分 类 号： TP311

学校代码： 11460

学　　号： 14550135

南京晓庄学院本科生毕业论文

基于Nodejs + Express的校园社区的设计与实现

Design And Implementation Of Campus Community Based On Nodejs and Express

所 属 学 院：信息工程学院

学 生 姓 名：周 涛 春

指 导 教 师：徐 家 喜

职　　称：高级工程师

研究起止日期：二〇一七 年 十 月 至 二〇一八 年 五 月

二〇一八 年 五 月

学位论文独创性声明

本人郑重声明:

1. 坚持以“求实、创新”的科学精神从事研究工作。

2. 本论文是我个人在导师指导下进行的研究工作和取得的研究成果。

3. 本论文中除引文外，所有实验、数据和有关材料均是真实的。

4. 本论文中除引文和致谢的内容外，不包含其他人或其它机构已经发表或撰写过的研究成果。

5. 其他同志对本研究所做的贡献均已在论文中作了声明并表示了谢意。

作者签名:

日　　期:

【摘要】

随着信息化社会的不断深入发展，智慧校园成为学校运营发展的一种新模式。本文在互联网+、移动互联网业务迅猛发展的背景下，分析了传统校园运营发展模式的缺点，包括信息量小、互动性弱、速度慢等缺点，提出了信息化校园的观点。

校园是信息交流比较活跃的环境之一，校园电子服务系统作为一种综合性的校园管理系统，更多的为学校教务性质的业务服务，对于校园中学生多样化生活，却有很少涉及；就目前现状，校园中正是缺少这样专注化的校园交流社区，为在校师生提供一个专属的多样化校园生活圈。

本文以Nodejs和Express为主要技术，使用Visual Studio Code开发工具，并结合MongoDB数据库进行数据存储设置，以此来实现一个基于互联网的校园信息交流社区系统；本系统包括校园论区、休闲区、表白墙、用户信息管理、管理员系统配置管理等功能。本文还详细介绍了，实现该系统的相关技术、系统分析、总体设计以及具体实现等内容。

【关键词】: Node.js、Express、MongoDB数据库、校园生活社区

【Abstract】

With the development of information society, smart campus has become a new mode of operation and development. In the background of the rapid development of Internet accelerated speed and mobile Internet, this paper analyzes the shortcomings of the traditional campus operation and development mode, including the disadvantages of small amount of information, weak interactivity and slow speed, and puts forward the viewpoints of informationization campus.

Campus is one of the more active environments for information exchange. As a comprehensive campus management system, campus electronic service system is more of a business service, but, there are few references to the diversity of life on campus. That's why it's the lack of focus, targeted campus interactions, to provide a unique and diverse campus life for the faculty and students.

This paper takes Nodejs and Express as the main technology, uses the Visual Studio Code development tool, and combines with the MongoDB database for data storage settings to realize an internet-based campus information exchange community system. This system includes campus discussion area, leisure area, confession wall, user information management, administrator system configuration management and other functions. This paper also introduces the related technology, system analysis, overall design and implementation of the system.

【Key words】：Node.js、Express、MongoDB database、Campus living community

目录

1 绪论

随着信息化社会的不断深入发展，智慧校园成为学校运营发展的一种模式，而校园社区作为一种新潮的获取和交流的渠道，不断的受到广大学校和师生的的欢迎。结合传统的校园教务系统，建立一个专注于互动性的生活休闲交流平台，是顺应时代要求的产品，本文将从本章开始进行课题的研究与分析。

1.1 课题背景和现状

校园是信息交流比较活跃的环境之一，校园广播、校园报刊、BBS 等作为校园信息的来源渠道都受到了在校师生的欢迎和广泛的使用。但随着互联网的发展，校园广播和校园报刊逐渐成为时代的后属品，而基于网络平台的 BBS 却得到越来越多的关注。校园电子服务系统作为一种综合性的校园管理系统，更多的是针对学校教务性质的管理，对于校园中学生的多样化生活，却有很少涉及；所以校园中正是缺少这样专注化针对性强的校园社区，来为在校师生提供了一个共享的生活圈。

目前，国内的主流论坛有腾讯、新浪、网易、百度等一些综合性比较强的论坛社区，用户可以很轻松在这些论坛中获取到大量的各方面的信息，但是这些论坛的商业性强，涉及面广，内容繁杂，不容易管理。国外有一些专注性和针对性都比较强的论坛社区，例如 Facebook、Twitter 等一些流行网站，也同样是受到很多用户的喜欢；所以并不是综合性强的论坛才会被用户所接受，一些专注性的论坛也同样是受到很多用户的青睐和喜欢。所以就目前我国的状态来看，我们急切的希望出现一些有针对性的论坛社区带给我们专业化的讨论和交流。

目前我国各大高校几乎都有自己的校园管理系统，也趋向于成熟，像晓庄学院的信息服务平台和 i 晓庄 App 的出现都给学生的生活带来了很大的方便，晓庄学院的信息服务平台主要是针对校园教务系统的管理平台，i 晓庄的出现是一个校园生活多样化的一个重大的改变，它不仅仅集成了晓庄学院信息服务平台中教务系统服务功能，而且还增添了一些其他的校园服务型模块，像校园的班车服务、付款码、手机充值等等一些贴近校园生活的服务，综合来说是一款非常强大的校园服务型移动 App。但是对于学生来说可能只是一个校园的服务平台，并不具有校园生活多样化的种种体现。就目前的情况来看，开发一款专注化针对性强的校园社区是非常有必要的一种选择。

1.2 课题研究目的和意义

校园社区旨在丰富在校师生的生活，让校园生活圈处在实时的信息化平台下。为在校师生提供信息、舆论、交友和休闲等一些互动平台。

从课题的现状分期看来，相对于传统形势下的校园运营发展模式，快节奏、强互动的“智慧校园”更加吸引师生的注意，不仅满足了他们在生活上的快节奏，也以高效、底本、便捷的优势丰富了校园的多样化生活；一方面，传统形式下，校园的信息通常是以报刊和广播的形式进行宣传交流，这样的信息形式不仅减慢了人们的生活节奏，而且在宣传成本方面也付出了一定的代价；想要得到实时的消息，就必须付出等价的成本通过一定的渠道来获取。而在网络化的今天，获取到实时的信

息，对我们来说却是轻而易举的事情；另一方面，现如今，很多高校都有自己的教务性质的管理系统，但是对于校园生活圈的社区论坛却有很少涉及。教务性质的系统针对性质强，系统以服务在校师生办理教务性质的业务为主，却忽略了广大师生在生活上的交流互动；所以建立这样一个生活休闲，交友互动的平台不仅仅是时代的要求，也是我们丰富休闲生活的必需品。

从高校生的生活角度出发，现如今在校大学生的生活更多的依赖于网络环境，善于用网络获得实时信息，他们在网络环境中活动主要集中在交友聊天、查找资料、收发邮件、浏览新闻、娱乐消遣等。网上生活成为大学生的业余生活的重要部分。但是现如今国内的论坛内容更多的趋向于商业和娱乐性质，缺少了校园生活的气息。开发一个以校园为中心、针对性强的校园休闲娱乐平台是目前现状不可或缺的需求。

综上，以校园为中心的休闲娱乐平台，不仅仅可以满足在校大学生的快节奏的生活，也增强了在校师生的之间的互动，是一种可持续发展的运营模式。

1.3 课题研究内容和方法

1.3.1 课题研究内容

1. 校园话题社区
   - 该模块主要针对的是校园话题论坛的实现；提供贴吧的关注、订阅、动态更新、评论回复、实时通信聊天等功能。
   - 主要研究点：websocket 实时通信、帖子的评论回复、数据库设计。
2. 表白墙
   - 该模块提供表白的功能:发表表白贴、支持、评论等功能。
   - 主要研究点：表白墙的支持率计算、表白贴的排序。
3. 生活休闲区
   - 该区域提供生活娱乐助手功能：校园新闻、音乐、星座、天气等。
   - 主要研究点：校园新闻的发表、数据库设计、请求第三方站点的跨域。
4. 个人信息模块
   - 该区域主要是是用户信息管理模块：账户设置、帖子管理、贴吧管理等。
   - 主要研究点：消息列表的实时更新。
5. 管理模块
   - 该模块提供系统的后台管理配置的功能：系统消息的发送、所有用户管理、所有帖子管理、校园新闻管理。
   - 主要研究点：Nodejs 操作 MongoDB 数据库。

1.3.2 课题研究方法

1. 文献书籍研究法

在进行开题报告和课题研究的初期，通过对本课题的分析和理解，在中国知网和校园图书馆查阅了相关的文献和书籍，同时也查找借鉴了相似的课题研究和设计方式，收集相关的资料，为开发阶段提供一些理论和技术的支持。主要针对 Nodejs 和 Express 程序设计案例、MongoDB 数据库存储技术进行一些探究和整理。

2. 调查法

针对校园的广大学生进行需求调查，以获得真实的用户体验和准确的数据，为解决问题提供依据。主要针对大学生对于校园生活互动的需求。

3. 经验总结法

一方面在研究的过程中，不断发现和总结问题，不断推敲实现方法的可行性和稳定性，争取做到程序运行如预期所想的一样顺利，在经验中学习，在总结中进步。另一方面通过和指导老师沟通交流，对需求明确化、细节化，做好开发前的准备工作。

