**成都锦城学院**

本科生毕业论文（设计）

题 目

二级学院 计算机与软件学院

专 业

学生姓名

学 号 年级

指导教师

教务处 制表

年 月 日

在线教育平台的设计与实现

**专业：**

**学生：鲜青松 指导老师：**

摘要：随着科技的迅猛发展，传统教育模式逐渐显露出诸多局限，如教学资源分布不均、学习时间与地点的限制，以及个性化教学的困难，这些问题显著影响了教育的覆盖面和教学效率。尤其在当今环境中，传统线下教育无法完全满足学习者的多样化需求，促使在线教育的兴起和广泛应用。因此，为了有效应对这些挑战，本项目设计并开发了一个基于 Spring Boot 2 和 MyBatis 的在线教育平台，旨在充分发挥在线教育的优势，合理配置教育资源，并优化用户的学习体验。

该平台提供了课程展示、视频学习、在线考试和互动评论等多种功能，满足用户的多样化学习需求。前端采用 Vue.js、element-plus、Pinia 技术栈进行开发，以实现动态和响应式的用户界面；后端则基于 Spring Boot 2 框架，结合 MyBatis 进行灵活的数据库操作，确保数据的高效处理以及数据存储。同时，通过引入 Redis 进行数据缓存优化，极大提升了平台的访问速度。为了保障用户信息的安全性，系统使用 JWT 技术进行身份验证，确保用户数据的保护，平台的系统设计包括多个模块：用户管理模块、课程管理模块、博客查看模块、视频播放模块和考试模块，学生可以查看学习课程，教师可方便地管理和更新课程内容。为了提高教育效果，平台还集成了数据分析功能，帮助教育机构基于成绩表现，从而实现个性化教育。

关键词：在线教育；Spring Boot；MyBatis；Vue；数据分析

**Design and Implementation of Online Education Platform**

**Major:**

**Student: Supervisor:**

**Abstract:** With the rapid development of technology, traditional education models have shown many limitations, such as uneven distribution of teaching resources, limitations on learning time and place, and difficulties in personalized teaching, which have significantly affected the coverage and efficiency of education. Especially in today's environment, traditional offline education cannot fully meet the diverse needs of learners, which has led to the rise and widespread use of online education. Therefore, in order to effectively address these challenges, this project designs and develops an online education platform based on Spring Boot 2 and MyBatis, aiming to fully leverage the advantages of online education, rationally allocate educational resources, and optimize the user's learning experience.

The platform provides functions such as course display, video learning, online examination, and interactive commenting to meet the diverse learning needs of users. The front-end is developed using Vue.js, element-plus, and Pinia technology stack to achieve dynamic and responsive user interfaces; the backend is based on the Spring Boot 2 framework and combined with MyBatis for flexible database operations to ensure efficient data processing and data storage. At the same time, Redis is introduced for data caching optimization, greatly improving the platform's access speed. To ensure the security of user information, the system uses JWT technology for identity verification to ensure the protection of user data, and the system design includes several modules: a user management module, a course management module, a blog viewing module, a video playback module, and an examination module. Students can view their learning courses, and teachers can conveniently manage and update course content. To improve the effectiveness of education, the platform also integrates data analysis functions to help educational institutions achieve personalized education based on performance.

**Key Words:** Online Education；Spring Boot；MyBatis；Vue；Data Analysis

目 录

[1 绪论 1](#_Toc179563381)

[1.1 研究背景与研究意义 1](#_Toc179563382)

[1.2 国内外研究现状 1](#_Toc179563383)

[1.3 论文主要研究内容 3](#_Toc179563384)

[1.4 论文组织结构 3](#_Toc179563385)

[2 关键技术和开发环境 5](#_Toc179563386)

[2.1 关键技术 5](#_Toc179563387)

[2.1.1 MySQL 技术 5](#_Toc179563388)

[2.1.2 MyBatis技术 5](#_Toc179563389)

[2.1.3 Redis技术 5](#_Toc179563390)

[2.1.4 JWT (JSON Web Token)技术 6](#_Toc179563391)

[2.1.5 Vue 3 6](#_Toc179563392)

[2.1.6 Vite 6](#_Toc179563393)

[2.1.7 Element Plus 6](#_Toc179563394)

[2.2 开发环境 6](#_Toc179563395)

[3 系统需求分析 7](#_Toc179563396)

[3.1 功能性需求 7](#_Toc179563397)

[3.1.1 账号模块 7](#_Toc179563398)

[3.1.2 商品模块 8](#_Toc179563399)

[3.1.3 订单模块 9](#_Toc179563400)

[3.1.4 购物车模块 10](#_Toc179563401)

[3.1.5 商品审核模块 10](#_Toc179563402)

[3.1.6 用户管理模块 10](#_Toc179563403)

[3.2 非功能性需求 11](#_Toc179563404)

[3.2.1 性能需求 11](#_Toc179563405)

[3.2.2 安全性需求 11](#_Toc179563406)

[3.2.3 数据保密需求 11](#_Toc179563407)

[3.2.4 易用性需求 11](#_Toc179563408)

[3.2.5 可用性需求 12](#_Toc179563409)

[3.2.6 可维护性需求 12](#_Toc179563410)

[3.3 本章小结 12](#_Toc179563411)

[4 系统概要设计 13](#_Toc179563412)

[4.1 系统总体设计 13](#_Toc179563413)

[4.1.1 系统模块设计 13](#_Toc179563414)

[4.1.2 系统架构设计 13](#_Toc179563415)

[4.2 数据库设计 13](#_Toc179563416)

[4.2.1 数据库整体设计 13](#_Toc179563417)

[4.2.2 数据库详细设计 13](#_Toc179563418)

[4.3 系统UI整体设计 13](#_Toc179563419)

[4.3.1 系统颜色设计 13](#_Toc179563420)

[4.3.2 系统字号、段落设计 13](#_Toc179563421)

[4.4 前端路由设计 13](#_Toc179563422)

[5 系统详细设计 14](#_Toc179563423)

[5.1 账号模块设计与实现 14](#_Toc179563424)

[5.1.1 注册 14](#_Toc179563425)

[5.1.2 登录 14](#_Toc179563426)

[5.1.3 查看订单 14](#_Toc179563427)

[5.2 商品模块设计与实现 14](#_Toc179563428)

[5.2.1 商品上下架 14](#_Toc179563429)

[5.2.2 修改商品信息 14](#_Toc179563430)

[5.3 订单模块设计与实现 14](#_Toc179563431)

[5.3.1 创建订单 14](#_Toc179563432)

[5.3.2 完成订单 14](#_Toc179563433)

[5.4 购物车模块设计与实现 14](#_Toc179563434)

[5.4.1 增减商品数量 14](#_Toc179563435)

[5.4.2 删除商品 14](#_Toc179563436)

[5.5 商品审核模块设计与实现 14](#_Toc179563437)

[5.5.1 下架商品 14](#_Toc179563438)

[5.5.2 修改商品信息 14](#_Toc179563439)

[5.6 用户管理模块设计与实现 14](#_Toc179563440)

[5.6.1 删除用户信息 14](#_Toc179563441)

[5.6.2 修改用户信息 14](#_Toc179563442)

[6 系统开发与实现 16](#_Toc179563443)

[6.1 账号模块开发与实现 16](#_Toc179563444)

[6.1.1 注册 16](#_Toc179563445)

[6.1.2 登录 16](#_Toc179563446)

[6.1.3 查看订单 16](#_Toc179563447)

[6.2 商品模块开发与实现 16](#_Toc179563448)

[6.2.1 商品上下架 16](#_Toc179563449)

[6.2.2 修改商品信息 16](#_Toc179563450)

[6.3 订单模块开发与实现 16](#_Toc179563451)

[6.3.1 创建订单 16](#_Toc179563452)

[6.3.2 完成订单 16](#_Toc179563453)

[6.4 购物车模块开发与实现 16](#_Toc179563454)

[6.4.1 增减商品数量 16](#_Toc179563455)

[6.4.2 删除商品 16](#_Toc179563456)

[6.5 商品审核模块开发与实现 16](#_Toc179563457)

[6.5.1 下架商品 16](#_Toc179563458)

[6.5.2 修改商品信息 16](#_Toc179563459)

[6.6 用户管理模块开发与实现 16](#_Toc179563460)

[6.6.1 删除用户信息 16](#_Toc179563461)

[6.6.2 修改用户信息 16](#_Toc179563462)

[7 系统测试 17](#_Toc179563463)

[7.1 测试方法说明 17](#_Toc179563464)

[7.2 测试用例及结果分析 17](#_Toc179563465)

[7.2.1 账号模块 17](#_Toc179563466)

[7.2.2 商品模块 17](#_Toc179563467)

[7.2.3 订单模块 17](#_Toc179563468)

[7.2.4 购物车模块 17](#_Toc179563469)

[7.2.5 商品审核模块 17](#_Toc179563470)

[7.2.6 用户管理模块 17](#_Toc179563471)

[8 总结与展望 18](#_Toc179563472)

[参考文献 19](#_Toc179563473)

[致谢 20](#_Toc179563474)

# 1 绪论

## 1.1 研究背景与研究意义

在当今互联网时代，迅猛发展的网络技术深刻改变了人们的学习与生活方式。随着信息技术的不断进步，教育领域持续向数字化和网络化转型。近年来，中国居民的收入与消费水平提升，教育消费规模逐步扩大，教育需求呈现多样化和个性化的趋势。传统教育模式虽然能够提供统一课程，但往往难以契合不同学习者的个性需求。尤其在疫情期间，这一问题更加凸显，促使许多学习者选择在线学习作为替代途径。

在线教育的兴起为学习者带来了前所未有的机遇与便利。依托互联网技术，学习者可以随时随地获取丰富的学习资源，包括视频课程、电子书籍、在线测试及互动讨论等。这种灵活性不仅打破了学习时间和地点的限制，还让教育资源的获取变得更加公平和普及。在此背景下，在线教育平台成为连接学生、教师与教育资源的关键纽带。

然而，当前的在线教育平台面临诸多挑战。首先，课程质量良莠不齐，影响学习者的学习体验，导致学习效果难以保证；其次，部分平台缺乏有效的互动机制，使学习者在学习过程中感到孤立，缺乏参与度。此外，在线教育的普及也引发了教育公平的讨论，一些偏远地区的学生因网络条件欠佳或缺少必要设备，无法享受优质的在线教育资源。

因此，设计一个高效且用户友好的在线教育平台至关重要。该平台需整合优质教育资源，提供个性化学习体验，增进学习者间的互动，促进学习者与教师的有效沟通。基于此背景，本论文将借助现代Web技术，如Vue、Element Plus、Mybatis、Pinia、Spring Boot等，设计并实现一个在线教育平台，以提升学习者的参与度和学习效果。

通过本研究，旨在为提高在线教育的质量与效率、推动教育公平作出切实贡献。期望构建一个涵盖课程浏览、注册、学习、测试和互动功能的在线教育平台，为学习者和教育工作者营造更加便捷高效的学习环境。同时，本研究还将为后续在线教育研究提供理论依据与实践经验，助力在线教育的发展与创新。

## 1.2 国内外研究现状

近年来，中国的在线教育平台迅速崛起，备受社会关注，成为教育研究的热门领域。尤其是在新冠疫情期间，全球封锁措施大大推动了在线学习的普及，各种在线教育平台不断涌现，深刻改变了传统教育格局。这一转变不仅在教育供给方式上带来了变革，也对学习者的学习习惯、环境和教育公平等方面产生了深远影响。

首先，在线教育市场正在逐步成为中国教育领域中不可或缺的重要组成部分。据统计数据，在线教育市场规模逐年增长，预计未来几年将持续保持强劲的增长态势。随着用户对教育质量和灵活性需求的日益增加，越来越多的人选择在线教育，这促进了各种在线教育平台的创新与发展。这一市场的迅猛扩张为教育机构带来了机遇，同时也加剧了市场竞争。

其次，技术的不断进步为在线教育的创新提供了坚实基础。近年来，人工智能（AI）、大数据和云计算等前沿技术在在线教育平台中的应用越来越普遍。这些技术的结合使教育内容更加个性化和针对性，显著提升了学习效率。例如，AI可以通过分析学习者的行为数据，提供个性化的学习建议和内容推荐；大数据则帮助教育机构深入了解学习者的需求与偏好，为课程设计提供支持。此外，云计算的应用使得教育资源的获取和管理变得更加高效，有效解决了传统教育模式下资源分配不均的问题。

在学习效果评估方面，研究者逐渐意识到多种因素会影响在线学习的成效。除了课程内容与教学设计外，学习者的自主学习能力、学习动机和学习环境等因素同样对学习效果有显著影响。大量研究表明，在线学习的有效性不仅取决于教育平台的设计与技术支持，还与学习者的积极参与和互动程度密切相关。这些研究为教育工作者在设计在线课程时提供了重要的参考，有助于激发学习者的学习动机。

教育公平与普及也是在线教育平台研究的关键议题。研究显示，在线教育为偏远地区的学生提供了接触优质教育资源的机会，推动了教育公平的实现。在中国，城乡教育差距问题亟待解决，而在线教育为此提供了新的解决方案。借助互联网，偏远地区的学生能够方便地获取来自一线城市和知名高校的优质课程，从而提升整体教育水平。在用户体验与满意度方面，研究表明在线教育平台的设计和内容呈现方式直接影响学习者的满意度。用户界面的友好性、课程内容的趣味性以及互动性都会对学习者的积极性和参与度产生影响。良好的用户体验不仅能提升学习者的满意度，还能增强他们的学习动力和效果。因此，越来越多的在线教育平台开始重视用户体验设计，通过调研和反馈不断优化平台的功能与内容。

此外，政策和法规的制定也在不断跟进。随着在线教育的快速发展，国家和地方政府逐渐认识到相关政策的重要性，以保障行业的健康发展。相关研究分析了国家政策在在线教育中的支持与监管，探讨如何在维护用户利益的同时，推动行业创新与发展。政策的引导与支持为在线教育平台提供了更广阔的发展空间，也为教育公平的实现提供了制度保障。

最后，通过对成功在线教育平台的案例分析，研究者总结了其在运营模式、课程设计和市场推广等方面的经验和教训。例如，一些在线教育平台通过营造社区化学习氛围，增强学习者之间的互动，取得了显著成效。这些实践研究为未来在线教育平台的发展提供了重要的参考和借鉴，有力推动了整个行业的创新与发展。

总之，在线教育平台的研究现状反映了中国教育行业转型的关键阶段。在线教育不仅是技术和市场的变革，更是教育理念和实践的深度创新。随着研究的不断深入，未来的在线教育将更加注重个性化、互动性和教育公平，为更广泛的学习者提供优质的教育服务。

## 1.3 论文主要研究内容

本文针对在线教育平台平台进行设计与实现，在实现的过程中主要工作内容如下：

（1）在网络上收集在线教育平台相关文献和资料，总结现有在线教育平台的优势与不足。

（2）对已有在线教育平台进行分析，确认网站将使用Vue.js、Element-Plus、SpringBoot2、Mybatis、Redis、MySQL数据库来实现。

（3）研究并分析在线教育平台的需求，将该平台分为6个模块：用户模块、课程模块、章节模块、评论模块、视频模块和图片上传模块。

（4）针对设计框架，该平台采用了MVVM架构，实现前后端分离。

（5）详细设计该平台涉及的数据库内容，主要包括用户数据表、学生课程表、学生作业表、课程信息数据表、评论数据表、章节数据表、章节视频表、分类表、作业表。

（6）测试平台各功能，排除问题，列举重要测试结果。

（7）分析并总结测试结果，找出不足，提出可改进方向。

## 1.4 论文组织结构

论文主体由七个章节组成，下面对每一个章节的主要内容作一个简单的说明。

第一章 绪论，本章讨论在线教育平台的研究背景及其重要性，概述国内外相关领域的发展情况，明确本论文的研究主题，并提供章节组织结构的概览。

第二章 关键技术与开发环境，本章首先介绍实现在线教育平台所需的关键技术，包括 Vue.js、Element-Plus、Spring Boot 2、MyBatis、Redis 和 MySQL。随后，简要描述该平台的开发环境。

第三章 系统需求分析，通过对在线教育平台的系统需求进行深入分析，本章阐述了平台的功能性需求与非功能性需求。

第四章 系统概要设计，本章从系统模块设计和架构设计的角度出发，提供在线教育平台的整体设计。接着，进行数据库的整体与详细设计，列出相关数据表；并针对用户界面设计了颜色、字号及段落格式；最后，完成了前端路由的设计工作。

第五章 系统详细设计，本章详细描述各模块的具体设计与实施方案，重点介绍每个模块所需实现的功能，并提供相应的时序图以展示模块间的交互。

第六章 系统开发与实现，本章涵盖主要功能的接口设计与核心代码，并展示平台部分功能的用户界面，体现系统的实际应用。

第七章 系统测试，本章对平台各项功能进行了测试，记录功能实现情况，并总结测试结果，以评估系统的稳定性与性能。

第八章 总结与展望，最后一章总结论文中所完成的工作，指出在线教育平台的不足之处，并提出未来改进的方向，以促进后续的研究与发展。

# 2 关键技术和开发环境

## 2.1 关键技术

本章将重点介绍在设计与实现在线教育平台过程中使用的关键技术。该平台采用了前后端分离的架构，并主要基于 MVVM 模式进行设计，以确保系统具有灵活性和可扩展性。

在前端开发中，平台主要运用 Vue 3 和 Element Plus。Vue 3 的响应式特性和组件化结构极大地提升了开发效率，同时减少了代码的重复编写。借助 Vite 构建工具，开发体验得到了优化，支持快速热重载，从而提升了开发过程中的响应速度。

