

重师计信在线平台

**《Python程序设计》课程设计报告**



**计算机与信息科学学院**

**作者：谭祎（2023051603132）**

**2023级计算机科学与技术56班**

《Python程序设计》课程设计报告

名称：重师计信在线平台——基于django/websocket的师生在线信息管理+跨平台聊天系统

院/系：计算机与信息科学学院

专业：计算机科学与技术

年级：2023级

班级：计算机科学与技术

姓名：谭祎

学号：2023051603132

时间：2025年1月1日

目录

[1、项目描述 1](#_Toc16840)

[1.1 项目背景 1](#_Toc12130)

[1.2 实现目的 2](#_Toc9427)

[1.3 系统描述 3](#_Toc11129)

[2、软件设计 3](#_Toc15751)

[2.1系统环境 3](#_Toc17543)

[2.1.1运行环境 3](#_Toc10875)

[2.1.2硬件环境 4](#_Toc12407)

[2.1.3软件环境 4](#_Toc15593)

[2.2 前端技术 4](#_Toc10897)

[2.2.1 HTML 5](#_Toc7693)

[2.2.2 CSS 5](#_Toc28143)

[2.2.3 Bootstrap框架 5](#_Toc8254)

[2.3 后端技术 5](#_Toc159)

[2.3.1 django框架 5](#_Toc8702)

[2.3.2 SQLite数据库 7](#_Toc15627)

[2.3.3 Websocket实时通信 8](#_Toc23124)

[2.4 系统架构 9](#_Toc13142)

[2.4.1 用户管理模块 10](#_Toc24877)

[2.4.2 课程管理模块 10](#_Toc2755)

[2.4.3 成绩查询模块 11](#_Toc7067)

[2.4.4 实时聊天室模块 11](#_Toc2935)

[2.4.5 系统前端与后端设计 12](#_Toc13092)

[3、设计结果 12](#_Toc31497)

[3.1 学生端 12](#_Toc14471)

[3.1.1登录与注册 13](#_Toc2743)

[3.1.2 学生主页 14](#_Toc21915)

[3.1.3 修改个人信息 15](#_Toc19524)

[3.1.4 修改密码 15](#_Toc20278)

[3.1.5 选课页面 16](#_Toc2763)

[3.1.6 聊天室 17](#_Toc24382)

[3.2 教师端 18](#_Toc3840)

[3.2.1教师主页 19](#_Toc28755)

[3.2.2 修改个人信息 20](#_Toc21186)

[3.2.3 修改密码 21](#_Toc20401)

[3.2.4 学生信息查询 21](#_Toc1508)

[3.2.5 学生成绩查询 22](#_Toc20282)

[3.2.6 聊天室 23](#_Toc5783)

[3.3 教务员端 25](#_Toc24784)

[4、总结与分析 26](#_Toc16868)

[4.1 项目效果 26](#_Toc15709)

[4.2 不足与改进方法 26](#_Toc5383)

[4.2.1 安全性问题 26](#_Toc32059)

[4.2.2 性能优化 27](#_Toc20152)

[4.2.3 用户体验 27](#_Toc29926)

[4.2.4 权限控制不完善 27](#_Toc18336)

[4.2.5 代码结构与模块化 28](#_Toc18)

[5、代码结构 29](#_Toc908)

[5.1 总体架构 29](#_Toc30366)

[5.2 前端架构 29](#_Toc22811)

[5.3 后端架构 30](#_Toc8701)

1、项目描述

1.1 项目背景

随着互联网的飞速发展，在线教育平台逐渐成为教育行业的重要组成部分。尤其是在近几年，随着技术的不断革新与信息化建设的推进，互联网在教育领域的应用越来越深入，给教师和学生带来了前所未有的学习和教学体验。2023年5月13日，“一带一路”TOP10影响力社会智库网经社电子商务研究中心联合网经社数字教育台发布《2023年中国数字教育市场数据报告》，报告显示，2023年数字教育市场规模为4133亿元，同比增长14.17%。此外，2019-2022年市场规模（增速）分别为3468亿元（21.47%）、4328亿元（24.79%）、3220亿元（-25.61）、3620亿元（12.42%），足以见得行业发展之迅猛，这也让如何利用现代科技的力量来提升教育效率、优化教学互动、扩展教学内容的传播途径，成为了教育信息化改革的重要课题。