2 开发工具及技术

上一章中分析了本课题的现状和本系统开发目的，对课题有了一定的了解，下面将对实现系统所需要相关开发工具和技术进行介绍。本系统主要基于 Node.js 和 Express 框架技术进行 web 端开发。开发工具主要是 Visual Studio Code 集成开发工具，以 Node.js 为主要的后台语言、前端主要以 Vue 为主要框架、存储平台主要采用 MongoDB 数据库，最后整个校园论坛采用 B/S 的架构形式进行开发，通过这样的方式为互联网提供一个完整的校园论区服务。

2.1 开发平台

2.1.1 Visual Studio Code 安装和配置（本节以 mac OS 系统为例进行安装配置）

1. 官网下载 Visual Studio Code 安装包，对照计算机系统下载正确的安装包 https://code.visualstudio.com/Download
2. 下载后，解压文件，直接打开 Visual Studio Code 文件。
3. Visual Studio Code 的配置。

   打开 Visual Studio Code 最左侧菜单的“扩展”进行插件的安装，安装插件： HTML Boilerplate、CSS Peek、HTML CSS Support、Auto Close Tag、JavaScript Snippet Pack、Vetur 等可以提高开发效率的插件。

2.1.2 Nodejs 安装和配置（本节以 mac OS 系统为例）

1. 安装 Homebrew
   - 打开终端输入：/usr/bin/ruby–e“$(curl–fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)”
2. 安装 Node.js
   - 打开终端键入：brew install node。
   - 检测 node 是否安装成功：终端键入：node –v，执行结果如图 2-2 所示。

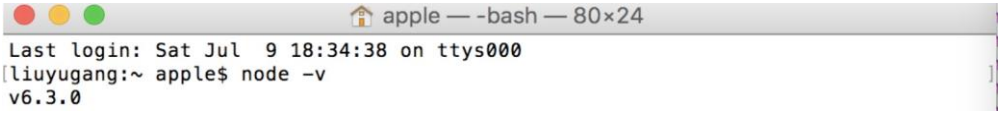

图 2-2 node 安装成功

2.1.3 MongoDB 数据库安装和配置

1. 终端输入：brew install mongodb
2. 在启动 MongoDB 之前，需要创建一个目录为 MongoDB 的默认的数据写入库，并赋予文件夹读和写的权限。
   - 终端键入：mkdir –p /data/db
   - 终端键入：chown ‘id -u’ /data/db

3. 执行启动

- 启动 MongoDB 数据库：终端中输入命令 mongodb
- 连接数据库：终端中输入命令 mongo

2.2 开发技术介绍

2.2.1 Node.js 编程语言

Node.js 是基于 Chrome JavaScript 运行时建立的一个平台，底层使用 C++语言编写，实际上它是对 Google Chrome V8 引擎进行了封装。它是一个 Javascript 运行环境(runtime)，虽然是使用 javascript 语言编写，但是它并不是 javascript 的框架，而是让 javascript 编写的脚本运行在服务端的开发平台，主要用于创建快速的、可扩展的网络应用。

Node.js 采用事件驱动和非阻塞 I/O 模型，以轻微、高效的优势构建运行在分布式设备的数据密集型的实时应用；Nodejs 官方提供很多模块，每个模块都是具有各自特定功能的 js 文件，如操作文件的 fs 模块，构建 http 服务的 http 模块等等。

2.2.2 基于 Nodejs 的 Express 框架

Express 是一个简洁而灵活的 Nodejs Web 应用框架，提供了一系列强大特性和丰富的 HTTP 工具。在 web 方面，Express 是一个基于 Node.js 平台的极简、灵活的 web 应用开发框架，它提供一系列强大的特性，帮助创建各种 web 和移动设备应用。在 API 方面，丰富的 HTTP 快捷方法和任意排列组合的 Connect 中间件，让创建健壮、友好的 API 变得既快速又简单。在性能方面，Express 不对 Node.js 已有的特性进行二次抽象，只是在它之上扩展了 Web 应用所需的基本功能。

Express 框架核心特性：

- 可以设置中间件来响应 HTTP 请求。
- 定义了路由表用于执行不同的 HTTP 请求动作。
- 可以通过向模板传递参数来动态渲染 HTML 页面。

2.2.3 MongoDB 数据库

MongoDB 是一个基于分布式文件存储的数据库。底层使用 C++语言编写。是介于关系型数据库和非关系型数据库之间的产品。它以数据结构松散、高性能、易部署、易使用、存储数据非常方便等特点为开发者熟知和使用，数据结构通常以 json 数据格式存储，因此可以存储较为复杂的数据类型。

MongoDB 数据库中没有“database（数据库）”和“table（表）”的概念，只有“document（文档）”和“collection（集合）”概念，集合的概念类似于关系型数据库里的表（table），不同的是它不需要定义任何模式，完全的模式自由。模式自由，就意味着对于存储在 MongoDB 数据库中的文件，我们不需要知道它的任何结构定义。如果需要的话，完全可以把不同结构的文件存储在同一个数据库里。

2.2.4 HTML + CSS + Javascript 前端技术

对于前台的基础的三门语言，浅显易懂。HTML 通过标记符号来标记要显示的网页中的各个部分；通过 HTML 标签可以根据自身喜好或想法来表现出各式各样的网站设计风格，来完美的实现个页面之间的跳转。HTML5 也新增了音频、本地存储等新功能特性 ，更精确的实现更多功能。是前端的结构层。

CSS（Cascading Style Sheet）可译为“层叠样式表”或“级联样式表”，它定义如何显示 HTML 元素，用于控制 web 页面的外观。是前端的样式层。

Javascript 是一种基于对象和事件驱动并具有相对安全性的客户端脚本语言，同时也是一种广泛用于客户端 web 开发的脚本语言，常用来给 HTML 网页添加动态功能。是前端的逻辑层。

2.2.5 Vue 前端框架

Vue 是一套用于构建用户界面的渐进式框架, 是一个轻巧、高性能、可组件化的 MVVM 库。与其它大型框架不同的是，Vue 被设计为可以自底向上逐层应用。Vue 的核心库只关注视图层，不仅易于上手，还便于与第三方库或既有项目整合。简而言之，Vue.js 是一个构建数据驱动的 web 界面的渐进式框架。Vue.js 的目标是通过尽可能简单的 API 实现响应的数据绑定和组合的视图组件。核心是一个响应的数据绑定系统。

数据驱动是 Vue 最大的特点。在 Vue 中，所谓的数据驱动就是当数据发生变化的时候，用户界面发生相应的变化，开发者不需要手动的去修改 DOM。随着前端技术的不断发展，之前流行的手动修改 DOM 结构的 jquery 框架已经逐渐被淘汰，前端框架越来越趋向于数据驱动 DOM 的方向发展。比如说我们点击一个 button，需要元素的文本值进行切换。在 jquery 刀耕火种的年代中，对于页面的修改我们一般是这样的一个流程，我们对 button 绑定事件，然后获取文案对应的元素 DOM 对象，然后根据切换修改该 DOM 对象的文案值。而对于 Vuejs 实现这个功能的流程，只需要在 button 元素上指明事件，同时声明对应文案的属性，点击事件的时候改变属性的值，对应元素的文本就能够自动的进行切换，我们不需要像以前那样手动的操作 DOM。简而言之，就是 Vuejs 帮我们封装了数据和 DOM 对象操作的映射，我们只需要关心数据的逻辑处理，数据的变化就能够自然的通知页面进行页面的重新渲染。

3 系统分析

搭建完成开发环境，确定好系统开发技术之后，就要开始为设计系统做准备。本章节主要从需求分析、系统结构分析、可行性分析进行介绍，完成这个阶段的关键在于开发人员和用户之间的沟通和交流，明确要解决的问题，对系统进行一个综合的分析，明确目标，尽可能找出各种可行性方案。

3.1 需求分析

随着互联网信息的发展，各大网络论坛平台都已给人们生活带来了无限的便利和活跃度，伴随着人们的生活节奏的加快，各大论坛在行业内部竞争也越来越激烈。本校园社区旨在丰富校师生的生活，让校园生活圈处在实时的信息化平台下。为在校师生提供信息、舆论、交友和休闲等一些互动平台，尽可能满足客户多样化的需求。

首先，应该先明确系统开发的目标和用户需求，通过浏览一些熟知的论坛去了解一个论坛应该具备的基本功能，从而获得本系统开发的基础，确定总体设计的具体要求和限制条件，再对详细的功能模块进行定义，从而达到协助系统开发的目的。

其次，系统在前端页面搭建的需求，主要是前台展示模块和后台管理模块两大界面相互配合，实现一个设计风格简约，观赏性舒适的系统界面；在操作流程方面的需求主要是做到用户体验良好；系统测试和维护的需求让各个模块还之间的耦合性和关联性较强，使得后期系统维护与功能更新起来更加容易。

最后，本系统在功能方面的需求，主要包括校论区、生活休闲区域、表白墙、用户管理等一些主要的模块。

- 校园话题社区

  主要功能：贴吧关注、订阅、动态更新、评论回复、实时通信聊天等功能；

- 表白墙

  主要功能：发表表白贴、支持、评论等功能；

- 生活休闲区

  主要功能：校园新闻、音乐、星座、天气等；

- 个人信息模块

  主要功能：账户设置，帖子管理、贴吧管理、消息管理；

- 系统管理配置模块

  主要功能：对系统的运行进行管理和配置；

3.2 系统结构分析

3.2.1 前端框架结构

考虑到前台页面的美观，前台页面主要由首页、人员信息模块、校园社区模块、表白墙模块、休闲生活模块等几大主页面组成，再由这几个大页面细分出相应的子页面；前端页面的搭建主要使用的是 Vue 框架，搭配上基础的 js 技术，实现相应的前端页面的编写和动画效果。