后端方面，采用了 Spring Boot 框架，充分利用其自动配置和开箱即用的特性，迅速搭建了微服务架构。通过 Spring Boot Starter Mail 组件，平台能够高效地处理用户通知、注册确认等邮件发送功能。

在数据存储方面，平台选用了 MySQL 数据库来管理结构化数据，确保了数据的一致性与完整性。同时，MyBatis 框架的使用简化了数据库操作，使得可以通过自定义 SQL 查询满足特定的业务需求。

为提升系统性能，平台采用 Redis 进行数据缓存，从而加快用户访问的响应速度。此外，借助 JWT 实现了安全的用户身份验证，确保用户信息的安全性。

### 2.1.1 MySQL 技术

MySQL 是一款广泛使用的开源关系型数据库管理系统，以其高效的性能和可靠的数据管理能力著称。它支持标准的 SQL 查询语言，能够有效存储和管理在线教育平台中的用户信息、课程数据和章节信息等结构化数据，确保数据的一致性与完整性。

### 2.1.2 MyBatis技术

MyBatis 是一款支持定制化 SQL、存储过程以及高级映射的持久层框架。它提供了对数据库操作的灵活控制，使开发者能够编写复杂的 SQL 查询并将结果映射到 Java 对象上。MyBatis 简化了数据库交互的复杂性，特别适合需要细粒度控制 SQL 的应用场景。

### 2.1.3 Redis技术

Redis 是一个开源的内存数据结构存储系统，作为高性能的键值数据库，广泛用于缓存、会话存储和实时数据处理。在在线教育平台中，Redis 可用于存储热点数据，快速响应用户请求，提高系统的整体性能和可扩展性。

Node.js服务器性能好，处理业务能力强。

### 2.1.4 JWT (JSON Web Token)技术

JSON Web Token (JWT) 是一种开放标准（RFC 7519），用于安全地在各方之间传输信息。它通过对用户身份进行编码，确保信息的真实性与完整性，广泛应用于用户身份验证和信息交换。JWT 使得在线教育平台能够在无状态的环境中实现安全和可扩展的用户认证。

### 2.1.5 Vue 3

Vue 3 是一款现代化的渐进式 JavaScript 框架，专注于构建用户界面。它采用了响应式的数据绑定和组件化开发方式，提供了灵活性和高效性，使得开发者能够快速构建出可维护的应用。Vue 3 的 Composition API 提供了更强的逻辑复用能力，适合大型应用的开发。

### 2.1.6 Vite

Vite 是一个快速的前端构建工具，致力于提升开发体验。其主要特点是即时热重载（HMR），能够在代码修改后快速反映到浏览器中，极大地提高了开发过程中的效率。Vite 支持现代 JavaScript 特性，能够根据项目需求进行按需编译和构建，适合现代前端开发。

### 2.1.7 Element Plus

Element Plus 是基于 Vue 3 的高质量 UI 组件库，专为开发者提供了一系列丰富的、易于使用的组件。其设计风格简洁美观，能够快速构建出用户友好的界面。Element Plus 具备良好的文档支持，帮助开发者高效使用组件，提高开发速度。

## 2.2 开发环境

在线教育平台的运行环境是Windows11，浏览器为Google Chrome，开发环境为IDEA 数据库环境为Linux 部署环境。

# 3 系统需求分析

## 3.1 功能性需求

此系统包括供用户使用的账号模块、课程模块、章节模块、博客模块、评论模块和为老师提供的课程编辑模块和用户管理模块和作业模块和仪表盘模块。其中，账号模块中有注册、登录功能。课程模块提供课程的展示、搜索和分类功能。用户可以浏览课程的详细信息、课程大纲和课程视频，支持按热门课程、推荐课程等排序。章节模块每个课程下包含多个章节，章节内可以有多节课或视频。用户可以按顺序学习章节内容，系统支持进度记录和回看功能。博客模块, 用户和教师可以撰写博客文章，分享学习心得、教学资源或讨论相关话题。评论模块, 为课程、博客文章等内容提供评论功能，支持多层级的嵌套评论。用户可以对评论进行回复，以提高互动性。课程编辑模块即为老师提供了删除和修改,查看用户课程信息的功能。 用户管理模块，教师可以查看学生的基本信息、学习进度和作业提交情况。作业模块，教师可以为课程设置作业，并定义作业的截止日期、评分标准和提交要求。学生可以提交作业，并查看评分和反馈。仪表盘模块，可以查看课程数据，用户数据清晰直观

平台整体用例图如图3-1所示。

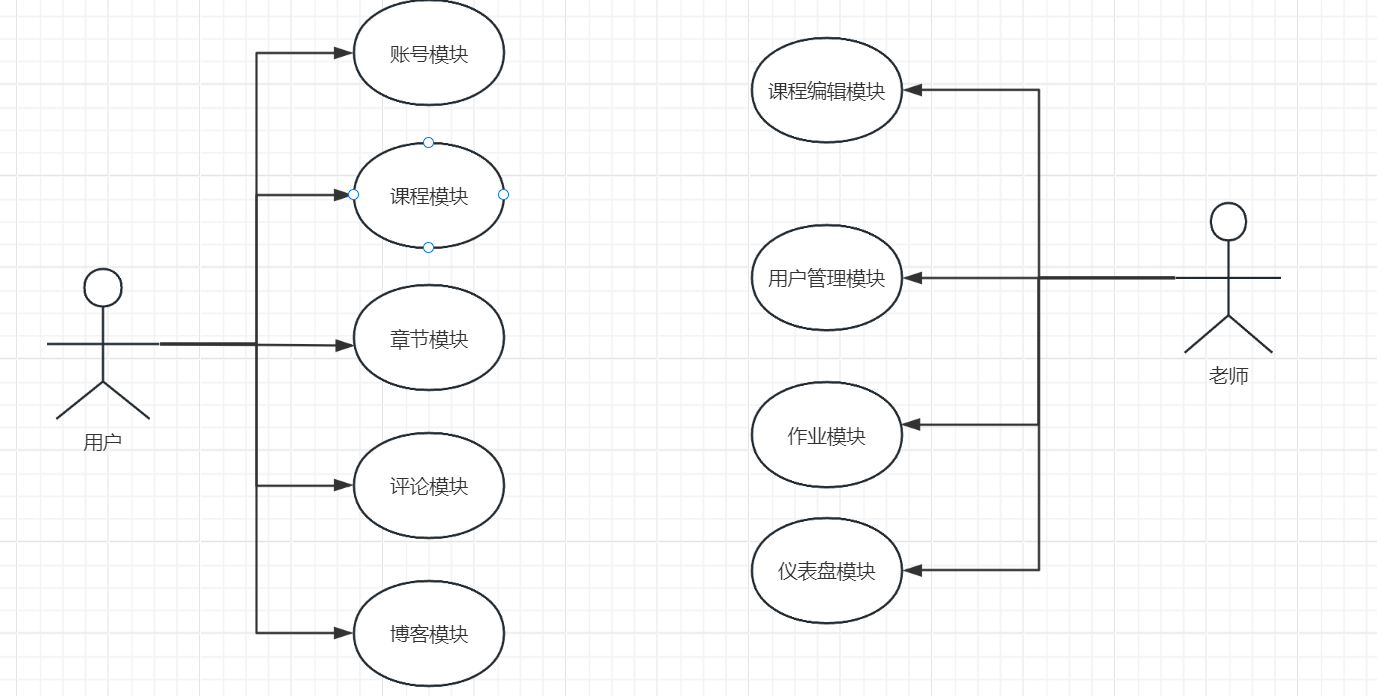


图3-1 平台整体用例图

### 3.1.1 账号模块

在账号模块中，用户可以进行注册、登录和查看订单。其中，注册是后两者的前提，在完成大部分操作之前，用户须进行注册并保持登录状态。

1、注册

功能包含：用户在登录页面点击立即注册，跳转到注册页面，通过填写用户名、邮箱、用户类型和8-12位由字母或数字构成的密码，即可完成注册。

2、登录

功能包含：已注册用户在登录页面正确填写注册所使用的邮箱和登录密码，点击登录按钮后即可完成登录。老师在登录页面中，进行同上操作即可完成登录。用户或老师完成登录后，网页将记录登录状态。

账号模块的用例图如图3-2所示。

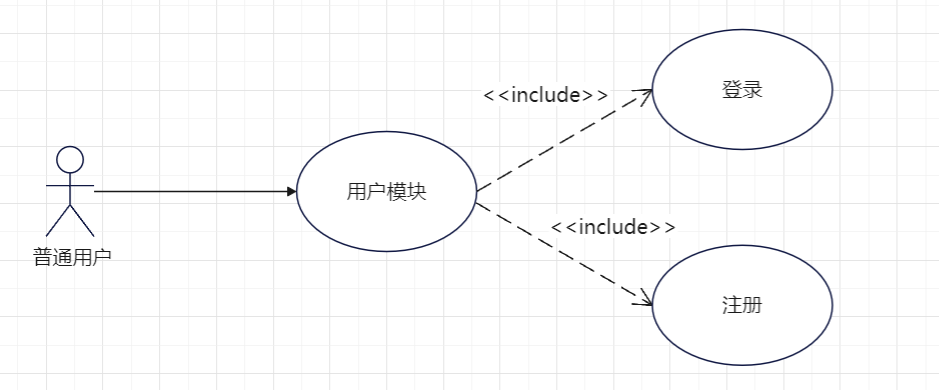


图3-2 账号模块用例图

### 3.1.2 课程模块

在课程模块中，用户可以浏览课程、查看课程详情和进行课程学习，用户还可以收藏课程和提交课程评价。

1、浏览课程

功能包含：所有用户可以在课程页面浏览不同类别的课程，课程按照分类展示，用户可以根据需要选择查看指定分类的课程。课程列表展示课程名称、讲师名称、简介、评分等信息。

2、查看课程详情

功能包含：用户点击某个课程后，跳转到课程详情页面。课程详情页面展示课程的详细信息，包括课程简介、课程大纲（章节列表）、评分、用户评价等。

3、学习课程

点击课程详情页的“开始学习”按钮进入课程学习页面。课程学习页面包含视频播放、章节导航等功能。

课程模块用例图如图3-3所示

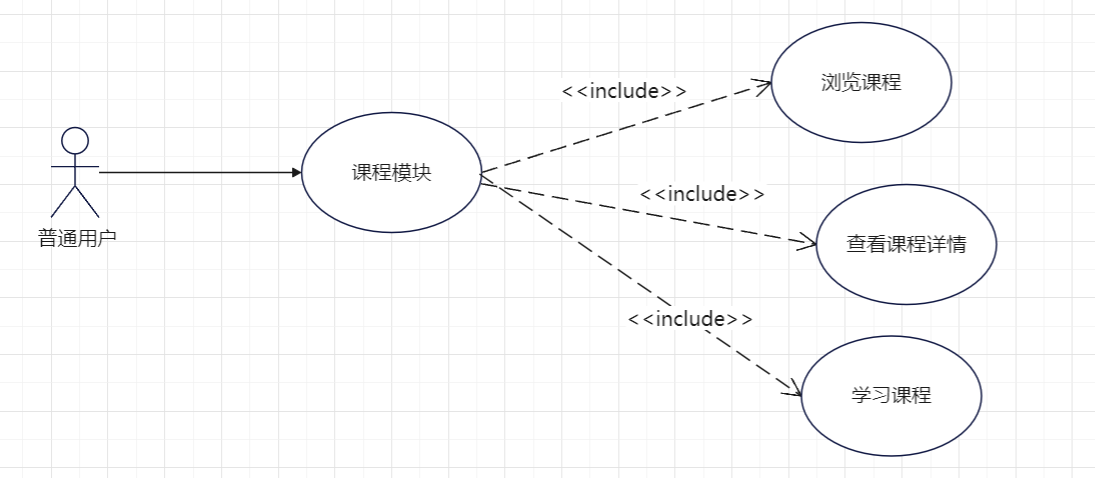


图3-3 商品模块用例图

### 3.1.3 章节模块

1、查看章节列表

功能包含：用户在课程详情页面可以看到课程的章节列表，章节按照课程大纲的顺序排列。每个章节显示章节标题、时长和简要介绍。

2、学习章节

功能包含：用户点击某个章节后，进入章节学习页面，播放该章节的视频内容。页面还显示章节的详细介绍，包含学习目标和重点内容。用户可以在播放视频时使用章节导航功能快速跳转到其他章节。

章节模块用例图如图3-4所示。

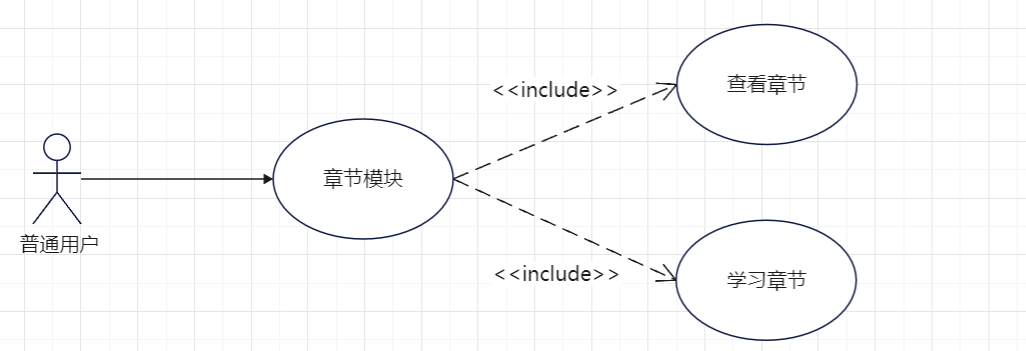


图3-4 章节模块用例图

### 3.1.4 评论模块

1、查看评论列表

功能包含：用户可以在课程详情页面或章节页面查看其他用户的评论。

2、提交评论

功能包含： 已登录用户可以在课程详情页或章节页面提交评论。

3、回复评论

功能包含：已登录用户可以在评论列表中选择回复他人的评论。回复内容将展示在原评论下方，形成评论的树状结构，便于查看讨论的上下文。

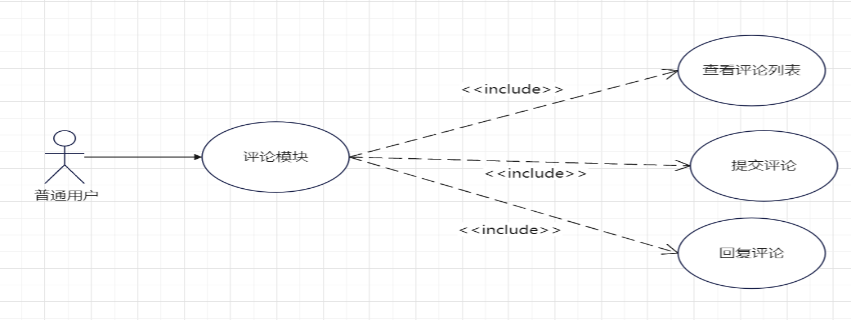
评论模块用例图如图3-5所示

图3-5 评论模块用例图

### 3.1.5 博客模块

1、查看博客文章

功能包含：所有用户可以在博客页面浏览平台内的博客文章。文章列表按照发布时间、点赞数或浏览量排序，用户可以选择分类标签来筛选感兴趣的文章。

2、发布博客文章

功能包含：已登录用户可以发布博客文章，内容包括标题、正文、分类标签和封面图片等。

博客模块用例图如图3-6所示。

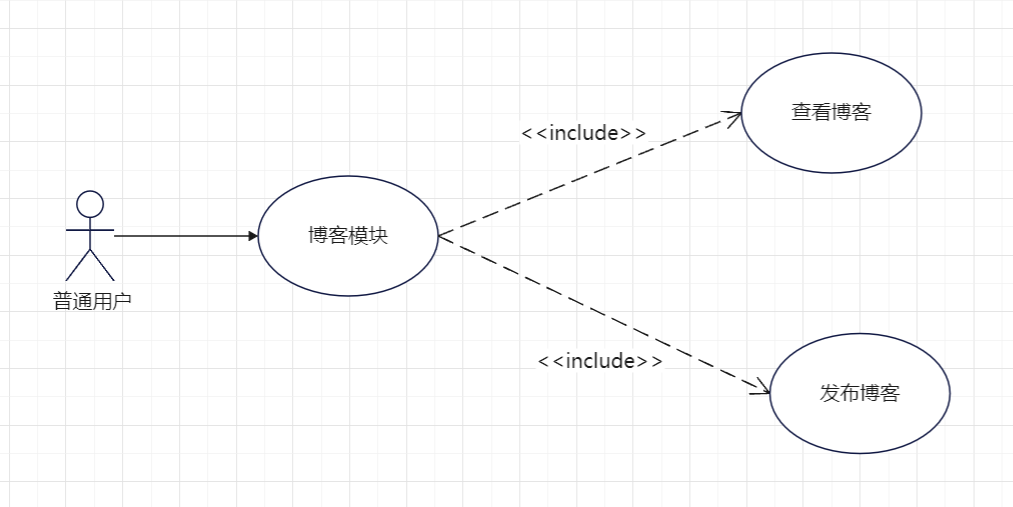


图3-6 章节模块用例图

### 3.1.6 课程编辑模块

1、创建课程

功能包含：具有老师管理权限的用户可以创建新课程。创建时需填写课程的基本信息，包括课程标题、简介、讲师信息、分类标签、课程封面图片等。课程创建后，用户可以继续编辑课程的详细信息和章节内容。

