东华理工大学长江学院

**本 科 生 毕 业 设 计**

|  |  |  |
| --- | --- | --- |
| 论文题目 | **：** | **基于Django的家校通的设计与实现** |
| 姓名 | **：** | **殷海** |
| 学号 | **：** | **20202031427** |
| 班级 | **：** | **2030314** |
| 年级 | **：** | **2020级** |
| 专业 | **：** | **计算机科学与技术** |
| 学院 | **：** | **信息工程系** |
| 指导教师 | **：** | **余恬** |
| 完成时间 | **：** | **2024年 \*月 \*\* 日** |

**作 者 声 明**

本人以信誉郑重声明：所呈交的学位毕业设计（论文），是本人在指导教师指导下由本人独立撰写完成的，没有剽窃、抄袭、造假等违反道德、学术规范和其他侵权行为。文中引用他人的文献、数据、图件、资料均已明确标注出，不包含他人成果及为获得东华理工大学或其他教育机构的学位或证书而使用过的材料。对本设计（论文）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。本毕业设计（论文）引起的法律结果完全由本人承担。

本毕业设计（论文）成果归东华理工大学长江学院所有。

特此声明。

毕业设计（论文）作者（签字）：

签字日期： 年 月 日

本人声明：该学位论文是本人指导学生完成的研究成果，已经审阅过论文的全部内容，并能够保证题目、关键词、摘要部分中英文内容的一致性和准确性。

学位论文指导教师签名：

年 月 日

**基于Django的家校通设计与实现**

**殷海**

**Design and Implementation of Home-School Communication System Based on Django**

**Yin Hai**

# 摘 要

本论文设计与实现的基于Django的家校通系统，旨在加强中小学家庭和班级之间的信息共享、学习管理和协作效率。家校通分为两种不同的普通用户角色，家庭用户和班主任用户，系统为他们提供了各种协作和管理的功能。Django自带的框架通过代码进行注册，系统将自动生成Django超级管理员用户，该用户拥有所有权限，方便管理用户，处理数据。家校通后端采用了Django框架，成功地构建了一个功能丰富、易于使用的家校通。前端使用Bootstrap框架，使家校通系统的界面设计更美观、响应更快、用户体验更好，提升了系统的社会可行性。数据库采用Sqlite，使得家校通系统更轻量级、易于部署和跨平台，提高了系统的运行可行性。本文采用了良好的软件工程实践，如模块化设计、代码重用和测试驱动开发，确保系统的可靠性和可维护性。系统的功能包括家庭用户注册和登录、签到、查看通知、事项讨论、查看考核成绩，在此功能基础上班主任用户可以发布通知、查看考勤统计记录、发布和评定考核记录、管理班级和管理成员。通过单元测试，系统的功能能够正确地实现。家校通系统操作简便，可以提高班级和家庭的协作效率和学习管理水平。未来的研究可以进一步完善和拓展家校通的功能，如添加手机短信通知和作业提交等功能。此外，可以通过用户反馈和数据分析来改进系统的性能和用户体验。

关键词： 家校通；Django；Bootstrap；SQLite

# Abstract

This paper presents the design and implementation of a home-school communication system based on Django framework, aimed at enhancing information sharing, learning management, and collaboration efficiency between primary and secondary school families and classes. The Home-School Communication system consists of two different types of regular user roles: home users and class teacher users, providing various collaboration and management functions for them. The Django framework's built-in features facilitate user registration through code, automatically generating a Django superuser with all permissions to manage users and process data. The backend of the Home-School Communication system is built using the Django framework, successfully constructing a feature-rich and user-friendly platform. The frontend utilizes the Bootstrap framework, improving the system's interface design, responsiveness, and user experience, enhancing its social feasibility. The database adopts Sqlite, making the system lightweight, easy to deploy, and cross-platform, thus increasing its operational feasibility. This paper adopts good software engineering practices such as modular design, code reuse, and test-driven development to ensure system reliability and maintainability. The system functionalities include home user registration and login, attendance tracking, notification viewing, discussion of matters, and viewing assessment scores. In addition, class teacher users can publish notifications, view attendance statistics records, publish and assess assessment records, manage classes, and manage members based on these functionalities. Unit testing ensures that the system functions correctly. The Home-School Communication system is user-friendly and can improve collaboration efficiency and learning management levels for classes and families. Future research can further enhance and expand the functionalities of the Home-School Communication system, such as adding features like mobile SMS notifications and homework submissions. Furthermore, system performance and user experience can be improved through user feedback and data analysis.

**Key words:** Home-School Communication System；Django；Bootstrap；SQLite

目 录

[**Yin Hai** 3](#_Toc161943412)

[摘 要 4](#_Toc161943413)

[Abstract 5](#_Toc161943414)

[1 绪论 1](#_Toc161943415)

[1.1 研究背景 1](#_Toc161943416)

[1.2 研究目的及意义 2](#_Toc161943417)

[1.3 国内外研究现状 2](#_Toc161943418)

[1.4 研究内容 3](#_Toc161943419)

[2 系统相关技术 4](#_Toc161943420)

[2.1 前端开发技术 4](#_Toc161943421)

[2.1.1 Bootstrap框架 4](#_Toc161943425)

[2.1.2 ECHARTS 4](#_Toc161943426)

[2.2 后端开发技术 5](#_Toc161943427)

[2.2.1 MVT模式 5](#_Toc161943428)

[2.2.2 Django框架 6](#_Toc161943429)

[2.2.3 SQLite 6](#_Toc161943430)

[2.2.4 ORM 6](#_Toc161943431)

[2.3 开发环境及开发工具 7](#_Toc161943432)

[2.3.1 Pycharm 7](#_Toc161943433)

[2.3.2 Navicat 8](#_Toc161943434)

[3 系统分析 9](#_Toc161943435)

[3.1 可行性分析 9](#_Toc161943437)

[3.1.1 技术可行性 9](#_Toc161943438)

[3.1.2 经济可行性 9](#_Toc161943439)

[3.1.3 社会可行性 9](#_Toc161943440)

[3.2 需求分析 11](#_Toc161943441)

[3.2.1 功能需求 11](#_Toc161943442)

[3.2.2 性能需求 11](#_Toc161943443)

[3.2.3 可扩展性需求 12](#_Toc161943444)

[4 系统设计 14](#_Toc161943445)

[4.1 系统模块设计 14](#_Toc161943447)

[4.1.1 普通用户模块 14](#_Toc161943448)

[4.1.2超级管理员模块 14](#_Toc161943449)

[4.1.3系统功能模块图 14](#_Toc161943450)

[4.2 数据库设计 15](#_Toc161943451)

[4.2.1 数据库设计原则 15](#_Toc161943452)

[4.2.2 数据库设计 16](#_Toc161943453)

[4.2.3 数据库表设计 16](#_Toc161943454)

[5 系统实现 20](#_Toc161943455)

[5.1 模块开发 20](#_Toc161943456)

[5.2 用户注册 20](#_Toc161943457)

[5.3 用户登录 22](#_Toc161943458)

[5.4 签到 23](#_Toc161943459)

[5.5 考勤统计 25](#_Toc161943460)

[5.6 通知栏 26](#_Toc161943461)

[5.7 通知栏管理 27](#_Toc161943462)

[5.8 考核管理 29](#_Toc161943463)

[5.9 考核评定记录 31](#_Toc161943464)

[5.10 事项 32](#_Toc161943465)

[5.11 班级管理 34](#_Toc161943466)

[5.12 成员管理 37](#_Toc161943467)

[5.13 退出登录 40](#_Toc161943468)

[5.14 Django超级管理员后台 41](#_Toc161943469)

[6 系统测试 44](#_Toc161943470)

[6.1 测试方法 44](#_Toc161943471)

[6.2 测试内容 44](#_Toc161943472)

[6.3 系统的测试结果 48](#_Toc161943473)

[结论 49](#_Toc161943474)

[致 谢 51](#_Toc161943475)

[参考文献 52](#_Toc161943476)

# 1 绪论

1.1 研究背景

在当前的教育环境中，家校通成为了一种重要的教育工具。中小学阶段，家校之间传统的口头和书面沟通模式，导致家长和学生不能及时了解学校通知、学生表现和成绩情况，有的学生遇到问题不敢询问。随着信息技术的快速发展和互联网的普及，即时通讯工具及各类软件平台应运而生，比如微信等群聊，非常实时便利，能发布通知和成绩信息、进行事项申请，但是容易因消息过多而错过，无法方便了解孩子的历史成绩和成绩趋势，以及在班级中的表现情况。同时，家长也可能受到孩子冒充的影响，导致沟通不畅或产生误解。为了解决这些问题，家校通应运而生，以满足新的教育需求。

传统的教学模式主要依赖于课堂教学和面对面交流，但存在资源有限和互动不足等问题。而家校通利用互联网技术提供了数字化的学习和合作平台，使得班主任、学生和家长能够更便捷地进行在线交流和合作。通过提供通知、讨论、成绩分析和签到等功能，家校通为教育提供了新的途径，拓展了教学和学习的可能性

家校通的课题时代背景主要包括以下几个方面：

1. 技术发展：随着互联网技术的普及和移动设备的普及，人们对于家校沟通的需求不断增加。微信等群聊具有一定的缺点和局限性。家校通利用先进的技术手段，满足了这一需求。

2. 教育改革：教育改革倡导个性化、合作性和探究性学习，传统的教学方式已经无法满足这些需求。家校通的出现为教育改革提供了新的工具和平台，促进了教学模式的创新。

3. 家校互联：家庭用户可在平台了解最新的通知，了解学生的成绩信息及表现，寻求老师的帮助等

4. 学生需求：现代学生对于交流的要求也发生了变化，他们更加习惯于数字化、在线化的交流方式。家校通提供了符合班级和家庭需求的学习环境，提高了学习的参与度和效果。

总的来说，家校通的课题时代背景源于教学模式的变革、技术的进步和家庭需求的变化。它为教育提供了更加灵活、便捷和个性化的学习方式，推动了教育的创新和进步。

1.2 研究目的及意义

研究目的：

1. 研究如何利用家校通改善家校沟通和班级管理，以满足教育领域的新需求。

2. 研究如何利用家校通促进教学模式的创新，提高家庭用户的参与度和学习效果。

研究意义：

1. 提供了一个数字化的学习和合作环境，使班主任、家庭用户（家长和学生）能够更加便捷地进行在线交流和协作。

2. 提供了事件通知、作业发布、讨论、成绩分析等功能，为教学和学习提供了更多的可能性。

3. 解决了传统家校口头传达和书面传达沟通模式的局限性，使家庭用户能够及时了解学校通知、孩子的表现情况和成绩情况。

4. 提供了一个安全可靠的平台，避免了孩子冒充家长在群聊中传达不实信息的问题。

5. 促进了教育改革，倡导个性化、合作性和探究性学习，推动了教学模式的创新和进步。

研究家校通的目的和意义在于探索如何利用现代技术和平台改善教育环境，提高教学质量和学生的学习效果，让班主任和家庭用户之间的有效沟通和合作。

1.3 国内外研究现状

早期的家校通信平台是由三大通信运营商为代表开发的，其主要功能是与用户的电话号码绑定，仅限于允许运营商使用该电话号码。家长接收和回复短信，实现家庭和学校之间的初步沟通。这个平台在一定程度上促进了家庭和学校之间的交流[13]，但随着技术的发展和需求的增加，家校交流平台也在不断更新。2016年1月，北京市教委将家校交流平台更名为“师生家校”，在北京市各小学推广。随后，武汉市洪山区一所小学率先试点了家校交流APP平台，并在全区推广使用[3][13]。这表明各地教育部门对家校通平台的认可，并希望通过这种方式促进家校沟通的效率和便利性。随着时间的推移，家校交流APP在手机端或电脑端的应用也在不断涌现。

美国在教育信息化方面一直走在世界前列。随着信息技术的不断发展，美国教育信息化的应用也走在了世界前列。美国历任总统都坚持利用信息技术促进教育改革和发展的战略方针[1]。直到20世纪90年代，美国大学教师开始使用电子邮件与学生家长沟通，并使用RFID门禁系统跟踪学生在校园的进出情况[2] 。在美国加州，使用Android技术开发幼儿园家校连接。家长可以通过这个平台轻松了解孩子在学校的情况[13]，包括学习进展、活动照片和视频等。2011年，硅谷诞生了Remind产品，成为美国最大的家校通之一。超过150万名班主任和2350万名学生家长使用Remind进行沟通，解决了班主任、家长和学生之间的沟通问题，促进了家校合作，提高了教学质量[4]。

1.4 研究内容

家校通项目提供了签到功能，在中小学的许多场景中具有重要作用。举例来说，学生离家上学时，家长可以进行签到代表学生已离家，系统将记录签到时间。学生放学回家时，家长可以进行签退操作，系统将记录签退时间，表示孩子已经安全到家。班主任通过这些信息可以了解学生是否安全到家，还可以计算从家到学校、从学校回家所花费的时间。

考勤分析统计功能可以通过计算签到到签退的总时长，并用柱状图显示时间。这样，班主任可以清楚地了解学生的出勤情况。

家庭用户可以查看班主任发布的通知，包括交费通知、放假通知、作业通知等。不同等级的通知会用不同颜色显示，代表通知的紧急程度。通知管理功能只能由班主任访问，家庭用户无权限。班主任可以在此发布通知，并设置通知的不同等级。

考核评定记录所有用户都可以访问，可以搜索历史考核记录名称，查看所有学生的成绩和评语。成绩分为优秀、良好、及格、不及格四个等级，每个学生的成绩按照等级分类，不同等级的成绩用不同颜色显示，以便区分。考核管理功能只能由班主任访问，班主任可以在此发布每次的考试成绩，并对每个学生写评语。还可以设置总分，如100、120、150，以便计算成绩的等级。

所有用户可以在事项功能区进行讨论。学生可以在此提出问题、讨论，并向班主任请假。班主任回复同意的消息将以绿色背景显示，不同意的将以红色背景显示。

在班级管理模块中，班主任可以修改班级名称，例如从四年级改为五年级。

在成员管理模块下，班主任可以修改学生的基本信息，包括姓名、联系方式等。为了安全起见，密码将显示为经过MD5加密的信息。

退出登录模块中，系统界面的所有元素将变灰，用户有十秒钟的时间来考虑取消或确认退出登录。超时后，系统将自动退出登录。

2 系统相关技术

2.1 前端开发技术

2. 1. 1. Bootstrap框架

Bootstrap是一种前端框架，基于HTML、CSS和JavaScript开发，以其简洁、灵活和高效而备受用户喜爱。该框架提供了丰富的样式和组件，但在实际开发中，有时并不需要全部样式或组件，或者Bootstrap提供的样式和组件无法满足实际需求。因此，我需要对基于Bootstrap的样式和组件进行相应的修改。Bootstrap 5相对于Bootstrap 4有很多改变，引入了大量的CSS变量和Sass变量，使得定制Bootstrap变得更加容易。在这里，我将分别介绍两种定制情况：一种是使用预编译版的编译好的Bootstrap框架，另一种是使用源码版[4]。

它具有以下特点和优势：

响应式布局：Bootstrap提供了响应式的网格系统，可以自动调整布局以适应不同设备的屏幕尺寸，使网站在各种设备上都能良好展示。

样式组件：Bootstrap提供了丰富的CSS样式组件，如按钮、表单、导航、警告框等，可以直接使用这些组件来构建页面，减少了开发者的工作量。

JavaScript插件：Bootstrap内置了多个常用的JavaScript插件，如轮播图、模态框、下拉菜单等，可以通过简单的HTML标记和CSS类来使用这些功能，提高了开发效率。

跨浏览器兼容性：Bootstrap经过广泛测试，能够在各种现代浏览器上良好运行，确保了网站在不同浏览器下的一致性和稳定性。

社区支持和文档资源丰富：Bootstrap拥有庞大的开发者社区，提供了大量的第三方资源、模板和插件。官方提供了详细的文档和示例，开发者可以轻松上手并获得支持。

通过使用Bootstrap，开发者可以快速构建具有现代化外观和良好用户体验的网站。它适用于各种规模的项目，无论是个人网站、企业网站还是Web应用程序，都可以从Bootstrap的特点和优势中受益。

使用该框架使我更加容易开发家校通的前端页面，让页面更加美观，符合审美。

* + 1. ECHARTS

ECharts是一款由百度开发的开源可视化图表库，ECharts能够在PC机和移动设备上流畅运行，并且兼容目前绝大部分浏览器。它提供直观、可交互的数据可视化图表，用户只需提供基本数据，ECharts就能通过适当的动画展示数据的变化[5]。

ECharts在数据分析、商业报告、可视化大屏、地理信息系统等领域有广泛的应用。它是一个功能强大、易于使用和高度可定制的图表库，为开发者提供了丰富的工具和功能来创建精美而交互式的数据可视化图表。

使用Echarts，让我在考勤统计分析页面，把数据直观地显示给班主任用户。能让班主任一眼清楚考勤的基本情况。

* 1. 后端开发技术
     1. MVT模式

当前，几乎所有基于Python的Web开发框架都鼓励或要求开发者采用MVC设计模式。Django作为一个全栈框架，采用了MTV模式，即模型（Model）、模板（Template）和视图（View）。相较于传统的MVC设计模式，Django对视图进行了改进，将其分为View模块和Template模块两部分，将动态逻辑处理与静态页面展示分离开来。此外，Django还采用了ORM技术来处理Model，它将关系型数据库表抽象成Python的类，使得数据库表的操作可以转换为Python类的操作，从而避免了编写复杂的SQL语句[6]。如图2-1。下面是MVT模式的流程：

1. 模型（Model）：模型负责定义数据结构和数据库的操作。在Django中，使用Python类来定义模型，并通过ORM（对象关系映射）将其映射到数据库表。模型定义了数据的结构和关系，包括字段、关联关系和数据验证规则。

2. 视图（View）：视图负责处理用户的请求和生成响应。在Django中，视图是一个Python函数或类，接收HTTP请求并返回HTTP响应。视图可以从模型中获取数据，处理业务逻辑，并渲染模板生成HTML页面或其他格式的响应。