3.2.2 后端框架结构

配合后台编程语言 Nodejs 的特性，搭建起 http 服务是一件轻而易举的事情，但是搭建好一个强壮稳定的 http 服务，使用原生 Nodejs 的 http 模块是一件很艰难的事情，而且在开发的后期代码的维护也需要花费大量的成本。所以在后台框架中我选择了 Express 这样的一款便捷的框架，不仅仅减少了代码的开发量，而且在开发效率上也得到了很大的提升。

3.3 可行性分析

3.3.1 技术可行性

本系统在开发技术上采用了 Nodejs 和 Vue 结合的方式，数据库部分使用 MongoDB 进行数据存储。其中，前端页面主要基于 Vue 框架，通过 CSS 和 JS 进行样式加工和页面互动性加工，让页面的布局和样式更加具有观赏性。后台使用 Nodejs 运行环境，使用稳定、高效的 MongoDB 数据库管理系统存储数据信息，这样的后台搭配使用率很高，同时也保证了系统的稳定性和安全性。综上，不管是前台页面编写还是后台服务搭建，本系统在技术方面都是可行的。

3.3.2 经济可行性

本系统在设计开发过程中涉及的到的环境和技术，如 Visual Studio Code、Studio 3T 等开发软件，都是开源软件，所以具备了免费试用和公布源代码的特点。搭建 Node 的 http 服务器也是一件简单易行的事情，使用 http 模块在本机上开启服务，以本机作为服务器，相对而言，开发过程中的成本开销就减少很多。综上，不管是在开发软件的成本，还是服务器的开销，在经济方面都是可行的。

3.3.3 操作可行性

通过浏览国内熟知的论坛网站，了解论坛网站的模块功能后，充分考量大学生查看网页的操作习惯，为大学生和管理员提供一个安全可行、快捷便利的系统操作环境。对系统功能的运行结果进行评估，要让用户能后很容易的掌握系统的操作，保证系统没有多余复杂的操作，能够让用户在尽可能端的是时间内熟悉整个系统的操作。所以，本系统在操作上是可行的。

3.4 本章总结

本章主要依据系统结构、需求、可行性以及功能对整个系统进行了初步的分析，为系统的设计和实现提供了帮助，我们明确了所需要解决的问题，排除了需求中的不确定因素，确定本系统的实现是可行的，从而有序地进行下一步系统的详细设计。

4 系统总体设计

本章节将通过结合类似的框架和功能的论坛系统对功能进行详细介绍和模块划分，目的是实现用户可以在互联网环境下进行信息管理、论坛发帖、回复评论、发表表白贴以及实时聊天等功能的操作，并且对系统的数据库表信息进行分析，进而完成系统的总体设计。

4.1 系统设计目标

本系统主要用于丰富高校生生活多样化，提高高校生的生活质量，通过这个平台拉近高校生之间的距离，方便高校师生交流，丰富校园生活的乐趣。系统设计的目标如下：

- 实现用户对帖子的创建、浏览、点赞、评论、回复、删除等操作。
- 实现用户之间在线聊天的功能。
- 实现用户在表白区域对表白贴的支持、评论；表白墙内表白贴的热度计算、排序。
- 实现用户对贴吧的创建，删除，管理等。
- 实现用户的信息管理、帖子管理、消息管理等个人中心模块。
- 最大限度的实现系统的易维护性和易操作性。
- 系统通过测试，能够安全稳定的进行操作和运行。

4.2 功能模块划分

本系统在功能上主要从前台用户体验性和后台代码的易维护性考虑；既要满足用户在此平台上操作的流畅性，也要实现后台运行的安全稳定性。所以依据以上基本想法，通过实现各功能模块之间的联系，完成一个能够满足高校生需求的校区论坛系统。如图 4-1 所示。

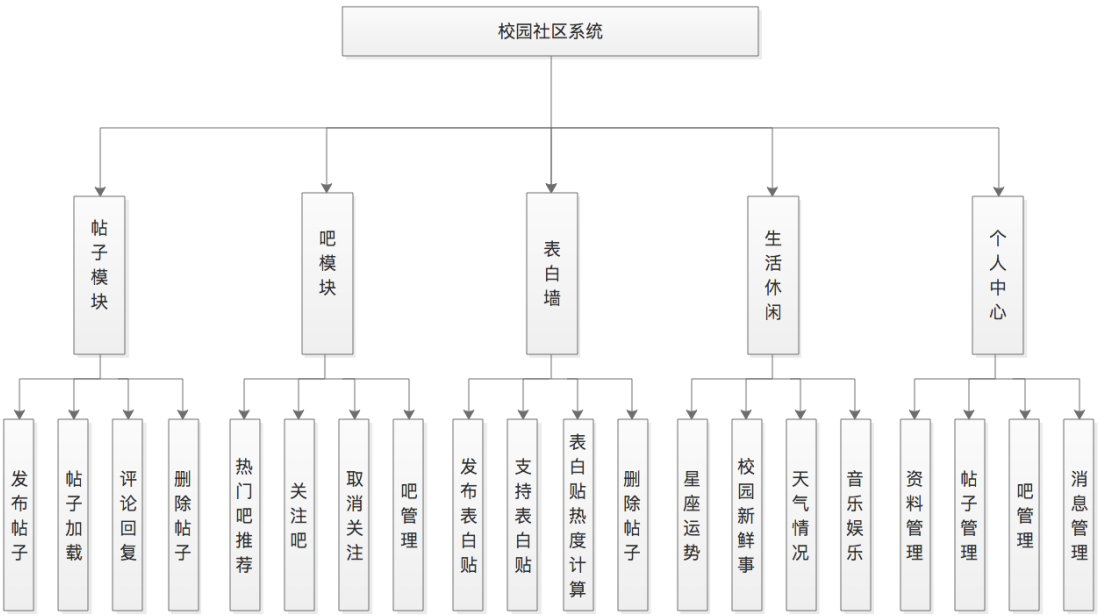

图 4-1 系统功能结构图

4.2.1 校论区

进入之后就可以刷新到最新的“已经关注贴吧”的所有帖子，同时可以切换加载“全贴吧”的所有帖子，帖子按照发布时间的降序排序，用户可以对每个帖子进行点赞、评论、回复等操作，后期还会提供帖子转发的功能。右侧栏展示个人信息的和热门贴吧推荐，用户可以便捷快速的查看到个人的基本信息和热门贴吧的关注。用户还可以由页面进入到贴吧推荐页面，贴吧推荐页面将会展示“热门贴吧推荐”、“全部贴吧”、“已经关注的贴吧”列表，用户可以对每个贴吧进行关注和取消关注的操作，点击相应的贴吧就会进入到本贴吧详情页面，在此页面用户可以看到贴吧的详细介绍以及一些基本信息。功能结构图如 4-2 所示。

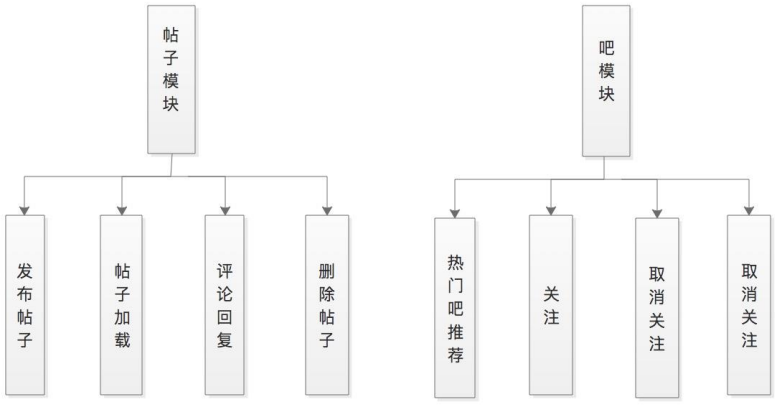

图 4-2 校论区功能结构图

4.2.2 表白区域

表白区页面有做过特别的动画效果，进入之后会看到各式各样的表白帖，支持用户对帖子的支持和评论操作。帖子排版的先后顺序将会按照表白贴的热度（支持率）来进行降序排序，排在前面的将会是支持度比较高的帖子。功能结构图如 4-3 所示。

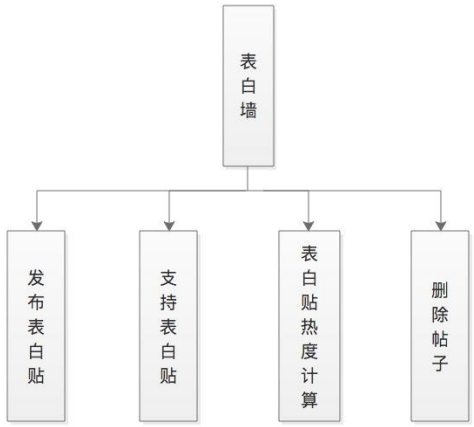

图 4-3 表白墙功能结构图

4.2.3 生活休闲区域

星座运势、天气、音乐等娱乐休闲的模块将会调用第三方接口使用它们的数据来展示相应的情况。校园新闻将会以管理员后台配置的“校园新鲜事”进行数据的读取展示，按照热度的降序排序。功能结构图如 4-4 所示。