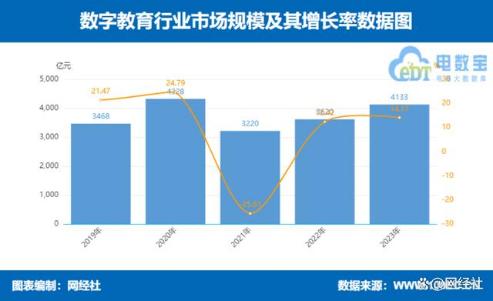


图1-1 数字教育行业市场规模及其增长率数据图

本项目旨在结合当前教育领域中的需求与挑战，通过 Django 框架实现一个综合性的在线教育平台，涵盖了从学生注册、课程管理到教师授课和成绩查询等多个方面，同时通过实时聊天室模块增强师生之间、学生与学生之间的互动，进一步推动教学的深度融合。通过这个平台，师生之间的互动将更加及时与有效，学生的个性化需求也能得到更好的满足，从而在教育信息化建设的道路上迈出重要的一步。

1.2 实现目的

本项目聚焦于设计并实现一个依托 Django 框架搭建的在线教育平台，集课程管理、成绩查询、学生注册以及在线互动聊天等多功能于一体。借助该平台，师生互动将更为顺畅，教学内容的传播也会更加高效快捷，从而有力推动教育信息化进程，为在线教育的发展开拓新路径。

在用户管理系统构建方面，借助 Django 的认证系统与自定义用户模型，学生和教师得以顺利注册、登录并管理各自的信息。学生能够完成注册流程，轻松查看自己的成绩，并根据兴趣挑选课程；教师则可便捷地管理所授课程，及时录入学生成绩，高效管控学生选课信息，确保教学管理的有序开展。

实时聊天室模块的设立显著提升了平台的互动性。传统课堂教学中，师生互动往往受时空束缚，而在此平台上，师生可随时通过聊天室交流，学生间也能自由讨论、分享学习体会，打破了传统课堂的限制，极大地增强了教学的互动性与灵活性，为教学活动注入新活力。

成绩查询系统的加入，让学生能够迅速查询自己的学习成绩，教师也能及时上传和管理成绩数据。系统以清晰直观的界面展示课程与成绩详情，保证信息的精准与透明，为师生营造优良的数据管理与查询环境，满足双方对成绩管理的需求。

平台在前后端设计上秉持简洁高效的理念。前端界面注重用户友好性，操作简便易懂；后端凭借 Django 框架的强大功能，确保系统稳定高效运行。整体设计以优化用户体验为出发点，让教师和学生在使用时尽享流畅便捷之感，使日常学习过程更加高效愉悦。

1.3 系统描述

本系统基于 Django 框架，设计了一个全面的在线教育平台，包含用户管理、课程管理、成绩查询和实时聊天室等模块。用户管理模块通过自定义用户模型区分学生和教师角色，提供注册、登录和信息管理功能。课程管理模块支持教师创建、编辑和分配课程，学生可以选择并参与感兴趣的课程。成绩查询模块允许教师录入学生成绩，学生可随时查询个人成绩。聊天室模块采用 WebSocket 技术，实现师生及学生间的实时互动，增强了学习的互动性和灵活性。系统前端简洁直观，后端采用 Django 强大的功能确保稳定运行，整体设计注重用户体验与数据安全，提供高效的学习与教学环境。

2、软件设计

2.1系统环境

**2.1.1运行环境**

1、前端运行环境

表2-1 前端运行环境

|  |  |
| --- | --- |
| **工具** | **版本号** |
| Edge | 任意版本 |
| Chrome | 任意版本 |

2、后端运行环境

表2-2 后端运行环境

|  |  |  |  |
| --- | --- | --- | --- |
| **包/库** | **版本号** | **包/库** | **版本号** |
| Django | 3.2.5 | django-bootstrap3 | 4.66.2 |
| Python | 3.9 | import-export | 1.3.2 |
| django-npm | 17.0.10 | django-simpleui | 2021.2.4 |
| djangorestframework | 2.1.0 | channels | 3.8.18 |
| channels-redis | 4.0.2 |  |  |

详细配置请见代码项目文件夹中的requirements.txt。

**2.1.2硬件环境**

表2-3 硬件环境

|  |  |
| --- | --- |
| CPU | 13th Gen Intel(R)Core(TM)i5-13490F 2.50 GHz |
| 硬盘 | 1T |
| 内存 | RAM 32G |
| 显卡 | NVIDIA GeForce RTX 4060 Ti |

**2.1.3软件环境**

表2-4软件环境

|  |  |
| --- | --- |
| 服务器操作系统 | Windows 11 |
| 开发工具 | Pycharm、Anaconda、SQlite |
| 开发语言 | Python、HTML+CSS+Django |
| 客户端 | Chrone或其他浏览器 |

2.2 前端技术

前端设计通常是指网站的视觉设计，主要包括HTML、CSS、JavaScript等技术。HTML负责定义页面结构，CSS决定页面元素的展示样式，JavaScript用于实现交互和动态效果。随着HTML5和CSS3的普及，前端框架的使用日益广泛，能够适应各种屏幕分辨率需求，并提高用户浏览体验。