3. 模板（Template）：模板负责定义HTML页面的结构和呈现。在Django中，使用Django模板语言（DTL）编写模板。模板可以包含动态数据和逻辑控制，通过模板标签和过滤器实现。视图可以将数据传递给模板，模板使用这些数据来生成最终的HTML页面。

MVT模式的流程如下：

1. 用户发送HTTP请求到Django应用程序。

2. Django的URL映射器根据请求的URL将请求转发给相应的视图函数或类。

3. 视图函数或类接收请求，处理业务逻辑，可能会与模型进行交互获取数据。

4. 视图将数据传递给模板，并使用模板渲染生成HTML页面。

5. 视图将最终的HTML页面作为HTTP响应返回给用户。

通过MVT模式如图2-1所示，基于Django的家校通实现了业务逻辑、数据处理和页面呈现的分离，使开发更加模块化和可维护。同时，Django提供了丰富的功能和工具，简化了开发过程，提高了开发效率。

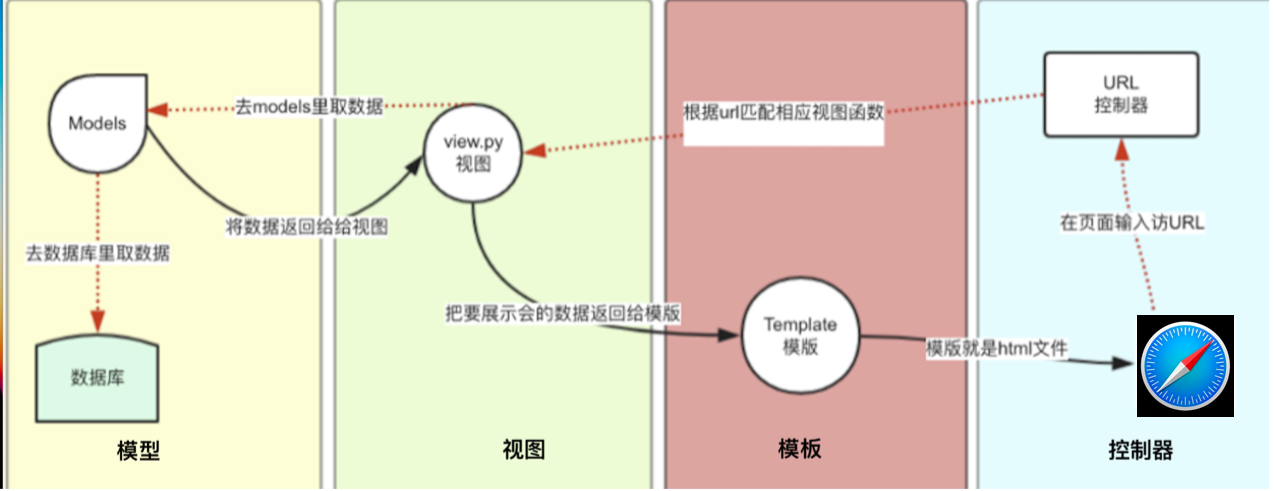


图2-1 MVT模式图表

* + 1. Django框架

Django是Python语言中比较成熟的Web框架。Django框架来了

具有更多的功能并在很大程度上简化了Web应用程序的开发过程。 django有可插拔形式的APP，可以创建APP将相应的功能拆分成不同的模块，一方面使得Web项目更加规范，易于开发并维护； 另一方面，它也使得 Django 开发 Web 服务变得可行。高效便捷。 开发者可以快速构建高质量的Web应用[7]。

* + 1. SQLite

SQLite是由D.Richard Shipp于2000年发明的一款功能强大的嵌入式关系数据库管理系统。首先，SQL 数据库引擎是一个完全可集成的零配置SQL数据库引擎，即使在系统故障或电源故障后也具有以下功能。其次，零配置，这意味着不需要安装和管理。第三，系统使用了SQL92的大部分标准。每个单独的硬盘文件可以保存一个完整的数据库，并且可以在不同的计算机之间自由共享。我可以看到，SQLite在嵌入式开发中很好地构建了上述功能。SQLite受版权许可，没有任何限制，包括商业产品[8]。

* + 1. ORM

ORM（对象关系映射）是一种技术，它将以Python类形式定义的数据模型与关系数据库进行连接，从而提供了一个简单易用的数据库API。通过使用ORM框架，开发者无需编写复杂的原生SQL语句，就可以对数据库进行操作。具体原理是将数据库中的表、行和字段映射为类、实例和属性。在Django中，当对数据库进行操作时，ORM框架会将对类、实例和属性的操作转化为相应的原生SQL语句。这样的设计使得开发者可以更方便地进行数据库操作，同时也提高了代码的可读性和可维护性[9]。用于在关系型数据库和面向对象编程语言之间建立映射关系。它可以简化开发过程、避免SQL注入、提高可维护性，并提供了跨数据库支持和对象化查询语言等优点。常见的ORM框架有Hibernate、Django ORM、Entity Framework和Ruby on Rails等。

使用 ORM 技术是的家校通能更加方便得用代码操作数据库，使我更加快速简便的完成家校通的开发。

* 1. 开发环境及开发工具
     1. Pycharm

JetBrains为软件创建智能、提高生产力的工具 开发人员简化其具有挑战性的任务并自动执行例程 的。JetBrains 广泛的开发人员生产力工具包括 最流行Python 开发工具：PyCharm[12]。

PyCharm是一款由JetBrains开发的集成开发环境（IDE），专为Python语言开发者设计。该系统用这个软件主要是因为它提供了丰富的功能和工具，旨在提高Python开发的效率和质量。

PyCharm具有许多功能，包括代码编辑器、调试器、版本控制集成、代码导航、自动完成、代码重构、单元测试工具等。它还支持多种Python框架和库，如Django、Flask、NumPy和Pandas，使开发者能够轻松地构建和管理Python项目。

PyCharm的代码编辑器具有智能代码完成和语法高亮等功能，可以帮助开发者编写准确和规范的Python代码。调试器可以帮助开发者在运行时检查和修复代码中的错误。版本控制集成使开发者能够方便地使用Git等工具进行代码版本管理。

此外，PyCharm还提供了丰富的插件生态系统，允许开发者根据自己的需求扩展和定制IDE。它还支持多种操作系统，包括Windows、Mac和Linux，使开发者能够在自己喜欢的平台上进行Python开发。

总之，我使用PyCharm是一款功能强大的Python集成开发环境开发家校通，为我提供了丰富的工具和功能，可以提高家校通开发的效率和质量。

* + 1. Navicat

Navicat Premium是Navicat系列产品中的一个，它提供了一个集中式的界面，可以连接多个数据库并进行管理操作。使用Navicat Premium，我可以方便地进行批量操作，如导入导出数据、创建查询工具、修改表字段、进行数据同步和备份等[10]。它支持多种数据库类型，使得数据库管理变得更加简单和高效。无论您是开发人员、数据库管理员还是数据分析师，Navicat Premium都是一个强大而实用的工具，可以帮助您更好地管理和操作数据库。

使用Navicat，我可以直观快速地查看数据库的改变，以验证我用代码操作数据库增加、删除、修改代码的正确性。

3 系统分析

1. 1. 可行性分析

3.1.1 技术可行性

使用Bootstrap、Django和SQLite的技术组合是可行的。Bootstrap提供了响应式和美观的界面设计工具，Django是一个功能强大的Python Web框架，而SQLite是一种轻量级的嵌入式数据库引擎。这些技术都有广泛的社区支持和文档资源，可以帮助开发者快速构建高效和可维护的Web应用程序。选择这些技术时，需要考虑项目需求、团队熟悉程度和可扩展性等因素。总体而言，使用Bootstrap、Django和SQLite可以提供良好的用户体验、高效的开发过程和可靠的数据存储。

3.1.2 经济可行性

家校通的经济层面可行性分析主要涉及以下几个方面：

1. 成本分析：对于家校通的开发和维护，需要考虑到相关的成本，包括开发人员的薪资、服务器和网络设备的费用、软件许可证的费用等。同时还需要考虑到长期运营和维护的成本，如服务器维护、软件更新等。

2. 收益预测：家校通的经济可行性还需要对其可能带来的收益进行预测。这包括可能的用户付费模式、广告收入、增值服务等。需要进行市场调研和用户调研，评估用户对该平台的需求和愿意支付的价格。

3. 市场竞争分析：在进行经济可行性分析时，还需要考虑到市场竞争的情况。家校通可能面临其他类似的产品或竞争对手，需要评估其在市场上的竞争力和差异化优势，以制定合适的定价和市场推广策略。

4. 风险评估：在经济可行性分析中，还需要评估可能的风险和不确定性因素。这可能包括市场需求波动、竞争加剧、技术变革等。需要制定相应的风险管理策略，降低风险对经济可行性的影响。

通过对以上方面的分析和评估，可以更好地了解家校通在经济层面的可行性，为决策提供依据，并制定相应的商业计划和策略

3.1.3 社会可行性

家校通系统在社会上具有很高的可行性。首先，签到签退功能可以提高学校管理的效率和准确性，有助于确保学生的安全和出勤情况。其次，发布和查看通知功能能够方便学校与家长之间的沟通，及时传达重要信息，促进家校合作。讨论功能则可以促进学生、家长和教师之间的交流与讨论，有助于共同解决问题和提高教育质量。成绩发布和历史成绩查看功能能够帮助家长和学生了解学习进展，及时发现问题并进行调整。最后，成绩登记划分功能有助于教师准确记录学生成绩，为学生个性化学习提供更好的支持。综上所述，家校通系统的功能设计满足了教育管理和家校沟通的需求，对提高教育质量和促进学生学习具有积极作用，因此在社会上具有很高的可行性。

* 1. 需求分析
     1. 功能需求

通过对中小学的线下走访调查，我们总结得出了平台的用户分为两种角色：班主任用户和家庭用户。家庭用户包括家长和学生，他们共用同一个账号，这是因为学生和家长的需求功能基本上一致，所以不需要将其拆分成两个账号处理。班主任拥有所有功能权限，而家庭用户则只需要基本功能即可。

（1）家庭用户需求

该平台为家庭用户提供了注册和登录功能，他们可以注册账号并登录平台，登录后即成为家庭用户角色。家庭用户能够进行签到操作，例如学生离开家时进行签到，学生回到家时进行签退。系统会记录这些时间，以便班主任了解学生的上学时间，并确认学生是否安全到家。

在平台中，家庭用户可以在通知栏查看已发布的通知，并在事项管理模块进行多功能操作，包括请假、讨论、提问，以及查看所有事项。此外，家庭用户还能够查看所有考核记录，例如考试成绩。这些成绩包括学生的所有历史考试成绩记录，以及对应的分数。每个考试成绩被分为优秀、良好、及格和不及格等等等级，系统会将所有学生的成绩按照分数等级分类，并用不同颜色直观展示，以方便家庭用户了解学生在考核中的表现情况。

（2）班主任需求

系统为班主任提供了丰富的功能，包括登录系统、查看学生的签到签退记录、发布和查看通知、事项管理、考核记录等。用户可以通过登录功能进入平台，角色为班主任。

签到功能包括查看自己班级的学生签到、签退情况以及时长统计，方便记录用户的活动情况。通过柱状图示签到总时长，用户能够清晰了解签到情况。

通知栏允许用户查看已发布的通知，并进行通知管理，包括发布通知和设置通知级别。

事项管理模块提供了多功能，包括请假、讨论、提问和查看所有事项。

考核记录和考核管理模块方便用户查看和管理考核情况，包括发布考核、评分和撰写评语。

此外，还有班级管理和成员管理功能，班主任可以对班级信息和成员信息进行管理和修改。

3.2.2 性能需求

根据用户需求，需要构建一个性能良好的家校通，提供流畅的用户体验和高效的操作响应。以下是需要考虑的一些因素：

1. 响应时间：平台应该能够快速响应用户的请求，避免长时间的等待和延迟。这可以通过优化代码、使用高效的算法和数据结构以及合理配置服务器来实现。

2. 扩展性：平台应该具备良好的扩展性，能够处理大量的用户和数据。这可以通过横向扩展服务器、使用负载均衡和缓存技术来实现。

3. 数据库性能：平台的数据库设计和查询应该经过优化，以提高数据的读写速度和查询效率。可以使用索引、合理拆分数据表和优化SQL查询来改善数据库性能。

4. 缓存机制：使用缓存技术可以减少对数据库的频繁访问，提高数据的读取速度。适当地使用缓存可以显著提升平台的性能。

5. 前端优化：前端页面应该经过优化，减少不必要的网络请求和资源加载时间。可以使用浏览器缓存、压缩静态资源和异步加载等技术来提高前端性能。

家校通项目选择了Python的Django框架作为后端开发工具，这是一个经过广泛应用和验证的成熟框架，具有稳定性和可靠性。Django框架提供了强大的功能，包括简化数据库调用、安全性、认证和授权机制等。通过使用Django，开发人员可以更专注于业务逻辑的实现，而不必花费太多精力在底层的技术细节上。

在前端方面，项目选择了Bootstrap框架，这是一个流行的前端框架，具有丰富的预设计组件，可以加快开发速度并提高用户界面的一致性和美观性。Bootstrap框架与Django框架相辅相成，使得开发人员可以在前端开发中更高效地创建用户界面，而不必从头开始设计和实现各种组件。

考虑到家校通项目的软件环境，macOS 13.6和Python 3.10都是当前流行和稳定的版本，因此在这样的环境下运行应用程序是可行的。另外，i5第八代处理器作为硬件环境也能够满足应用程序的运行要求，因为家校通这类普通的教育管理系统并不需要特别高的硬件性能。

综上所述，家校通项目基于Python的Django框架和Bootstrap框架的技术选择在技术上是可行的。它们能够提供稳定、高效的开发环境，使得开发团队可以更加专注于实现项目的功能和需求，而不必过多关注底层的技术细节。

3.2.3 可扩展性需求

家校通的可扩展性是确保平台能够满足未来需求和功能的重要因素。以下是一些考虑因素：

1. 模块化设计：平台应该采用模块化的设计，将功能划分为独立的模块或组件。这样，在需要添加新功能时，可以更容易地扩展和集成新模块，而不会对整个系统造成过多的影响。

2. 松耦合架构：平台的各个模块应该是松耦合的，即彼此之间的依赖性应尽量降低。这样，在引入新功能或修改现有功能时，可以最小化对其他模块的影响，提高扩展性。

3. 可插拔的插件系统：平台可以提供插件系统，允许开发者开发和集成第三方插件。这样，用户可以根据自己的需求选择安装和使用特定的插件，实现个性化的功能扩展。

4. 水平扩展：平台应该支持水平扩展，即通过增加更多的服务器节点来分担负载。这可以通过负载均衡和分布式架构来实现，确保平台在用户量增加时仍能保持高性能和可用性。

5. 异步处理：平台应该支持异步处理机制，将一些耗时的操作转移到后台进行处理，避免阻塞用户界面。这样可以提高系统的并发能力和响应速度。

6. 数据库扩展性：平台的数据库设计应考虑到未来的数据增长和查询需求。可以采用分库分表、数据库集群等技术来提高数据库的扩展性和性能。

4 系统设计

1. 1. 系统模块设计

4.1.1 普通用户模块

所有用户都拥有的功能:

（1）注册模块：可注册账号登录平台，注册账号为家庭用户类型，注册所需要的信息包括学号、用户名、密码、手机号码、Email，以及由班主任提供的班级代码。

（2）登录模块：在用户登录时，系统验证邮箱和密码，验证用户类型，分配用户权限，用户可访问的自己有权限访问的页面。

（3）签到功能：签到、签退。

（4）通知栏：查看已经发布的通知，不同级别的通知会显示出不同的颜色

（5）事项管理：本模块具有多种用途，包括请假、讨论、提问等等。

（6）考核记录：查看所有考核内容及其对应的所有成员分数，不同等级的成绩进行分类，并用不同的颜色显示出来以便区分。

班主任用户拥有的额外功能:

（1）签到统计：柱状图显示签到总时长，让班主任一目了然考勤情况。

（2）考核管理：发布考核，评定分数，对每个人写评语。

（3）班级管理：对班级信息管理。

（4）成员管理；修改成员信息，删除成员、查询成员，可设置成员类别。

4.1.2超级管理员模块

超级管理员模块是 Django 框架自带的管理功能，需要手动用代码注册，拥有最高权限。以下是超级管理员的基本模块：

（1）登录模块：系统验证用户名密码，访问后台所有数据。

（2）app模块：这是所有用户界面的数据，可被超级管理员操作。

（3）认证和授权：对该模块的用户进行添加、删除、授权。

4.1.3系统功能模块图

本系统设计为前台和超级后台两个模块，前台可供班主任、家庭用户访问，这两种用户的类别不同，所拥有的功能不同。超级管理功能是 django框架自带功能，需要在后端代码进行注册配置才能正常使用，拥有所有界面数据的添加、删除、编辑、查询，对超级管理员进行授权。全部功能如图4-1所示。登录程序流程图如图4-2所示。