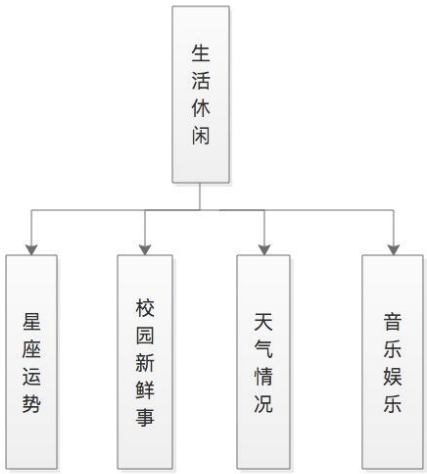

图 4-4 生活休闲功能结构图

4.2.4 个人中心

个人信息管理，对个人信息的更新等操作；帖子管理，对已经发布的帖子进行查看和删除等操作；贴吧管理，对创建的贴吧进行基本信息的修改和基本的配置等操作；消息管理，对消息列表的读取和删除等操作功能。结构图如图 4-5 所示

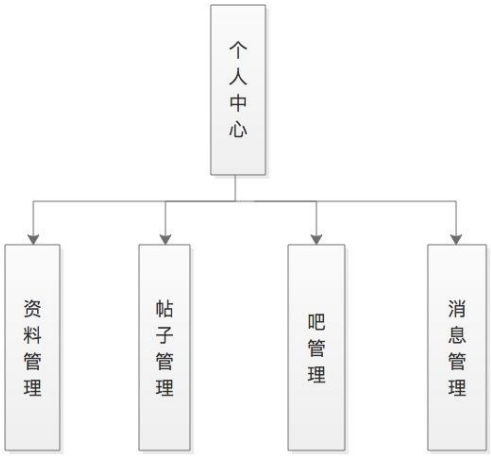

图 4-5 个人中心功能结构图

4.3 数据库的分析和设计

4.3.1 数据库的分析

在对数据库进行设计之前，一定要根据一些数据库的设计和使用经验，提出一些可行的数据库设计的原则，在进行系统设计时，学会善于识别和正确处理数据表之间字段的关联，正确认识数据的冗余，分清表与表之间的挂钩关系，在做到基本的数据库设计原则之后，尽可能的提高数据库运行效率的办法。数据库在设计阶段尤为重要，需要花费精力去研究和学习，否则可能会影响系统实际的运行结果，分析总结的数据库原则如下：

1. 原始单据于实体之间的关系

   可以是一对一、一对多、多对多的关系；一般情况下，他们是一对一的关系，即一张原始单据只对应一个实体，在特殊的情况下它们可能是多对多的关系，即一张原始单据对应多个实体。这里的实体可以理解为数据表。明确好表与表之间的对应关系之后，对我们数据库的设计有很大的帮助。

2. 主键与外键

   一般来说，一个实体都有一个主键或者一个外键。数据库主键，指的是一个列或多列组合，其值可以唯一的标识表中的每一行，通过它可以强制表的实体完整性。数据库的外键，用于与另外一张表关联，是能确定另一张表记录的字段。主键和外键的设计是数据库设计步骤中很重要的一部分。主键与外键的连接，表示了表与表之间的关联。所以在设计表结构时要非常注重表主键和外键的设计。

3. 数据冗余

   为了避免提高系统开发的难度，就要正确看待和分析数据的冗余问题，数据冗余主要是指数据之间的重复性，简单来说就是同一个数据存储在不同的存储文件中，减少数据库的数据冗余不仅在减少了存储空间，也提高了系统的开发效率。

4. 结合实际情况合理选择数据库

   通过自身的需求和实际情况去选择数据库，而不是盲目的想当然，否则不利于后期数据库的操作，可能会影响数据库和系统性能和功能。

4.3.2 数据库设计概念

谈到数据库的设计概念，自然就离不开数据库的实体图和 E-R 图，E-R 图又叫做实体-联系图，它提供了表示实体型、属性和联系的方法，在现实世界和信息世界之间架起了桥梁。实体图和 E-R 图概念帮助我们构建数据库表与表（实体和实体）之间的关系，使得数据库模型和数据表之间的关系清晰可见。以下是校园社区中的部分主要实体-联系图。

1. 用户的实体图，主要包括用户的基本信息，如图 4-6 图所示，其中红色标志的是主键。

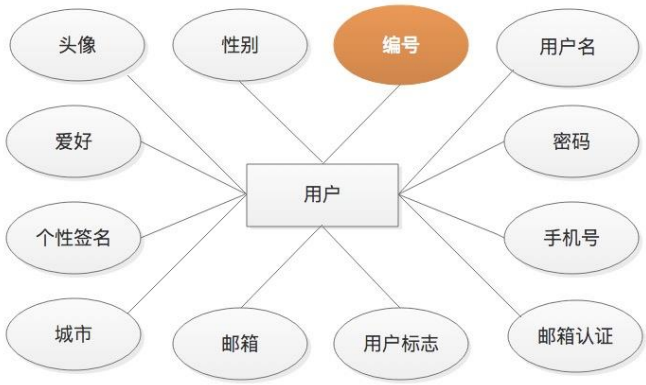

图 4-6 用户实体图

2. 帖子的实体图，主要包含帖子所包含的帖子内容的一些属性，如图 4-7 所示，其中红色为主键，绿色为外键。

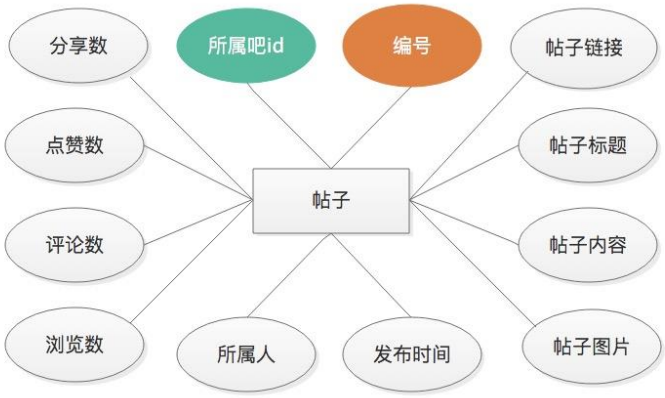

图 4-7 帖子实体图

3. 贴吧的实体的图，主要包含贴吧所包含的属性，主要包括贴吧主题的一些属性，如图 4-8 所示，其中红色为主键。

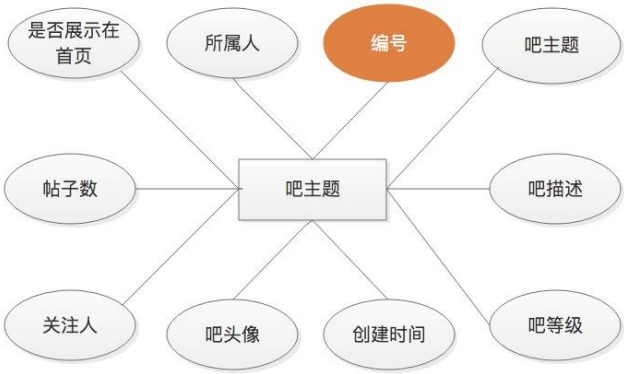

图 4-8 贴吧实体图

4. 表白墙实体图，如图 4-9 所示，其中红色为主键。

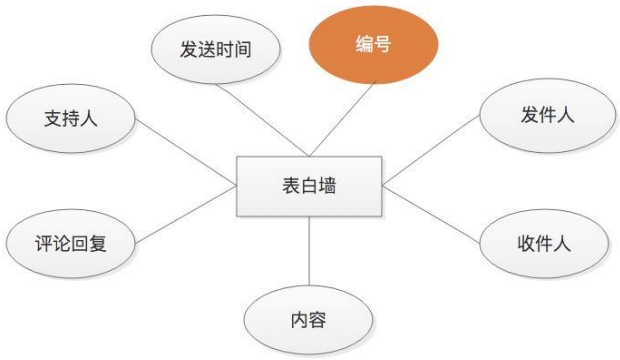

图 4-9 表白墙实体图

5. 用户相关操作的 E-R 图，体现了用户与帖子、贴吧、表白墙、个人管理的关联性。如图 4-10 所示。

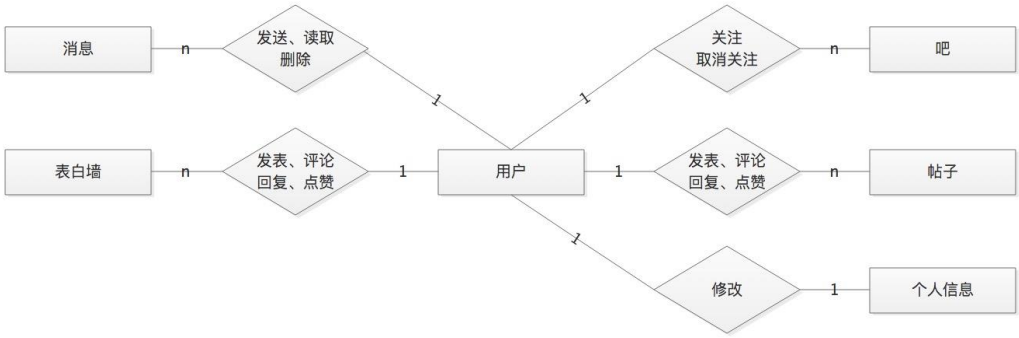

图 4-10 用户操作 E-R 图

4.3.3 数据库基本表的设计

根据设计的 E-R 图，开始对本系统的数据库中的数据表进行分析和设计。结合上面数据设计原则和数据库实体之间的关系，进行数据表整体的设计和分析。数据库使用最近比较流行的 MongoDB 数据库，MongoDB 数据库以 json 数组的形式进行数据存储，基本类型有 String、Number、Array、Object、Null、Boolean 等基本数据类型，MongoDB 数据库在第三章已经有详细的介绍，此处就不在多做介绍。下面是各数据表的详细结构，分别进行说明和分析。下面将对系统关键数据表进行分析说明。