**2.2.1 HTML**

HTML是网页制作中最常用的标记语言，已发展到HTML5版本。它通过标签和属性定义网页结构，创建用户界面。在电影推荐系统中，HTML被用来构建用户信息输入界面和电影信息展示界面，简洁的网页设计提升了系统的使用体验。

**2.2.2 CSS**

CSS（层叠样式表）用于网页风格设计，目前已更新至CSS3版本。它定义HTML元素的表现，如背景颜色、边框、内容位置等，并支持渐变、透明、圆角等特效。在职位推荐系统中，CSS被用来设计页面主题色、字体、排版等，提升页面的视觉效果和用户体验。

**2.2.3 Bootstrap框架**

Bootstrap是一个集HTML、CSS和JavaScript于一体的前端框架，旨在简化Web开发过程。本项目就基于Bootstrap框架进行了前端开发，它提供了规范化的HTML和CSS样式，确保跨浏览器的一致性。它的响应式设计使得本次设计能够快速创建适应不同设备的页面，广泛应用于各类网站开发中。

2.3 后端技术

**2.3.1 django框架**

Django框架是起源于开源社区的Python编程语言驱动的Web应用程序框架。Django具有较强的可扩展性，是因为在Django框架中包含了很多功能强大的第三方插件。目前，Django框架在国内的网站中应用较为广泛，比如大家熟知的网站有豆瓣、知乎等，该框架开源免费，是由社区进行维护，因为它展现出的强大生命力，吸引着许多Web开发者。使用该框架优势如下：

1、通用性

Django框架的模板系统强大且易扩展，是可以与所有客户端框架一起使用的，还可以提供多种形式的代码。

2、安全性

该框架提供一种较为智能安全的文件结构与管理方法，实际数据存储在数据库表中，防范了多种漏洞，很大程度上保障了网站的安全性。

3、灵活性

该框架由Python语言编写，不受服务器平台的限制，开发者不必去了解具体技术，因为Django中内嵌了ORM框架，鼓励开发者快速开发，遵循MVC设计，运用MTV框架（模型、视图和模板）模型。它们各自的职责如表2.5所示：

表2-5MVT框架职责表

|  |  |  |  |
| --- | --- | --- | --- |
| 层次 | 模型（Model）  即数据存取层 | 模板（Template）  即表现层 | 视图（View）  即业务逻辑层 |
| 职责 | 处理与数据相关的所有事务。 | 处理与表现相关的决定。 | 存取模型及调取恰当模板的相关逻辑、模型、模板的桥梁。 |

Django框架的MTV各个组织协作图如图2-1所示：

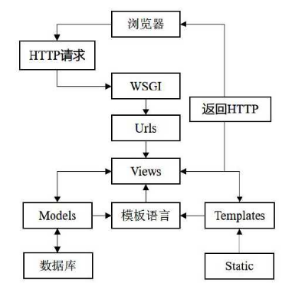


图2-1 MTV协作流程图

**2.3.2 SQLite数据库**

SQLite是django系统原生的一种轻型的，基于文件的嵌入式数据库。由于SQLite是一款开源软件，商业和用户都可以免费下载使用。它是专门为嵌入式设备而设计的，以文件的形式存在。SQLite支持大小至2TB的数据库，一个完整的SQLite数据库完全存储在单一的磁盘文件中，SQLite可以直接将文件读写到磁盘中，而且数据库文件格式是可以跨平台的，可以与C#、Java、PHP等多种编程语言相结合，具有占用资源少、零配置、可移植性好、运行快速可靠等特点,目前广泛应用在移动端小型设备的数据库中。本系统设计的客户端的全部数据信息都是通过SQLite数据库进行存储。SQLite的体系结构如图2-2所示。

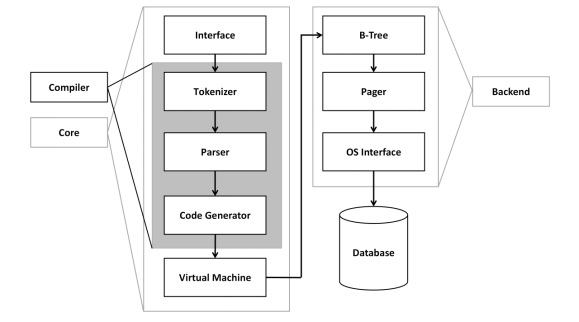


图2-2 SQLite体系结构

**2.3.3 Websocket实时通信**

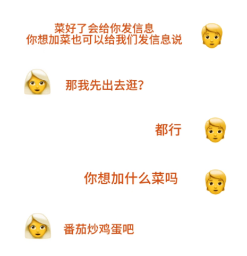
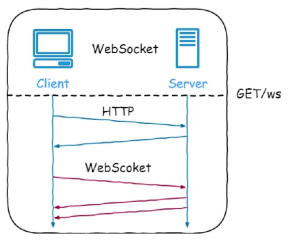
****