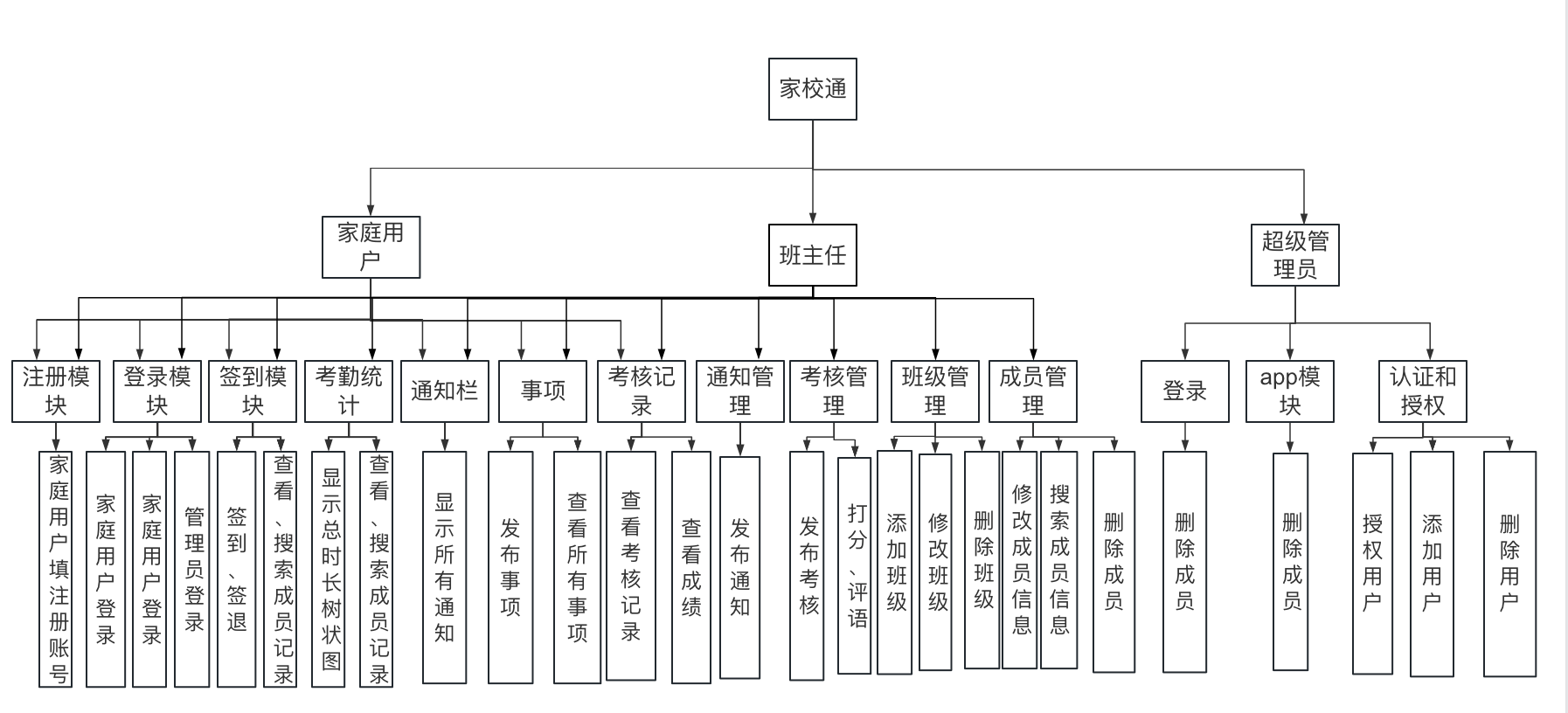


图4-1 系统功能模块图

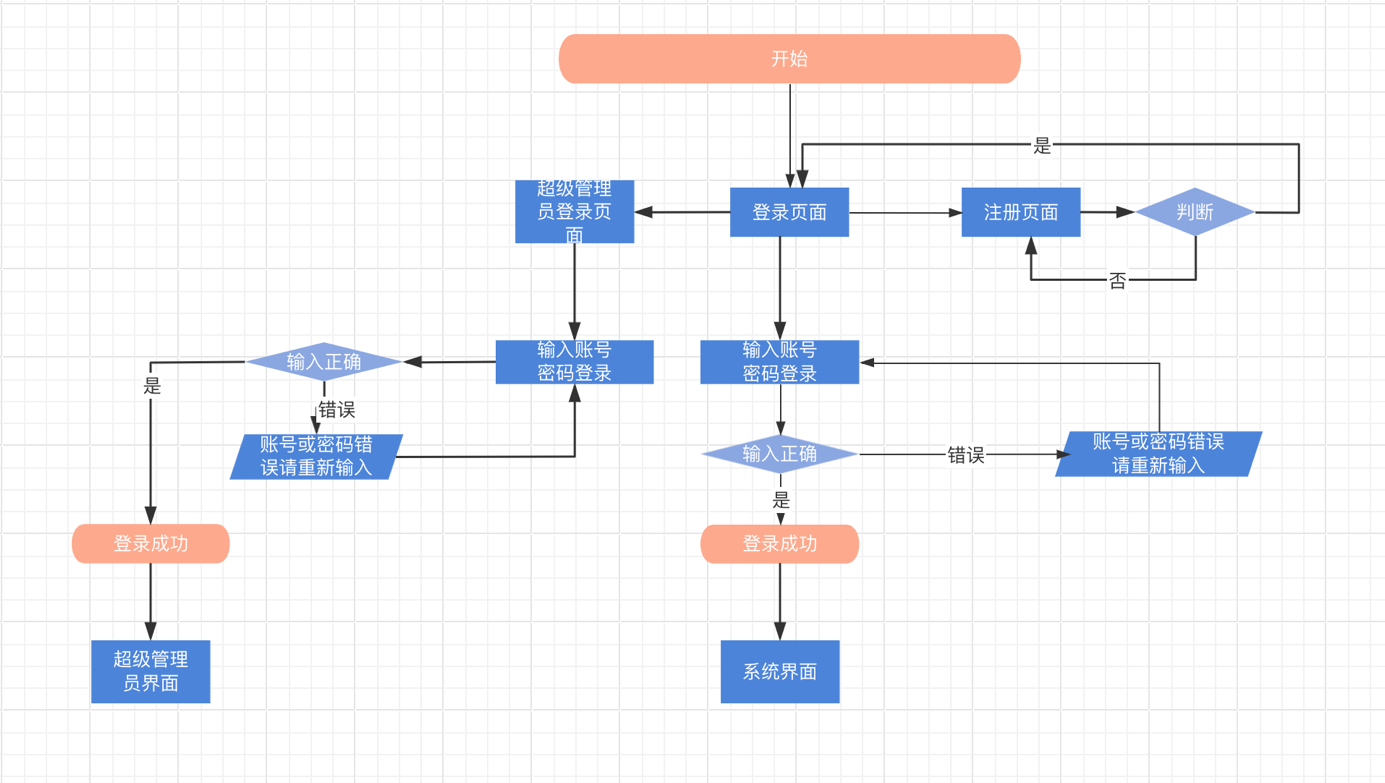


图4-2 登录程序流程图

* 1. 数据库设计
     1. 数据库设计原则

数据库,顾名思义是用来存放数据的一个仓库，数据按照其特有的规律被我放置在计算机的储存中，可以有效实现对很多数据的存储，也能消除不同客户因为实际需求不同带来的影响,可以满足家校通的基本处理要求。

数据库设计实际上就是对项目设计的一个整体规划，因为数据库的设计要从全局的角度考虑，而且数据库中表的结构都是与程序紧密相关的，如果数据库设计如果没有到位，程序的设计也会受到一定的影响。

* + 1. 数据库设计

根据需求分析，设计了八张表来存储数据，E-R图如图4-3所示

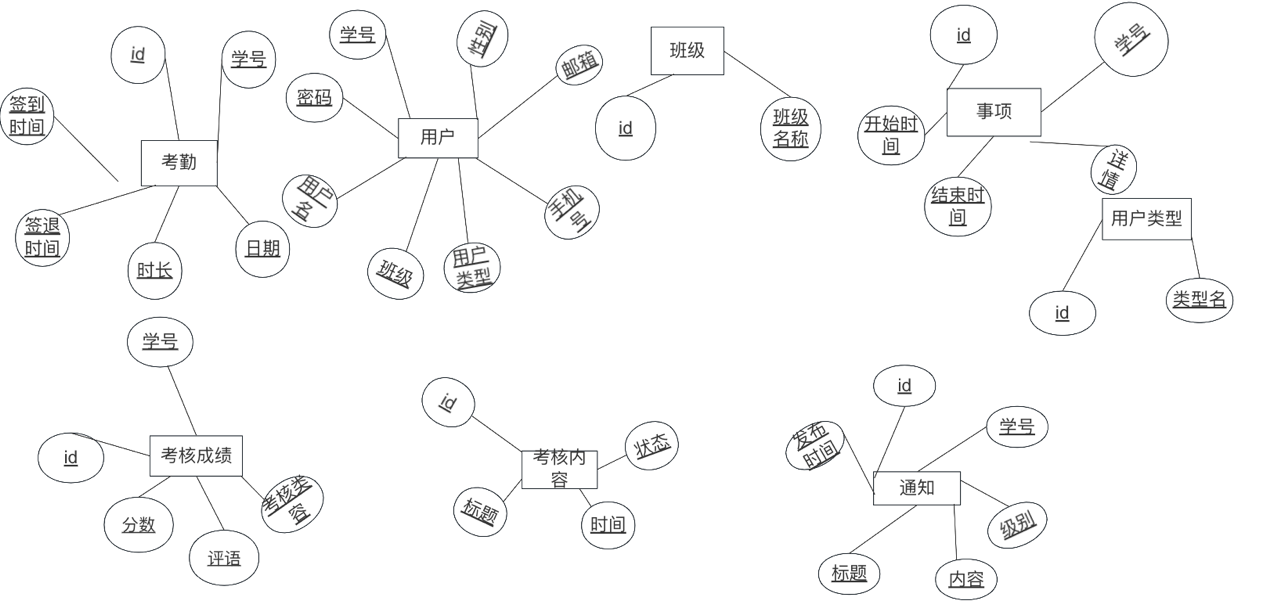


图4-3（a） 系统 E-R 图

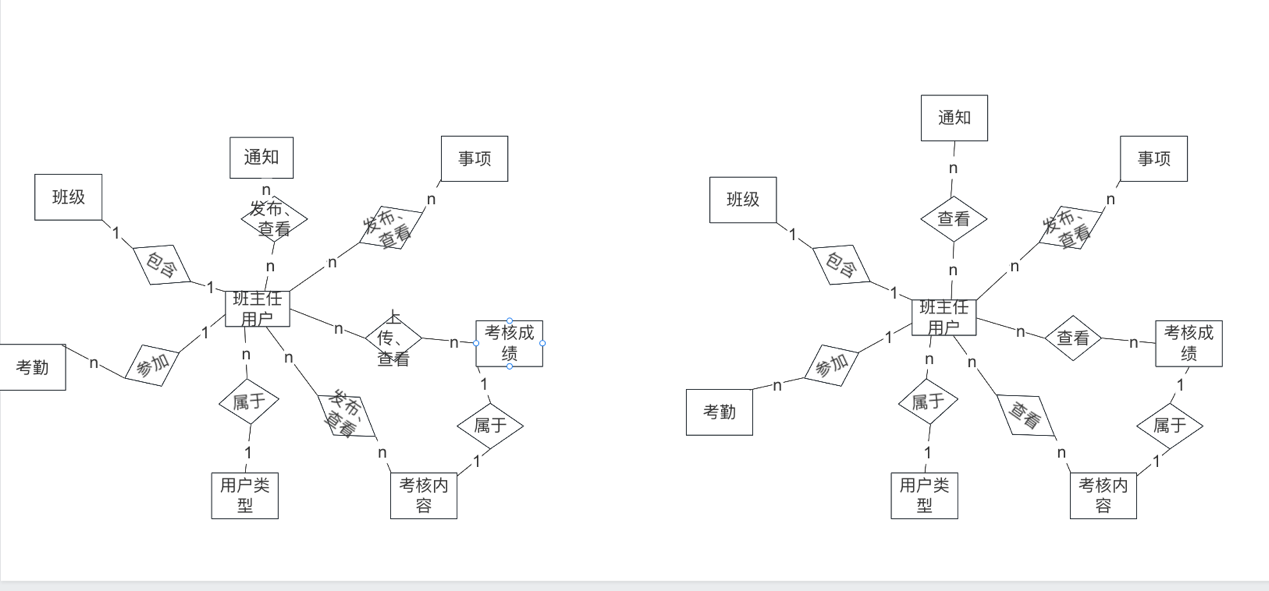


图4-3（b） 系统 E-R 图

* + 1. 数据库表设计

1. 用户类型表（app\_usertype表）：

该表存储用户类型，包括序号id和用户类型caption。其中，序号id是该表的主键，保证了每个用户类型的唯一性，当系统登录时，通过此表进行用户类型校验，检查用户的类型，获得不同的功能权限。

表4-1 用户类型表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否可为空 | 是否主键 | 说明 |
| id | INTEGER | 4 | 否 | 是 | 序号自动增 |
| caption | varchar | 10 | 否 | 否 | 用户类型 |

（2）班级表（app\_classinfo）：

该表存储用户类型，包括序号id和班级名name。其中，序号id是该表的主键，保证了每个班级类型的唯一性。

表4-2 班级表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否可为空 | 是否主键 | 说明 |
| id | INTEGER | 4 | 否 | 是 | 序号自动增 |
| name | varchar | 20 | 否 | 否 | 班级名 |

（3）用户信息表（app\_userinfo）：

该表存储用户信息，包括学号studentNum、密码password、用户名username、班级cid、用户类型user\_type、手机号phone、邮箱email和性别gender。其中，学号studentNum是该表的主键，保证了每个用户的唯一性，班级 cid是表ClassInfo 的外键，用户类型user\_type是表UserType的外键，当系统登录时，通过此表进行登录校验，检查邮箱和密码是否对应。

表4-3 用户信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否可为空 | 是否主键 | 说明 |
| studentNum | varchar | 15 | 否 | 是 | 学号 |
| password | varchar | 64 | 否 | 否 | 密码 |
| username | varchar | 15 | 否 | 否 | 用户名 |
| cid | INTEGER | 4 | 否 | 否 | 班级 |
| user\_type | INTEGER | 4 | 否 | 否 | 用户类型 |
| phone | varchar | 11 | 否 | 否 | 手机号 |
| email | varchar | 20 | 允许空 | 否 | 邮箱 |
| gender | INTEGER | 4 | 否 | 否 | 性别 |

（4）考勤表（app\_attendence)：

该表存储班主任信息，序号id、签到时间start\_time、签退时间end\_time、时长duration、日期date和学号stu。其中，序号id是该表的主键，保证了每个签到数据的唯一性，学号stu是表UserInfo的外键，当系统顶点击签到时，通过此表进行签到。

表4-4 考勤表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否可为空 | 是否主键 | 说明 |
| id | INTEGER | 4 | 否 | 是 | 序号自动增 |
| start\_time | datetime | 8 | 否 | 否 | 签到时间 |
| end\_time | datetime | 8 | 否 | 否 | 签退时间 |
| duration | decimal | 10 | 否 | 否 | 时长 |
| date | date | 3 | 否 | 否 | 日期 |
| stu | varchar | 15 | 否 | 否 | 学号 |

（5）考核成绩表（app\_exam）：

该表存储考核成绩信息，包括序号id、分数point、评语detail、考核内容content\_id和学号user。其中，序号id是该表的主键，保证了每个考核成绩的唯一性，学user是表UserInfo的外键，考核内容content是表ExamContent的外键，当系统查看成绩时，通过此表进行查看所有考核内容及其分数。当系统提交成绩时，通过此表进行储存成绩信息。

表4-5 考核成绩表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否可为空 | 是否主键 | 说明 |
| Id | INTEGER | 4 | 否 | 是 | 序号自动增 |
| point | decimal | 10 | 否 | 否 | 分数 |
| detail | TEXT | 200 | 否 | 否 | 评语 |
| content | INTEGER | 4 | 否 | 否 | 考核内容 |
| user | varchar | 15 | 否 | 否 | 学号 |

（6）考核内容表（app\_examcontent)：

该表存储班主任信息，包括序号id、标题title、时间date和状态state。其中，序号id是该表的主键，保证了每个考核内容的唯一性，当系统发布考核时，通过此表进行存储成考核内容。

表4-6 考核内容表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否可为空 | 是否主键 | 说明 |
| Id | INTEGER | 4 | 否 | 是 | 序号自动增 |
| title | TEXT | 200 | 否 | 否 | 标题 |
| date | date | 3 | 否 | 否 | 时间 |
| state | bool | 2 | 否 | 否 | 状态 |

（7）事项表（app\_leave）：

该表存储班主任信息，包括开始时间start\_time、结束时间end\_time、详情explain、用户名字user和序号id。其中，序号id是该表的主键，保证了每个事项的唯一性，用户user是表UserInfo中username的外键，当系统发布事项时，通过此表存储事项和界面显示所有事项。

表4-7 事项表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否可为空 | 是否主键 | 说明 |
| start\_time | date | 3 | 否 | 否 | 开始时间 |
| end\_time | date | 3 | 否 | 否 | 结束时间 |
| explain | TEXT | 500 | 否 | 否 | 详情 |
| user | varchar | 15 | 否 | 否 | 用户 |
| Id | INTEGER | 4 | 否 | 是 | 序号自动增 |

（8）通知表(app\_notice)：

该表存储班主任信息，包括序号id、发布时间post\_time、标题head、内容content、级别level和用户姓名author。其中，序号id是该表的主键，保证了每个通知的唯一性， 用户名author是表UserInfo中username的外键，当系统发布通知时，通过此表进行存储通知信息等。当系统查看通知时，通过此表进行存储通知信息等。

表4-8 通知表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否可为空 | 是否主键 | 说明 |
| Id | INTEGER | 4 | 否 | 是 | 序号自动增 |
| post\_time | datetime | 8 | 否 | 否 | 发布时间 |
| head | TEXT | 200 | 否 | 否 | 标题 |
| content | TEXT | 500 | 否 | 否 | 内容 |
| level | INTEGER | 4 | 否 | 否 | 级别 |
| author | varchar | 15 | 否 | 否 | 用户名 |

5 系统实现

5.1 模块开发

家校通采用模块化开发的主要目的在于提高软件的可维护性、可扩展性和可重用性。模块化开发将整个软件系统分解为独立的、功能清晰的模块，每个模块专注于处理特定的功能或任务，因此代码结构更为清晰，易于理解和维护。通过模块化设计，团队成员可以在不同的模块上独立工作，从而提高了团队的协作效率。

此外，模块化设计还促进了代码的重用。通用的功能被封装在独立的模块中，可以在整个项目中或者将来的项目中被重复使用，从而减少了重复编写代码的工作量，提高了开发效率。同时，当需要对系统进行修改或扩展时，模块化设计使得修改范围更加明确，降低了对整个系统的影响，减少了引入错误的风险。