2、编辑课程

功能包含：用户可以修改已有课程的信息，如课程标题、简介、封面图片、分类标签和讲师信息等。用户还可以对课程大纲进行调整，包括添加、编辑或删除章节。课程的内容和结构可根据实际教学需求进行灵活调整。

3、删除课程

功能包含：用户可以将不再需要的课程删除。删除操作会将课程从平台中移除。删除课程时，会提示用户确认操作，以避免误删

4、管理章节

功能包含：在课程编辑页面，用户可以对课程的章节进行管理，如添加章节，编辑章节，删除章节。

课程编辑模块用例图如图3-7所示。

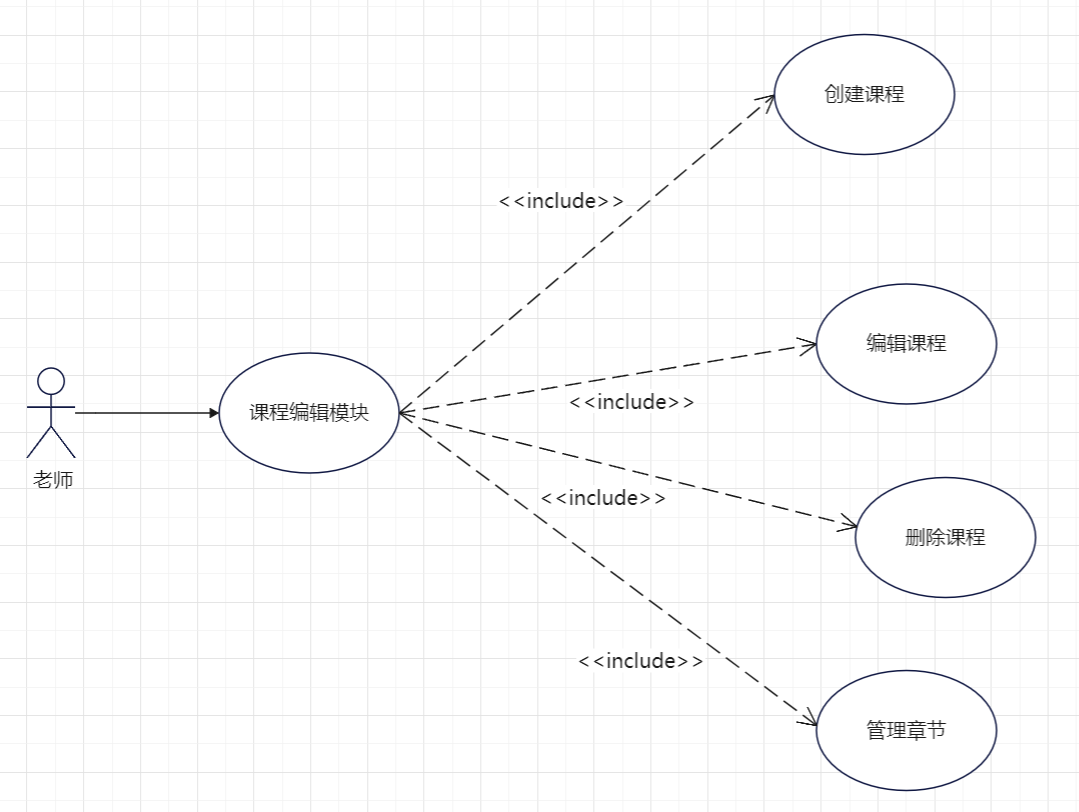


图3-7 课程编辑模块用例图

### 3.1.7 用户管理模块

1、查看学生列表

功能包含：老师可以查看所有学生的列表，列表展示学生的基本信息，如学生姓名、学号、邮箱、注册时间和课程进度等。老师可以通过搜索框根据学生姓名进行搜索，以快速找到特定学生。

2、查看学生详情

功能包含：老师可以点击学生列表中的某个学生，查看该学生的详细信息。

3、给学生发送消息

功能包含：老师可以通过系统向学生发送消息或通知，告知学生课程安排、作业要求、学习建议等。该功能有助于促进师生之间的沟通和互动。

4、修改学生信息

功能包含：老师可以编辑学生的基本信息，如更新邮箱或更改课程状态等。修改信息后，系统会保存更改并反馈操作成功信息。

用户管理模块用例图如图3-8所示。

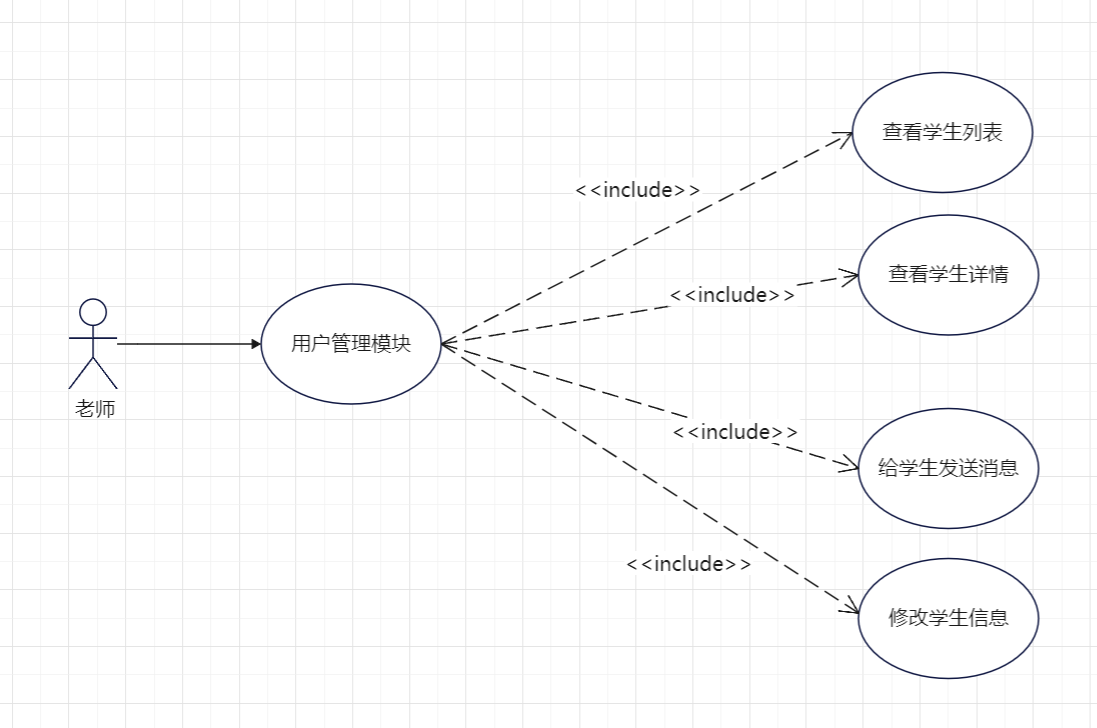


图3-8 用户管理模块用例图

### 3.1.8 作业模块

1、发布作业

功能包含：教师可以向学生发送关于新发布作业。

2、批改作业

功能包含：教师可以查看学生提交的作业并进行批改。

3、发布作业通知

功能包含：教师可以向学生发送关于新发布作业的通知。

作业模块用例图如图3-9所示。

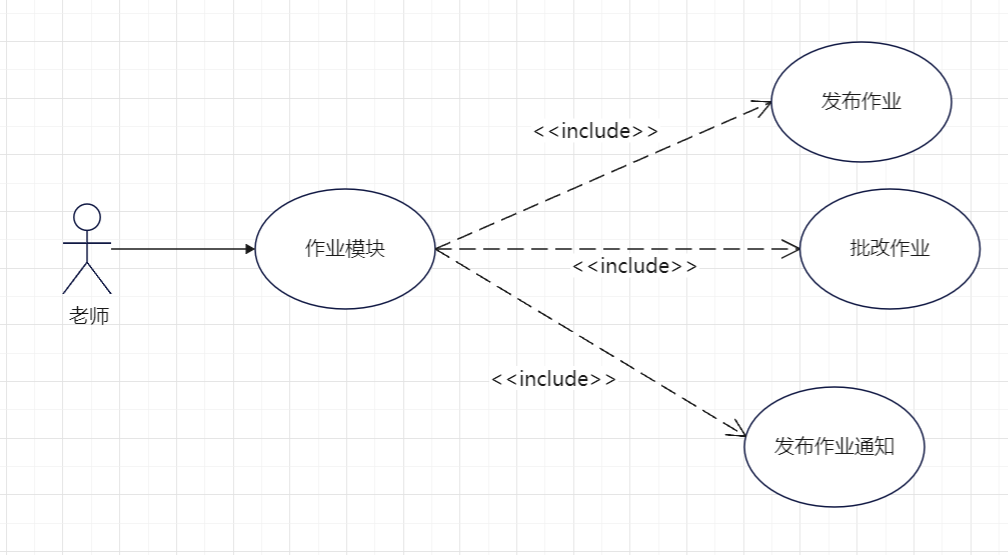


图3-9 作业模块用例图

### 3.1.9 仪表盘模块

1、概览统计

功能包含：用户可以查看整体的课程和学生数据概览。

2、作业完成情况

功能包含：用户可以查看作业的提交和完成情况。

3、图表和图形展示

功能包含：教师可以查看可视化的图表和数据。

仪表盘模块用例图如图4-0所示。

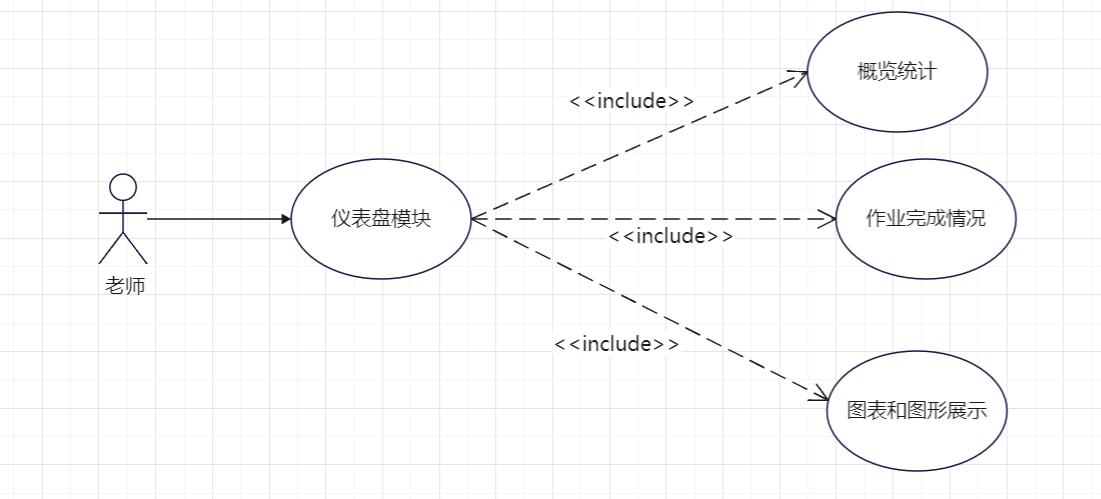


图4-0 仪表盘模块用例图

## 3.2 非功能性需求

### 3.2.1 性能需求

（1）响应时间：用户在进行任何操作（如注册、登录、课程浏览、视频播放）时，系统响应时间应不超过2秒，页面加载时间应不超过3秒，且在高负载情况下也能保持在此范围内。

（2）数据处理能力：系统应能在6秒内处理用户提交的作业和评论，并实时更新相关数据。

（3）缓存策略：系统应使用缓存机制（如 Redis），以减少数据库查询频率，提高数据访问速度。

（4）资源使用率：系统应优化资源使用，CPU和内存利用率应保持在合理范围内（如 CPU < 80%，内存 < 75%）。

### 3.2.2 安全性需求

（1）用户权限：用户仅能使用用户权限，老师能使用老师相关权限，只能在权限范围内进行操作。

（2）提供不同登陆方式：用户使用邮箱进行登录，老师使用被派发的账号进行登录。

（3）定期进行安全审计和漏洞扫描，以识别和修复潜在的安全风险。

（4）所有用户数据（如密码、个人信息）应使用加密存储和传输

### 3.2.3 可维护性需求

（1）代码可读性：所有代码应遵循统一的编码规范，确保易于阅读和维护。

（2）模块化设计：系统应采用模块化设计，各个功能模块应独立，便于后续扩展和维护。

（3）错误处理：系统应提供统一的错误处理机制，确保错误信息对用户友好且便于开发人员排查。

3.2.4 监控与日志需求

（1）实时监控 系统应实施实时监控，跟踪关键性能指标和用户行为，以便及时发现和处理问题。

（2）日志记录 系统应记录用户活动和系统事件的日志，以便后续审计和分析。日志应包括详细的信息（如用户ID、操作类型），便于故障排查和用户行为分析

### 3.2.5 可用性需求

（1）简化注册流程：注册流程应简洁，用户只需填写必要的信息（如用户名、邮箱和密码），且能够提供错误提示以指导用户完成注册。

（2）界面设计：用户界面应直观且易于导航，确保用户能够轻松找到所需的功能和信息。

（3）计划停机时间：计划外的系统停机时间应限制在每月不超过1小时，计划内维护时应提前通知用户，避免影响使用。

### 3.2.6 可维护性需求

（1）BUG修正时间 80%的BUG修正应在2个工作日内完成，其余BUG修正应在7个工作日内完成。

（2）数据库一致性 版本更新后，数据库中的个人信息应保持不变。

## 3.3 本章小结

本章节对在线教育平台进行了全面的需求分析，从用户和教师两个角色出发，详细阐述了系统的功能需求，如课程浏览、注册、学习管理以及教师的课程发布和作业管理。同时，分析了非功能性需求，包括性能、安全性、数据保密、易用性、可用性和可维护性需求，确保系统能够在高并发情况下稳定运行，保护用户数据，提供良好的用户体验，并便于后续维护与升级。这些分析为平台的设计与实现设定了明确的目标，确保满足各类用户的需求。以此为该平台的实现设定了目标。

# 4 系统概要设计

## 4.1 系统总体设计

### 4.1.1 系统模块设计

在线教育平台的系统设计经过细致的需求分析，划分为九个主要模块：账号模块、课程模块、章节模块、博客模块、评论模块、课程编辑模块、用户管理模块、作业模块和仪表盘模块。每个模块都承担着特定的功能，以提升用户体验并支持教师与学生之间的互动。

账号模块是在线教育平台的基础功能，主要负责用户的注册、登录和个人信息管理。普通学生和教师可以通过填写用户名、邮箱和密码进行注册，系统会发送验证邮件以确保邮箱的有效性。登录功能允许用户通过邮箱和密码快速访问平台，系统还提供了记住密码的选项，增强用户便利性。用户可以随时查看和编辑个人信息，包括昵称、头像和学习历史。。

课程模块是在线教育平台的核心部分，旨在为用户提供丰富的课程内容。教师可以创建新课程，填写课程标题、简介、讲师信息、分类标签和封面图片等基本信息。此外，用户能够搜索和浏览课程，以便选择适合自己的学习资源。课程模块还支持课程的发布和下架，教师可以根据需要灵活管理课程。

章节模块为每门课程提供结构化的内容管理功能。教师可以在课程内添加、编辑或删除章节，以便更好地组织和展示课程材料。每个章节可以包含详细的学习内容、视频讲解和相关的学习资源，帮助学生更清晰地掌握课程的结构和进度。

博客模块允许教师和学生分享学习经验和心得体会。用户可以撰写博客文章，记录学习过程中的思考和感悟。其他用户可以对博客进行评论和点赞，促进知识的交流和分享。该模块为用户提供了一个互动的平台，增强了学习社区的活跃度。

评论模块为用户提供了对课程进行评价和反馈的渠道。学生可以对已注册课程进行星级评分和文字评论，帮助其他学生更好地了解课程质量。教师可以查看学生的评论，了解课程的优缺点，并根据反馈调整教学内容和方法。

课程编辑模块为教师提供了创建和管理课程的功能。教师可以在此模块中创建新课程，编辑课程信息，管理课程章节和内容。此模块支持教师灵活调整课程结构，以满足教学需求并优化学习效果。

用户管理模块主要用于管理学生和教师的账户信息。教师可以查看和修改自己的用户信息，确保资料的准确性。系统还允许教师管理学生的参与情况，例如查看学生的学习进度和作业提交情况，以便给予相应的指导和支持。

作业模块是教师与学生互动的重要部分。教师可以为每门课程发布作业，设定作业内容、截止日期和提交要求。学生可以在截止日期前上传作业，教师则可以查看作业并进行批改、评分及反馈。这一过程确保学生能获得及时的指导和评估。

仪表盘模块为教师和学生提供了数据分析和学习情况的监控功能。教师可以查看课程的整体统计数据，包括注册学生人数、课程参与率和作业完成情况等信息。学生则可以在仪表盘中跟踪自己的学习进度、完成的课程和作业情况，帮助他们制定更有效的学习计划。