图 2-3 socket通信原理

Socket 是一种基于 TCP 的网络协议，与 HTTP 协议类似。其原理基于可靠的 TCP 传输，确保数据准确无误地在网络中传递。

Socket 具有双向全双工通讯的特性，这意味着客户端和服务器端能够自由地相互收发消息。此外，Socket 采用长连接机制。当客户端与服务器建立连接后，连接不会轻易断开，服务器可以持续地与客户端互相收发消息。长连接避免了频繁连接和断开操作所带来的网络开销，提高了数据传输效率。

在聊天系统开发中，Socket 的这些特性具有显著优势。基于 TCP 的特性确保了聊天信息传递的准确性和完整性，避免了信息丢失或出错；双向全双工通讯能够实现聊天用户间的实时交互，保证聊天体验的流畅性；此外，长连接机制使得用户在聊天过程中不会因连接中断而影响体验，能够持续地接收和发送消息。

我们可以发现，这样能实现实时通讯的协议——Socket 非常适合用于聊天室的开发，这也是本项目实时聊天板块的核心技术。

**2.4 系统架构**

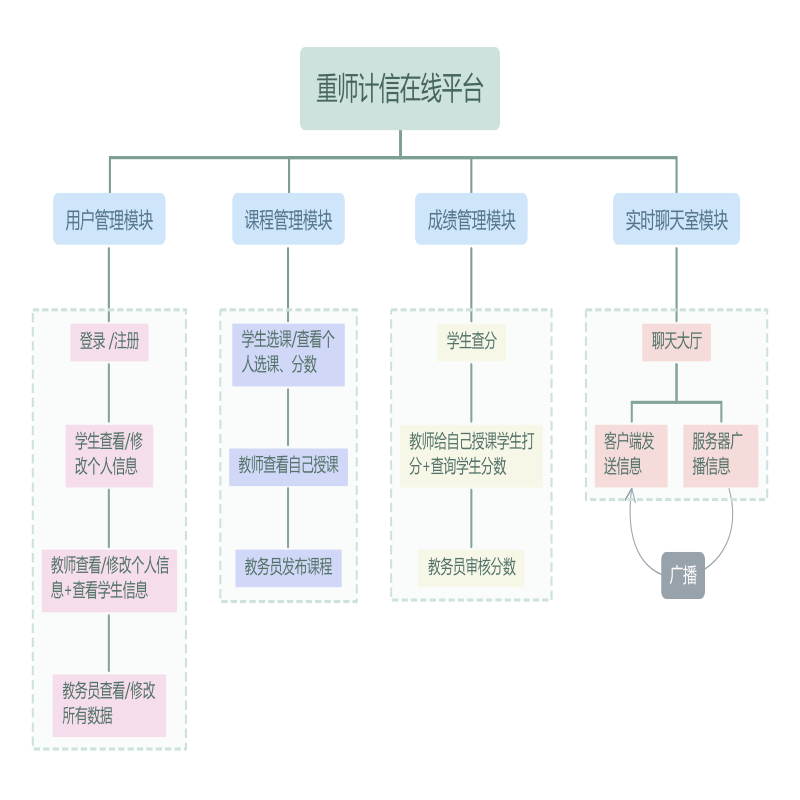
本系统以 Django 框架为基础，结合现代教育需求和技术发展趋势，开发出一个功能完整的在线教育平台。系统采用模块化设计，包含多个模块，涵盖了用户管理、课程管理、成绩查询以及实时在线聊天等功能，旨在为学生和教师提供一个高效便捷的在线教学与学习环境。以下图2-4为该系统的模块化设计图：

图 2-4 系统模块设计图

**2.4.1 用户管理模块**

系统通过 Django 的内置用户认证功能，并结合自定义用户模型，设计了一个完善的用户管理模块。学生和教师可以通过注册、登录等方式进入平台，管理个人信息。学生能够查看个人成绩、选课以及获取学习资源；教师则能够管理所教授的课程，记录学生成绩，并进行课程安排。

学生和教师的角色通过权限控制加以区分，确保每个用户只能访问与自己相关的内容。学生可以选择感兴趣的课程并参与其中，而教师则负责课程的管理和学生成绩的录入。此外，系统还支持角色权限的扩展和修改，确保能够灵活应对不同的使用场景。