最重要的是，模块化设计提高了系统的稳定性和可靠性。由于模块之间的耦合度较低，当一个模块出现问题时，不会影响到其他模块的正常运行，从而提高了系统的容错能力和稳定性。综上所述，家校通选择模块化开发，旨在构建一个更易于维护、扩展和稳定的软件系统，以满足日益复杂和变化的需求。

5.2 用户注册

用户注册界面如图5-1所示，当用户点击"确认注册"按钮时，前端系统调用Ajax向服务器发送注册请求，并将学号、用户名、密码、手机号码、Email发送到后端系统。使用hashlib库的sha1算法对密码进行加密，将加密后的密码和其他信息一起创建一个新的UserInfo对象并设置用户类型为家庭用户，将新创建的对象保存到数据库中。



图5-1 用户注册界面

后端用户注册View核心代码如下：

@csrf\_exempt #@csrf\_exempt表示这个视图函数不需要CSRF验证。

def register(request):

if request.method == 'POST':

#判断请求是否为Ajax请求。

if request.is\_ajax():

#获取前端传来的学号。

stu\_num\_v = request.POST.get('stu\_num\_verify')

#如果该学号已经被注册过，返回False。

if UserInfo.objects.filter(studentNum=stu\_num\_v):

ret = {'valid': False}

#创建一个字典，包含"valid": False或"valid": True。

else:

ret = {'valid': True}

#将字典转换成JSON格式并返回给前端

return HttpResponse(json.dumps(ret))

else:#如果请求不是POST方法或不是Ajax请求，返回渲染后的register.html模板。

return render(request, 'account/register.html')

def register\_verify(request):

if request.method == 'POST':

print('验证成功')

username = request.POST.get('username')

email = request.POST.get('email')

stu\_num = request.POST.get('stu\_num')

pwd = request.POST.get('password')

cid\_id = request.POST.get('class')

m1 = hashlib.sha1()

m1.update(pwd.encode('utf8'))

pwd = m1.hexdigest()

phone = request.POST.get('phone')

a=UserInfo.objects.create(username=username, email=email,cid\_id=cid\_id,studentNum=stu\_num,password=pwd,phone=phone, user\_type\_id=2)

a.save()

return HttpResponse('OK')

5.3 用户登录

用户登录界面如图5-2所示，当用输入户邮箱和密码，当用户登录时，如果是POST请求，获取用户输入的邮箱和密码，对密码进行加密。调用check\_login函数验证邮箱和密码。如果验证成功，创建一个重定向对象response，将用户的邮箱和密码分别存储在名为'qwer'和'asdf'的cookie中，并设置这两个cookie的过期时间为3600秒。如果验证失败，将错误信息传递给模板并渲染出登录页面。如果是GET请求，调用check\_cookie函数检查是否存在有效的登录cookie。如果存在有效的登录cookie，重定向到首页。如果不存在有效的登录cookie，渲染出登录页面。

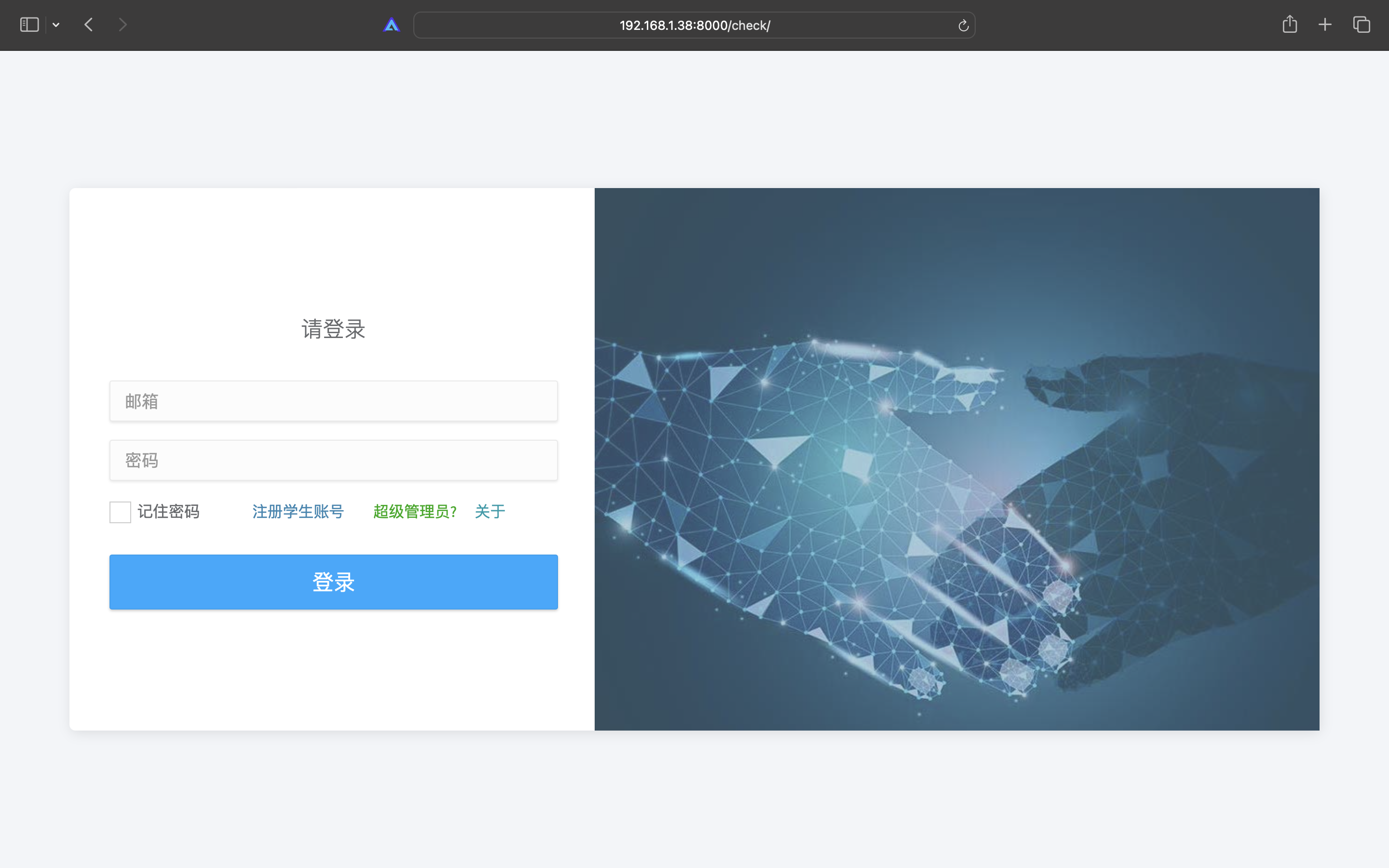


图5-2 用户登陆界面

后端用户登录的View核心代码如下：

@csrf\_exempt

def login(request):

if request.method == 'POST':

email = request.POST['email']

password = request.POST['password']

m1 = hashlib.sha1()

m1.update(password.encode('utf8'))

password = m1.hexdigest()

if check\_login(email, password):

response = redirect('/index/')

response.set\_cookie('qwer', email, 3600)

response.set\_cookie('asdf', password, 3600)

return response

else:

return render(request, 'account/page-login.html', {'error\_msg': '账号或密码错误请重新输入'})

else:

(flag, rank) = check\_cookie(request)

print('flag', flag)

if flag:

return redirect('/index/')

return render(request, 'account/page-login.html', {'error\_msg': ''})

5.4 签到

签到界面如图5-3所示，当用户签到时，前端设置flag的值，判断变量 flag 是否为真。如果为真，则进入下一层条件判断。判断请求的方法是否为 POST 方法。如果是 POST 方法，则获取请求中的 'sign' 参数的值。如果 sign\_flag 的值为 'True'，则创建一个 Attendence 对象，其中 stu 属性为 user，start\_time 属性为当前时间。如果 sign\_flag 的值为 'False'，则获取当前用户最后一次签到记录中 end\_time 为空的记录。获取当前时间，并计算出当前时间与最后一次签到记录的开始时间之间的时长（以小时为单位），并保留一位小数。更新最后一次签到记录的 end\_time 属性为当前时间，并将时长保存到 duration 属性中。

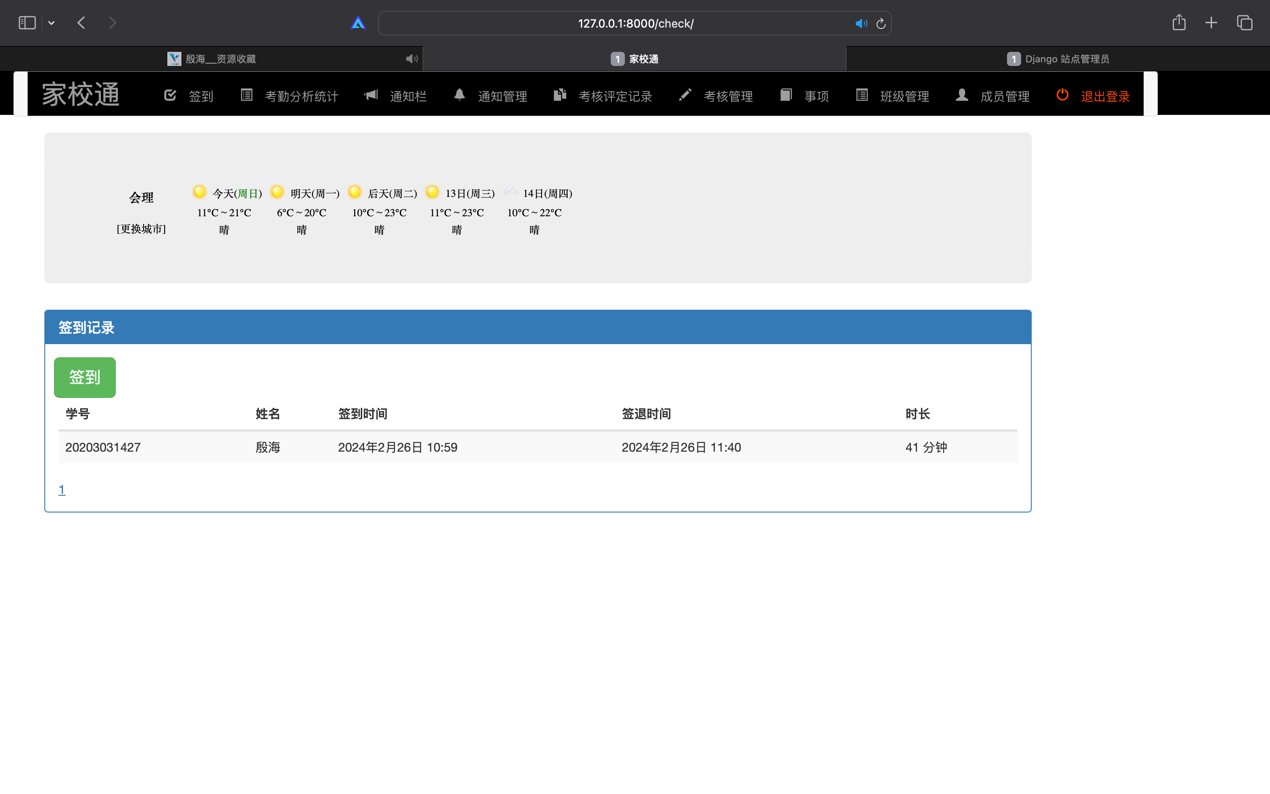


图5-3 签到模块

后端用户签到View核心代码如下：

def check(request):

(flag, rank) = check\_cookie(request)

user = rank

if flag:

if request.method == 'POST':

sign\_flag = request.POST.get('sign')

print('sign\_flag', type(sign\_flag), sign\_flag)

if sign\_flag == 'True':

Attendence.objects.create(stu=user, start\_time=datetime.datetime.now())

elif sign\_flag == 'False':

cur\_attendent = Attendence.objects.filter(stu=user, end\_time=None)

tmp\_time = datetime.datetime.now()

duration = round((tmp\_time - cur\_attendent.last().start\_time).seconds / 3600, 1)

cur\_attendent.update(end\_time=tmp\_time, duration=duration)

return HttpResponse(request, '操作成功')

else: # 查询上一个签到的状态

pre\_att = Attendence.objects.filter(stu=user).order\_by('id').last()

if pre\_att:# 如果当前时间距上次签到时间超过六小时，并且上次签退时间等于签到时间

if (datetime.datetime.now() - pre\_att.start\_time.replace(tzinfo=None)).seconds / 3600 > 6 and pre\_att.end\_time == None:

pre\_att.delete()

sign\_flag = True

elif (datetime.datetime.now() - pre\_att.start\_time.replace(tzinfo=None)).seconds / 3600 < 6 and pre\_att.end\_time == None:

sign\_flag = False

else:

sign\_flag = True

else:

sign\_flag = True

if user.user\_type.caption =='班主任':

att\_list = Attendence.objects.filter(stu\_\_cid=user.cid).order\_by('-id')

else:

att\_list = Attendence.objects.filter(stu=user).order\_by('-id')

return render(request, 'attendance/check.html', locals())

return render(request, 'account/page-login.html', {'error\_msg': ''})

5.5 考勤统计

考勤分析界面如图5-4所示，当班主任登录时，后端获取当前日期和星期几，计算本周的第一天和最后一天，查询Attendence模中日期在本周范围内的记录，并按照学号、家庭用户的用户名和班级名称进行分组，计算总时长。将查询结果转换为JSON格式，使用json.dumps()方法将查询结果列表转换为JSON字符串。使用HttpResponse()函数将JSON字符串作为响应内容返回给客户端。渲染名为'total.html'的模板，使用render()函数将模板和变量传递给模板引擎进行渲染。使用HttpResponse()函数将渲染后的页面作为响应内容返回给客户端。

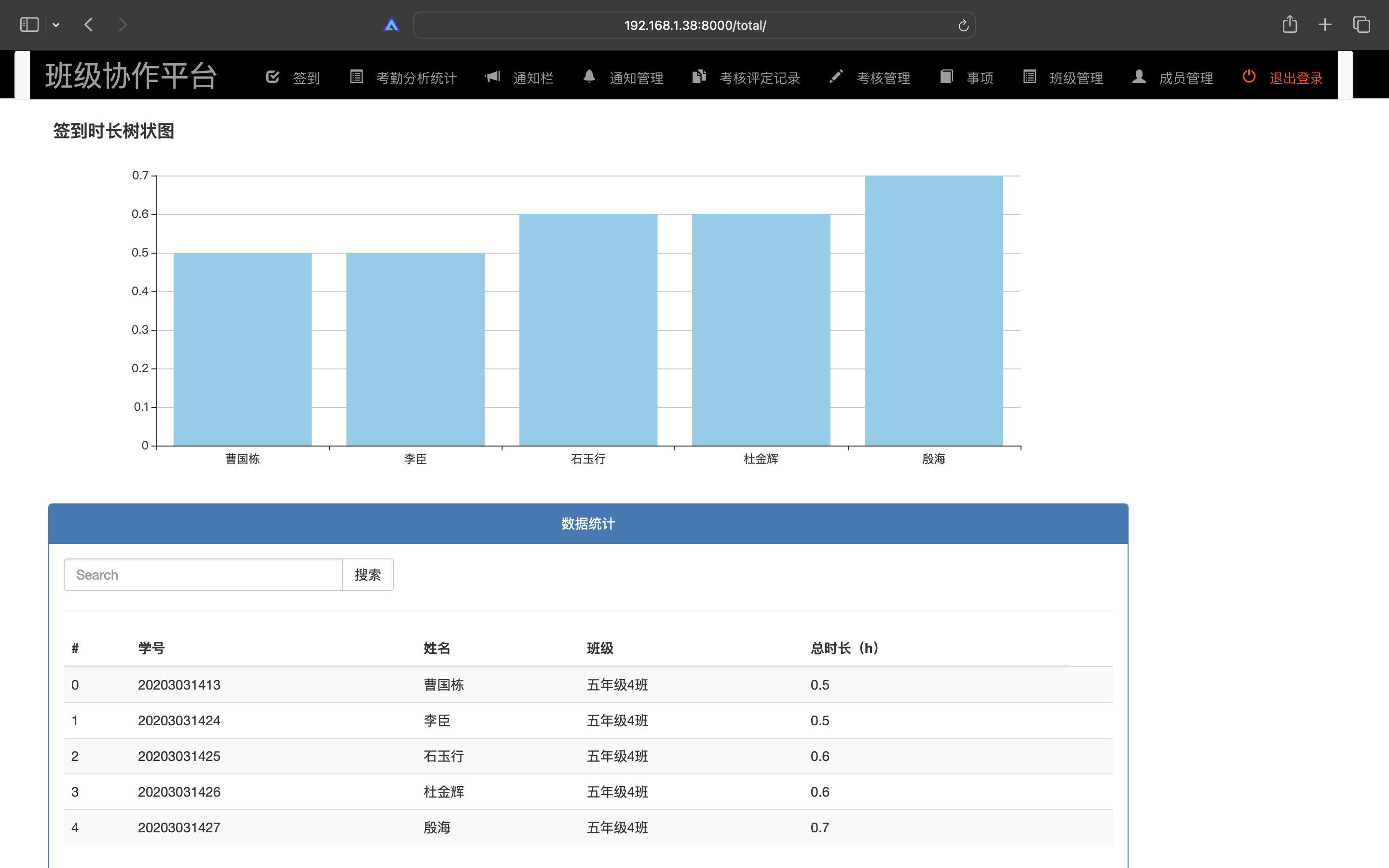


图5-4

后端用户考勤统计View核心代码如下：

@is\_login

def total(request):

(flag, user) = check\_cookie(request)

(flag, rank) = check\_cookie(request)

user = rank

if rank.user\_type.caption == '班主任':

if request.method == 'POST':

nowdate = datetime.datetime.now()#获取当前日期和星期几。

weekDay = datetime.datetime.weekday(nowdate)

firstDay = nowdate - datetime.timedelta(days=weekDay)

lastDay = nowdate + datetime.timedelta(days=6 - weekDay)

info\_list = Attendence.objects.filter(date\_\_gte=firstDay, date\_\_lte=lastDay,stu\_\_cid=user.cid).values( \

