成功登陆系统 | |
| 测试目标状态及测试数据 | | 达到预期效果 | |
| 序号 | 测试项 | 输入说明(操作) | 输出说明(预期结果) |
| 1 | 创建社团 | 正确输入选择数据并提交 | 系统提示申请成功并正在等待管理员审核 |
| 输入错误格式数据 | 系统提示申请失败 |
| 空白提交 | 提示请输入数据 |
| 2 | 加入社团 | 正确输入理由并提交 | 提示申请成功，正在等待管理员审核。 |
| 空白输入提交 | 提示请填写申请理由 |

**2) 审核社团测试**

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 04 | 测试需求项 | 管理员后台审核社团申请 |
| 测试需求标识 | Club |
| 前置条件 | | 管理员已成功登陆后台系统 | |
| 测试目标状态及测试数据 | | 达到预期效果 | |
| 序号 | 测试项 | 输入说明(操作) | 输出说明(预期结果) |
| 1 | 审核社团 | 在待审核社团中执行审核操作 | 系统提示审核成功并成功创建社团 |
| 选择已审核社团进行审核 | 系统提示该社团已经审核 |

### 7.2.3活动模块测试

**1) 创建/加入活动测试**

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 05 | 测试需求项 | 用户创建/加入活动 |
| 测试需求标识 | Acticity |
| 前置条件 | | 用户已成功登陆系统 | |
| 测试目标状态及测试数据 | | 达到预期效果 | |
| 序号 | 测试项 | 输入说明(操作) | 输出说明(预期结果) |
| 1 | 申请活动 | 正确输入活动名称，内容等信息 | 系统提示申请成功并正在等待管理员审核 |
| 输入不符格式的活动信息 | 系统提示申请失败并请重新输入 |
| 空白提交 | 提示请输入数据 |
| 2 | 加入活动 | 选择未开始活动，输入加入理由并提交 | 提示加入活动成功并返回个人主页面 |
| 选择已结束活动提交申请 | 系统提示活动已结束请重新选择尚未开始的活动 |
| 空白提交 | 提示请重新输入数据 |

**2) 审核创建/加入活动测试**

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 06 | 测试需求项 | 管理员审核创建或加入活动申请 |
| 测试需求标识 | Acticity |
| 前置条件 | | 管理员成功登陆后台管理系统 | |
| 测试目标状态及测试数据 | | 达到预期效果 | |
| 序号 | 测试项 | 输入说明(操作) | 输出说明(预期结果) |
| 1 | 审核创建活动申请 | 正确选择待审核活动并执行操作 | 系统提示审核成功，活动创建成功 |
| 选择已审核活动并执行操作 | 系统提示该活动已被审核并要求重新选择 |
| 空白提交 | 提示请输入数据 |
| 2 | 审核加入活动 | 正确选择待加入会员执行审核 | 系统提示审核成功并返回浏览页面 |
| 选择已审核会员执行审核 | 系统提示该会员已被审核并提示重新审核 |
| 空白提交 | 提示请重新输入数据 |

## 7.3本章总结

本章重在对系统进行测试，本章选用了6个测试用例，重点对用户模块，社团模块和活动模块这几个核心模块进行测试，经过测试发现，所有的测试结果都符合预期。

# 第8章 总结展望

## 8.1相关难点及创新点

本次毕业设计中，由于许多编程技术是第一次使用，在第一次运行时都出现了不少问题。比如初次用Entity Formwork连接SqlServer2012时，出现了数据库无法创建，数据库创建之后无法连接，数据库中出现乱码等问题，通过上网查阅资料解决。另外在前端方面，由于相关技术不熟悉，为了尽量保持页面美观，也查阅了不少相关资料。本系统在编写时，运用了一些创新点如下。

1) 运用积分制来实行综合排名，总结出相应明星社团和僵尸社团。积分主要来自于社团互动数量，社员参与度等。

2) 后台管理员根据社团排名，定期清理垫底的僵尸社团，从而提高社团参与积极性。

## 8.2问题以及改进措施

本次毕设由于仅由一人完成，因此存在很多不完善的地方，例如该系统的论坛版块功能较为欠缺，社区互动气氛营造较差等问题，在以后的项目改进方面，我将着重改进以下三方面的内容。