在线教育平台的功能模块拆分图如图4-1所示。

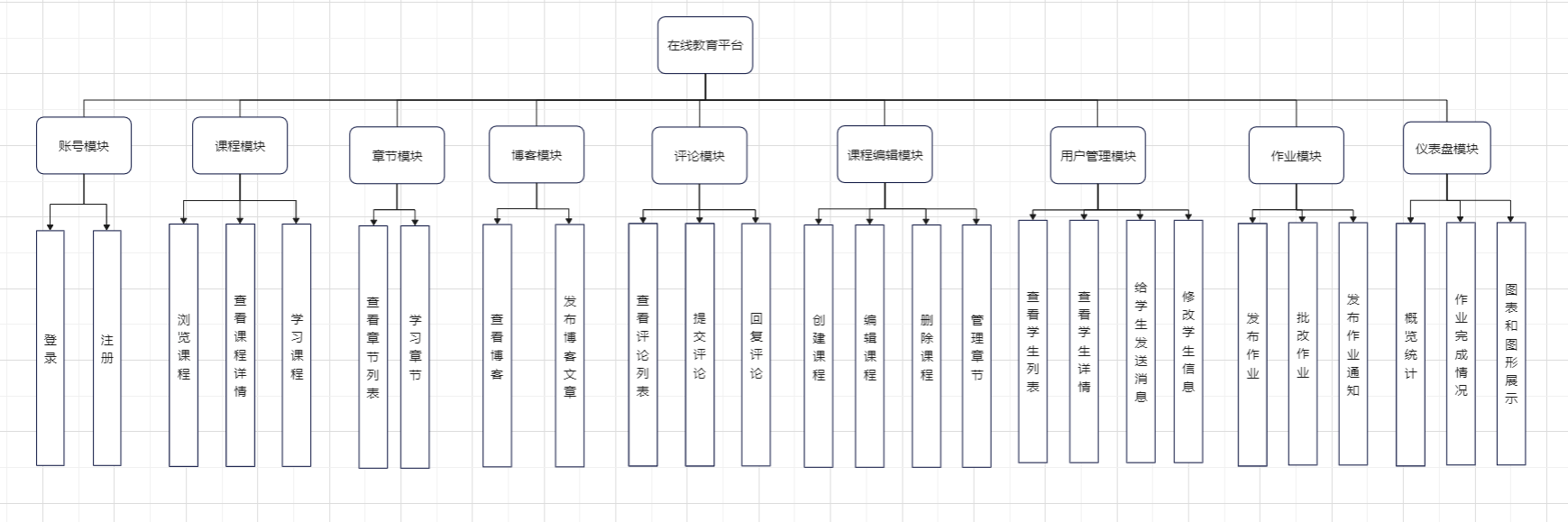


图4-1 在线教育平台的功能模块拆分图

### 4.1.2 系统架构设计

根据第三章节对需求进行分析并结合实际情况，本系统将在线教育平台划分为前端UI层、后端处理层、数据层和数据存储层共四个层次。

1、前端UI层

该层主要使用Vue框架为学生以及老师提供逻辑清晰、易操作性强的交互界面。学生用户及老师用户可根据各自角色的不同选择相应相对应的页面进入。

2、后端处理层

通过Ajax技术向服务器发送请求，采用后端拦截器进行权限验证，根据用户角色权限为用户提供相应的业务服务。为提升代码复用性和处理并发请求的性能，使用Spring Boot框架搭建后端业务，结合MySQL数据库存储业务数据，MyBatis进行数据层的访问和映射，并使用Redis缓存提高查询性能和响应速度。后端通过Service层向数据层发出命令，确保业务逻辑的实现和数据的一致性。

3、数据层

为降低高并发时直接访问数据库的压力，系统会先尝试从Redis缓存获取数据，如果命中缓存，就直接返回结果；若未命中，则从MySQL中查询，并将结果存入缓存，以加快后续请求的响应速度。借助MyBatis实现对数据库的操作，简化增删改查的过程，并通过存储过程优化复杂数据查询的执行效率。数据层专注于数据管理，结合缓存机制显著提升系统的性能和用户请求的处理速度。

4、数据存储层

数据信息采用MySQL数据库进行存储，可在后端直观展示。

平台的系统架构设计图如图4-2所示。

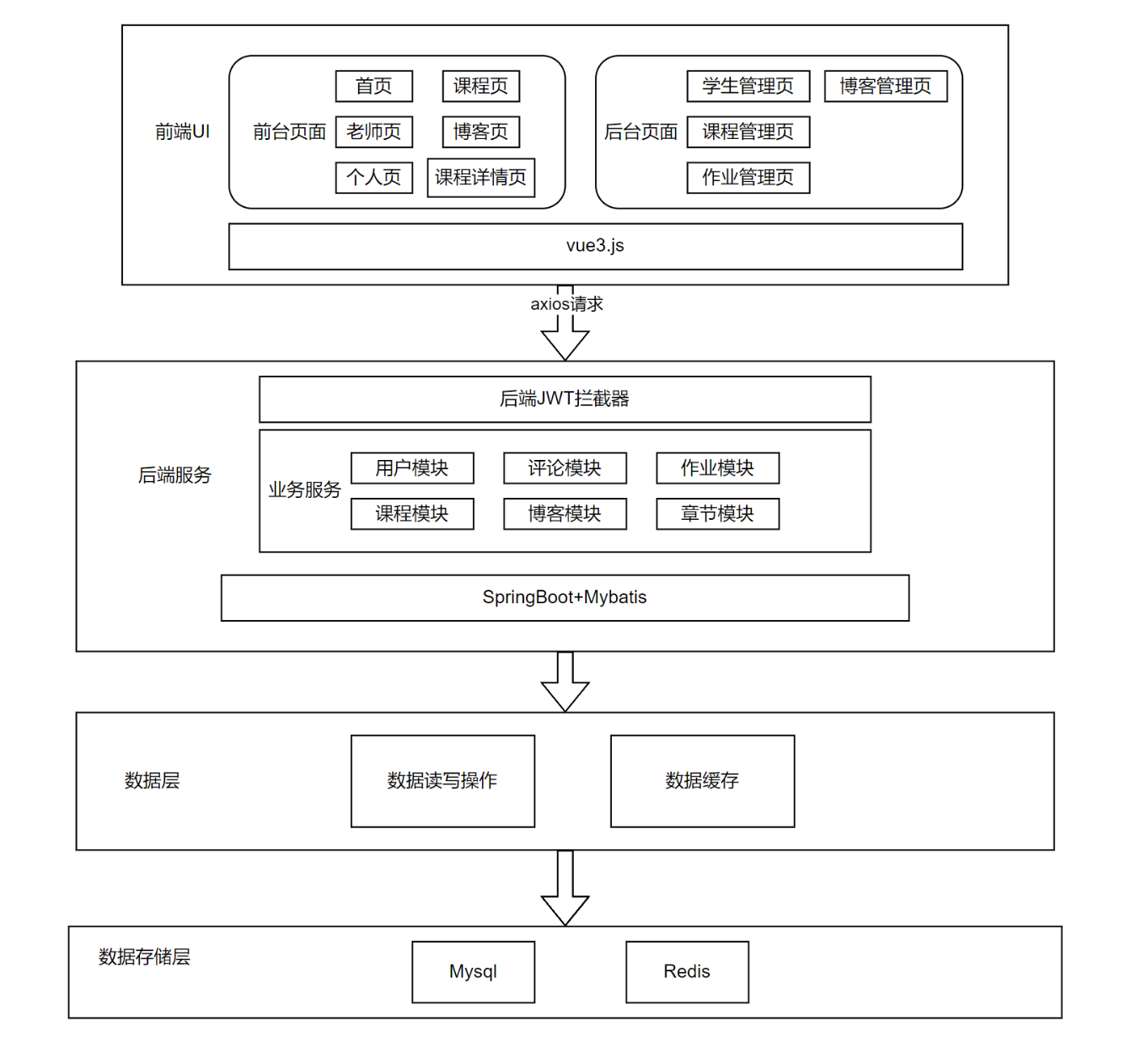


图4-2 系统架构设计图

## 4.2 数据库设计

将系统总体设计完善后，再设计数据库。本章节主要分别介绍了从数据库整体设计与数据库详细设计。

### 4.2.1 数据库整体设计

根据对该在线教育平台的研究，划分了用户、管理员、订单、购物车、商品共五个实体。

1、用户

用户实体，存储用户的基本信息，包括用户ID、用户名、邮箱、密码、用户类型（学生、教师）、用户标签、用户描述、以及头像。用户的E-R图如图4-3所示。

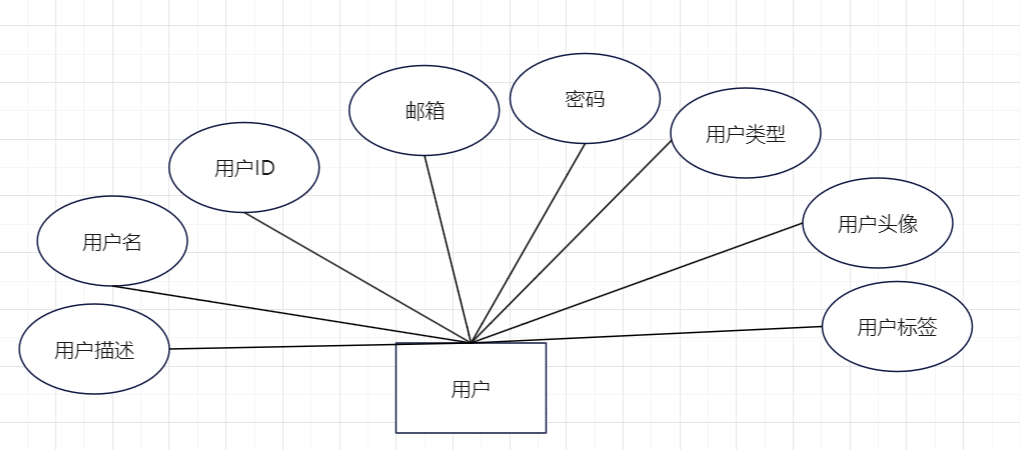
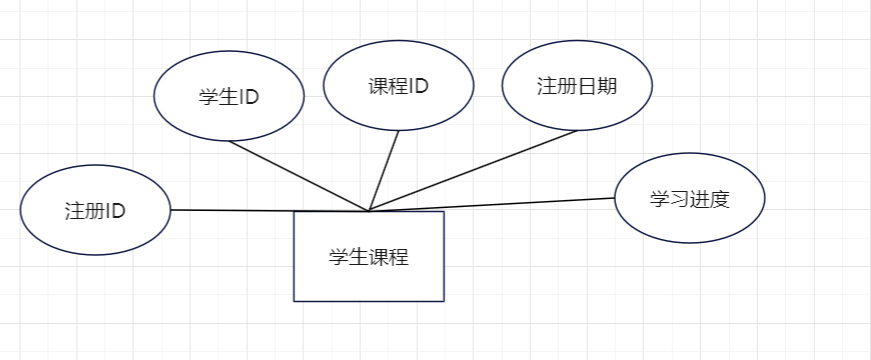


图4-3 用户的E-R图

2、学生课程

在学生课程实体中，用于记录学生的课程注册情况，包括注册ID、学生ID、课程ID、注册日期和学习进度。管理员的E-R图如图4-4所示。



**图**4-4 学生课程的E-R**图**

3、学生作业

学生作业实体中，记录学生提交的作业信息，包括提交ID、学生ID、作业ID、提交链接、提交日期、评分和教师反馈。订单的E-R图如图4-5所示。

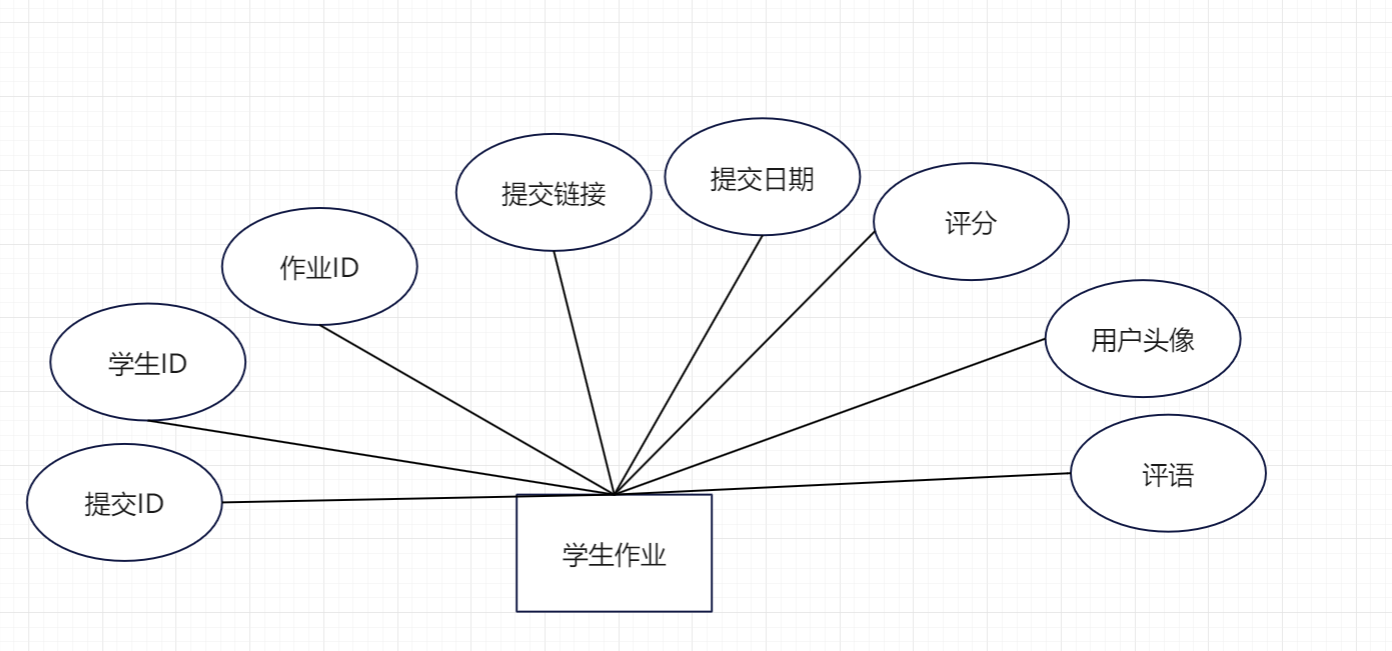


图4-5 学生作业的E-R图

4、论坛

论坛表实体中存储用户创建的讨论主题及其内容，包括讨论主题ID、用户ID、标题、内容和创建日期。购物车的E-R图如图4-6所示。

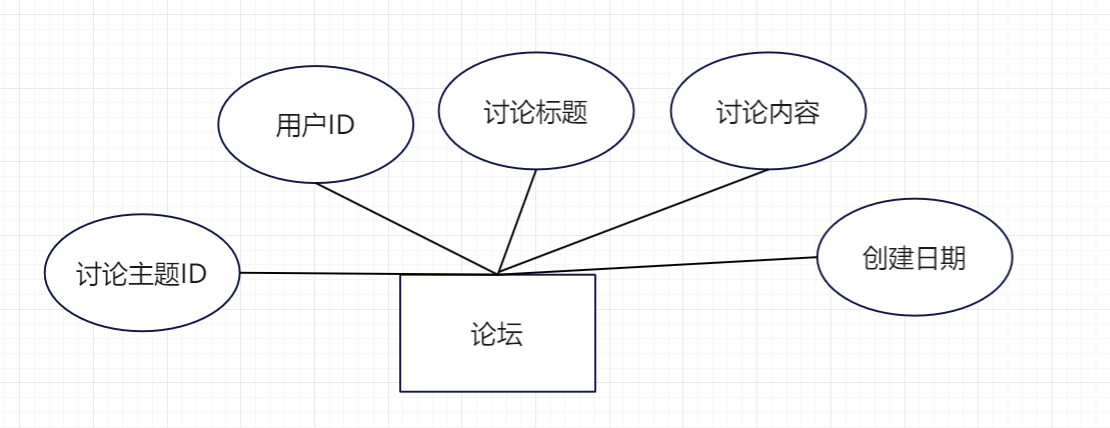
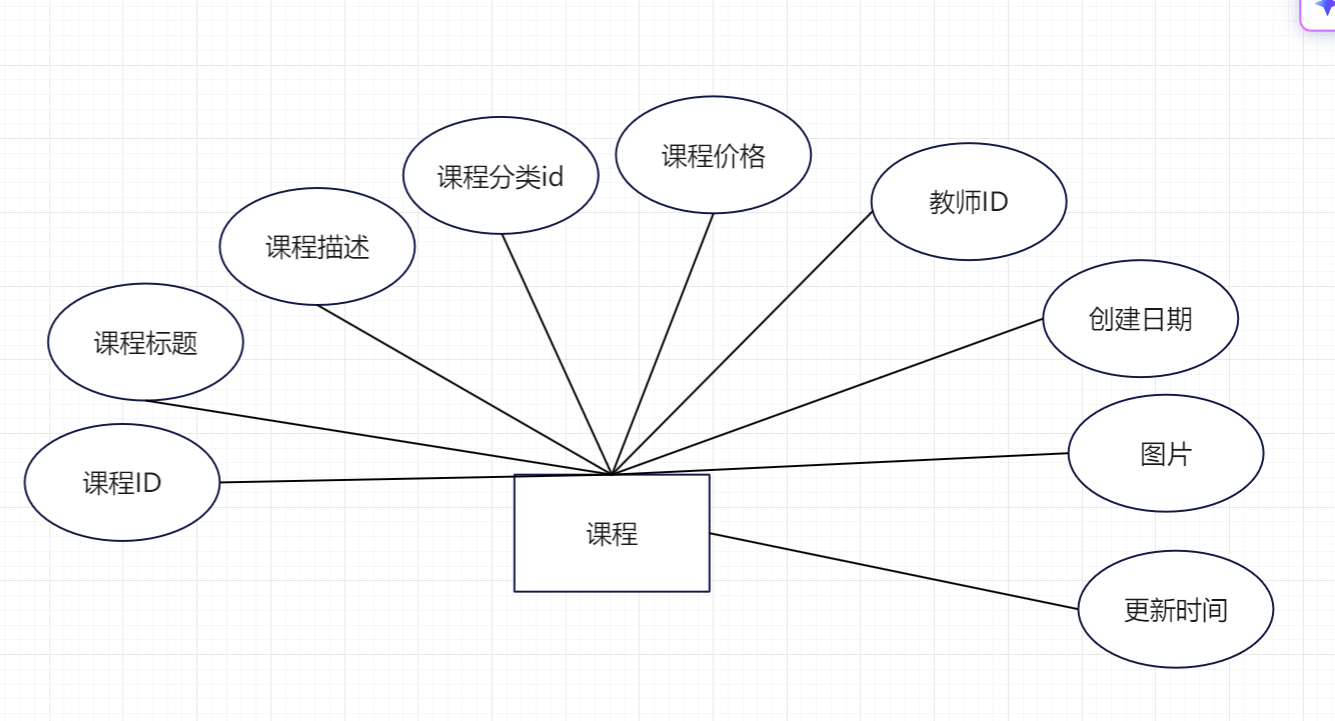