'stu', 'stu\_\_username', 'stu\_\_cid\_\_name', ).annotate(total\_time=Sum('duration')).order\_by()

info\_list = json.dumps(list(info\_list), cls=DecimalEncoder)

return HttpResponse(info\_list)

else:

nowdate = datetime.datetime.now()

weekDay = datetime.datetime.weekday(nowdate)

firstDay = nowdate - datetime.timedelta(days=weekDay)

lastDay = nowdate + datetime.timedelta(days=6 - weekDay)

info\_list = Attendence.objects.filter(date\_\_gte=firstDay, date\_\_lte=lastDay,stu\_\_cid=user.cid).values('stu', 'stu\_\_username',

'stu\_\_cid\_\_name').annotate(

total\_time=Sum('duration')).order\_by()

return render(request, 'attendance/total.html', locals())

else:

return render(request, 'denied.html')

5.6 通知栏

用户通知栏界面如图5-5所示，当用户查看通知时，该函数的功能是获取所有的Notice对象，并按照发布日期的倒序进行排序，然后将结果存储在info\_list变量中。最后，使用render函数将请求、模板路径和局部变量作为参数，返回一个渲染后的HTML页面给用户。

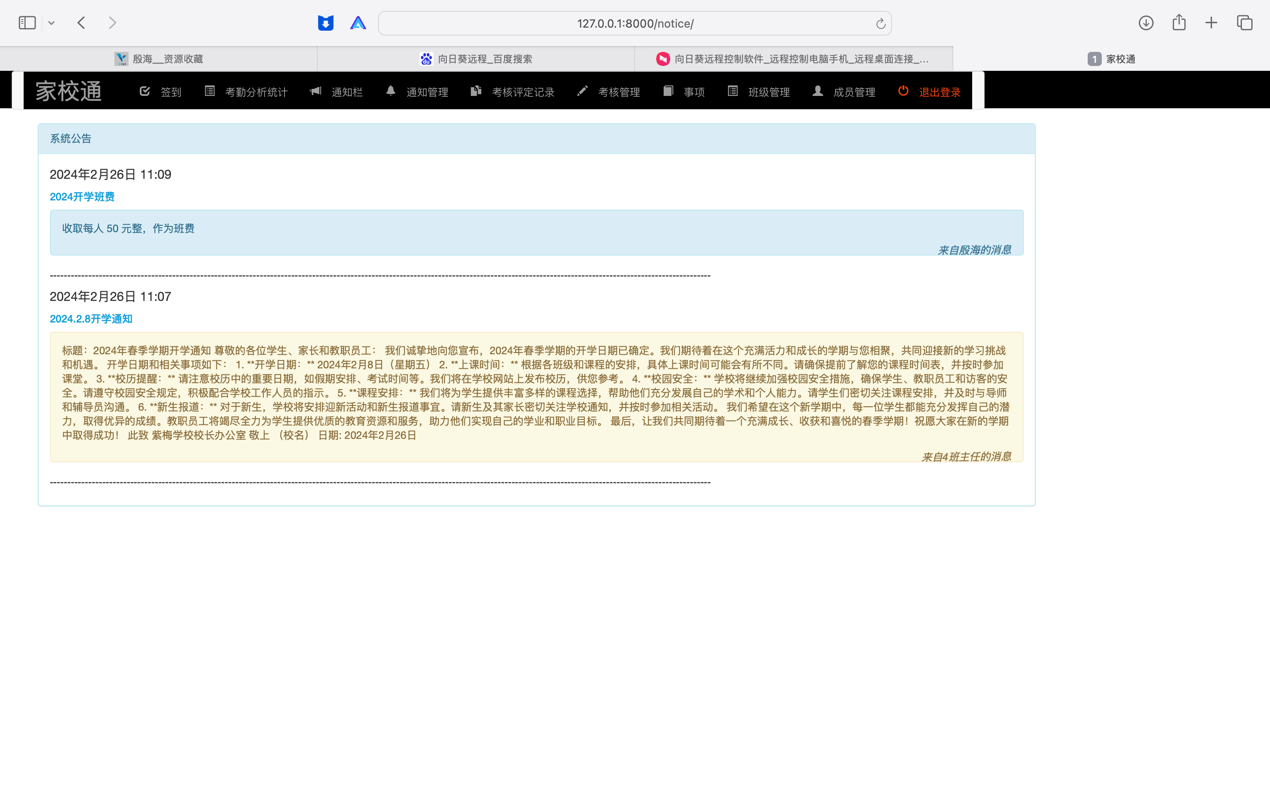


图5-5 通知栏

后端用户通知栏View核心代码如下：

@is\_login

def notice(request):

(flag, user) = check\_cookie(request)

user\_class = user.cid

info\_list=Notice.objects.filter(author\_\_cid=user\_class).order\_by('-post\_date')

return render(request, 'notice/notice.html', locals())

5.7 通知栏管理

通知管理界面如图5-6所示，当用户发布通知时，这段代码的作用是管理通知。首先，它会检查用户是否已经登录，如果没有登录，则会跳转到登录页面。然后，它会检查用户是否为班主任。如果是，则会根据请求方法创建通知或者显示通知管理页面。如果不是班主任，则会显示无权访问通知管理页面如图5-7。

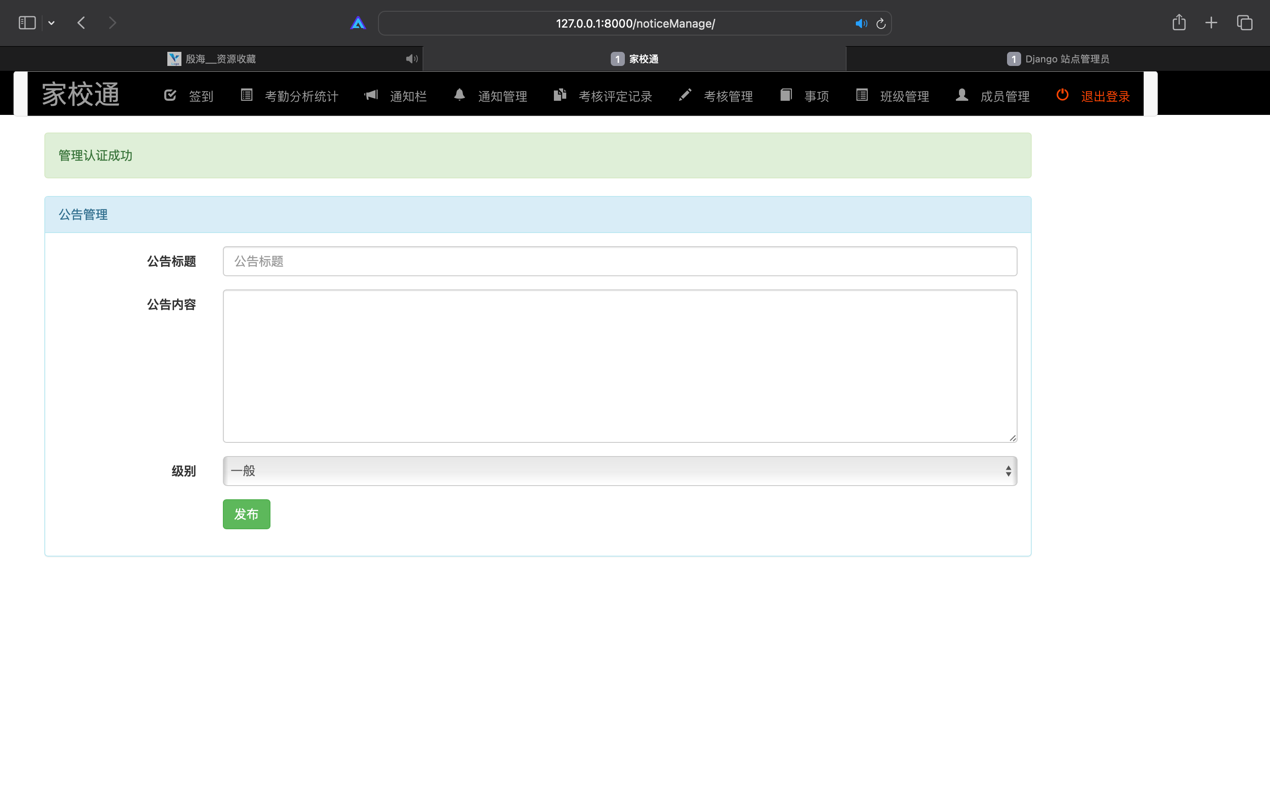
图5-6 发布通知



图5-7 学生用户在此界面无权限

后端用户通知栏管理View核心代码如下：

@is\_login

def noticeManage(request):

(flag, user) = check\_cookie(request)

if user.user\_type.caption == '班主任':

if request.method == 'POST':

title = request.POST.get('title')

content = request.POST.get('content')

level = request.POST.get('selectLevel')

Notice.objects.create(head=title, content=content, level=level, author=user)

return render(request, 'notice/notice\_manage.html')

else:

return render(request, 'notice/notice\_manage.html')

else:

return render(request, 'denied.html')

5.8 考核管理

考核管理界面如图5-8，图5-9所示，当用户发布和批阅时，代码判断用户类型是否为班主任，如果是，则继续执行相应的操作。如果不是，则返回相应的无权访问页面如图5-10。

对于考核管理，代码会根据请求方法创建考核内容或者显示考核管理页面。如果请求方法为 POST，则会获取请求中的 title 参数，并根据 title 创建一个 ExamContent 对象。如果 title 为空，则获取所有用户的数量，并获取请求中的 exam\_id 参数。然后，使用一个循环来获取每个用户的得分、学生ID 和详情，并根据这些信息创建一个 Exam 对象。最后，通过 ExamContent 的 filter 方法将对应的考核内容的状态更新为 True。

最后，代码会将考核内容列表和用户列表传递给相应的模板进行渲染。

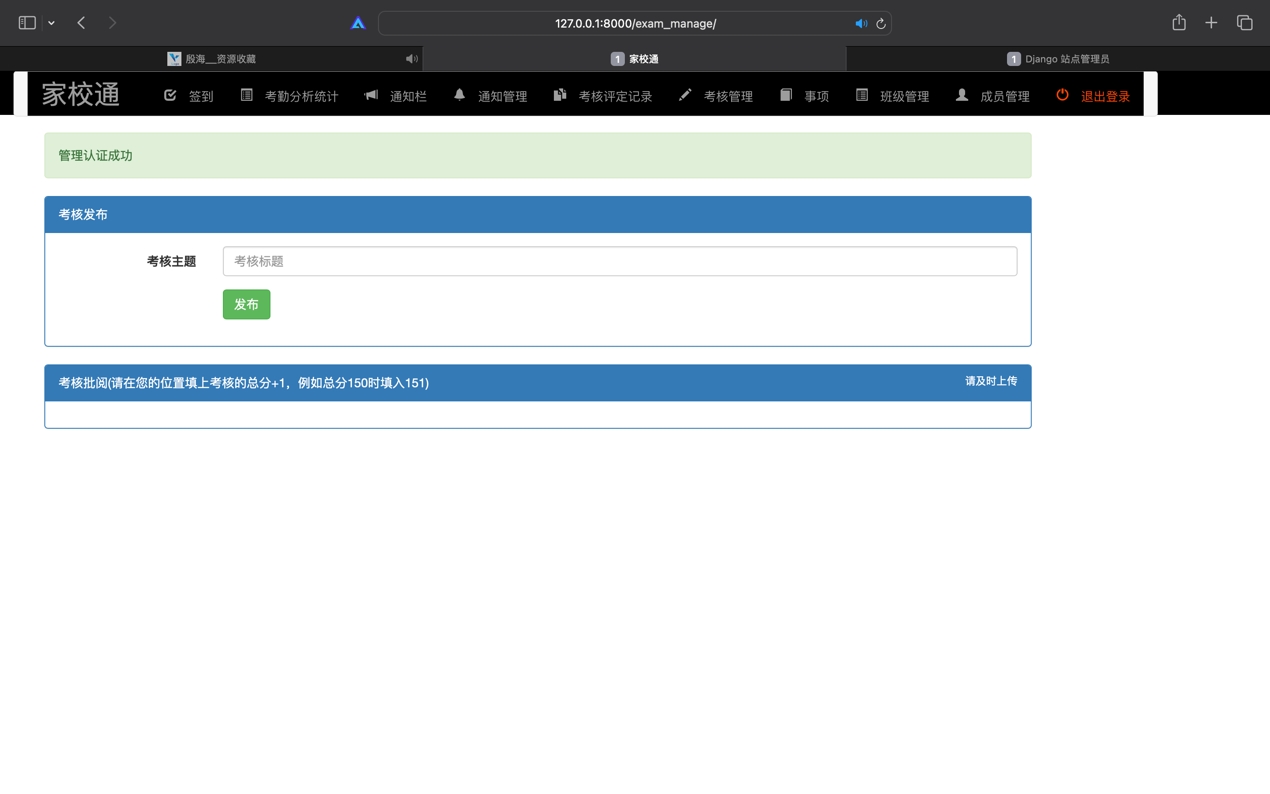


图5-8 发布考核

图5-9 考核打分和写评语



图5-10 学生无权限操作

后端用户考核管理View核心代码如下：

# 考核管理

@is\_login

def exam\_manage(request):

(flag, user) = check\_cookie(request)

if user.user\_type.caption == '班主任':

if request.method == 'POST':

title = request.POST.get('title')

if title:

ExamContent.objects.create(title=title)

else:

count = UserInfo.objects.filter(cid=user.cid).count()

content\_id = request.POST.get('exam\_id')

for i in range(count):

point = request.POST.get('point{}'.format(i))

stuID = request.POST.get('stu{}'.format(i))

detail = request.POST.get('detail{}'.format(i))

Exam.objects.create(point=point, content\_id=content\_id, user\_id=stuID, detail=detail)

ExamContent.objects.filter(id=content\_id).update(state=True)

check\_list = ExamContent.objects.filter(state=False).order\_by('-id')

user\_list = UserInfo.objects.filter(cid=user.cid).order\_by('studentNum')

return render(request, 'exam/exam\_manage.html', locals())

else:

return render(request, 'denied.html')

5.9 考核评定记录

用户所有考核名称如图5-11所示，当用户查看时，这段代码根据请求的查询参数 exam\_id 来查询相关的考试内容和考试结果如图5-12，并根据最高分（班主任设置默认为满分）计算每个结果的分数比例。然后，将查询到的数据和定义好的分数范围列表传递给模板进行渲染。