**2.4.2 课程管理模块**

课程管理模块是本平台的核心功能之一。教师可以通过该模块创建、编辑和删除课程，并且能够为每个课程分配合适的学生。学生可以查看所有可用课程，并选择自己感兴趣的课程进行学习。在课程管理过程中，教师能够对课程进行启用或禁用，从而有效地管理课程资源。

该模块的设计不仅方便了课程的管理，还提高了学生选课的效率。学生通过系统查看课程信息后，能够快速选择并参与自己感兴趣的课程，而教师也可以随时了解每个课程的学生选课情况，方便教学安排。

**2.4.3 成绩查询模块**

成绩查询模块实现了学生成绩的录入与查询功能。教师可以通过该模块录入学生的成绩，学生则可以随时查询自己的成绩，了解自己的学习进展。这一功能不仅提高了学生和教师之间的信息透明度，也为学生提供了一个便捷的成绩查询渠道。

成绩查询模块的数据安全性非常重要，系统对学生成绩进行严格的权限控制，确保只有学生本人和相关教师能够查看成绩信息。这不仅保护了学生的隐私，还保证了成绩数据的准确性与可靠性。

**2.4.4 实时聊天室模块**

实时聊天室是本系统的一大亮点。与传统的教学模式不同，教师和学生之间的沟通不再仅限于课堂上，系统提供了一个实时的互动平台，使得师生可以随时进行交流。学生之间也可以在聊天室内讨论课程内容、交流学习心得，进一步增强了学习的互动性。

聊天室采用了 WebSocket 技术，实现了实时消息推送。当学生或教师发送消息时，系统会立即显示，无需刷新页面，确保了信息的及时传递。此外，系统还提供了消息的发送、接收和历史记录功能，使得用户能够方便地查看之前的聊天内容。

**2.4.5 系统前端与后端设计**

在前端设计上，系统力求简洁直观，用户界面清晰、操作简单，确保不同用户能够快速上手。前端采用了 HTML、CSS 和 JavaScript 等技术，保证了页面的响应式设计和良好的用户体验。

后端则基于 Django 框架，利用其强大的 ORM、权限管理和视图控制功能，实现了整个系统的稳定运行。数据的存储和管理依赖于 sqlite 数据库，确保了数据的安全与高效访问。

3、设计结果

3.1 学生端

“君子生非异也，善假于物也。”为力求达到优良的体验效果的同时，达到“跨平台”的最终目的，同时在开发时间和工作量上尽可能经济——选择使用django架构的初衷也在于此，我计划将“浏览器”作为跨平台的门户。

这样，只要我在主机上启动了服务器，任何平台的用户都可以在连接重师校园网后，通过我给出的地址（即我的实时校园网ip：8000）就可以在自己的设备的浏览器上访问我的网站。