1) 强化论坛版块，尽量营造友好交流的社团活跃气氛。

2) 开启推送机制，设置实时推送。在会员申请社团发起后，申请消息会实时推送到管理员处，管理员会马上给予相关回复。

3) 继续美化前端页面。

# 参考文献

1. 杨帆, 李朝阳, 许庆豫. 高校学生社团的学生评价与影响因素[J]. 教育研究, 2015, 37(12):43-51.
2. 胡继东. 我国高校学生社团发展动力及其引导策略研究[D]. 大连: 大连理工大学, 2012.
3. 王珩. 高校学生社团发展调查报告——以浙江省高校为例[J]. 中国青年政治学院学报, 2007, 26(03):35-39.
4. 常青. 高校学生社团问题研究[D]. 吉林: 东北师范大学, 2012.
5. 王运东, 刘桂宇. 社会管理理念视域下的高校学生社团管理模式创新[J]. 思想教育研究, 2013,25(09):102-105.
6. 黄宁. 当前高校社团面临的困境与对策研究[J]. 长沙大学学报, 2006, 20(04):156-158.
7. 陈星. 论高校学生社团管理的信息化建设[J]. 旅游纵览(下半月)，2013，3(11):216.
8. 殷人昆, 郑人杰. 实用软件工程(第三版) [M]. 北京:清华大学出版社, 2014:46-47.
9. 罗福强, 白忠建, 杨剑. Visual C#.NET程序设计教程(第2版) [M]. 北京:人民邮电出版社, 2012:32-33.
10. 邹琼俊. ASP.NET MVC企业级实战[M]. 北京:清华大学出版社, 2017:1-2.
11. 马骏. ASP.NET MVC程序设计教程（第3版）[M]. 北京:人民邮电出版社, 2015:1-2.
12. Jon Galloway, Brad Wilson Scott Allen, David Matson. ASP.NET MVC 5 高级编程（第五版）[M]. 北京:清华大学出版社, 2015:3-4.
13. Jon Skeet. 深入理解C#(第三版)[M]. 北京:人民邮电出版社, 2015:1-2.
14. 蒋金楠. ASP.NET MVC 5 框架揭秘[M]. 北京:电子工业出版社, 2014:1-2.
15. Jamie Munro. 基于Bootstrap和Knockout.js的ASP.NET MVC开发实战[M]. 北京:中国电力出版社, 2015:36-37.
16. Benjamin Perkins, Jacob Vibe Hammer, Jon Dried. C#入门经典(第7版) [M]. 北京:清华大学出版社, 2016:8-9.
17. 沈金菽，谷敬华. Visio在科技期刊插图制作中的应用 [J].编辑学报, 2001, 13(01):45-47.
18. 闫旭. 浅谈SQL Server数据库的特点和基本功能[J].价值工程, 2012 , 31(22):229-231.
19. 江凌，杨平利，杨梅，袁媛. 基于ADO.NET技术访问SQL Server数据库的编程实现[J]. 现代电子技术, 2015,38(08):95-98.
20. 秦继伟. 当代大学生网络自律意识教育研究[D]. 湖南: 湖南师范大学, 2013.

致谢

在毕业设计（论文）即将完成时，回望毕设和论文编撰的2个多月时间里，我十分感谢指导老师和学院其他老师对我项目论文的指导，无论是在毕设的选题，毕设的制作或是论文的编写过程中，我向学院老师们咨询了大量与业务或技术相关的问题，都得到了老师们耐心细致的解答，对此我深表谢意。同时我也感谢与我一起完成毕设的同学，大家共同讨论，相互帮忙，虽然各自的毕设题目不一致，但是大家互相协作，为了共同的目标一起奋斗，加油。

另外，我也想感谢我自己，感谢自己这两个多月的付出，所有努力也最终凝结到了这篇毕业论文上。

最后，我由衷感谢参与评阅本篇论文的各位专家老师们。

# 附录A 部分源代码

**1) 社团模块核心代码**

public class ClubController : Controller

{

// GET: Club

private ClubDbContextEntities db = new ClubDbContextEntities();

private static int PAGE\_SIZE = 3;

private static string Clubid;

//进入首页按键

[HttpGet]

public ActionResult Index()

{

var Club = (from u in db.Clubs select u).ToList<Clubs>();

ViewBag.ClubDataSource = Club.Take(PAGE\_SIZE);

ViewBag.ClubRecordCount = Club.Count();

return View();

}

//点击下方翻页按键

[HttpPost]

public ActionResult Club\_PageIndexChanged(JArray Club\_fields,int Club\_pageIndex)

{

var Club = (from u in db.Clubs select u).ToList<Clubs>();

var grid= UIHelper.Grid("Club");