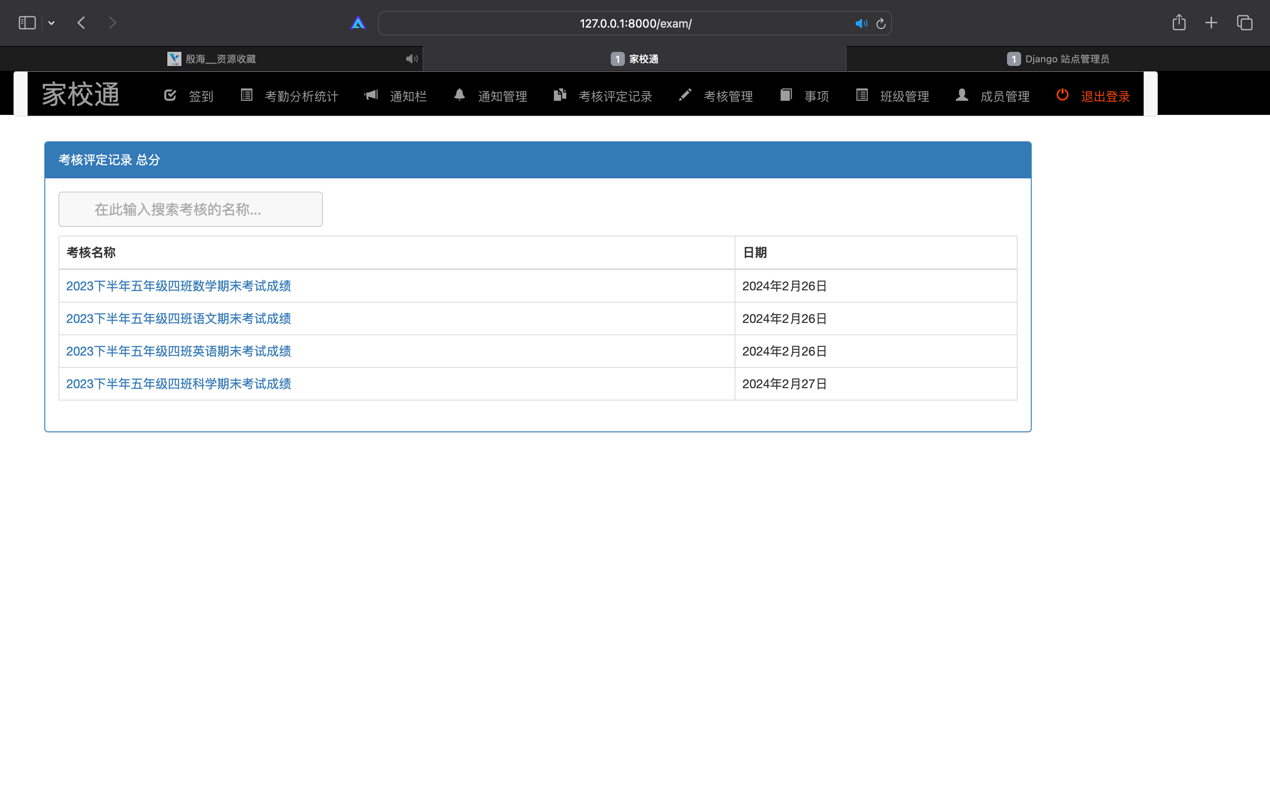


图5-11 所有考核主题

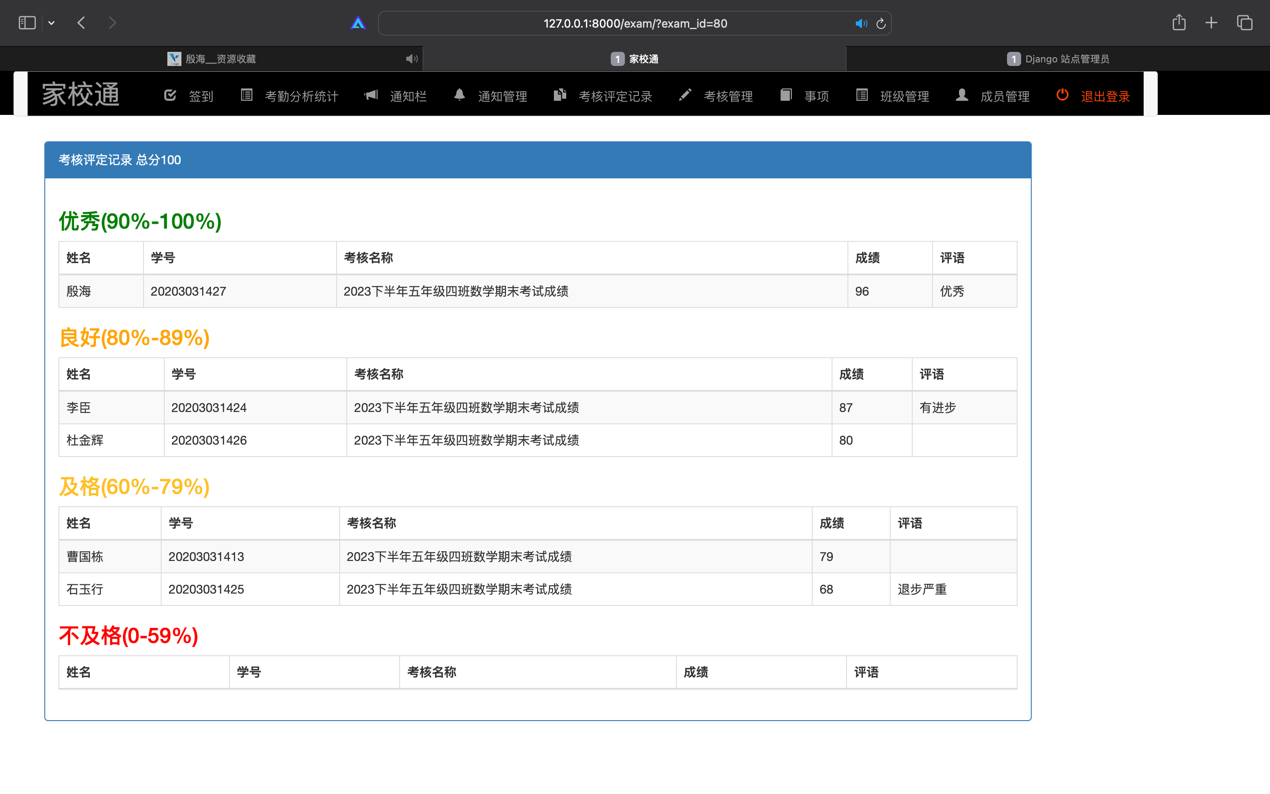


图5-12 考核分数详情

后端考核记录View核心代码如下：

@is\_login

def exam(request):

exam\_list = ExamContent.objects.all()

exam\_id = request.GET.get('exam\_id')

if exam\_id:

user\_list = Exam.objects.filter(content\_id=exam\_id).all()

total\_point = max(item.point for item in user\_list)

ratio\_list = [item.point / total\_point for item in user\_list]

user\_list\_with\_ratio = list(zip(user\_list, ratio\_list))

grades = [

{"name": "优秀(90%-100%)", "color": "#008000", "min\_point": 0.8999, "max\_point": 1.00},

{"name": "良好(80%-89%)", "color": "#FFA500", "min\_point": 0.7999, "max\_point": 0.8999},

{"name": "及格(60%-79%)", "color": "#FFC125", "min\_point": 0.6999, "max\_point": 0.7999},

{"name": "不及格(0-59%)", "color": "#FF0000", "min\_point": 0.00001, "max\_point": 0.5999}

]

return render(request, 'exam/exam.html', locals())

5.10 事项

用户事项界面如图5-13所示，当用户查看或发布事项时，获取所有的事项记录， 如果请求方法是POST，表示用户提交了事项表单，执行以下操作： 获取开始时间和结束时间，并打印开始时间。 根据开始时间和结束时间计算事项天数。 获取事项说明。

创建一个新的事项记录，并将开始时间、结束时间、用户和说明保存到数据库中。最后，渲染leave.html模板，并将请求对象和局部变量传递给模板进行渲染。

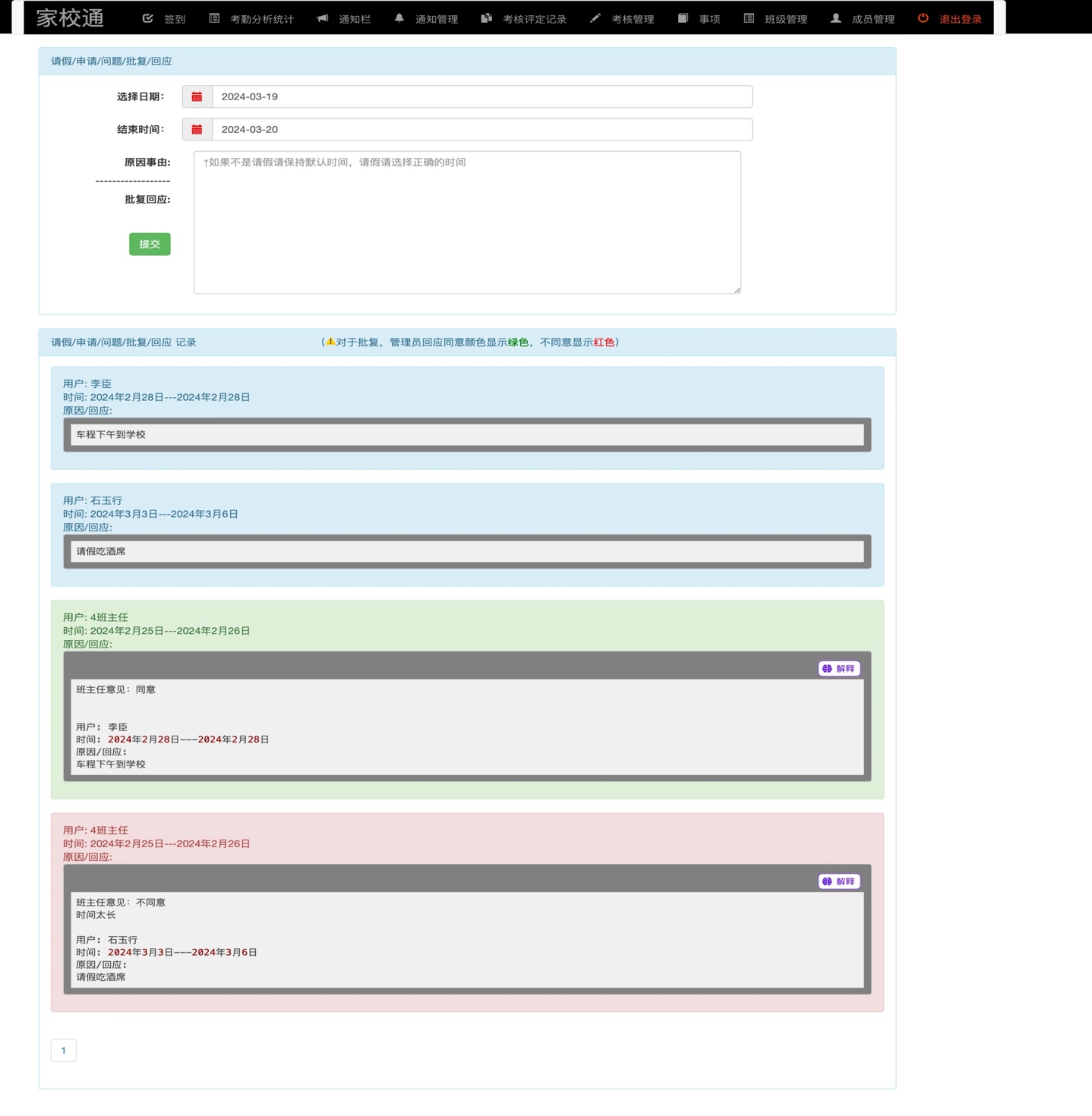


图5-13 事项

后端事项View核心代码如下：

@is\_login

def leave(request):

(flag, user) = check\_cookie(request)

user\_class = user.cid

leave\_list = Leave.objects.filter(user\_\_cid=user\_class)

if request.method == 'POST':

starttime = request.POST.get('starttime')

endtime = request.POST.get('endtime')

print(starttime)

a = int(datetime.datetime.strptime(starttime, '%Y-%m-%d').day - datetime.datetime.strptime(endtime,'%Y-%m-%d').day) +1

explain = request.POST.get('explain')

Leave.objects.create(start\_time=starttime, end\_time=endtime, user=user, explain=explain)

return render(request, 'attendance/leave.html', locals())

5.11 班级管理

班级界面如图5-14所示，当用户查看班级时，后端根据cookie验证的结果和用户的权限，渲染不同的模板页面。如果验证通过且用户权限为班主任，则显示所有的ClassInfo对象；如果验证通过但用户权限不为班主任，则显示一个拒绝访问的页面如图5-15；如果验证未通过，则显示登录页面。

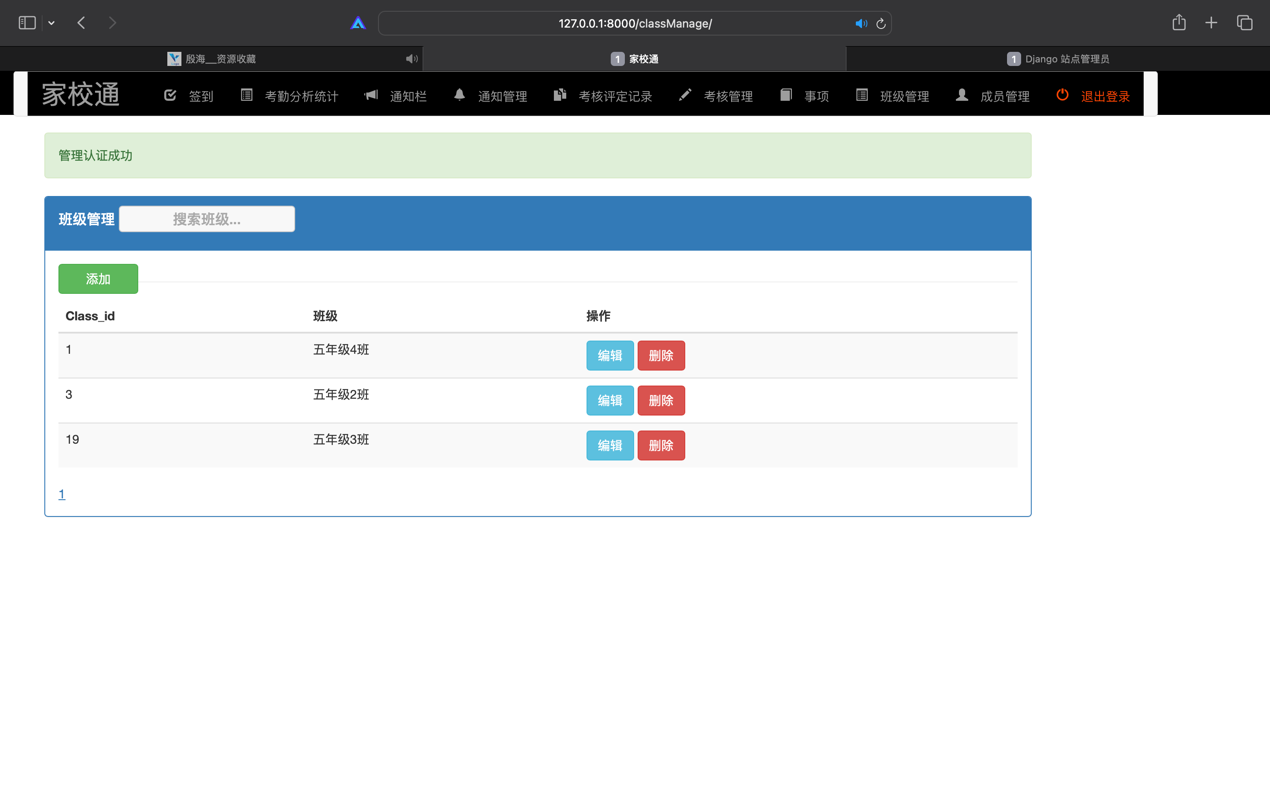


图5-14 所有班级信息



图5-15 非班主任用户类型无权限操作本模块

后端查看班级View核心代码如下：

def classManage(request):

(flag, rank) = check\_cookie(request)

print('flag', flag)

if flag:

if rank.user\_type.caption == '班主任':

class\_list = ClassInfo.objects.all()

return render(request, 'class/classManage.html', {'class\_list': class\_list})

else:

return render(request, 'denied.html')

else:

return render(request, 'account/page-login.html', {'error\_msg': ''})

当用户编辑班级时，根据请求中的参数add\_class\_name，判断是否已经存在同名班级。如果存在，则返回'班级已存在'；如果不存在，则将该班级添加到数据库中，并返回'添加成功'。

后端编辑班级View核心代码如下：

@csrf\_exempt

def edit\_class(request):

(flag, rank) = check\_cookie(request)

print('flag', flag)

if flag:

if rank.user\_type.caption == '班主任':

if request.method == 'POST':

pre\_edit\_id = request.POST.get('edit\_id')

class\_name = request.POST.get('edit\_class\_name')

temp\_flag = ClassInfo.objects.filter(name=class\_name)

print('pre\_edit\_id1', pre\_edit\_id)

pre\_obj = ClassInfo.objects.get(id=pre\_edit\_id)

if not temp\_flag and class\_name:

pre\_obj.name = class\_name

pre\_obj.save()

return HttpResponse('班级修改成功')

class\_list = ClassInfo.objects.all()

return render(request, 'class/classManage.html', {'class\_list': class\_list})

# return HttpResponse('编辑班级')

else:

return render(request, 'denied.html')

else:

return render(request, 'account/page-login.html', {'error\_msg': ''})

当用户添加班级时，后端根据cookie验证的结果和用户的权限，渲染不同的模板页面。如果验证通过且用户权限为班主任，则显示所有的ClassInfo对象；如果验证通过但用户权限不为班主任，则显示一个拒绝访问的页面；如果验证未通过，则显示登录页面。

后端添加班级View核心代码如下：

@csrf\_exempt

def add\_class(request):

if request.method == 'POST':

add\_class\_name = request.POST.get('add\_class\_name')

flag = ClassInfo.objects.filter(name=add\_class\_name)

if flag:

return HttpResponse('班级已存在')

pass

# print('已有数据，不处理')

else:

if add\_class\_name:

ClassInfo.objects.create(name=add\_class\_name).save()

return HttpResponse('添加成功')

当用户删除班级时，后端处理班级信息的删除请求，只有班主任身份的用户才有权限执行删除操作。

后端删除班级View核心代码如下：

def delete\_class(request):

(flag, rank) = check\_cookie(request)

print('flag', flag)

if flag:

if rank.user\_type.caption == '班主任':

# class\_list=ClassInfo.objects.all()

delete\_id = request.GET.get('delete\_id')

ClassInfo.objects.filter(id=delete\_id).delete()

return redirect('/classManage/')

else:

return render(request, 'denied.html')

else:

return render(request, 'account/page-login.html', {'error\_msg': ''})

5.12 成员管理

成员管理界面如图5-16，图5-17所示，当用户删除成员时，后端处理成员信息的删除请求，只有班主任身份的用户才有权限执行删除操作。当用户查看成员时，前端系统调用Ajax向服务器发送登录请求，并将学号、用户名、密码、手机号码、Email发送到后端系统。