图4-6 论坛的E-R图

5、课程

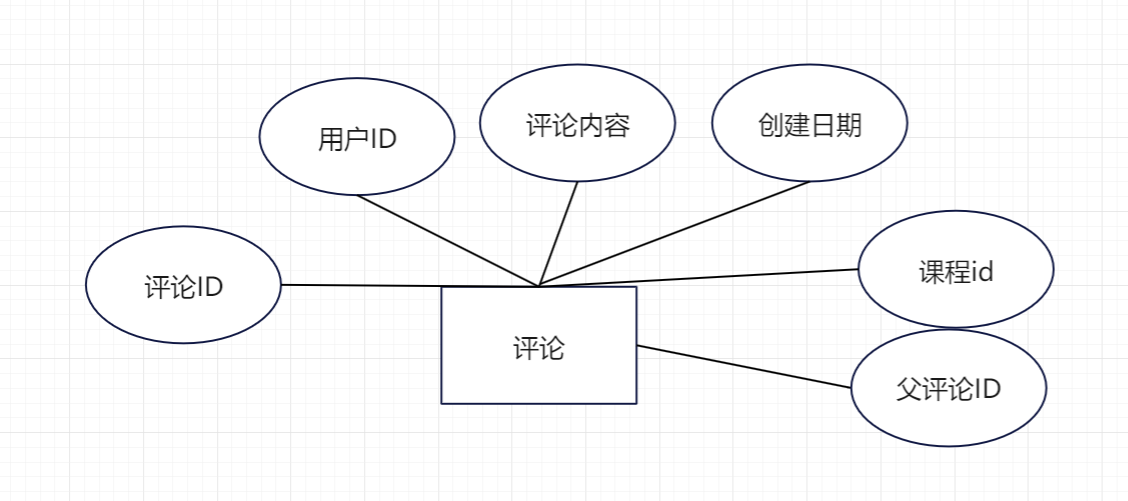
课程实体的属性主要有：包含课程的详细信息，如课程ID、标题、描述、分类ID、价格、教师ID、创建日期、更新时间、以及课程图片。商品的E-R图如图4-7所示。



**图4-7** 课程**的E-R图**

6、评论

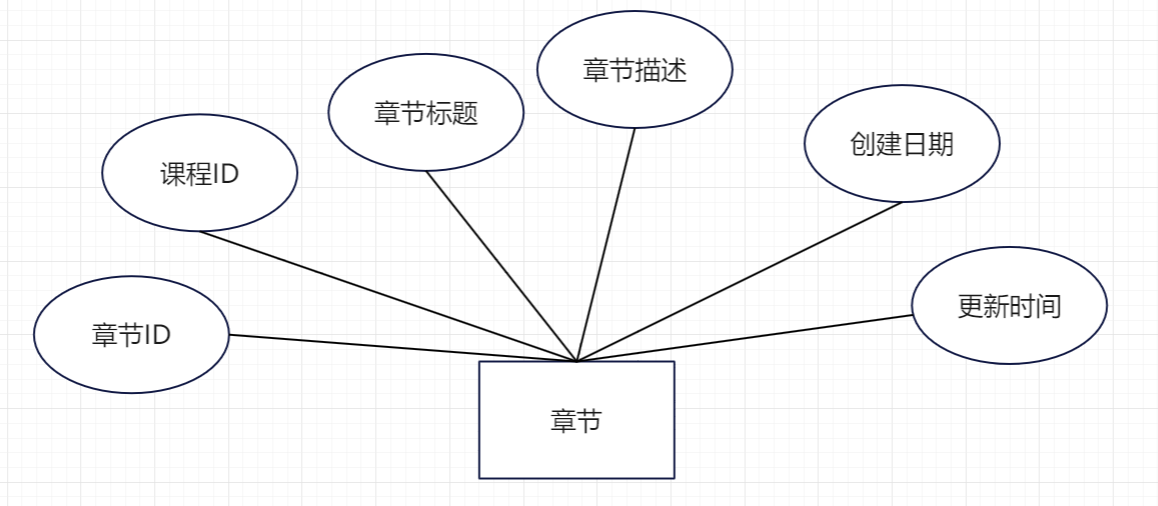
评论实体的属性主要有：记录用户对课程的评论信息，包括评论ID、用户ID、评论内容、创建日期、课程ID和父评论ID（用于表示评论的层级关系）。商品的E-R图如图4-7所示。



**图4-7** 评论**的E-R图**

7、章节

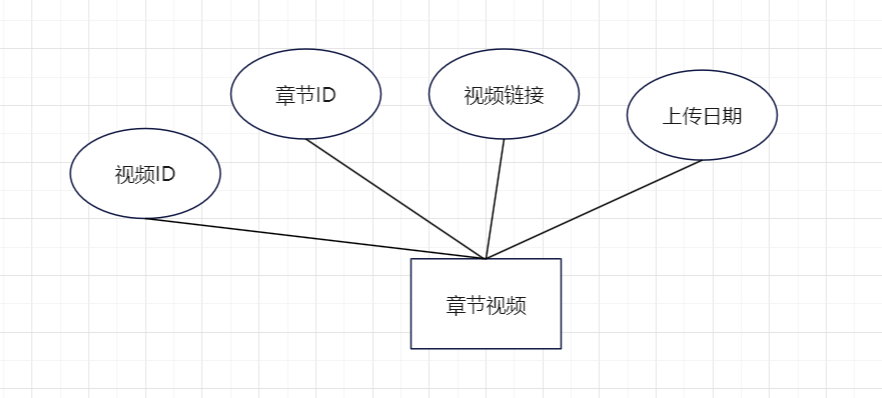
章节实体的属性主要有：存储课程的章节信息，包括章节ID、课程ID、章节标题、章节描述、创建日期和更新时间。商品的E-R图如图4-7所示。



**图4-7** 章节**的E-R图**

8、章节视频

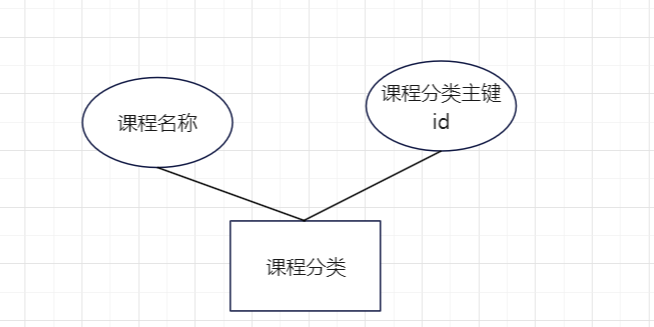
章节视频实体的属性主要有：记录课程章节相关的视频信息，包括视频ID、章节ID、视频链接、上传日期和视频编码。商品的E-R图如图4-7所示。



**图4-7** 章节视频**的E-R图**

9、课程分类

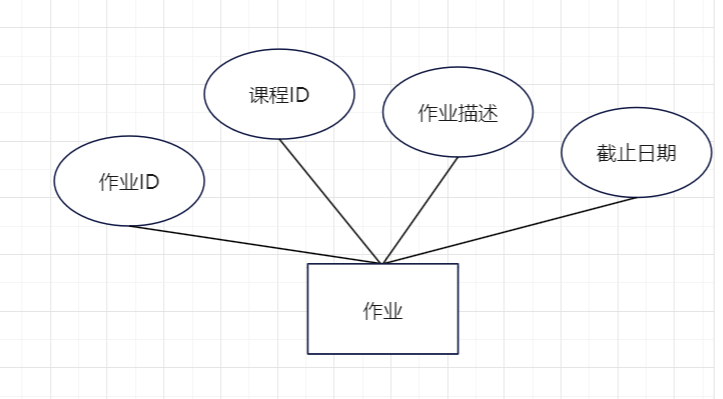
课程分类实体的属性主要有：包含课程分类的基本信息，如分类ID和分类名称。商品的E-R图如图4-7所示。



**图4-7** 课程分类**的E-R图**

10、作业

作业实体的属性主要有：记录课程的作业信息，包括作业ID、课程ID、作业描述和截止日期。商品的E-R图如图4-7所示。



**图4-7** 作业**的E-R图**

11、总体E-R图

将各实体的属性及关系进行整理：用户可以注册多个课程，而每个课程只能被多个用户注册。每个用户可以提交多个作业，而每个作业只能由一个用户提交。每个用户可以发起多个讨论主题，而每个讨论主题仅由一位用户创建。课程可以包含多个章节，而每个章节只能属于一个课程。每个章节可以包含多个视频，而每个视频只能属于一个章节。此外，课程可以有多个评论，而每个评论仅对应一个课程。最后，课程可以包含多个作业，而每个作业只能属于一个课程。

该平台总体E-R图如图4-8所示。

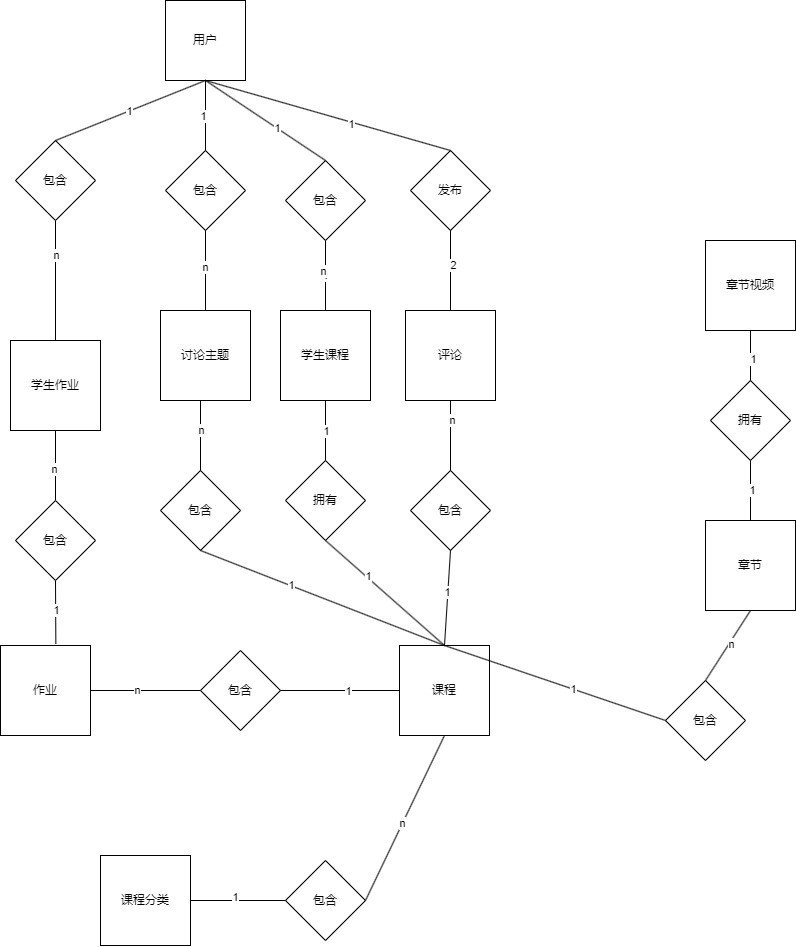


图4-8 平台总体E-R图

### 4.2.2 数据库详细设计

根据以上设计的E-R图分析可以得到：该数据库所包含的数据表主要有：用户数据表、课程数据表、讨论主题数据表、学生课程数据表、评论数据表、作业数据表、章节数据、章节视频数据表、学生数据表、课程分类数据表。本章节将对其进行详细说明。

1、用户数据表

用户数据表，主要包括以下字段：用户ID、用户名、邮箱、密码、用户类型、注册日期、更新时间、用户标签、用户描述和头像。用户ID是主键，用于唯一标识每个用户。邮箱字段设置为唯一，确保每个邮箱只能注册一个账号。

用户数据表结构如表4-1所示。

表4-1 用户数据表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 类型 | 属性 | 描述 |
| 1 | user\_id | INT | 主键 | 用户ID |
| 2 | username | varchar(50) | 否 | 用户名 |
| 3 | email | varchar(100) | 否 | 邮箱 |
| 4 | password | varchar(255) | 否 | 密码 |
| 5 | user\_type | enum ('student', 'teacher') | 否 | 用户类型 |
| 6 | created\_at | timestamp | 否 | 注册日期 |
| 7 | updated\_at | timestamp | 否 | 更新时间 |
| 8 | user\_tag | varchar(10) | 否 | 用户标签 |
| 9 | description | varchar(255) | 否 | 用户描述 |
| 10 | avatar | varchar(255) | 否 | 用户头像 |

2、学生课程表

记录学生选修的课程，主要包括以下字段：注册ID、学生ID、课程ID、注册日期和学习进度。注册ID是主键，学生ID和课程ID分别外键关联到用户表和课程表，确保数据的完整性。

学生课程表结构如表4-2所示。

**表4-1 购物车数据表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 类型 | 属性 | 描述 |
| 1 | registration\_id | INT | 主键 | 注册ID |
| 2 | student\_id | int | 外键 | 学生ID |
| 3 | course\_id | int | 外键 | 课程ID |
| 4 | registered\_at | timestamp | 否 | 注册日期 |
| 5 | progress | int | 否 | 学习进度 |

3、学生作业表

存储学生提交的作业信息，主要包括以下字段：提交ID、学生ID、作业ID、提交链接、提交日期、评分和评语。提交ID是主键，学生ID和作业ID外键关联到用户表和作业表。

学生作业表结构如表4-3所示。

表4-2 学生作业表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 类型 | 属性 | 描述 |
| 1 | submission\_id | INT | 主键 | 提交ID |
| 2 | student\_id | int | 外键 | 学生ID |
| 3 | assignment\_id | int | 外键 | 作业ID |
| 4 | submission\_url | varchar(255) | 否 | 提交链接 |
| 5 | submitted\_at | timestamp | 否 | 提交日期 |
| 6 | grade | float | 否 | 评分 |
| 7 | feedback | text | 否 | 评语 |

4、讨论论坛表

讨论论坛表，用于存储用户发起的讨论主题，主要包括以下字段：讨论主题ID、用户ID、讨论标题、讨论内容和创建日期。讨论主题ID是主键，用户ID外键关联到用户表。

讨论论坛表结构如表4-4所示。

表4-3 讨论论坛表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 类型 | 属性 | 描述 |
| 1 | forum\_id | INT | 主键 | 讨论主题ID |
| 2 | user\_id | int | 外键 | 用户ID |
| 3 | title | varchar(255) | 否 | 讨论标题 |
| 4 | content | text | 否 | 讨论内容 |
| 5 | created\_at | timestamp | 否 | 创建日期 |

5、课程表

课程数据表，记录平台上的课程信息，主要包括以下字段：课程ID、课程标题、课程描述、课程分类ID、课程价格、教师ID、创建日期、更新时间和课程图片。课程ID是主键，教师ID外键关联到用户表，表示课程的授课教师。

课程数据表结构如表4-5所示。

表4-4 课程数据表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 类型 | 属性 | 描述 |
| 1 | course\_id | INT | 主键 | 管理员账号 |
| 2 | title | varchar(255) | 否 | 管理员密码 |
| 3 | description | text | 否 | 课程描述 |
| 4 | category\_id | int | 否 | 课程分类id |
| 5 | price | decimal(10, 2) | 否 | 课程价格 |
| 6 | created\_at | timestamp | 否 | 创建日期 |
| 7 | updated\_at | timestamp | 否 | 更新时间 |
| 8 | image | varchar(255) | 否 | 图片 |
| 9 | teacher\_id | int | 外键 | 教师ID |

6、评论表

评论信息，主要包括以下字段：评论ID、用户ID、评论内容、创建日期、课程ID和父评论ID。评论ID是主键，用户ID外键关联到用户表，课程ID外键关联到课程表。

评论表结构如表4-5所示。

表4-4 评论表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 类型 | 属性 | 描述 |
| 1 | comment\_id | INT | 主键 | 评论ID |
| 2 | user\_id | int | 否 | 用户ID |
| 3 | content | text | 否 | 评论内容 |
| 4 | created\_at | timestamp | 否 | 创建日期 |
| 5 | course\_id | int | 外键 | 课程id |
| 6 | parent\_id | int | 否 | 父评论ID |

7、章节表

记录课程的章节信息，主要包括以下字段：章节ID、课程ID、章节标题、章节描述、创建日期和更新时间。章节ID是主键，课程ID外键关联到课程表。

章节表结构如表4-5所示。

表4-4 章节表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 类型 | 属性 | 描述 |
| 1 | chapter\_id | INT | 主键 | 章节ID |
| 2 | course\_id | int | 否 | 课程ID |
| 3 | title | varchar(255) | 否 | 章节标题 |
| 4 | description | text | 否 | 章节描述 |
| 5 | created\_at | timestamp | 否 | 创建日期 |
| 6 | updated\_at | timestamp | 否 | 更新时间 |