ViewBag.ClubDataSource=Club.OrderBy(m=>m.ClubName).Skip(Club\_pageIndex \* PAGE\_SIZE).Take(PAGE\_SIZE);

ViewBag.ClubRecordCount = Club.Count();

grid.RecordCount(ViewBag.ClubRecordCount);

grid.DataSource(ViewBag.ClubDataSource,Club\_fields);

return UIHelper.Result();

}

//点击详细按钮编辑方法

[HttpGet]

[Authorize(Roles="1")]

public ActionResult Edit(string id)

{

Guid G\_uid = new Guid(id);

Clubs club = db.Clubs.SingleOrDefault(m => m.ClubId == G\_uid);

if (club == null)

{

return HttpNotFound();

}

else

{

return View(club);

}

}

**2) 活动模块核心代码**

public class ActivityController : Controller

{

// GET: Activity

private ClubDbContextEntities db = new ClubDbContextEntities();

private static int PAGE\_SIZE = 3;

/// <summary>

/// 活动主页页面

/// </summary>

/// <returns></returns>

//GET

[HttpGet]

public ActionResult Index()

{

var Activity = (from u in db.Activities select u).ToList<Activities>();

ViewBag.ActivityDataSource = Activity.Take(PAGE\_SIZE);

ViewBag.ActivityRecordCount = Activity.Count();

return View();

}

//翻页方法

[HttpPost]

public ActionResult Activity\_PageIndexChanged(JArray Activity\_fields, int Activity\_pageIndex)

{

var Activity = (from u in db.Activities select u).ToList<Activities>();

var grid = UIHelper.Grid("Activity");

ViewBag.ActivityDataSource = Activity.OrderBy(M=>M.ActivityName).Skip(Activity\_pageIndex \* PAGE\_SIZE).Take(PAGE\_SIZE);

ViewBag.ActivityRecordCount = Activity.Count();

grid.RecordCount(ViewBag.ActivityRecordCount);

grid.DataSource(ViewBag.ActivityDataSource, Activity\_fields);

return UIHelper.Result();

}

# 附录B 英文翻译

Introduction to ASP.NET Core

ASP.NET Core is a cross-platform, high-performance, open-source framework for building modern, cloud-based, Internet-connected applications. With ASP.NET Core, you can:

* Build web apps and services, IoT apps, and mobile backends.
* Use your favorite development tools on Windows, macOS, and Linux.
* Deploy to the cloud or on-premises.
* Run on .NET Core or .NET Framework.

Why use ASP.NET Core?

Millions of developers have used (and continue to use) ASP.NET 4.x to create web apps. ASP.NET Core is a redesign of ASP.NET 4.x, with architectural changes that result in a leaner, more modular framework.

ASP.NET Core provides the following benefits:

* A unified story for building web UI and web APIs.
* Integration of modern, client-side frameworks and development workflows.
* A cloud-ready, environment-based configuration system.
* Built-in dependency injection.
* A lightweight, high-performance, and modular HTTP request pipeline.
* Ability to host on IIS, Nginx, Apache, Docker, or self-host in your own process.
* Side-by-side app versioning when targeting .NET Core.
* Tooling that simplifies modern web development.
* Ability to build and run on Windows, macOS, and Linux.
* Open-source and community-focused.

ASP.NET Core ships entirely as NuGet packages. Using NuGet packages allows you to optimize your app to include only the necessary dependencies. In fact, ASP.NET Core 2.x apps targeting .NET Core only require a single NuGet package. The benefits of a smaller app surface area include tighter security, reduced servicing, and improved performance.

Build web APIs and web UI using ASP.NET Core MVC.

ASP.NET Core MVC provides features to build web APIs and web apps:

* The Model-View-Controller (MVC) pattern helps make your web APIs and web apps testable.
* Razor Pages (new in ASP.NET Core 2.0) is a page-based programming model that makes building web UI easier and more productive.
* Razor markup provides a productive syntax for Razor Pages and MVC views.
* Tag Helpers enable server-side code to participate in creating and rendering HTML elements in Razor files.
* Built-in support for multiple data formats and content negotiation lets your web APIs reach a broad range of clients, including browsers and mobile devices.
* Model binding automatically maps data from HTTP requests to action method parameters.
* Model validation automatically performs client- and server-side validation.

Client-side development

ASP.NET Core integrates seamlessly with popular client-side frameworks and libraries, including Angular, React, and Bootstrap. For more information, see Client-side development.