图 3-1 查询ip地址



图 3-2 启动服务器

故学生端我们在手机上测试，也是为了展示项目的跨平台效果。

**3.1.1登录与注册**

学生可通过学号和用户名登录，若没有账户可进行注册，填写学号、用户名和密码后完成注册，确保身份验证后进入平台使用各项功能。目前注册默认只能注册得到学生账户。

测试学生账号：

账号：23001，密码：123456。



图 3-3 学生端手机登录/注册

**3.1.2 学生主页**

学生登录后可访问个人主页，查看个人信息、已选课程、成绩以及参与的聊天房间，主页提供清晰的导航功能，便于学生查看学习情况。

同时，若不想使用网页了，也可以退出登录。



图 3-4 学生主页

**3.1.3 修改个人信息**

学生可以修改个人资料，包括用户名、学号、真实姓名等信息，确保个人信息的准确性和完整性。



图 3-5 学生端修改个人信息

**3.1.4 修改密码**

学生可以修改个人资料，包括用户名、学号、真实姓名等信息，确保个人信息的准确性和完整性。



图 3-6 学生端修改密码

**3.1.5 选课页面**

学生通过选课页面浏览和选择感兴趣的课程，可以查看课程信息与解说，方便学生根据自己的需求进行选课。

同时，出分后也可以对应查看成绩。

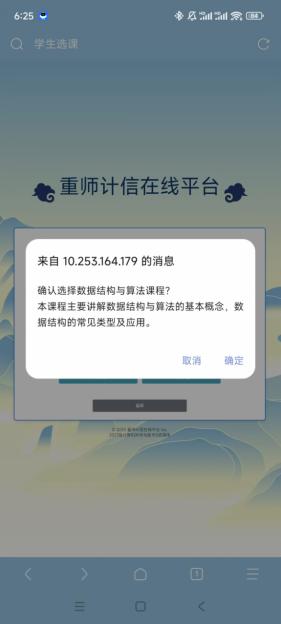


图 3-7 学生选课界面

**3.1.6 聊天室**

学生可进入聊天室与同学和教师进行实时沟通，讨论课程内容或学习问题，聊天室增强了学习的互动性与便捷性。



图 3-8 学生端手机加入聊天

3.2 教师端

为了保证信息的安全性，同时真正满足组织管理需要，目前教师账号只允许教务员创建并发放。

测试教师账号：

账号：teacher，密码：tea123456。

教师端我们用Edge浏览器展示：



图 3-9 教师主页

**3.2.1教师主页**

教师登录后进入个人主页，查看授课课程列表、学生选课情况以及已发布的成绩。主页提供对已选学生、课程内容的清晰展示，教师可便捷管理课程安排与与学生互动的功能。此外，教师还可以快速进入聊天功能与学生沟通。



图 3-10 教师主页

**3.2.2 修改个人信息**

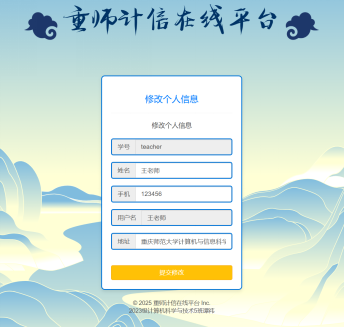
教师可通过个人主页修改个人信息，包括用户名、邮箱、联系方式等，确保资料的准确性。修改信息后，教师的教学管理、通讯和课程安排将基于最新的资料，方便与学生和同事之间的沟通。

图 3-11 教师端修改个人信息

**3.2.3 修改密码**

为了保障账户的安全，教师可以随时修改登录密码。修改密码后，教师需要使用新密码进行登录，确保账号安全性并防止未经授权的访问。



图 3-12 教师端修改密码

**3.2.4 学生信息查询**

教师可以查看所授课程下学生的基本信息，包括学号、姓名等，方便教师了解学生的情况。通过该功能，教师能够对学生进行有效的管理、了解学生的个人情况，以及时为学生提供帮助。



图 3-13 教师端查询学生信息

**3.2.5 学生成绩查询**

教师可通过此功能查看学生的成绩，按课程和学生进行分类管理。教师能够为每个学生输入成绩并进行更新，还可以查看学生成绩历史，帮助教师更好地跟踪学生的学术表现，便于进行教学调整。



图 3-14 教师端查询学生成绩

**3.2.6 聊天室**

教师可以进入聊天室与学生进行即时交流，解答学术问题、讨论课堂内容。聊天室作为一种互动工具，促进了教师与学生之间的沟通，增强了学习的互动性和及时反馈。

在主页点击“进入聊天室”后，首先会进入到“聊天大厅”。



图 3-15 教师端进入聊天室——聊天大厅

你需要和你想要聊天的用户，在同一时间输入同样的口令，以进入对应聊天室。聊天室中自己的视图下，自己发出的信息是绿色气泡，别人发出的信息是白色气泡，符合日常使用情况。



图 3-16 教师端在线聊天室

3.3 教务员端

登录账户名“admintanyi”，输入密码123456，即可进入管理后台。



图 3-17 登录教务员账密

登录后会自动跳转到django管理界面，真正实现最高权限的管理（如您所见，admintanyi是一个超级用户，默认只有这一个教务员，如果您有需要，也可以手动创建一个）。



图 3-18 教务员（超级用户管理后台）

4、总结与分析

4.1 项目效果

本项目基于 Django 开发，旨在为用户提供一个集成的在线平台

主要功能包括用户管理、课程管理、成绩查询、聊天和文件上传等。通过自定义用户模型，系统能够实现不同角色的管理，如学生、教师和管理员，确保了权限控制和认证的灵活性。

课程管理模块允许用户创建、查看和选择课程，学生与教师之间的关联通过外键和多对多关系得以管理。此外，成绩管理功能使得教师可以为学生录入成绩，学生也能查询到自己的成绩信息，进一步加强了平台的教学功能。在实时通讯方面，项目通过 Django Channels 实现了 WebSocket 功能，支持用户在聊天室内即时交流。

同时，本项目在实现这些功能时保持了模块化设计，功能模块独立且易于维护，为用户提供了一个完整的线上教学管理和互动平台。

4.2 不足与改进方法

**4.2.1 安全性问题**

当前系统在用户认证和数据传输等方面存在安全隐患。用户登录仅依靠简单的账号密码组合，缺乏如多因素认证等强化手段，易遭受暴力破解攻击，导致用户信息泄露。在数据传输过程中，未全面采用加密技术，敏感信息（如成绩数据、用户个人信息等）在网络中传输时面临被窃取风险。改进方法是引入多因素认证，如短信验证码或指纹识别等，增强用户登录安全性；同时，对系统内所有数据传输启用 加密协议，确保数据在网络传输中的保密性和完整性，防止信息被非法获取与篡改。