1. users（用户信息表），如表 4-1 所示。

表 4-1 users 用户表

序号	字段	类型	备注	描述
1	_id	ObjectID	主键	由 MongoDB 自动生成的主键
2	username	String	非空	用户名
3	password	String	非空	登录密码（md5 加密）

4	phone_number	String	可为空	手机号
5	sex	String	可为空	性别
6	check	Boolean	非空	邮箱是否验证
7	birthday	Date	可为空	生日
8	city	String	可为空	城市
9	email	String	非空	邮箱
10	pwd_question	String	可为空	忘记密码问题
11	pwd_answer	String	可为空	忘记密码问题答案
12	avator	String	可为空	头像
13	tag	Number	非空	标志 1 管理员，2 普通用户
14	sign	String	可为空	个性签名
15	hobby	String	可为空	爱好

users 表中需要解释的字段有 check、tag 两个字段。首先 check 字段标志的是邮箱是否已经验证，如果注册的邮箱没有验证，那么就无法登录系统。其次 tag 字段，是用户分类的标志，tag 为 1 代表的是管理员，就有权限访问系统管理页面，如果用户 tag 字段为 2 时，表示是普通用户，那么，就没有权限访问系统管理页面。

2. infos（消息表），如表 4-2 所示。

表 4-2 infos 消息表

序号	字段	类型	备注	描述
1	_id	ObjectID	主键	由 MongoDB 自动生成的主键
2	user_name	String	非空	用户名
3	avator	String	非空	头像
4	receiveMsg	Array	可为空	收到的消息
5	sendMsg	Array	可为空	发送的消息

其中的 receiveMsg 和 sendMsg 字段存储形式是数组（json 数组），数组中存储的是一个一个 josn 对象，每个对象就是用户所发送或者收到的一个整体消息实体，数组中的每个对象包括以下字段，如下表 4-3 所示。

表 4-3 消息数组中 json 对象表

序号	字段	类型	备注	描述
1	from_user_name	String	非空	消息发送人
2	to_user_name	String	非空	消息接收人
3	content	String	非空	消息内容
4	time	Date	非空	时间

3. posts（帖子表），如 4-4 表所示。

表 4-4 posts 帖子表

序号	字段	类型	备注	描述
1	_id	ObjectID	主键	由 MongoDB 自动生成的主键
2	subject_id	String	非空	帖子所属的 subject
3	subject_title	String	非空	帖子所属 subject 的名称
4	post_title	String	可为空	帖子标题
5	post_content	String	可为空	帖子内容
6	post_photos	Array	可为空	帖子发的表图片
7	link	String	可为空	帖子链接
8	user_name	String	非空，外键	用户名
9	avator	String	可为空	用户头像
10	time	Date	非空	发表时间
11	share	Number	非空	转发分享数
12	look	Number	非空	浏览数
13	hearts	Number	非空	点赞数
14	replys_num	Number	非空	评论回复数

4. subjects（贴吧表），如表 4-5 所示。

表 4-5 subjects 贴吧表

序号	字段	类型	备注	描述
1	_id	ObjectID	主键	由 MongoDB 自动生成的主键
2	subject_id	String	非空	帖子所属的 subject 的 id
3	subject_title	String	非空	帖子所属 subject 的名称
4	post_title	String	可为空	帖子标题
5	post_content	String	可为空	帖子内容
6	post_photos	Array	可为空	帖子发的表图片
7	link	String	可为空	帖子链接
8	user_name	String	非空，外键	用户名
9	avator	String	可为空	用户头像
10	time	Date	非空	发表时间
11	share	Number	非空	转发分享数
13	hearts	Number	非空	点赞数
14	replys_num	Number	非空	评论回复数

5. replys（评论回复表），如表 4-6 所示。

表 4-6 replys 评论回复表

序号	字段	类型	备注	描述

1	_id	ObjectID	主键	由 MongoDB 自动生成的主键
2	post_id	String	非空, 外键	帖子所属的 subject
3	post_title	String	可为空	帖子标题
4	post_content	String	可为空	帖子内容
5	post_photos	Array	可为空	帖子图片
6	replys	Array	可为空	帖子评论回复

其中 replys，为本帖子的评论回复内容，存储为 json 数组的形式，数组由 json 对象组成，如下表 4-7 所示。

表 4-7 评论回复 json 数组中对象表

序号	字段	类型	备注	描述
1	from_user_name	String	非空	评论人
2	to_user_name	String	非空	接收人
3	content	String	非空	评论回复内容
4	time	Date	非空	时间

6. white_wall（表白墙表），如表 4-8 所示。

表 4-8 white_wall 表白墙表

序号	字段	类型	备注	描述
1	_id	ObjectID	主键	由 MongoDB 自动生成的主键
2	from_user_name	String	非空	告白人
3	to_user_name	String	可为空	被告白人
4	supports	Array	可为空	支持人
5	time	Date	非空	发表时间
6	replys	Array	可为空	评论回复

其中 replys 为 json 数组的形式，数组中 json 对象的字段信息和表 4-7 信息是一致的，请参见表 4-7 了解 replys 数组中对象的存储信息。

7. school_news（校园新鲜事表），如表 4-9 所示。

表 4-9 school_news 校园新鲜事表

序号	字段	类型	备注	描述
1	_id	ObjectID	主键	由 MongoDB 自动生成的主键
2	title	String	可为空	新闻标题
3	content	String	可为空	新闻内容
4	image	String	可为空	新闻图片
5	look	Nnmber	非空	浏览数
6	like	Number	可为空	喜欢数

7	replys	Array	可为空	评论回复
8	owner	String	可为空	最后编辑人
9	support	Array	可为空	支持人
10	hot	Number	可为空	热度

4.4 本章总结

本章主要通过分析系统的模块划分和数据库的设计来完成对系统的整体设计，根据系统分析的情况来看，系统的总体设计实现是简单可行的。综上，系统的整体设计全部完成，接下来将完成实现系统模块功能了。

5 系统实现

通过上面四章节的分析，不仅对系统的整体架构以及功能模块进行了初步的分析和了解，还对实现系统的各项技术以及数据的存储进行了深入的介绍。接下来我们将对系统功能的实现进行深入分析，对功能模块代码块进行介绍。

5.1 开发系统文件介绍

如图 5-1 所示系统开发主要目录

图 5-1 系统开发主要目录

系统开发目录介绍：

1. controller 文件夹：后台控制层，主要是针对功能逻辑实现的控制。

   说明：controller.js 文件中使用“exports”关键字导出，在其他文件中只需要引入 controller.js 文件，就可以直接使用 controller.js 文件中所导出的一些逻辑函数。

2. db 文件夹：用来存储连接数据库的 js 文件，也包括了针对每个数据表所做的操作存储。
   说明：db.js 用来连接数据库，并且导出连接数据库的函数。以供全局使用
   其他 js 文件主要是针对不同的数据表进行的一些增删改查的操作。
3. public：此文件夹下主要包含了一些公共使用的文件如 css 文件、js 文件、图片文件等静态文件。
4. router 文件夹：此文件夹下包含了一个 router.js 文件，用来对路由的控制。
5. tool 文件夹：工具类。主要用来编写使用到的公共的工具方法。并且使用“exports”导出工具函数。
6. views 文件夹：页面视图文件夹，包含了所有页面的 ejs 文件（最后会解析成 html 文件）。
7. test 文件夹：测试文件夹，只要用来在开发过程中的测试，主要包括数据库操作测试，页面请求测试等，
8. app.js：启动文件，包括 Express 框架配置、session 配置、路由配置等等配置。

5.2 Express 框架搭建 Node 服务介绍

5.2.1 启动文件全局配置介绍

1. Express 框架搭建 http 服务实现。
   - 实现原理

     Express 搭建 http 服务，只需要三步就可以简单快速的搭建起 node 服务，第一步：引入 Express 模块文件；第二步：获取 Express 对象，实际上就是调用了 Express 模块文件中的 createApplication()方法；第三步：利用 Express 对象监听 http 端口，在这里我们以 3000 端口为例。在启动服务后，用户在浏览器中访问 3000 端口就会被监听到。这样，简单的 http 服务就搭建好了。继而就可以进行 Express 的路由注册，为 web 页面注册相应的路由，将不同的路径请求区分到不同的模块中去。在路径匹配正确后，就会调用其中的回调函数，执行相应的逻辑函数；回调函数有两种执行结果，第一种执行结果是：向前台渲染页面；第二种执行结果是：向前台返回数据。前台在接收到回调函数的第一种执行结果后，会在 views 文件夹中找到相应的.ejs 文件并进行页面的渲染；前台在接收到回调函数的第二种执行结果时，会解析数据并且进行数据的展示（页面 DOM 结构的修改）。
   - 搭建 http 服务，实现前后台交互，流程图如图 5-9 所示。