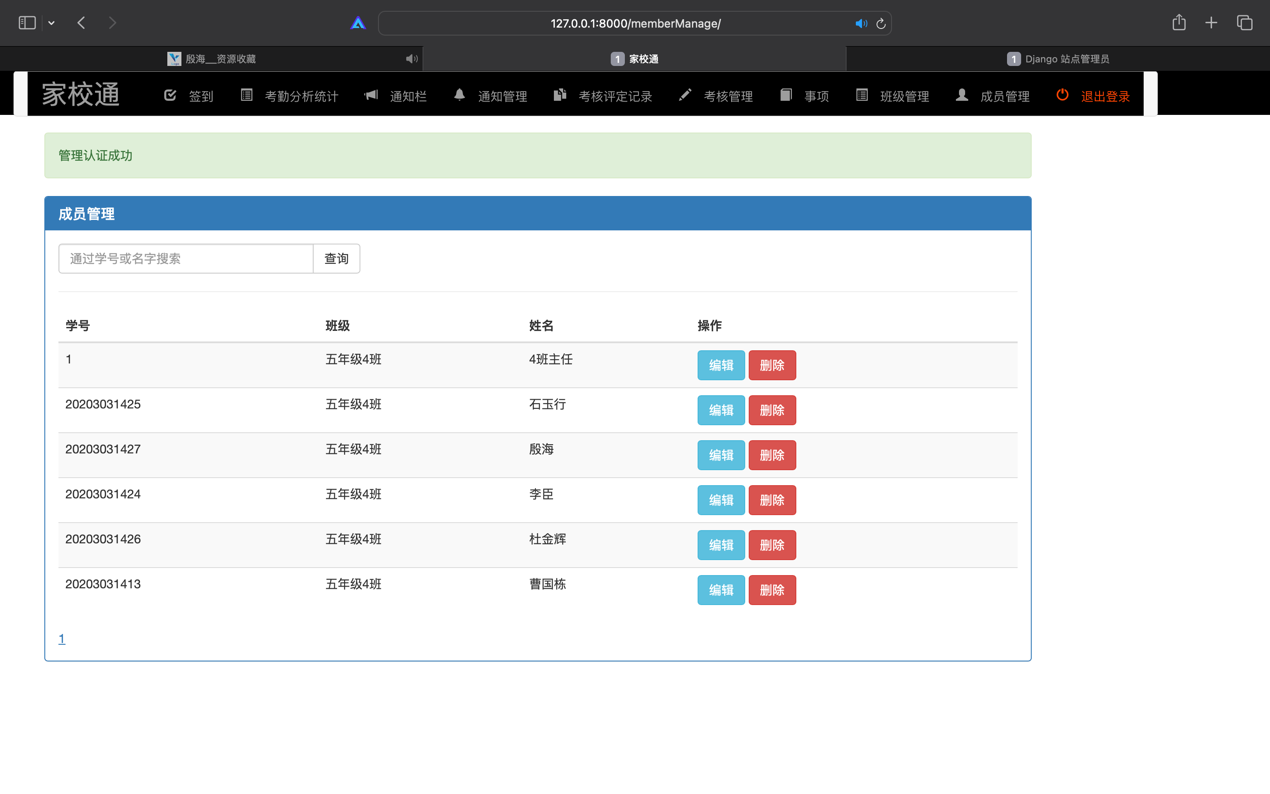


图5-16 成员信息管理

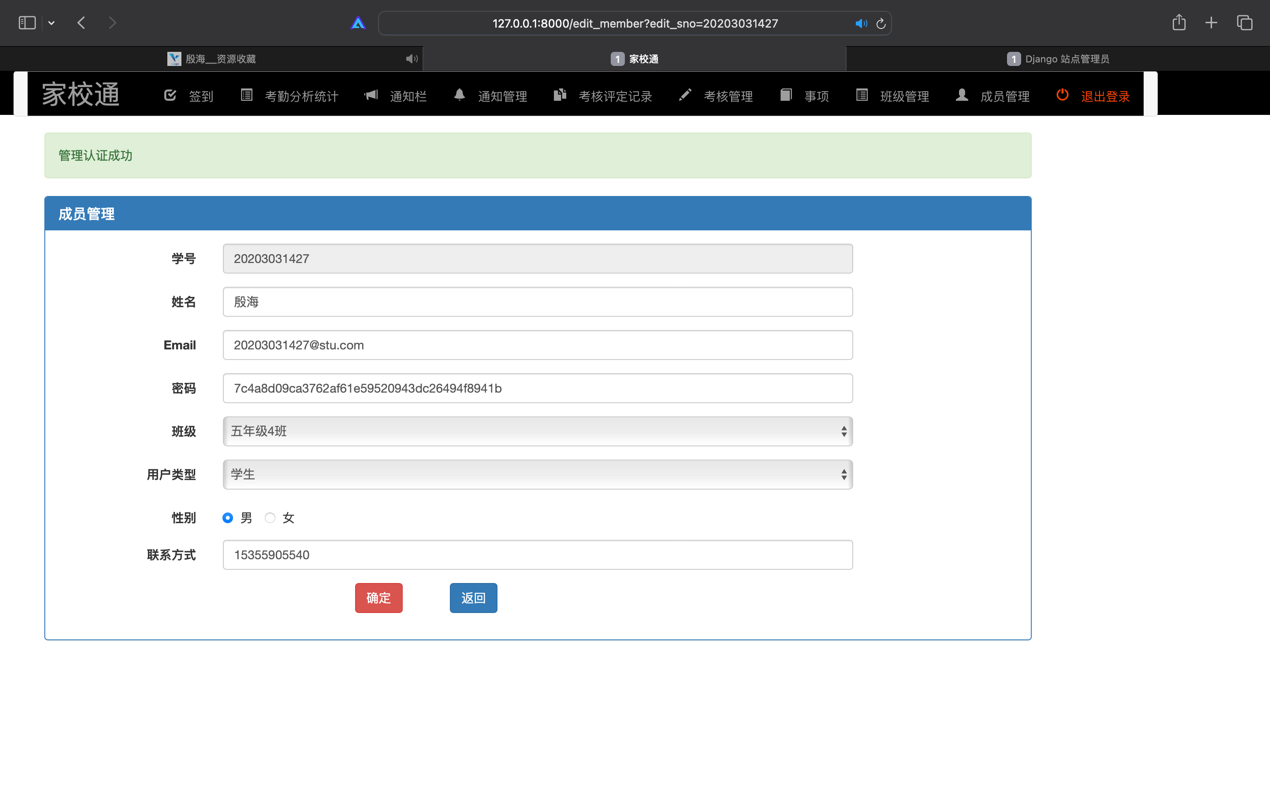


图5-17 成员编辑



图5-18非班主任用户类型无权限操作本模块

后端查看成员View核心代码如下：

def member\_manage(request):

(flag, rank) = check\_cookie(request)

(flag, user) = check\_cookie(request)

if flag:

if rank.user\_type.caption == '班主任':

member\_list = UserInfo.objects.filter(cid=user.cid)

return render(request, 'member/member\_manage.html',{'member\_list': member\_list})

else:

return render(request, 'denied.html')

else:

return render(request, 'account/page-login.html', {'error\_msg': ''})

后端删除成员View核心代码如下：

def delete\_member(request):

(flag, rank) = check\_cookie(request)

if flag:

if rank.user\_type.caption == '班主任':

delete\_sno = request.GET.get('delete\_sno')

UserInfo.objects.get(studentNum=delete\_sno).delete()

member\_list = UserInfo.objects.all()

return render(request, 'member/member\_manage.html', {'member\_list': member\_list})

else:

return render(request, 'denied.html')

else:

return render(request, 'account/page-login.html', {'error\_msg': ''})

当用户编辑成员时，首先，通过检查用户的cookie来验证用户是否登录。

如果验证通过，则进一步判断用户的身份是否为班主任。

如果是班主任，并且请求方法是POST，则获取表单中的学生编号、用户名、电子邮件、性别、班级、用户类型和电话等信息。

根据学生编号在数据库中查找相应的用户对象，并将获取到的信息更新到该对象中。获取所有用户信息的列表，并将其传递给名为"member\_list"的变量。重定向到"/memberManage/"页面，并将"member\_list"作为参数传递。

如果请求方法不是POST，则获取名为"edit\_sno"的参数，该参数表示要编辑的学生编号。获取所有用户类型和班级的列表。

根据学生编号在数据库中查找相应的用户对象，并将其赋值给变量"edit\_stu\_obj"。渲染名为"edit\_member.html"的模板，并传递当前的请求对象和局部变量。如果用户的身份不是班主任，则渲染名为"member\_manage\_denied.html"的模板。如果用户的cookie验证未通过，则渲染名为"page-login.html"的模板，并传递一个空的错误消息。

后端编辑成员View核心代码如下：

# 编辑成员

def edit\_member(request):

(flag, rank) = check\_cookie(request)

if flag:

if rank.user\_type.caption == '班主任':

if request.method == 'POST':

student\_num = request.POST.get('student\_num')

username = request.POST.get('username')

email = request.POST.get('email')

gender = int(request.POST.get('gender'))

cls = ClassInfo.objects.get(name=request.POST.get('cls'))

usertype = UserType.objects.get(caption=request.POST.get('user\_type'))

phone = request.POST.get('phone')

password = request.POST.get('password')

edit\_obj = UserInfo.objects.filter(studentNum=student\_num)

passq = edit\_obj.values('password').first()

if password==passq['password']:

edit\_obj.update(studentNum=student\_num, username=username,email=email, cid=cls, user\_type=usertype, gender=gender, phone=phone,)

else:

a = hashlib.sha1(password.encode("utf-8")).hexdigest()

edit\_obj.update(studentNum=student\_num, username=username,email=email, password=a, cid=cls,user\_type=usertype,gender=gender, phone=phone,)

member\_list = UserInfo.objects.all()

return redirect('/memberManage/', {'member\_list':member\_list})

else:

edit\_member\_id = request.GET.get('edit\_sno')

stu\_type\_list = UserType.objects.all()

cls\_list = ClassInfo.objects.all()

edit\_stu\_obj = UserInfo.objects.get(studentNum=edit\_member\_id)

return render(request, 'member/edit\_member.html',locals())

else:

return render(request, 'denied.html')

else:

return render(request, 'account/page-login.html', {'error\_msg': ''})

5.13 退出登录

用户退出登录如图5-19所示，当用户退出登录，注销用户并删除与用户相关的两个cookie。最后，将用户重定向到登录页面。

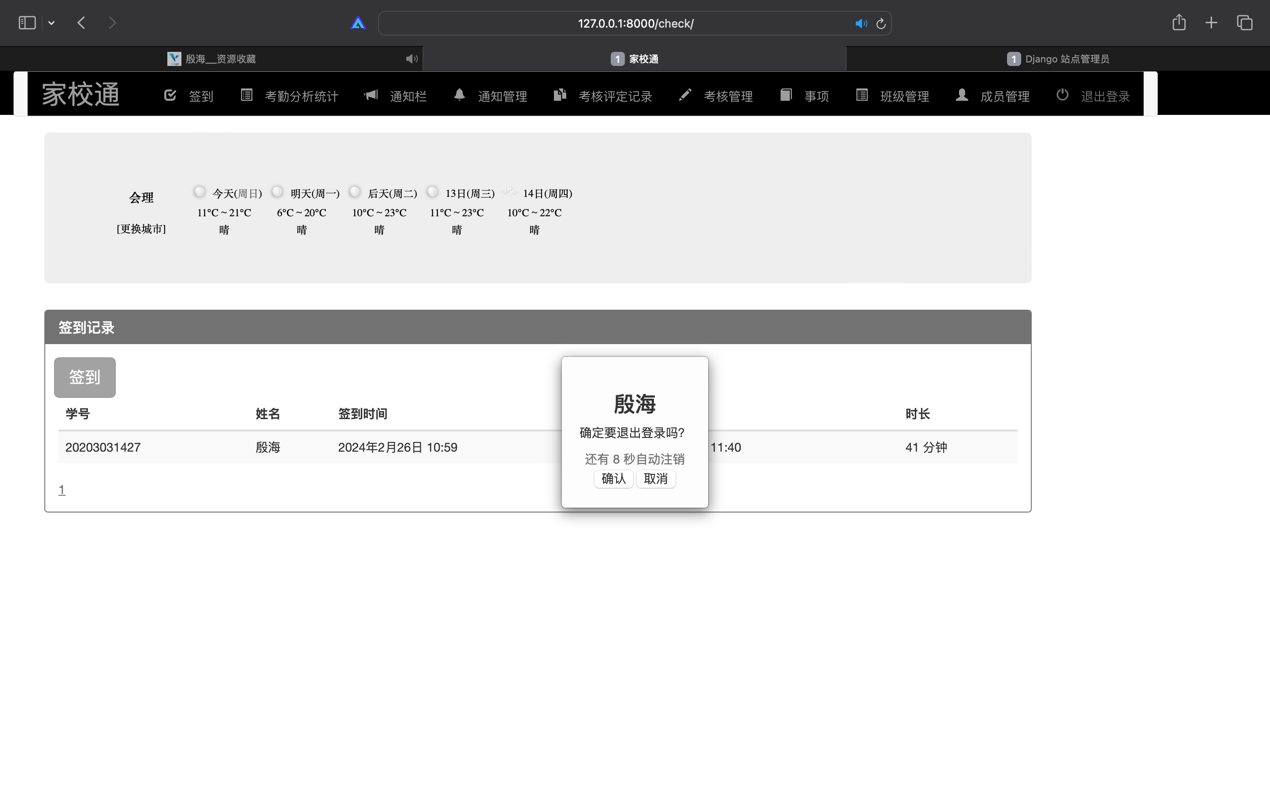


图5-19 退出登录

后端用户退出登录View核心代码如下：

def logout(request):

req = redirect('/login/')

req.delete\_cookie('asdf')

req.delete\_cookie('qwer')

return req

5.14 Django超级管理员后台

Django超级管理员后台登录界面如图5-20所示， Django超级管理员后台管理界面如图5-21所示

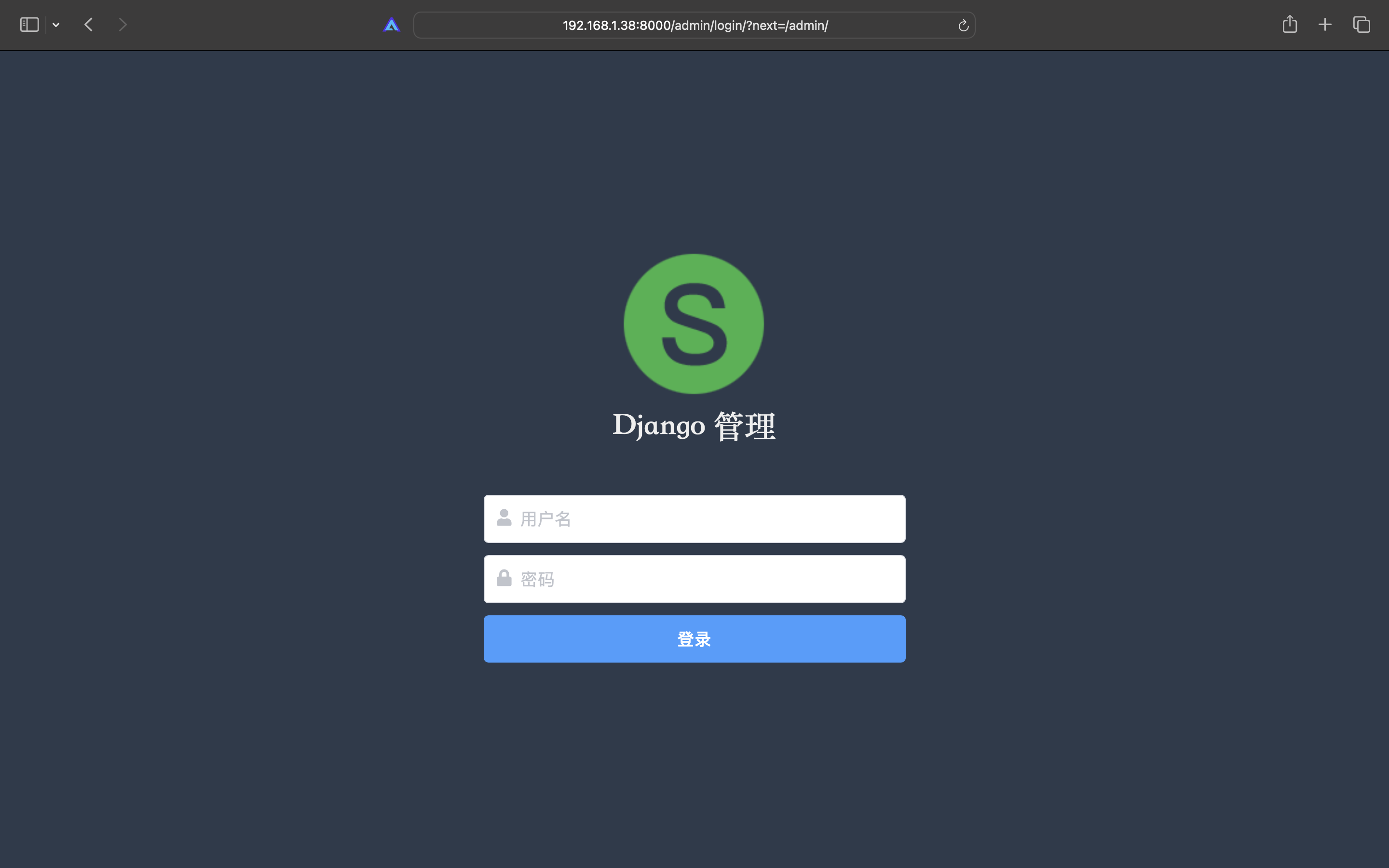


图5-20 Django超级管理员后台登录

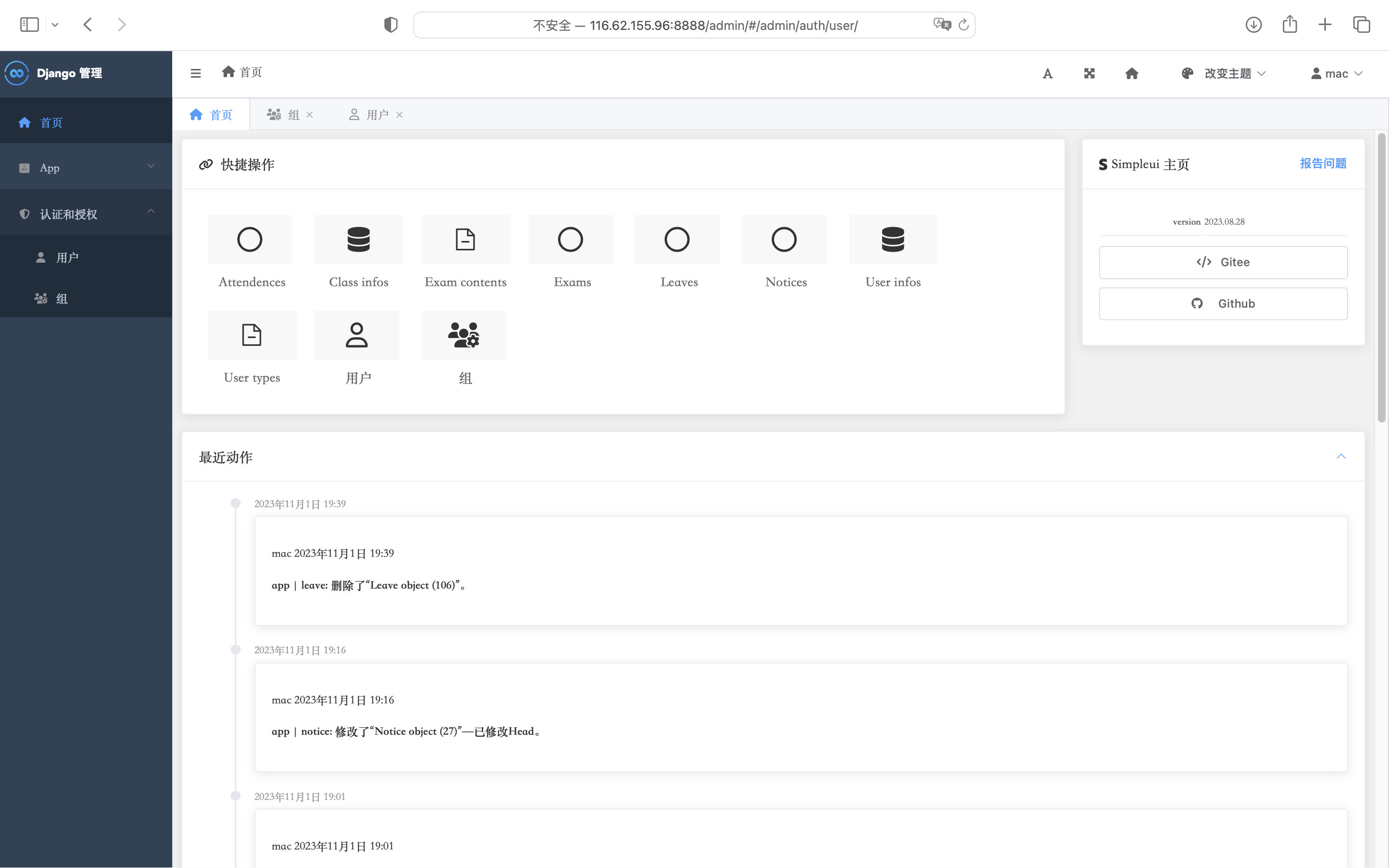


图5-21 Django超级管理员后台主界面

在使用之前，您需要创建一个超级用户。您可以通过运行命令`python manage.py createsuperuser`来创建超级管理员。超级管理员具有最高权限，可以管理其他用户、数据、组等[11]。创建超级用户后，您将能够使用其凭据登录系统，并拥有更高级别的权限来执行管理任务。这样可以确保系统的安全性和灵活性，使您能够更好地管理和控制系统中的用户和数据。这是Django框架自带的平台，需要配置注册才能使用。首先，导入了需要注册的模型，包括UserInfo、UserType、ClassInfo、Attendence、Notice、Leave、Exam和ExamContent。定义了一系列Admin类，每个类都指定了要在后台显示的字段。接下来，通过调用admin.site.register()方法，将每个模型和对应的Admin类进行注册。