**4.2.2 性能优化**

随着平台用户和课程数据的不断增长，系统性能面临挑战。数据库查询效率逐渐降低，复杂查询操作（如查询大量学生成绩或多门课程信息时）响应时间延长，影响用户体验。服务器资源分配未充分优化，在高并发访问（如选课高峰期或聊天室活跃时段）时容易出现卡顿甚至崩溃现象。后续计划采用数据库索引优化技术，针对频繁查询的字段建立合适索引，提升数据检索速度。

**4.2.3 用户体验**

前端界面设计在交互性和友好度上仍有提升空间。操作流程不够简洁明了，如学生选课过程步骤繁琐，需要多次点击和页面跳转才能完成选课操作；教师在成绩录入和课程管理时，操作界面布局不够直观，影响操作效率。此外，系统缺乏个性化设置，不能满足用户多样化需求。改进方向为重新设计操作流程，简化选课等关键操作步骤，减少不必要的页面切换；优化界面布局，将相关功能模块集中展示，方便用户操作；增加个性化设置功能，允许用户根据自身习惯调整界面主题、字体大小等，提升用户使用的舒适度和便捷性。

**4.2.4 权限控制不完善**

目前的权限管理体系在角色权限细分和动态权限调整方面存在不足。教师和管理员在部分功能权限上存在重叠或界定不清的情况，可能导致数据管理混乱。同时，系统无法根据业务场景变化灵活调整用户权限，如特殊课程或教学活动中需要临时授予或限制某些用户权限时难以实现。为解决此问题，需重新梳理各角色在所有业务功能中的权限需求，明确教师、管理员等不同角色的操作权限边界；结合用户属性、环境因素等实现动态权限分配与管理，确保权限控制精准有效，保障系统数据安全与业务秩序。

**4.2.5 代码结构与模块化**

项目中部分代码的模块化程度欠佳，部分视图函数与业务逻辑过度耦合，使得代码维护和扩展困难。（这也是开发过程中报错的主要原因之一）例如，在用户管理和课程管理相关功能中，部分代码分散且相互依赖，当修改其中一个功能时，可能引发连锁反应影响其他功能的正常运行。后续应按照单一职责原则对代码进行重构，将紧密关联的功能代码封装成独立模块或类，降低模块间耦合度，降低其维护难度。

5、代码结构

5.1 总体架构

本项目遵循了现代 web 应用的分层设计，后端主要负责数据处理、业务逻辑和 API 提供，前端则通过接口与后端进行数据交互，并负责呈现用户界面。

通过后端的路由设置，整体上实现了前后端分离设计。

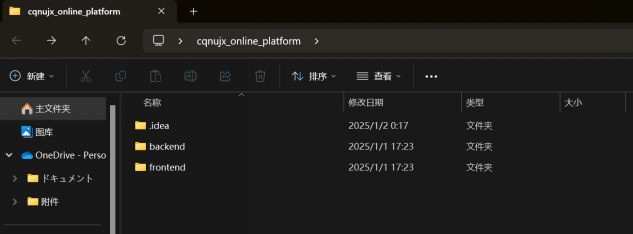


图 5-1 代码总结构

5.2 前端架构

前端文件结构如下：

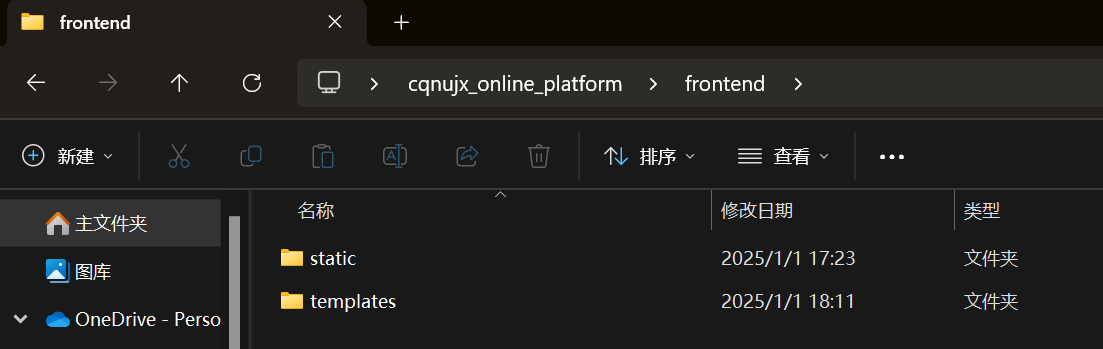


图 5-2 前端文件结构

frontend:.