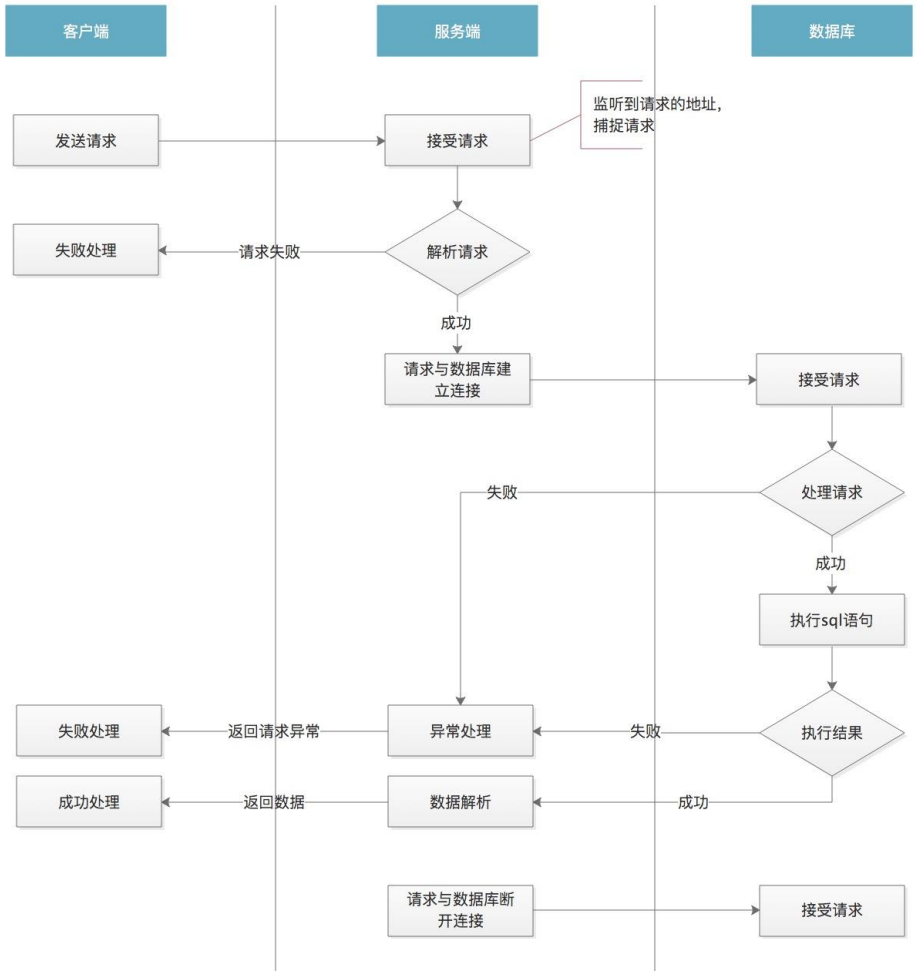

图 5-9 Express 前后台交互流程图

- 关键代码块

```
//引入 Express 模块
var express = require("express");
var app = express();
var http = require("http").Server(app);
//use ejs
 // 监听路由,前台使用 get 方法访问/router 路径，就会被捕捉到并执行 function
 函数
app.get("/router", function(req,res){});
// 监听端口
 http.listen(3000);
```

- 代码块解读

首先使用 require()引入了 Express 框架，接着使用 app.set()方法设置了使用 ejs 文件，这段代码的配置就是告诉服务器所有的视图文件放在根目录的 views 文件夹下，并且视图文件的后缀名为.ejs。最后 http.listen(3000)这段代码是监听 http 3000 的端口。一旦有访问 3000 端口的请求，就会被检测到，接着就开始调用相应的 ejs 文件渲染前台页面。只需要在当前目录下的键入 node app. js 就可以将 http 服务提起。

2. session 使用的配置；在开发过程中不可避免的需要使用到 session 存储。下面将介绍 session 的配置过程。

- session 配置原理

  session 实际上就是在服务端开辟出来的一个内存空间。一般情况下，会将用户的操作信息存储到 session 中。在本系统中，在用户登录之后会将用户的登录信息存储到 session 中，这样就避免了用户重复的输入用户名和密码登录系统的情况，也保证了用户登录的安全性。相应的，session 会有失效的情况，在用户点击退出系统时，就会将此 session 置为失效状态。

- 关键代码块

```
//引入 session 的模块
var session = require("express-session");
//session 配置
app.use(session({
      secret: 'keyboard cat',
      resave: false,
      saveUninitialized:  true
 }));
```

- 代码块解读

  根据官网的配置，实现对 session 的存储和使用，只需要简单的使用 request.session 语句就可以对 session 进行数据存储和读取操作。

3. 公共文件的路径配置。对于前台需要引用的 css、js、图片等静态文件，Express 可以对其访问路径进行一些配置，以方便访问这些公共文件。

- 配置静态文件原理

  使用 Express 框架，在前台访问静态文件，会是一件比较麻烦的操作，因为 Express 框架会将静态文件的路径配置成很复杂的形式，所以需要将静态文件的路径配置成简单的路由模式，方便前台访问。例如，在本系统中，有头像图片的上传，头像图片保存在一个统一文件夹中，需要将此头像文件夹配置 public 文件夹的形式，方便前台获取图片。

- 关键代码块

```
//static resource
app.use(express.static("./public"));
app.use(express.static("./postImages"));
 app.use(express.static("./avator"));
```

- 代码块解读

  app.use(express.static('./public'))这段代码的意思就是根目录的 public 文件夹下的所有文件都配置在了系统访问的根目录下，如 js 文件夹的 jquery.js 文件，在前台.ejs 视图文件中就可以使用"/js/jquery.js"路径访问到此 jquery 文件。这个方法经常用来配置静态文件。

5.3 功能模块实现介绍

关于功能模块的实现介绍，主要介绍功能实现的方式和介绍代码实现的逻辑。倾向于难点功能的实现和分析，对于一般的简单点实现操作就不在进行过多的分析介绍。所以，在本节内容中，将主要针对各个模块的难点进行功能的实现和代码的分析介绍。

5.3.1 人员信息模块

1. 找回密码功能模块

- 找回密码原理

  找回密码主要是向注册的邮箱中发送邮件，邮件内容是一条链接地址，此链接地址中会带有经过加密后的用户名和新密码信息，用户点击此链接地址，会返回到本系统中，系统会将链接中的用户名和新密码重新写入数据库中。如果用户在规定时间内没有点击邮箱中链接，那么视为失效链接。

- 找回密码操作流程图，如下图 5-10 所示

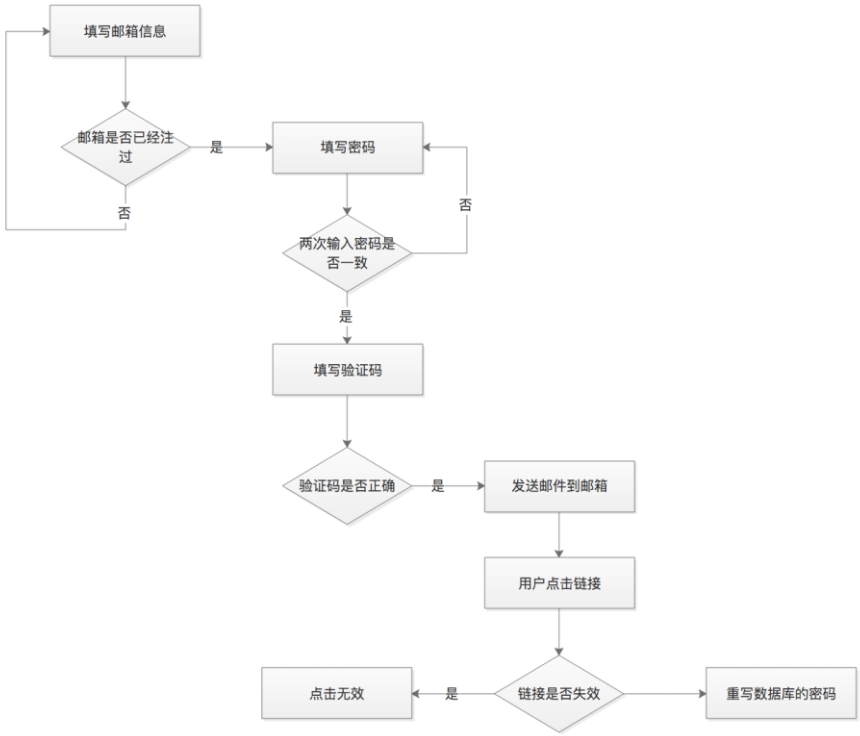

图 5-10 找回密码流程图

- 关键代码块

```
// 发送邮箱
function sendEmail(email,  username, password, tag) {
      // 发送 email
      // 时间戳，超过一定的时间，用户没有点击此链接，则此链接视为失效状态
      var now = Date.parse(new Date())
      var href = 'http://localhost:3000/loginCheckEmail?username='  + username +
'&password=' + password + '&tag=' + tag + '&timestamp=' + now
            // 发送方
```

```
    var transporter = nodemailer.createTransport({
            service: 'qq',
            auth: {
                user: 'ztchun@qq.com',
                pass: 'bsdhtkkqjkbyhaia'
            }
        })
        // 接收方
    var mailOptions = {
        from: 'ztchun@qq.com',
        to: email,
        subject: '校园社区邮箱验证',
        text: '您好，此邮件为校园社区的邮箱认证，点击下方链接认证邮箱有效性',
        html: '<h2>您好，此邮件为校园社区的邮箱认证，请点击下方链接认证邮箱有效性</h2>' +
            '<div><a href=\'' + href + '\'>链接</a></div>'
    }
    transporter.sendMail(mailOptions, function(error, info) {
        if (error) {
        } else {
        }
    });
}
```