后端配置注册admin核心代码如下：

from django.contrib import admin

from .models import UserInfo, UserType, ClassInfo, Attendence, Notice, Leave, Exam, ExamContent

# Register your models here.

class UserInfoAdmin(admin.ModelAdmin):

list\_display = ['studentNum', 'username', 'cid', 'password', 'gender', 'phone', 'email',]

class UserTypeAdmin(admin.ModelAdmin):

list\_display = ['id', 'caption']

class ClassInfoAdmin(admin.ModelAdmin):

list\_display = ['id', 'name']

class MajorInfoAdmin(admin.ModelAdmin):

list\_display = ['id', 'name', ]

class AttendenceAdmin(admin.ModelAdmin):

list\_display = ['id', 'stu', 'date', 'start\_time', 'end\_time','is\_leave', 'duration', 'detail']

class NoticeAdmin(admin.ModelAdmin):

list\_display = ['id', 'author', 'head', 'content', 'level']

class LeaveAdmin(admin.ModelAdmin):

list\_display = ['id', 'start\_time', 'end\_time', 'explain']

class ExamAdmin(admin.ModelAdmin):

list\_display = ['id', 'user', 'content', 'point', 'detail']

class ExamContentAdmin(admin.ModelAdmin):

list\_display = ['id', 'title', 'date', 'state']

# 接下来，通过调用admin.site.register()

# 方法，将每个模型和对应的Admin类进行注册。

admin.site.register(UserType, UserTypeAdmin)

admin.site.register(UserInfo, UserInfoAdmin)

admin.site.register(ClassInfo, ClassInfoAdmin)

admin.site.register(Attendence, AttendenceAdmin)

admin.site.register(Notice, NoticeAdmin)

admin.site.register(Leave, LeaveAdmin)

admin.site.register(ExamContent, ExamContentAdmin)

admin.site.register(Exam, ExamAdmin)

6 系统测试

6.1 测试方法

采用单元测试，是对系统设计中的各个功能模块进行独立测试的过程。它的目标是验证每个模块是否符合需求标准，并找出不符合用户需求的功能点。通过编写测试用例，测试人员可以模拟各种输入情况，包括正常情况、边界情况和异常情况，以确保每个模块在各种情况下都能正确运行。通过进行单元测试，可以及早发现和修复功能模块中的问题和缺陷，提高代码质量，并确保系统的各个功能模块能够按照用户需求的标准进行工作。

6.2 测试内容

下列是单元测试内容：

验证登录功能：测试用户登录功能的准确性和安全性，确保用户能够成功登录系统，同时防止未经授权的访问。

表6-1 登录界面用例测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试单元 | 测试页面 | 包含功能 | 输入 | 网页响应 | 是否达标 |
| 用户注册 | register.html | 用户注册 | 正确格式的学号、用户名、密码、确认密码、手机号码Email | 注册成功，跳转到登录界面 | 是 |
| 错误格式的学号（为空或不是11位） | 提示请输入11位学号或学号为空 | 是 |
| 错误格式的用户名（为空） | 提示用户名为空 | 是 |
| 错误格式的密码（为空或密码位数不满足6-30） | 提示密码为空或密码位数不满足6-30 | 是 |
| 错误格式的确认密码（两次密码不一致） | 提示两次密码不一致 | 是 |
| 错误格式的手机号（为空或不足11位或手机号不是1开头，第二位不  是3，5，6，7，8，9） | 提示输入11为手机号或手机号格式不正确 | 是 |
| 错误格式的Email（为空或邮箱地址不是xx@xx.xx） | 提示邮箱为空或格式不正确 | 是 |

表6-2 登录界面用例测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试单元 | 测试页面 | 包含功能 | 输入 | 网页响应 | 是否达标 |
| 用户登录 | page-login.html | 用户登录 | 正确的密码和用户名 | 成功登录跳转到系统界面 | 是 |
| 正确的密码和用户名 | 提示用户名或密码错误请重新输入 | 是 |

表6-3 签到用例测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试单元 | 测试页面 | 包含功能 | 点击 | 网页响应 | 是否达标 |
| 签到 | check.html | 签到，显示用户签到、签签退时间，记录时长 | 第一次点击 | 提示操作成功，显示用户信息和签到时间 | 是 |
| 第二次点击 | 提示操作成功，显示用户信息和签退时间，记录时长 | 是 |

表6-4 考勤分析统计模块用例测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试单元 | 测试页面 | 包含功能 | 搜索 | 网页响应 | 是否达标 |
| 考勤分析统计 | total.html | 显示所有用户的信息，考勤总时长，柱  状图显示每个人总时长 | 模糊搜索姓名 | 显示模糊搜索的所有学生 | 是 |
| 模糊搜索学号 | 显示模糊搜索的所有学生 | 是 |

表6-5 通知栏模块用例测试

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试单元 | 测试页面 | 包含功能 | 网页响应 | 是否达标 |
| 通知栏 | notice.html | 显示所有通知，包括时间，标题，内容，发布人，不同级别的通知显示成不同的颜色 | 全部正确按照功能显示 | 是 |

表6-6 通知管理模块用例测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试单元 | 测试页面 | 包含功能 | 输入 | 网页响应 | 是否达标 |
| 通知管理 | notice\_manage.html | 发布通知，标题、内容、级别 | 内容为空 | 提示输入的内容为空 | 是 |
| 正确输入 | 发布成功，并显示在通知栏里 | 是 |
| denied.html | 用户类型为家庭用户无权限 | 无 | 提示无权限操作 | 是 |

表6-7 考核管理模块用例测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试单元 | 测试页面 | 包含功能 | 输入 | 网页响应 | 是否达标 |
| 考核管理 | exam\_manage.html | 发布考核内容 | 内容空 | 提示内容空 | 是 |
| 有内容 | 发布成功，并在考核批阅显示出来 | 是 |
| 考核批阅  打分  写评语 | 分数格式正确 | 完成批阅，批阅成功，参加将在考核评定记录显示 | 是 |
| 格式不满足常规分数百分制或为空 | 提示内容为空分数分数的值应该是一个十进制的数字 | 是 |
| denied.html | 阻止班主任用户类型操作 | 无 | 提示无权限操作 | 是 |

表6-8 考核评定记录模块用例测试

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试单元 | 测试页面 | 包含功能 | 网页响应 | 是否达标 |
| 考核评定记录 | exam.html | 查看所有考核类型，成绩 | 显示所有记录 | 是 |

表6-9 事项模块用例测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试单元 | 测试页面 | 包含功能 | 输入 | 网页响应 | 是否达标 |
| 事项 | leave.html | 发布事项，包括时间、内容，显示所有事项 | 内容为空 | 提示空 | 是 |
| 有内容发布 | 显示在下方的记录中，有同意字眼的显示绿色，不同意红色 | 是 |

表6-10 班级管理模块用例测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试单元 | 测试页面 | 包含功能 | 操作 | 网页响应 | 是否达标 |
| 班级管理 | classManage.html | 增加、删除、编辑 | 添加 | 添加成功 | 是 |
| 编辑 | 编辑成功 | 是 |
| 删除 | 删除成功 | 是 |
| denied.html | 阻止非班主任用户类型操作 | 提示 | 提示无权限操作 | 是 |

表6-11 成员管理模块用例测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试单元 | 测试页面 | 包含功能 | 操作 | 网页响应 | 是否达标 |
| 成员管理 | member\_manage.html | 搜索、删除、 | 按学号或姓名模糊搜索 | 显示所有符合搜索条件的成员 | 是 |
| 删除 | 删除成功 | 是 |
| edit\_member.html | 编辑 | 编辑 | 编辑成功 | 是 |
| denied.html | 阻止非班主任用户类型操作 | 提示 | 提示无权限操作 | 是 |

：

表6-12 考勤分析统计模块用例测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试单元 | 测试页面 | 包含功能 | 点击 | 网页响应 | 是否达标 |
| 注销登录 | base.html | 注销登录，提示确认或取消，10秒内无操作自动确认 | 注销 | 提示确认或取消，网页自动确认10秒倒计时 | 是 |
| 确认 | 退出登录，回到登录界面 | 是 |
| 倒计时10秒未点击 | 自动退出，回到登录界面 | 是 |
| 取消 | 返回提示框消失，取消退出 | 是 |

6.3 系统的测试结果

通过对家校通的单元测试，发现无任何功能性问题，以下是测试结果总结：

1. 注册功能：注册功能正常，用户可以成功注册新账号。

2. 登录功能：登录功能正常，用户可以使用正确的凭据成功登录系统。

3. 签到功能：签到功能正常，用户可以进行签到操作，并记录签到状态。

4. 考勤统计功能：考勤统计功能正常，系统能够准确统计用户的考勤情况。

5. 通知栏功能：通知栏功能正常，用户可以收到系统发布的通知。

6. 通知管理功能：通知管理功能正常，班主任可以发布通知。

7. 考核管理功能：考核管理功能正常，班主任可以创建和管理考核内容。

8. 考核评定记录功能：考核评定记录功能正常，系统能够显示用户的考核评定结果。

9. 事项功能：事项功能正常，用户可以发布事项。

10. 班级管理功能：班级管理功能正常，班主任可以管理班级信息。

11. 成员管理功能：成员管理功能正常，班主任可以管理班级成员的信息。

综上所述，家校通的各项功能经过测试均正常运行。

# 结论

本论文旨在设计和实现一个基于Django的家校通，通过使用Django框架作为后端，用Boostrap框架作前端，Sqlite作为数据库，为我成功地构建了一个功能丰富、易于使用、美观的家校通系统，为家庭用户提供了丰富的功能，为班主任提供了各种交流和管理工具。

通过本研究，我验证了使用Django框架构建家校通的可行性和优势。Django框架提供了一种快速、高效的方式来构建Web应用程序，并具有强大的安全性和可扩展性，非常适合我的家校通的需求。

在设计和实现过程中，我充分利用了Django框架提供的功能和特性，如模型、视图、模板和表单等。我采用了良好的软件工程实践，如模块化设计、代码重用和测试驱动开发，确保平台的可靠性和可维护性。我也充分利用了Bootstrap框架提供的功能和特性，如响应式设计、组件库和网格系统等。我采用了良好的前端开发实践，如组件化设计、样式重用和响应式布局，确保页面的一致性和易维护性。在设计和实现中，我充分发挥了SQLite数据库的轻量级、零配置和跨平台等特性。我遵循了规范化的数据模型设计，通过使用约束，我确保了数据的完整性和一致性，从而保证了系统的稳定性和可靠性

通过平台的功能和设计，我提供了家庭用户注册和登录；签到、签退方便为班主任掌握学生的上学、放学到家时间，初步了解孩子的离校安全情况。家庭用户还可以查看班级发布的通知，参与事项讨论，查看家庭用户中学生的成绩等功能。为为班主任班主任考勤统计，柱状图的展示出学生的出勤情况。班主任可以通知发布，参与事项讨论与同学交流和管理。我为班主任考核任务发布，比如考试和考试分数的登记、管理班级和成员管理等一系列功能。这些功能旨在提高班级内的信息共享、方便交流、协作效率和学习管理，促进家庭用户的参与和成长，方便班主任用户的管理、及时发布消息，为所有用户提供所有历史成绩的查看等等功能

家校通的实际应用具有重要意义。它可以提供一个集中的平台。通过平台的功能，家庭用户可以更方便地参与讨论、和查看成绩，班主任可以更好地管理家庭用户的学习任务和评估家庭用户的表现。这将有助于提高班级的整体学习效果和家庭用户的学习动力。我对前端搜索进行了优化，数据搜索不再经过后端调用数据再显示出搜索的内容，而是根据前端技术直接在显示的元素查找，提高了响应速度，减小对服务器的负担。然而，本研究还存在一些局限性，可以通过用户调研和反馈收集更多的需求和改进建议，不断优化平台的功能和用户体验。家校通目前数据库设计只能用于一个学校，未来将考虑支持多个学校。家校通未来还将支持成绩由 Excel 文件导入，手机短信通知。基于Django的家校通为家庭用户和班主任提供了一个强大的工具，促进了家庭和班级内的信息共享、协作效率和学习管理。它具有广泛的应用前景和发展潜力，有助于提高班级的整体学习效果和家庭用户的学习动力。

# 致 谢

在完成这篇家校通论文的过程中，我想向许多人表达我最深切的感激之情。他们的支持、鼓励和帮助是我能够顺利完成这项研究工作的重要动力和资源。

首先，我要衷心感谢我的指导老师。他在整个研究过程中给予了我无私的指导和悉心的关怀。他的专业知识、丰富经验和深入见解对我的研究起到了至关重要的推动作用。他耐心地解答我的问题，引导我思考问题的深层次，帮助我克服困难和挑战。在每次讨论和指导中，他们给予我宝贵的建议和指引，使我能够不断提高和完善我的研究。没有他们的指导和支持，我无法完成这篇论文。

其次，我要感谢我的同学们，特别是那些与我一同参与研究的同学们。我共同努力、协作和交流，相互启发和督促，共同追求卓越。在研究过程中，我一起面对挑战，共同解决问题，共享知识和经验。他们的智慧、勇气和友谊使我感到无比幸运和骄傲。我互相支持和鼓励，共同成长，形成了一种深厚的团队精神。

此外，我要感谢那些愿意接受我的访谈和调研的学生们和班主任们。他们的宝贵意见和反馈为我的研究增添了重要的实践和现实的观点。他们的参与和合作使我的研究更具有可行性和适用性。他们的真诚和耐心使我能够更好地理解他们的需求和期望，从而更好地设计和改进家校通。我要向他们表达我最诚挚的感谢和敬意。

最后，我要感谢我的家人和朋友们。他们一直以来对我的支持和鼓励是我前进的动力和勇气。他们无私地给予我关爱和理解，为我提供了宝贵的心灵支持。他们的陪伴和鼓励使我在研究的道路上永远不感孤单和迷茫。我要向他们表达我最深深的感激和爱意。

在这里，我要向所有支持和帮助过我的人们表示衷心的谢意。没有你们的支持和鼓励，我无法完成这篇论文。你们的帮助和支持是我研究工作的动力和源泉。我将永远珍视你们的帮助，并将继续努力不懈，为教育事业做出更大的贡献。

最后，我要再次向所有支持我的人们致以最诚挚的谢意！

# 参考文献

1. 梁云真，李俊“家校通”相关研究述评[J].软件导刊(教育技术),2011,06:34-36
2. 邹强.国外家校合作问题研究及其启示[J].教学与管理,2011(10):86-88.
3. 汪时冲,张鸽.大学生手机日常使用情况调查与分析——以某市属高校为例[J].教育现代化,2018,5(53):324-327.DOI:10.16541/j.cnki.2095-8420.2018.53.116.
4. 赵丙秀.Bootstrap5的定制化研究[J].电脑编程技巧与维护,2023,(09).
5. 夏小翔.基于Echarts学生成绩管理系统设计[J].鄂州大学学报,2023,30(05).
6. 郭鹤楠.基于Django和Python技术的网站设计与实现[J].数字通信世界,2023,(06).
7. Xiya Y ,Xianhe L ,Changping W , et al.Design and Deployment of Django-based Housing Information Management System[J].Journal of Physics: Conference Series,2023,2425(1).
8. Junlong S .SQLite Encryption Method for Embedded Databases Based on Chaos Algorithm[J].Journal of Applied Mathematics,2023,2023.
9. 蹇常林.ORM在Django操作数据库中的应用[J].技术与市场,2020,27(01).
10. 梁国达.基于VMware+Navicat实现空管自动化的基础数据迁移[J].科学技术创新,2020,(08).
11. 邱红丽,张舒雅.基于Django框架的web项目开发研究[J].科学技术创新,2021,(27).
12. Anaconda Inc.; Anaconda and JetBrains Join Forces to Launch 'PyCharm for Anaconda'[J].Journal of Engineering,2019.
13. 邱睿.基于微信小程序的家校通系统开发与实现[D].上海师范大学,2019.