8、章节视频表

存储每个章节下的视频信息，主要包括以下字段：视频ID、章节ID、视频链接、上传日期和视频代码。视频ID是主键，章节ID外键关联到章节表。

章节视频表结构如表4-5所示。

表4-4 章节视频表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 类型 | 属性 | 描述 |
| 1 | video\_id | INT | 主键 | 视频ID |
| 2 | chapter\_id | int | 否 | 章节ID |
| 3 | video\_url | varchar(255) | 否 | 视频链接 |
| 4 | uploaded\_at | timestamp | 否 | 上传日期 |
| 5 | video\_code | varchar(255) | 否 | 视频编码 |

9、课程分类表

课程分类数据表，用于记录课程的分类信息，主要包括以下字段：课程分类主键ID和课程分类名称。分类ID是主键，用于唯一标识每个分类。

课程分类表结构如表4-5所示。

表4-4 课程分类表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 类型 | 属性 | 描述 |
| 1 | category\_id | INT | 主键 | 课程分类主键名称 |
| 2 | category\_name | VARCHAR(50) | 否 | 课程名称 |

10、作业表

作业数据表，用于记录课程的作业信息，主要包括以下字段：作业ID、课程ID、作业描述和截止日期。作业ID是主键，课程ID外键关联到课程表。

作业表结构如表4-5所示。

表4-4 作业数据表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 类型 | 属性 | 描述 |
| 1 | assignment\_id | INT | 主键 | 作业ID |
| 2 | course\_id | INT | 外键 | 课程ID |
| 3 | description | text | 否 | 作业描述 |
| 4 | due\_date | date | 否 | 截止日期 |

## 4.3 系统性能设计

### 4.3.1 系统缓存技术设计

内存缓存：使用Redis等内存缓存技术，对常用的静态数据（如课程信息、用户信息）进行缓存，从而减少对数据库的频繁访问，加快响应速度。

查询结果缓存：对数据库的查询结果进行缓存，特别是复杂的查询操作，例如某课程的所有章节和资源列表，可以将查询结果缓存一定时间，避免每次都访问数据库。

### 4.3.2 系统数据库性能设计

索引：为常用的查询字段（如用户ID、课程ID、课程标题等）建立索引，优化数据库查询性能。

SQL查询优化：避免使用低效的查询（如全表扫描、子查询等），优先采用索引、分页查询或预先计算结果的方式进行优化。

分区和分表：对于数据量特别大的表，可以采用水平分区或垂直分区的方式，将数据分散存储在多个表或数据库实例中，降低单表的数据量，提升查询性能。

### 4.3.3 异步处理和队列技术

异步任务处理：对于一些耗时的操作（如视频转码、邮件发送等），可以采用异步处理，将任务放入消息队列中（如RabbitMQ、Kafka、ActiveMQ等），后台服务异步执行，避免阻塞主线程。

批量处理：对于高频率的请求或操作（如日志记录、积分更新等），可以采用批量处理的方式，减少数据库的访问次数和压力。

## 4.4 前端路由设计

该前端路由设计使用Vue Router来有效地管理应用程序的页面导航，确保用户能够顺畅地在不同功能模块间切换。路由配置包括多个子路由，使得应用结构更加清晰，并便于维护和扩展。应用的主要路径设定在一个布局组件（Layout）下，包含了多个功能模块，如仪表盘（Dashboard）、课程列表（CourseList）、作业发布（PublishHomework）等。这种嵌套路由设计不仅提升了用户体验，还便于模块化管理，使得不同的页面逻辑和状态能够独立处理。

此外，系统实现了全局路由守卫功能，通过检测用户的登录状态来控制访问权限。在用户访问页面时，首先检查是否存在有效的Token，以确定用户是否已登录。若用户已登录且试图访问登录页面，则会被重定向到首页，避免不必要的重复登录。而如果用户未登录，则需要检查目标路由是否在白名单中。只有白名单中的页面（如登录页面）可以被访问，其他页面将被重定向到登录页面。这种设计确保了应用的安全性，防止未授权用户访问敏感数据或功能，维护了系统的整体安全架构。

# 5 系统详细设计

## 5.1 用户模块设计与实现

账号模块主要包含注册、登录和登出。用户使用已注册账号进行登录后才能使用平台其余功能。

### 5.1.1 注册

该模块主要负责完成用户的注册功能。

在用户注册流程中，用户首先在注册页面输入必要的注册信息，包括用户名、密码和邮箱等，并在输入框失焦时，前端会对这些信息进行初步的格式校验，以确保输入符合规定的要求。如果输入信息不符合规范，系统会立即在输入框下方显示相应的错误提示，要求用户进行修改。若所有信息均符合规范，注册页面将通过 API 将注册请求发送至后端。后端会转发请求至用户服务，首先查询数据库以验证用户名的唯一性，如果用户名已存在，则会提示用户“用户名已存在”。接下来，若用户名可用，系统将继续检查邮箱是否已注册，如果邮箱已存在，系统会返回“邮箱已存在”的提示。只有在用户名和邮箱均未被注册的情况下，系统才会将用户信息保存至数据库，并确认保存成功。同时，注册成功后，用户信息会被推送到消息队列，以便进行后续的异步处理，例如发送欢迎邮件。最终，注册结果会返回至注册页面，用户将看到“注册成功”的消息。

用户成功注册时序图如图5-1所示。

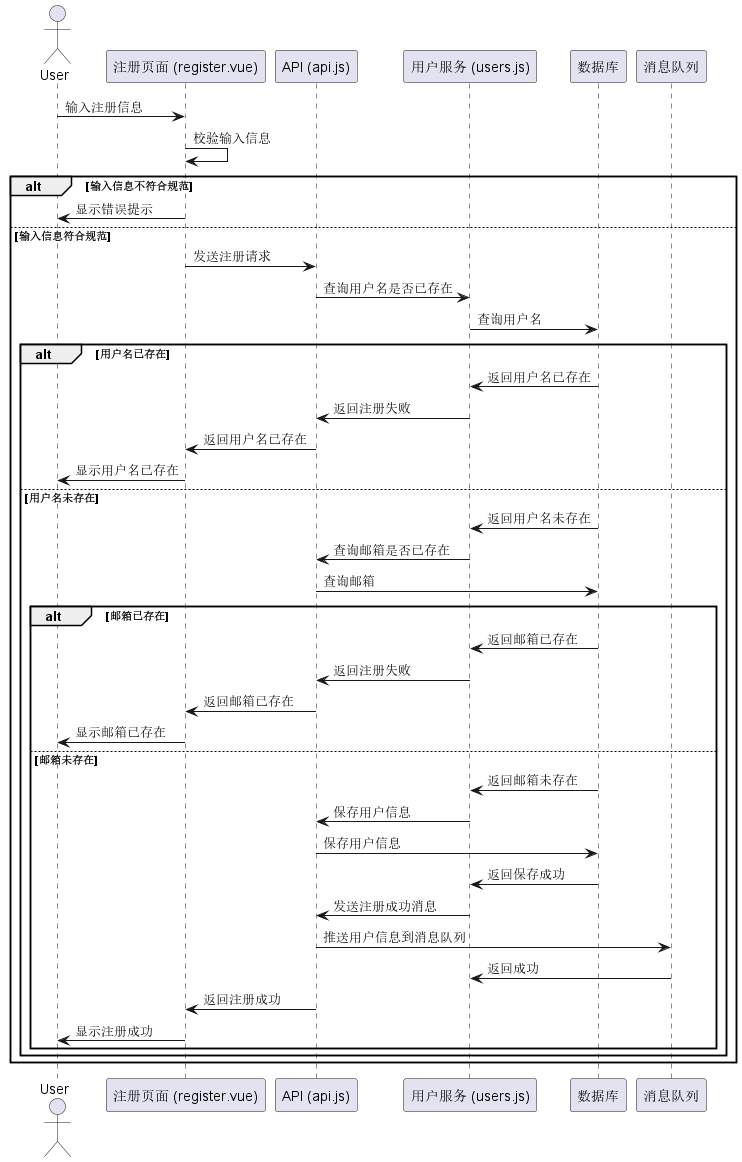


图5-1 用户成功注册时序图

### 5.1.2 登录

该模块主要负责完成用户的登录功能。

在用户登录流程中，用户首先在登录页面输入用户名和密码。页面会立即进行输入信息的格式校验，确保输入不为空。如果校验通过，系统会通过 API 发送登录请求至后端。

后端会接收到请求并将其转发给用户服务进行处理。用户服务会首先查询数据库以获取用户信息。如果用户存在，系统将验证密码是否正确。如果密码匹配成功，系统会生成一个 token，并将用户的信息存储到 Redis 中以便后续使用。

存储过程完成后，用户服务会将生成的 token 返回至 API，最终 API 将登录成功的信息传回至前端页面，并向用户展示“登录成功”的消息。

这个流程确保了用户的登录信息安全，并允许系统对用户身份进行有效管理。

用户登录成功时序图如图5-2所示。

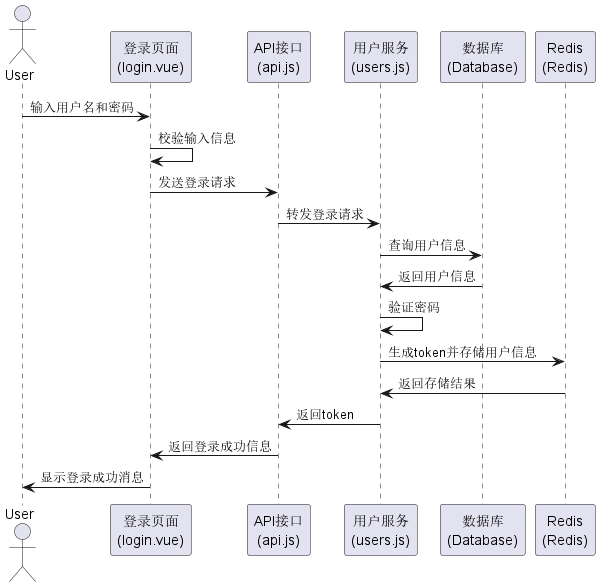


图5-2 用户成功登录时序图

### 5.1.3 登出

在用户决定退出登录时，首先点击登录页面上的“退出登录”按钮。该操作会触发一个退出请求，通过 API 发送至后端。

后端接收到请求后，将其转发至用户服务进行处理。用户服务从上下文中获取当前用户的信息，特别是用户的 ID。接下来，用户服务会调用 Redis 来删除与该用户 ID 相关联的 token。

如果 Redis 成功删除了 token，用户服务将返回成功的结果至 API，最后 API 将退出成功的消息反馈给前端页面，用户随后将看到退出成功的提示。

这个流程保证了用户的安全性，确保在用户退出后，系统会及时清除用户的登录状态。

用户登出时序图如图5-3所示。

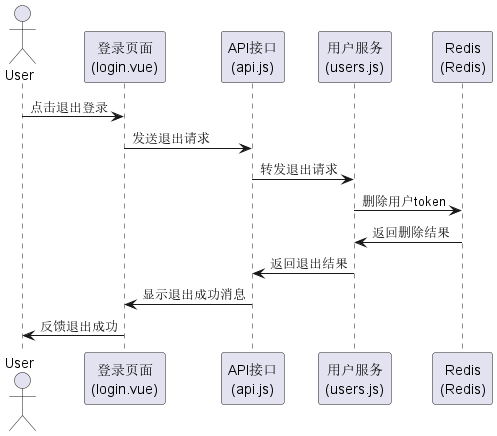


图5-3 登出时序图

## 5.2 课程模块设计与实现

课程模块的主要作用是学生能学习课程老师能添加课程，该模块主要包括老师添加课程和用户界面展示课程和学生收藏课程两个功能。

### 5.2.1 添加课程

在系统中，教师通过前端界面填写课程信息并点击“添加课程”按钮，前端将这些信息通过API调用发送到后端。后端接收到请求后，首先验证当前用户的身份是否为教师，并检查课程信息的有效性（如教师ID和分类ID是否合理）。若验证通过，后端将课程信息保存到数据库中并返回成功响应。前端在收到成功信息后，更新课程列表，展示新增的课程。

添加课程时序图如图5-4所示。

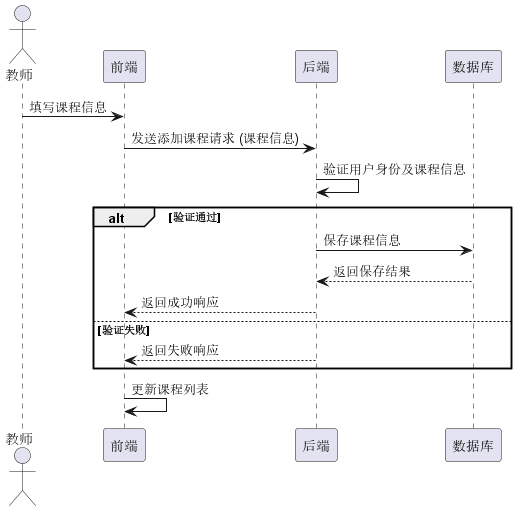


图5-4 添加课程时序图

用户进行商品下架时序图如图5-5所示。

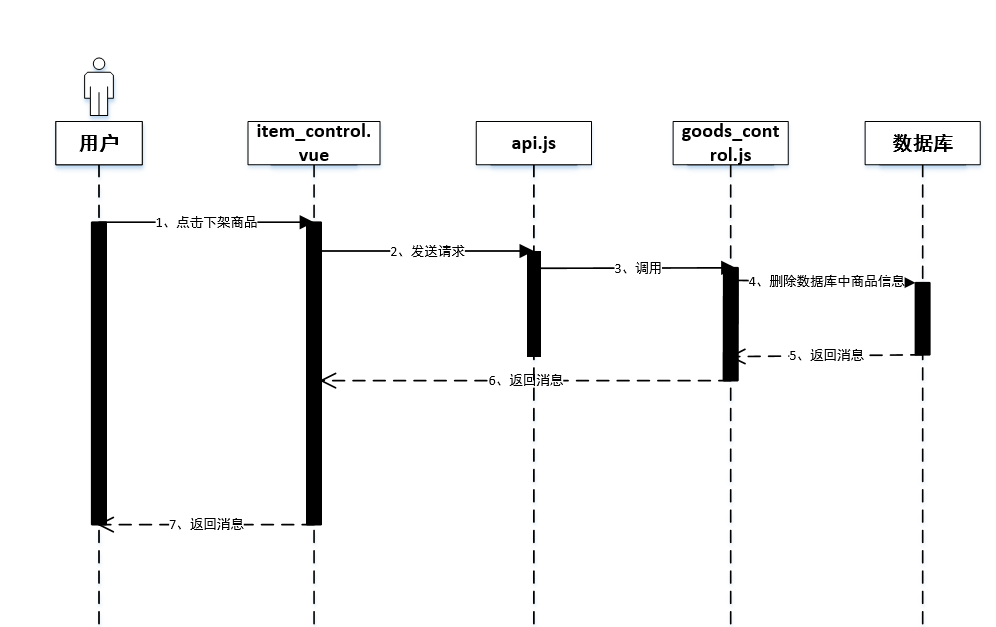


图5-5 用户下架时序图

### 5.2.2 展示课程

用户访问课程展示页面时，系统会从后端 API 获取所有课程分类和热门课程信息，并展示在页面上。用户可以输入课程名称进行搜索，系统会根据用户的输入过滤课程并显示相应的结果。此外，用户还可以点击课程分类标签，系统会根据所选的分类更新并展示对应的课程列表。每个课程卡片上显示课程的图片、标题和描述，用户可以通过点击课程进入学习页面。

展示课程时序图如图5-6所示。

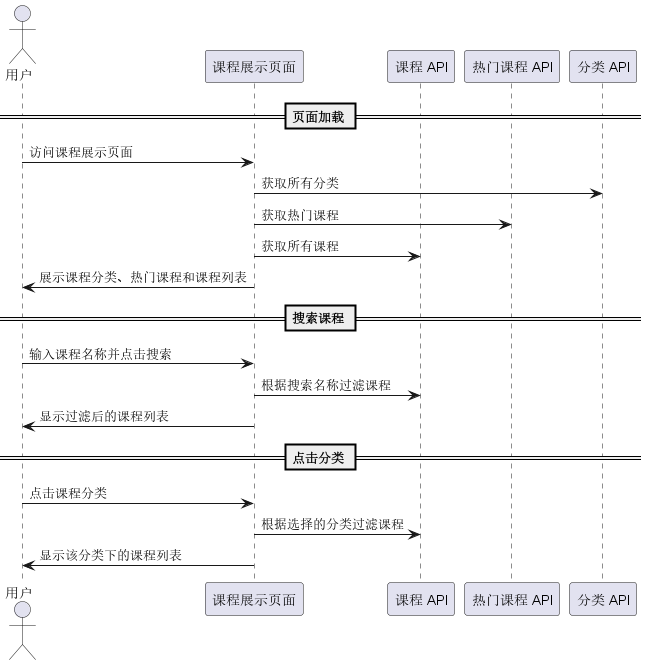


图5-6 展示课程时序图

### 5.2.3 收藏课程

已登录用户点击“加入课程”按钮后，系统首先获取当前用户的信息，包括用户 ID 和用户类型。系统会检查该用户是否为学生，如果不是，抛出一个异常，提示用户不能加入课程。如果用户是学生，则系统会查询该学生已加入的课程，确保用户不能重复加入同一课程。如果用户尚未加入该课程，系统会调用数据库操作将课程与学生关联，成功后更新页面状态。。