├─static

│ ├─bootstrap

│ │ ├─css

│ │ ├─fonts

│ │ └─js

│ ├─css

│ │ └─libs

│ │ └─images

│ ├─img

│ └─js

│ └─libs

└─templates

└─chat

5.3 后端架构

后端文件结构如下：

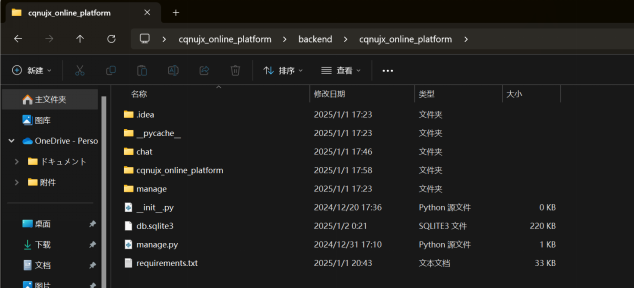


图 5-3 后端文件结构

backend:.

│ db.sqlite3

│ manage.py

│ requirements.txt

│ \_\_init\_\_.py

│

├─.idea

│ │ .gitignore

│ │ cs\_wisdom\_platform.iml

│ │ misc.xml

│ │ modules.xml

│ │ workspace.xml

│ │

│ └─inspectionProfiles

│ profiles\_settings.xml

│

├─chat

│ │ admin.py

│ │ apps.py

│ │ consumers.py

│ │ models.py

│ │ routing.py

│ │ tests.py

│ │ urls.py

│ │ views.py

│ │ \_\_init\_\_.py

│ │

│ ├─migrations

│ │ │ \_\_init\_\_.py

│ │ │

│ │ └─\_\_pycache\_\_

│ │ \_\_init\_\_.cpython-39.pyc

│ │

│ └─\_\_pycache\_\_

│ admin.cpython-39.pyc

│ apps.cpython-39.pyc

│ consumers.cpython-39.pyc

│ models.cpython-39.pyc

│ routing.cpython-39.pyc

│ urls.cpython-39.pyc

│ views.cpython-39.pyc

│ \_\_init\_\_.cpython-39.pyc

│

├─cqnujx\_online\_platform

│ │ asgi.py

│ │ settings.py

│ │ urls.py

│ │ wsgi.py

│ │ \_\_init\_\_.py

│ │

│ └─\_\_pycache\_\_

│ asgi.cpython-39.pyc

│ settings.cpython-39.pyc

│ urls.cpython-39.pyc

│ wsgi.cpython-39.pyc

│ \_\_init\_\_.cpython-39.pyc

│

├─manage

│ │ \_\_init\_\_.py

│ │

│ ├─course

│ │ │ admin.py

│ │ │ apps.py

│ │ │ models.py

│ │ │ tests.py

│ │ │ views.py

│ │ │ \_\_init\_\_.py

│ │ │

│ │ ├─migrations

│ │ │ │ 0001\_initial.py

│ │ │ │ \_\_init\_\_.py

│ │ │ │

│ │ │ └─\_\_pycache\_\_

│ │ │ 0001\_initial.cpython-39.pyc

│ │ │ \_\_init\_\_.cpython-39.pyc

│ │ │

│ │ └─\_\_pycache\_\_

│ │ admin.cpython-39.pyc

│ │ apps.cpython-39.pyc

│ │ models.cpython-39.pyc

│ │ views.cpython-39.pyc

│ │ \_\_init\_\_.cpython-39.pyc

│ │

│ ├─user

│ │ │ admin.py

│ │ │ apps.py

│ │ │ auth.py

│ │ │ managers.py

│ │ │ models.py

│ │ │ tests.py

│ │ │ views.py

│ │ │ \_\_init\_\_.py

│ │ │

│ │ ├─migrations

│ │ │ │ 0001\_initial.py

│ │ │ │ \_\_init\_\_.py

│ │ │ │

│ │ │ └─\_\_pycache\_\_

│ │ │ 0001\_initial.cpython-39.pyc

│ │ │ \_\_init\_\_.cpython-39.pyc

│ │ │

│ │ └─\_\_pycache\_\_

│ │ admin.cpython-39.pyc

│ │ apps.cpython-39.pyc

│ │ managers.cpython-39.pyc

│ │ models.cpython-39.pyc

│ │ views.cpython-39.pyc

│ │ \_\_init\_\_.cpython-39.pyc

│ │

│ └─\_\_pycache\_\_

│ \_\_init\_\_.cpython-39.pyc

│

└─\_\_pycache\_\_

\_\_init\_\_.cpython-39.pyc