ASP.NET Core targeting .NET Framework

ASP.NET Core can target .NET Core or .NET Framework. ASP.NET Core apps targeting .NET Framework aren't cross-platform—they run on Windows only. There are no plans to remove support for targeting .NET Framework in ASP.NET Core. Generally, ASP.NET Core is made up of .NET Standard libraries. Apps written with .NET Standard 2.0 run anywhere that .NET Standard 2.0 is supported.

There are several advantages to targeting .NET Core, and these advantages increase with each release. Some advantages of .NET Core over .NET Framework include:

* Cross-platform. Runs on macOS, Linux, and Windows.
* Improved performance
* Side-by-side versioning
* New APIs
* Open source

We're working hard to close the API gap from .NET Framework to .NET Core. The Windows Compatibility Pack made thousands of Windows-only APIs available in .NET Core. These APIs weren't available in .NET Core 1.x.

译文

ASP.NET Core 简介

ASP.NET Core 是一个跨平台的高性能开源框架，用于生成基于云且连接 Internet 的新式应用程序。 使用 ASP.NET Core，您可以：

* 建置 Web 应用程式和服务、IoT 应用和移动后端。
* 在 Windows、macOS 和 Linux 上使用喜爱的开发工具。
* 部署到云或本地。
* 在 .NET Core 或 .NET Framework 上运行。

为何使用 ASP.NET Core

数百万开发人员使用过（并将继续使用）ASP.NET 4.x 创建 Web 应用。 ASP.NET Core 是重新设计的 ASP.NET 4.x，更改了体系结构，形成了更精简的模块化框架。

ASP.NET Core 具有如下优点：

* 生成 Web UI 和 Web API 的统一场景。
* 集成新式客户端框架和开发工作流。
* 基于环境的云就绪配置系统。
* 内置依赖项注入。
* 轻型的高性能模块化 HTTP 请求管道。
* 能够在 IIS、Nginx、Apache、Docker 上进行托管或在自己的进程中进行自托管。
* 定目标到 .NET Core 时，可以使用并行应用版本控制。
* 简化新式 Web 开发的工具。
* 能够在 Windows、macOS 和 Linux 进行生成和运行。
* 开放源代码和以社区为中心。

ASP.NET Core 完全作为 NuGet 包的一部分提供。 借助 NuGet 包，可以将应用优化为只包含必需的依赖项。 实际上，定目标到 .NET Core 的 ASP.NET Core 2.x 应用只需要使用一个 NuGet 包。 较小的应用图面区域的优势包括：提升安全性、减少维护和提高性能。

使用 ASP.NET Core MVC 生成 Web API 和 Web UI

ASP.NET Core MVC 提供生成 Web API 和 Web 应用所需的功能：

* Model-View-Controller (MVC) 模式 使 Web API 和 Web 应用可测试。
* ASP.NET Core 2.0 中新增的 Razor 页面是基于页面的编程模型，可简化 Web UI 生成并提高工作效率。
* Razor 标记提供了适用于 Razor 页面和 MVC 视图的高效语法。
* 标记帮助程序使服务器端代码可以在 Razor 文件中参与创建和呈现 HTML 元素。
* 内置的多数据格式和内容协商支持使 Web API 可访问多种客户端，包括浏览器和移动设备。
* 模型绑定自动将 HTTP 请求中的数据映射到操作方法参数。
* 模型验证自动执行客户端和服务器端验证。

客户端开发

ASP.NET Core 与常用客户端框架和库（包括 Angular、React 和 Bootstrap）无缝集成。 有关详细信息，请参阅客户端开发。

面向 .NET Framework 的 ASP.NET Core

ASP.NET Core 可以面向 .NET Core 或 .NET Framework。 面向 .NET Framework 的 ASP.NET Core 应用无法跨平台，它们仅在 Windows 上运行。 没有计划删除 ASP.NET Core 中对面向 .NET Framework 的支持。 通常，ASP.NET Core 由 .NET Standard 库组成。 使用 .NET Standard 2.0 编写的应用可在 NET Standard 2.0 支持的任何位置运行。

面向 .NET Core 有以下几个优势，并且这些优势会随着每次发布增加。 与 .NET Framework 相比，.NET Core 的部分优势包括：

* 跨平台。 在 macOS、Linux 和 Windows 上运行。
* 提高的性能
* 并行版本控制
* 新 API
* 打开源

我们正努力缩小 .NET Framework 与 .NET Core 的 API 差距。 Windows 兼容性包使数千个仅 Windows API 可在 .NET Core 中使用。 这些 API 在 .NET Core 1.x 中不可用。