收藏课程时序图如图5-6所示。

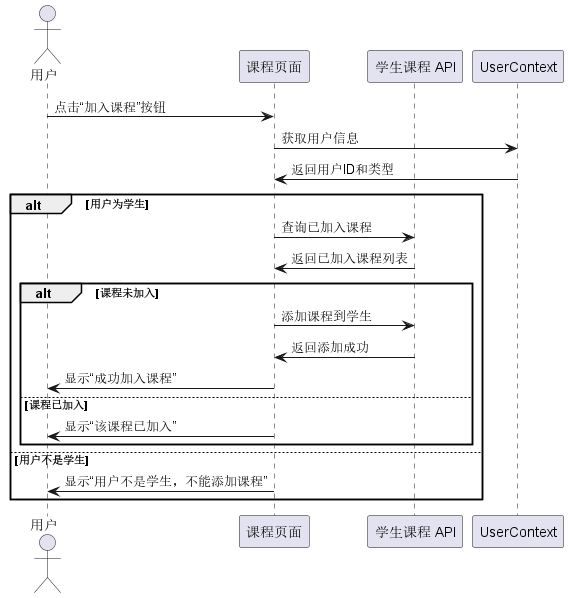


图5-6 用户收藏课程时序图

5.3 评论模块设计与实现

评论模块主要提供发送评论和回复用户评论两个功能。

### 5.3.1 发送评论

发送评论时，首先对传入的评论数据进行参数校验，确保评论内容和课程ID不为空。获取当前用户的ID，构建评论对象，并根据是否为回复设置相应的父评论ID（如为一级评论则设为0）。最后，将评论对象保存到数据库中，并返回操作结果。

发送评论时序图如图5-7所示。

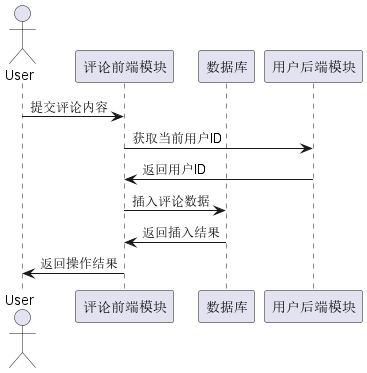


图5-7 发送评论时序图

### 5.3.2 回复用户评论

展示评论时，从数据库中获取所有相关评论数据，并将其整理成适合前端显示的格式，包括用户名和用户头像等信息。在展示时，如果评论为回复，则根据父评论ID关联显示，确保回复评论能正确归属到原评论下方。通过前端组件动态渲染评论列表，允许用户查看和回复特定评论。

回复用户评论时序图如图5-8所示。

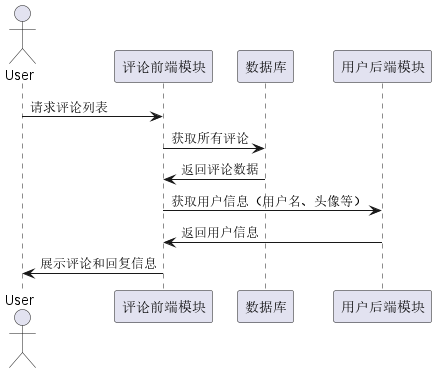


图5-8 回复用户时序图

5.4 博客模块设计与实现

博客模块主要包含两个功能：发布博客功能和查看博客功能。

### 5.4.1 发布博客

普通注册用户登录后，点击上方导航栏“购物车”按钮跳转至购物车页面，在购物车中，点击商品数量栏的“+”或“-”按钮，ShoppingCart.vue发送请求至api.js，api.js调用shoppingcart.js实现修改数据库中的商品数量。

用户增减商品数量时序图如图5-9所示。

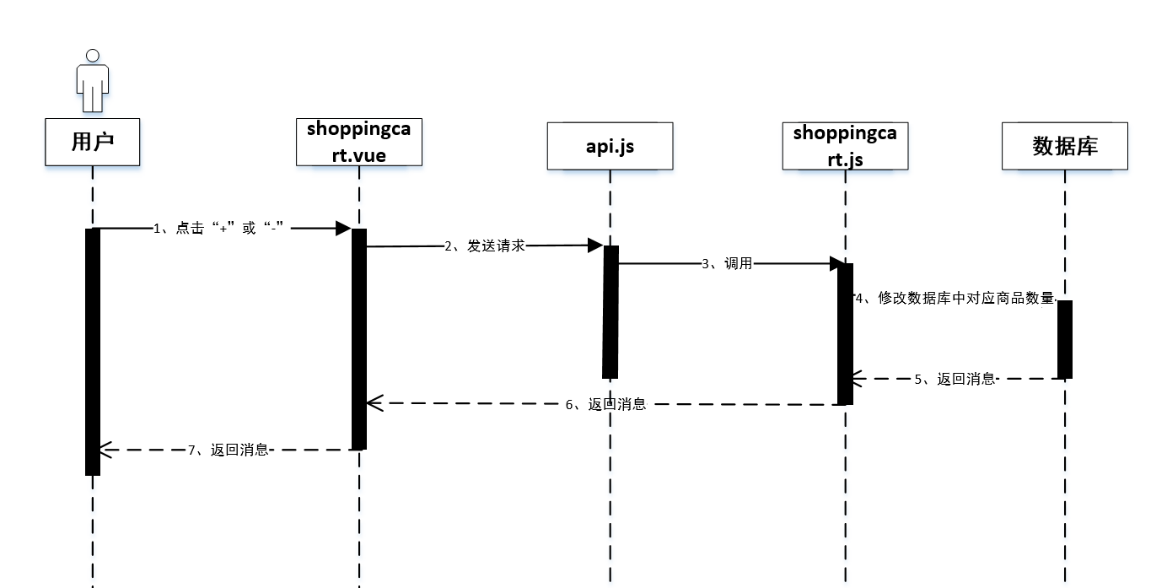


图5-9 增减商品数量时序图

### 5.4.2 查看博客

普通注册用户登录后，点击导航栏中“购物车”按钮即可跳转至购物车页面，在该页面中，点击商品右侧“删除”按钮，ShoppingCart.vue发送请求至api.js，api.js调用shoppingcart.js实现删除数据库中的商品即可完成该操作。

用户删除商品时序图如图5-10所示。

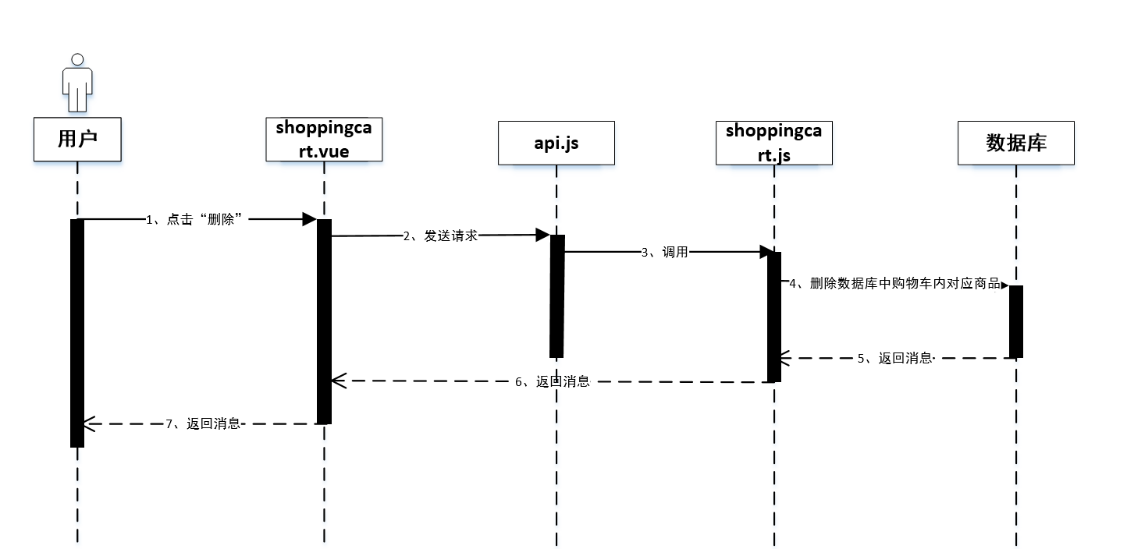


图5-10 删除商品时序图

## 5.5 作业模块设计与实现

商品审核模块主要包括老师发布作业，学生收到作业通知，功能，仅为管理员所用。

### 5.5.1 老师发布作业

已登录的管理员用户点击左侧导航栏“商品管理”进入商品审核页面，可以对所有已上架商品进行审核，下架不符合规范的商品。当管理员单击下架按钮并确认时，GoodsList.vue就会向api.js发送请求，api.js通过调用goods\_control.js实现删除数据库中对应商品。

管理员用户进行商品下架时序图如图5-11所示。

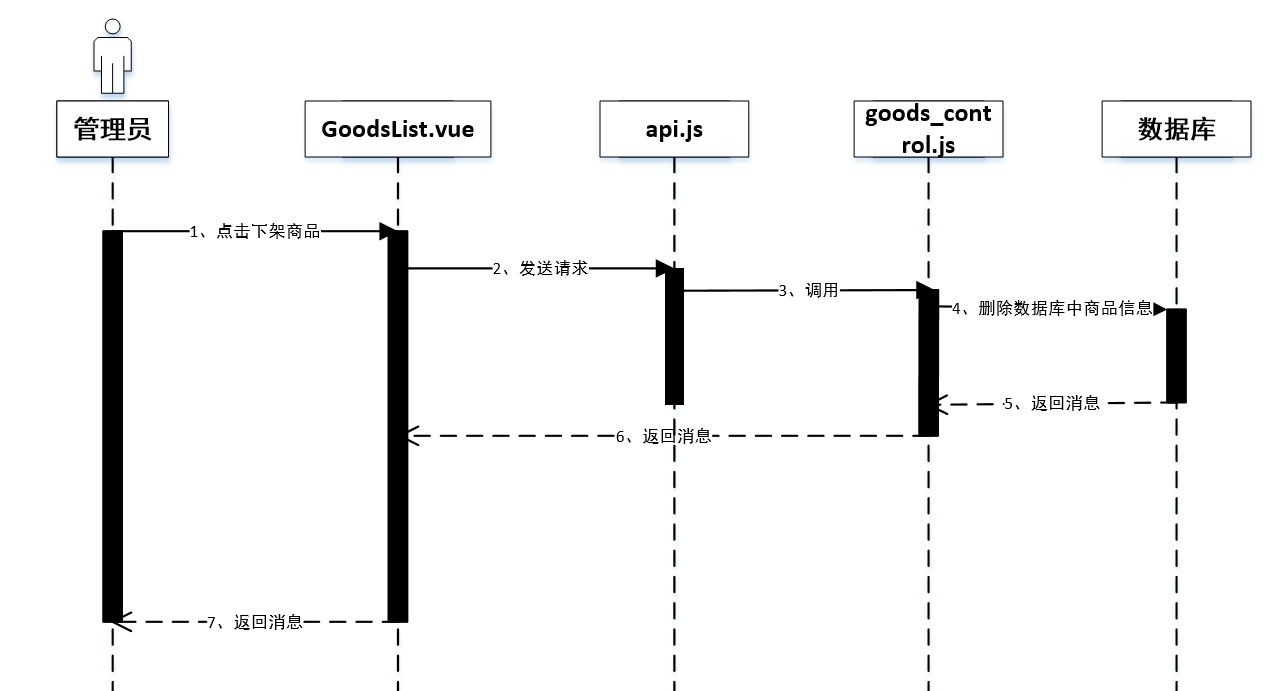


图5-11 管理员用户下架商品时序图

### 5.5.2 学生收到作业通知

已登录的管理员用户点击左方导航栏中的“商品管理”按钮，可以跳转进入商品审核页面，在该页面查看所有已上架商品。在此列表中，对于部分不规范的已上架商品信息，管理员用户可以点击编辑按钮，填写好信息并提交后，GoodsList.vue向api.js发送请求，api.js调用goods\_control.js实现更改数据库中对应商品的信息。

管理员用户修改商品信息时序图如图5-12所示。

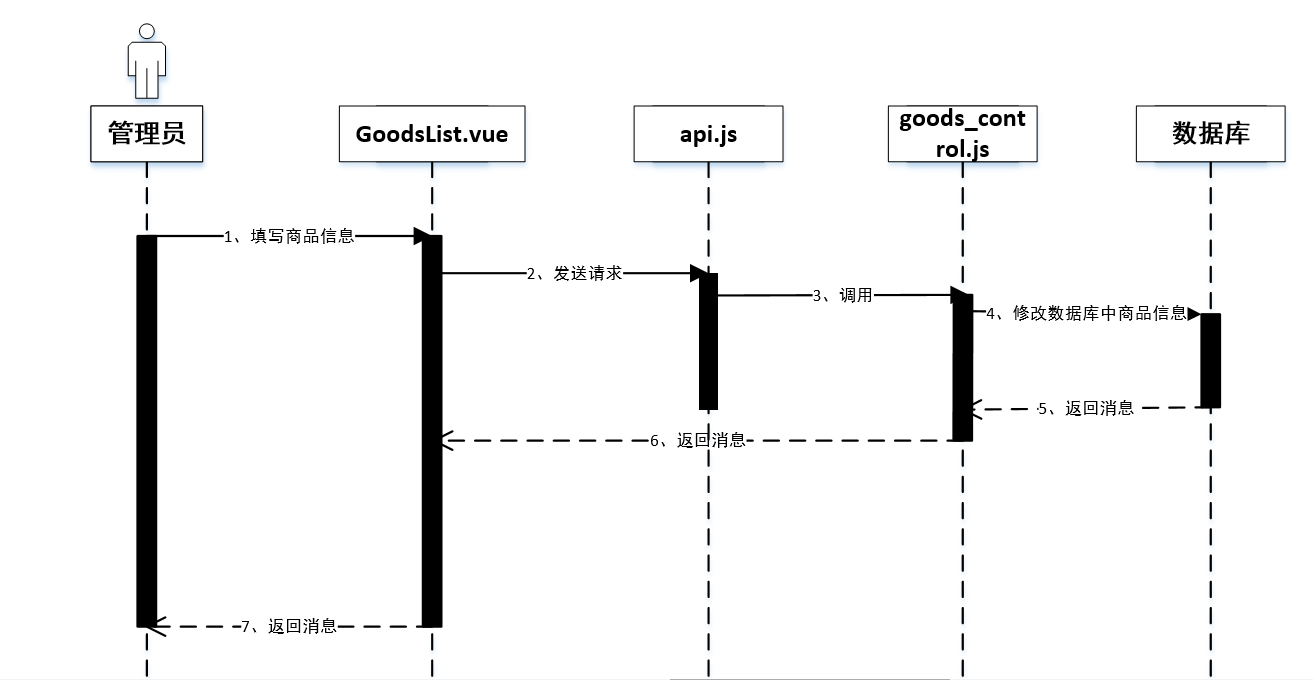


图5-12 管理员用户修改商品信息时序图

## 5.6 章节模块设计与实现

章节模块主要包括两个功能：在课程下添加章节和上传对应章节视频。

### 5.6.1 课程下添加章节

处于已登录状态的管理员用户单击左方导航栏中的“用户管理”按钮，跳转进入用户信息页面，在该页面里可以查看所有已注册用户的信息。管理员可以将违反相关法律法规的用户信息进行删除。在这个用户列表中，管理员单击用户信息右方的“删除”按钮，UserList.vue就会向api.js发送请求，api.js将调用users.js来实现删去数据库中该违规用户信息的操作。

管理员用户删除用户信息时序图如图5-13所示。

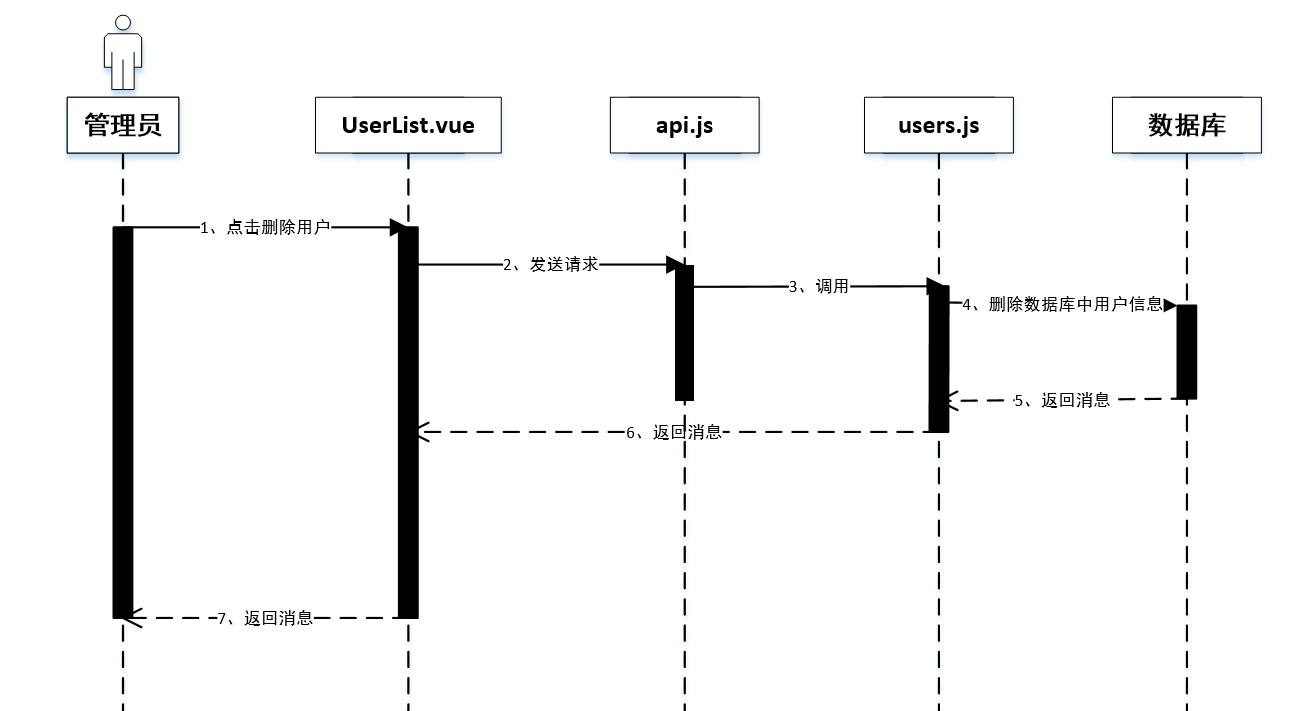


图5-13 管理员用户删除用户信息时序图

### 5.6.2 上传对应章节视频

已登录的管理员用户点击左侧导航栏“用户管理”即可跳转至用户信息页面查看所有已注册用户的信息。在此页面中，如有用户的信息不合相关法律法规，则点击编辑按钮，填写后好信息并提交后，UserList.vue向api.js发送请求，api.js调用users.js修改数据库中对应用户的信息。