- 代码解析

  此代码是向邮箱发送邮件，主要使用的是 nodemailer 模块实现，nodemailer 模块是 Nodejs 第三方封装的 js 库，主要用于向特定的邮箱发送特定的邮件消息。实现的三步骤是：配置发送方信息、配置接收方消息、发送消息。

- 界面实现图，如图 5-11 所示

图 5-11 找回密码界面图

2. 记住密码功能模块

- 实现原理

  本系统中的记住密码主要是由前台代码实现。用户登录时，如果勾选了“记住密码”选择框，系统会将加密过的用户名和密码存入 cookie 中，并且将 cookie 的生命周期设置为 7 天，7 天之后此 cookie 就会置为无效状态；如果用户没有勾选“记住密码”或者取消了“记住密码”勾选框，那么就会删除 cookie 中的用户信息。相应的，页面加载期间，如果 cookie 中存在了用户信息，说明用户在之前的登录操作中勾选了“记住密码”选项，那么就需要把用户名和密码框自动填充，并将“记住密码”选项设置为勾选状态。

- 登录操作流程图，如图 5-12 所示

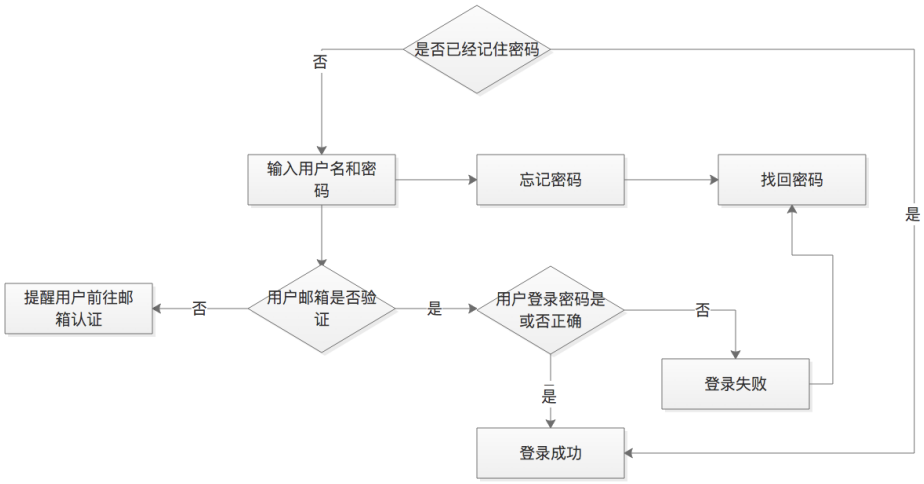

图 5-12 登录操作流程图

- 关键代码块

```
//页面初始化时，如果帐号密码 cookie 存在则勾选“记住密码”
if (getCookie('username') && getCookie('password')) {
    $(".username").val(getCookie('username'))
    $(".password").val(getCookie("password"))
    if (getCookie('username') && getCookie('password')) {
        $(".check")[0].checked = true
    }
}
//复选框勾选状态发生改变时，如果未勾选则清除 cookie
$(".check").change(function() {
    if (!$(this)[0].checked) {
        delCookie("username")
        delCookie("password")
    }
})
```

- 代码块解析

以上功能由前端代码实现，主要使用 jquery 框架实现。页面加载，需要检测 cookie 中是否有用户的登录信息，如果有用户的信息，那么用户无需输入用户名和密码就可以直接登录。当“记住密码”状态更新的时候，出发 checkbox 的 change 事件，并在 change 方法中根据选择框状态进行用户信息的存储和删除操作。

- 登录注册页面界面图, 如图 5-13 所示

图 5-13 登录注册页面图

5.3.2 校论区模块

1. 上传图片功能模块

- 实现原理

实现图片上传主要使用了 Nodejs 中的自带的 fs 文件操作模块和第三方封装的 formidable 表单数据转化模块。其中 foridable 模块主要用于上传图像和视频，对于高版本的 Express 模块，其中已经封装了 formidable 模块，但是低版本必须手动引入才可以使用。文件操作模块 fs 模块，在这部分的主要作用就是修改上传文件的命名，修改上传文件的命名对于前台引用图片有非常重要的好处。

- 关键代码块

```
function uploadImage(req, res, dir, filePath) {
    // 从 formidable 模块中获取封装对象，此时图片存在 form 对象中
    var form = fd.IncomingForm()
    // 修改上传文件的存储文件夹
    form.uploadDir = path.normalize(__dirname + dir);
    form.parse(req, function(err, fields, files) {
        var oldpath = files.file.path;
        var time = Date.parse(new Date())
        // 修改上传文件的命名
        var newpath = path.normalize(__dirname + dir + filePath + ".jpg");
        fs.rename(oldpath, newpath, function(err) {
            if (err) {
```

```
                    res.send("0")
                    return;
                }
                res.send(newpath)
            })
        })
    }
```

- 代码解析

  见上一项代码块的注释。

- 界面实现图，如图 5-14 所示

图 5-14 图片上传界面图

2. websocket 实时通信

（1） 实现原理

websocket 是 html5 提供的一种全双工通信的网络协议，websocket 在浏览器和服务器之间架起了一条实时的通道，此通道实现了实时快速的信息交流。此功能的实现上主要依赖于 Nodejs 的 websocket 模块。Websocket 工作方式通俗讲就是在前后台相应的模块中配置发送和监听消息的动作。例如，在前台点击按钮向后台发送了消息，此时后台正处于监听状态，当捕捉到此消息后就会将此消息发送给相应的接受人，此消息又会返回到前台，同时，前台也有相应的监听代码，监听后台推送的消息并将此消息展示在前台页面；这样，简单的实时通信就建立了。

（2） 后台实时通信功能的实现

- 后台服务端代码块

```
// 后台引入 websocket 模块
var io = require("socket.io")(http);
// 前后台连接
io.on("connection", function(socket) {
    // 监听 chat 事件
    socket.on("chat", function(msg) {
        // 将消息存到数据库中
        users.findData({
            username: msg.to_user_name
        }, function(data) {
            var message = {
                // message 对象 此处省略
            }
            var message1 = Object.assign({ readed: true }, message)
            infos.updateDataBy({
                user_name: msg.from_user_name,
            }, {
                $push: {
                    sendMsg: message1
                }
            }, function(data) {
                if (data.result.ok) {
                    var message2 = Object.assign({ readed: false }, message)
                    infos.updateDataBy({
                        user_name: msg.to_user_name
                    }, {
                        $push: {
                            receiveMsg: message2
                        }
                    }, function(data) {
                        if (data.result.ok) {
                            // 向前台发送 answer 消息
                            io.emit("answer", message);
                        }
                    })
                }
            })
        })
    })
})
```

- 代码解析

  代码解析：实时通信服务端代码的实现，主要是监听到前台发送消息之后，接受参数并存储消息，然后使用 io.emit('answer',message)代码块主动向前台推送消息内容。实时通信的实现主要依靠 websocket 协议，此处功能的实现主要依赖于 Nodejs 的 io 模块，此模块是对 websocket 的封装，大大简化了实时通信的功能

（3） 前台实时通信功能的实现

- 实时通信前台代码块

```
// 引入 socket 文件
<script type="text/javascript" src="/socket.io/socket.io.js"></script>
//获取对象
var socket = io();
```

```
$(".chat-send").click(function(){
        var message = {
                // message 对象 此处省略
        }
    // 向后台发送 chat 消息
        socket.emit("chat",message);
        $(".msg").val('')
})
//监听 answer 事件
socket.on("answer",function(msg){
        if (msg.from_user_name == user.username){
                // 发送的消息
                var $chat = ‘’
                $(".message").append($chat)
        } else {
                // 收到的消息
                if (msg.to_user_name == user.username){
                        var $chat =’’
                        $(".message").append($chat)
                }else{
                        return
                }
        }
})
```

- 代码解析

实时通信前台功能的实现。首先引入 socket.io.js 文件，然后通过 var socket = io()取得 socket 对象之后，就可以在 socket 对象上进行事件的触发和监听。在点击按钮之后通过 socket.emit()代码触发 socket 的“chat”事件，并向后台发送消息。继而后台接收到 chat 事件之后，进行一些逻辑处理，处理结束后向前台发送“answer”事件，在前台利用 socket 对象监听“answer”事件，在监听到“answer”事件后前台就会作出一些处理。

（4） 实时通信实现的界面截图，如图 5-15

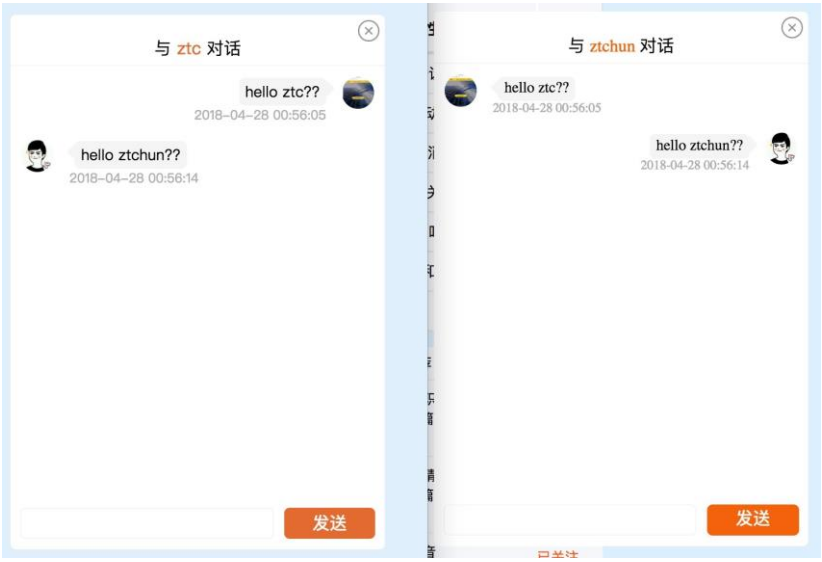

图 5-15 实时通信界面图

3. 帖子上拉加载功能模块

（1） 实现原理

前台监听滚动事件，当检测到页面拉倒最底部的时候，就发送帖子请求到后台，后台进行数据库的查询，将数据返回给前台，前台就行数组的拼接。

（2） 前台上拉加载实现

- 关键代码块

```
// 监听滚动时间
window.addEventListener("scroll", this.handleScroll)
// 检测是否滚动到页面底部
handleScroll() {
    if ($(window).scrollTop() >= $(document).height() - $(window).height()) {
        this.initPost(this.start, 10)
    }
}
```

（3） 后台上拉加载的实现

- 关键代码块

```
function findDataSort(data, sort, start, limit, callback){
    var result = []
    db._connnection(function(db){
        var cursor =
db.collection("posts").find(data).sort(sort).skip(start).limit(limit)
        cursor.each(function(err,doc){
            if(err){
                console.log("排序失败", err)
                db.close();
                return
            }
            //遍历游标
            if(doc != null){
                result.push(doc);
            }else{
                //遍历结束，没有更多的文档了，那么这个时候就需要使用回
调函数了
                callback(result);
            }
        })
    })
}
```

- 代码解析

上面代码主要输数据库的分页功能，function 中传入了五个参数分别是 data, sort, start, limit, callback，其中 data 是定位作用，用来指定数据库的查询对象；sort 是排序作用，数据库按照哪个字段进行排序；start 和 limit 是数据库查询的起始位置。最后通过 db.collection("posts").find(data).sort(sort).skip(start).limit(limit)执行数据库的分页查询。

（4） 帖子上拉加载界面实现图，如图 5-16 所示

图 5-16 帖子上拉加载界面实现图

5.3.3 表白墙模块

1. 表白贴的评论和回复

- 实现原理

  此模块功能主要依赖 MongoDB 数据库中对数组字段的增删改查的操作。Nodejs 操作 MongoDB 数据库的 sql 语句的简单，只需要通过一些 sql 的关键字如$push、$pull、$set 等就可以快速的实现数据库的增、删、改、查、排序、分页等操作。

- 关键代码块

```
// 提交评论
exports.commitComment = function(req, res, callback) {
    var form = fd.IncomingForm()
    form.parse(req, function(err, fields) {
        console.log(req.session.username, fields.post_user_name)
        tools.showTime(function(time) {
            // 更新 replys
            replys.updateReplys({
                post_id: fields.post_id
            }, {
                $push: {
                    replys: {
                        // 对象省略
```

```
                    }
                }
            }, function(data) {
                // 回调函数中向前台返回数据
            })
        })
    })
}
```

- 代码分析

  通过上面的代码不难发现，Nodejs 操作 MongoDB 数据库的 sql 语句的简单性，上面代码实现了向 replys 数据表中 replys 数组字段进行增加的操作，使用到了$push 关键字。简单的实现了更新数据表的操作。

- 表白贴评论回复界面实现图，如图 5-17 所示

图 5-17 表白贴评论回复界面图

5.4 本章总结

本章主要介绍了系统实现的过程，首先介绍了 Nodejs + Express 框架开发目录，讲解了项目目录文件夹的地位和作用；紧接着，介绍了 Nodejs 开发 web 服务的流程，对 Nodejs 开发 web 服务做出了系统的流程阐释；最后，对功能复杂的模块实现，进行了原理讲解和代码分析。综上，对于本系统功能的实现全部结束。下面将介绍本人对系统的总结和展望。

6 总结与展望

从课题的研究现状到系统的实现，这期间的工作一直循序渐进的进行着，目前，该系统已经能够按照预期的需求正常运行，实现了前台的展示和后台服务器的搭建。虽然系统有些细节性的用户体验有待改进和提高，但是基本上还是能满足高校生对于生活多样化的追求。本文重点介绍了校园论坛的设计以及实现的过程，主要完后了以下工作：

- 从目前高校校园电子服务方面和国内外论坛的状况阐述了本系统在当前环境下的开发目的和意义，其次又明确了本研究课题的内容和方法。
- 简单介绍了该系统在开发过程中用到的开发工具和开发技术。
- 对本系统进行了需求分析，明确了开发需求。继而对系统开发的可行性进行分析。同时确立了本系统在功能模块上的划分。
- 对本系统的数据库设计进行分析和探究，明确了数据库模型和实体之间的联系。
- 详细的介绍了本系统各个功能的实现，对系统难点功能的实现进行了代码块的粘贴和说明。

同时，本系统也存在了一些不足的地方，在这里，将对系统的不足给出意见，这些意见也是本系统接下来做出的改进。

- 登录失效的检测。用户登录成功之后，如果一定时间内在页面没有做出任何操作，将 session 结束，登录失效。
- 向用户推荐贴吧，可以适当的增加算法。
- 帖子的转发和帖子浏览数计数。

完成这次毕业设计论文，于个人来说不仅是一件很有成就感的事，而且在系统开发技术上也提高了自己的开发经验。在开发过程中难免会遇到困难，但是在自己的努力下和老师的指导下，不断突破自我，获取源源不断的灵感。在这几个月的工作中，毕业设计和毕业论文对我的专业技能和理论知识都有了很多的积累和提高。从某种意义上来说，毕业设计也是一种学习方式，这种学习方式完全依靠个人的自主性和坚韧性。自主学习，遇到困难不断克服，不断从开发过程中积累经验和学习开发技巧。

最后，希望我的毕业设计和毕业论文，不仅仅是向别人展示我的毕业作品，也希望我自己能够在此次开发过程中看到自己的优缺点，并且保持优点，改良缺点，让自己能够在计算机这条道路上走的更好更扎实。

参考文献

[1]. 朴灵.深入浅出 Nodejs[M].北京：人民邮电出版社，2013，3-43.

[2]. Mike Cantelon，TJ Holowaychuk，Nathan Rajlich. Nodejs 实战[M].北京：人民邮电出版社，2014，23-67.

[3]. Nicholas C.Zakas.JavaScript 高级程序设计[M].北京：人民邮电出版社，2012，21-354.

[4]. 单东林，张晓菲，魏然.锋利的 JQuery（第二版）[M].北京：人民邮电出版社，2012，2-87.

[5]. Steve Souders.高性能网站建设指南[M].北京：电子工业出版社，2008，3-78.

[6]. 张容铭.JavaScript 设计模式[M].北京：人民邮电出版社，2013，45-78.

[7]. 李东博.HTML5+CSS3 从入门到精通[M].北京：清华大学出版社，2013，80-132.

[8]. 王伶俐、张国传.基于 Nodejs+Express 的轻应用定制平台的设计与实现 [J].计算机科学 Computer Science 期刊.2017，S2 期 1-4.

[9]. Riordan.Head First Ajax（中文版）.北京：中国电力出版社，2010，67-96.

[10]. 张文盛、郑汉华.基于 MongoDB 构建高性能网站技术研究[J].吉林师范大学学报(自然科学版)期刊.2013，01 期 1-3.

[11]. 吕林.基于 MongoDB 的应用平台的研究与实现.博士论文学位.北京：北京邮电大学，2014.

[12]. 许会元、何利力.Nodejs 的异步非阻塞 I/O 研究[J].工业控制计算机期刊.2015，01 期，1-3.

致 谢

我很荣幸能够独立完成此次的南京晓庄学院毕业设计和论文，也很感激在此过程中帮助我的老师和同学，下面我将对帮助过我的老师和同学表示感谢。

首先要感谢的是我的指导老师徐老师，在老师的指导下，我不断更正错误和改进需求，在选题期间，老师让我们充分发挥想象力，尽可能的想出有创新意义的课题，在和老师不断的沟通下，才确立了我的课题。在分析和开发阶段，在老师的督促下，严格按照开发进度进行毕业设计的编写。在此过程中，得到了老师很多关心，也从中汲取到很多的经验和建议。不仅仅在毕业设计方面得到了很多老师的帮助，在实习期间遇到难以决策的事情，老师都会帮我分析利弊，给我建设性的意见。在此真挚的说一声，老师，谢谢！

在实习期间，在单位师傅的带领下，也学到了很多的开发经验，大大提高了此次的毕业设计的编写的效率。在工作实习期间，师傅还会经常带领我走进项目的讨论会议，在每一次的会议过程中，都会学到很多的不同的开发方法和实现技巧，这不仅教会了我解决问题付方式方法，更重要的是培养了我解决问题的信心和能力。

同时，在开发过程中也会请教有开发实习经验的同学，也得到了很多的指点和理解。

能够顺利的完成此次的毕业设计，给我最大的帮助就是我的指导老师，再次，感谢我的老师，其次就是我的实习单位的师傅，感谢在实习期间教会了我开发技巧和解决问题的能力。最后，感谢帮助我的同学，有了你们的帮助，让我在开发中少走了很多的弯路。再次表达衷心的感谢并祝福你们今后有更好的发展。