管理员用户修改用户信息时序图如图5-14所示。

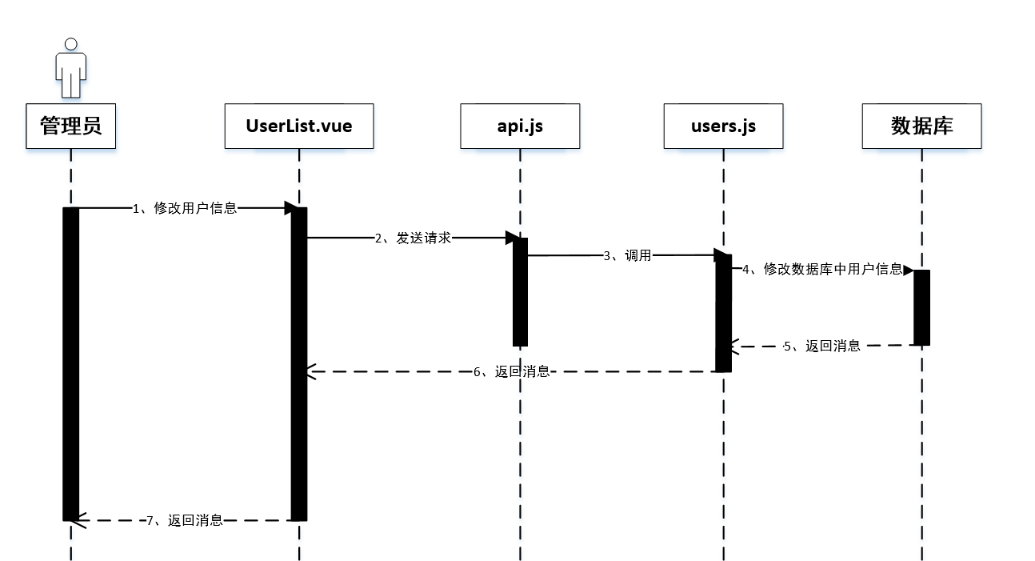


图5-14 管理员用户修改用户信息时序图

# 6 系统开发与实现

## 6.1 用户模块开发与实现

### 6.1.1 注册

注册接口设计表如表6-1所示：

表6-1 注册接口设计表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口名称 | 注册的接口 | | | |
| 接口描述 | 用户注册时调用该接口，在数据库中增加一条用户信息 | | | |
| URL | /api/users/register | | | |
| Method | POST | | | |
| 请求参数 | 手机号：telephone:12345654321 | 用户名：  name:nihao | 密码：password:123456 | 身份证号：cardId:123456789123456789 |
| 返回参数 | {status:200,message:’账号注册成功’} | | {status:400,message:’手机号已被注册’} | |

注册功能后端核心代码如下：

代码6-1 注册代码

|  |
| --- |
| router.post("/register", (req, res) => {  const { telephone, password, name, cardId } = req.body;  User.findOne({ where: { telephone } }).then((user) => {  if (user) {  return res.status(400).json("手机号已被注册！");  } else {  let cipher = crypto.createCipheriv(  "aes-128-cbc",  "123456789abcdefg",  "abcdefg123456789"  );  let newPWD =  cipher.update(password, "binary", "base64") + cipher.final("base64");  const newUser = User.build({  name,  telephone,  cardId,  password,  });  bcrypt.genSalt(10, (err, salt) => {  bcrypt.hash(newUser.dataValues.password, salt, function (err, hash) {  if (err) throw err;  newUser.password = hash;  newUser  .save()  .then((user) => res.json(user))  .catch((err) => console.log(err));  });  });  }  });  }); |

注册界面效果示意图如图6-1所示：



图6-1 用户注册图

### 6.1.2 登录

登录接口设计表如表6-2所示：

表6-2 登录接口设计表

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名称 | 登录的接口 | | |
| 接口描述 | 用户登录时调用该接口，在数据库中查询用户输入的信息 | | |
| URL | /api/users/login | | |
| Method | POST | | |
| 请求参数 | 手机号：telephone:12345654321 | 用户名：name:nihao | 密码：Password:123456 |
| 返回参数 | {status:200,message:’登录成功’} | {status:404,message:’用户不存在’} | {status:400,message:’密码错误’} |

登录功能后端核心代码如下：

代码6-2 登录代码

|  |
| --- |
| router.post("/login", (req, res) => {  const { telephone, password } = req.body;  User.findOne({ where: { telephone } }).then((user) => {  if (!user) {  return res.status(404).json("用户不存在！");  }  bcrypt.compare(password, user.password).then((isMatch) => {  if (isMatch) {  const rule = {  id: user.id,  name: user.name,  telephone: user.telephone,  cardId: user.cardId,  password: user.password,  intro: user.intro,  headpic: user.headpic,  };  jwt.sign(rule, "secret", { expiresIn: 3600 }, (err, token) => {  if (err) throw err;  res.json({ success: true, token: "Bearer " + token });  });  } else {  return res.status(400).json("密码错误！");  }  });  });  });}); |

登录界面效果示意图如图6-2所示：

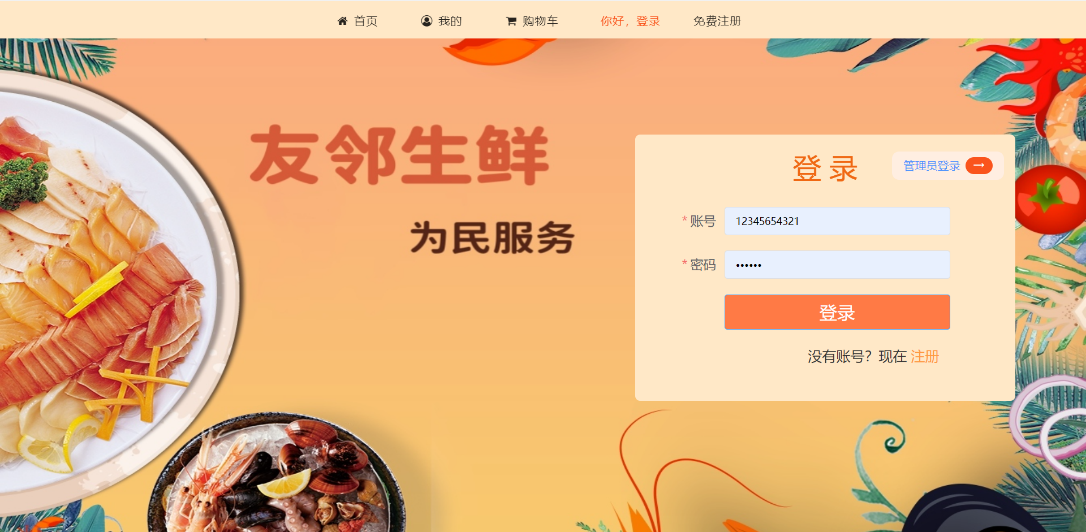


图6-2 用户登录图

### 6.1.3 查看订单

查看订单接口设计表如表6-3所示：

表6-3 查看订单接口设计表

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 接口名称 | 查看订单的接口 | | | | | | | | | |
| 接口描述 | 用户查看订单时调用该接口，在数据库中查询对应用户订单信息 | | | | | | | | | |
| URL | /api/order | | | | | | | | | |
| Method | POST | | | | | | | | | |
| 请求参数 | 订单总价：Totalprice:30 | 订单id：number:11111111 | | 商品名称：goodsname:西瓜 | | 商品种类：宁夏西瓜 | | 商品单价：cprice:30 | | 卖家账号：sellnumber:12345678910 |
| 商品数量：num:1 | | 商品图片：url:宁夏西瓜.jpg | | 订单状态：goodsstate:待收货 | | 交易操作：goodsoperation:确认收货 | | 买家账号：buynumber:12345654321 | |
| 返回参数 | {status:200,message:’查询成功’} | | | | | | | | | |

查看订单功能后端核心代码如下：

代码6-3 查看订单代码

|  |
| --- |
| router.post("/order", (req, res) => {  const params = req.body;  const selectAllOrders =  "select \* from `orders` where buynumber=? order by CONVERT( goodsstate USING gbk ) COLLATE gbk\_chinese\_ci ASC,number";  mysql.query(selectAllOrders, params.buynumber, function (err, data) {  if (err) throw err;  return res.send(data);  });  }); |

查看订单界面效果示意图如图6-3所示：



图6-3 查看订单图

## 6.2 课程模块开发与实现

### 6.2.1 商品上下架

上下架商品接口设计表如表6-4所示：

表6-4 上下架商品接口设计表

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 接口名称 | 上下架商品的接口 | | | | | | | | | | | | | |
| 接口描述 | 用户上架商品时调用该接口，在数据库中添加一行商品信息/删除一行商品信息 | | | | | | | | | | | | | |
| URL | /api/control/addGoods，/api/control/delGoods | | | | | | | | | | | | | |
| Method | POST | | | | | | | | | | | | | |
| 请求参数 | 卖家账号：sid:12345654321 | | 商品id：gid:1 | | | 商品名称：gname西瓜 | | 商品种类：goodstype:宁夏西瓜 | | 商品图片3：Imgurl0\_3:shop3.jpg | | | 商品数量：num:1 | |
| 商品图片0：Imggurl:0:shop6.jpg | Describe:很甜 | | | 商品价格：unitprice:30 | | 商品简介：describe:5kg，很甜 | | 商品图片1：Imgurl0\_1:shop7.jpg | | | 商品图片2：Imgurl0\_2:shop8.jpg | |
| 返回参数 | {status:200,message:’加入成功’} | | | {status:0,message:’加入失败’} | | | | {status:0,message:’删除失败’} | | | {status:200,message:’删除成功’} | | | |

### 6.2.2 修改商品信息

修改商品信息接口设计表如表6-5所示：

表6-5 修改商品信息接口设计表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口名称 | 修改商品信息的接口 | | | |
| 接口描述 | 用户修改商品信息时调用该接口，在数据库中修改一行商品信息 | | | |
| URL | /api/control/updateGoodsInfo | | | |
| Method | POST | | | |
| 请求参数 | 商品id：gid:1 | 商品名称：gname:西瓜 | | 商品种类：goodstype:宁夏西瓜 |
| 商品数量：num:1 | Describe:很甜 | | 商品单价：unitprice:40 |
| 返回参数 | {status:200,message:’修改成功’} | | {status:0,message:’修改失败’} | |

## 6.3 评论模块开发与实现

### 6.3.1 创建订单

创建订单接口设计表如表6-6所示：

表6-6 创建订单接口设计表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口名称 | 创建订单的接口 | | | |
| 接口描述 | 用户创建订单时调用该接口，在数据库中添加一行订单信息 | | | |
| URL | /api/order | | | |
| Method | POST | | | |
| 请求参数 | 手机号：id:12345654321 | 订单号：number:11111111 | 商品名称：goodsname:西瓜 | 商品种类：宁夏西瓜 |
| 商品数量：num:1 | 商品总价：totalprice:30 | 商品状态：goodsstate:待付款 | 订单操作：goodsoperation:进行付款 |
| 返回参数 | {status:200,message:’提交成功’} | | | |

### 6.3.2 完成订单

完成订单接口设计表如表6-7所示：

**表6-7 完成订单接口设计表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口名称 | 完成订单的接口 | | | |
| 接口描述 | 用户完成订单时调用该接口，在数据库中修改一行订单信息 | | | |
| URL | /api/order | | | |
| Method | POST | | | |
| 请求参数 | 买家账号：Buynumber:12345654321 | 卖家账号：Sellnumber:22222222222 | | 订单id：number:11111111 |
| 订单状态：goodsstate:待收货 | | 交易操作：goodsoperation:确认收货 | |
| 返回参数 | {status:200,message:’修改成功’} | | | |

完成订单功能后端核心代码如下：

代码6-4 完成订单代码

|  |
| --- |
| //修改待付款到待收货  router.post("/orderpay", (req, res) => {  const params = req.body;  const pay =  "update orders set goodsstate='待收货' , goodsoperation='确认收货' where number=?";  mysql.query(pay, [params.number], function (err, data) {  if (err) throw err;  return res.send(data);  });  });  //删除已签收订单  router.post("/orderdel", (req, res) => {  const params = req.body;  const del = "delete from `orders` where number=?";  mysql.query(del, [params.number], function (err, data) {  if (err) throw err;  return res.send(data);  });  }); |

完成订单界面效果示意图如图6-4所示：



图6-4 完成订单界面效果示意图

## 6.4 博客模块开发与实现

### 6.4.1 增减商品数量

增减商品数量接口设计表如表6-8所示：

表6-8 增减商品数量接口设计表

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名称 | 增减商品的接口 | | |
| 接口描述 | 用户增减商品数量时调用该接口，在数据库中修改一行购物车信息 | | |
| URL | /api/addreduce/${id} | | |
| Method | POST | | |
| 请求参数 | 手机号：telephone:12345654321 | 商品id：goodsid:1 | 商品数量：num:2 |
| 返回参数 | {status:200,message:’修改成功’} | | |

### 6.4.2 删除商品

删除商品接口设计表如表6-9所示：

表6-9 删除商品接口设计表

|  |  |  |
| --- | --- | --- |
| 接口名称 | 删除商品的接口 | |
| 接口描述 | 用户删除购物车中商品时调用该接口，在数据库中删除一行购物车信息 | |
| URL | /api/delete/${id} | |
| Method | POST | |
| 请求参数 | 手机号：id:12345654321 | 商品号：sid:1 |
| 返回参数 | { status:200,message:’删除成功’} | |

## 6.5 作业模块开发与实现

### 6.5.1 下架商品

管理员用户下架商品接口设计表如表6-10所示：

表6-10 管理员用户下架商品接口设计表

|  |  |  |
| --- | --- | --- |
| 接口名称 | 管理员用户下架商品的接口 | |
| 接口描述 | 管理员下架商品时调用该接口，在数据库中删除一行商品信息 | |
| URL | /api/control/delGoods | |
| Method | POST | |
| 请求参数 | 商品号：gid:1 | |
| 返回参数 | {status:200,message:’下架成功’} | {status:0,message:’下架失败’} |

管理员下架商品功能后端核心代码如下：

代码6-5 管理员下架商品代码

|  |
| --- |
| *router.post("/delGoods", (req, res) => {*  *const goods\_id = req.body.gid;*  *let sqlStr = "DELETE FROM goodsList WHERE gid =" + goods\_id;*  *mysql.query(sqlStr, (error, results, fields) => {*  *if (error) {*  *res.json({ err\_code: 0, message: "下架失败!" });*  *} else {*  *let sqlStr2 = "DELETE FROM shoppingCart WHERE gid =" + goods\_id;*  *mysql.query(sqlStr, (error, results, fields) => {*  *if (error) {*  *res.json({ err\_code: 0, message: "下架失败!" });*  *} else {*  *res.json({ success\_code: 200, message: "下架成功!" });*  *}*  *});*  *}*  *});*  *});* |

### 6.5.2 修改商品信息

修改商品信息接口设计表如表6-11所示：

表6-11 修改商品信息接口设计表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口名称 | 修改商品信息的接口 | | | |
| 接口描述 | 管理员修改商品信息时调用该接口，在数据库中修改一行商品信息 | | | |
| URL | /api/control/updateGoodsInfo | | | |
| Method | POST | | | |
| 请求参数 | 商品id：gid:1 | 请求参数 | 商品id：gid:1 | 请求参数 |
| 商品数量：num:1 | 商品照片：shop6.jpg | 商品数量：num:1 | 商品单价：unitprice:40 |
| 返回参数 | {status:200,message:’修改成功’} | | {status:0,message:’修改失败’} | |

修改商品信息功能后端核心代码如下：

代码6-6 修改商品信息代码

|  |
| --- |
| router.post("/updateGoodsInfo", (req, res) => {  const goods\_id = req.body.gid;  const goods\_name = req.body.gname;  const goods\_desc = req.body.describe;  const price = req.body.unitprice;  const category = req.body.type;  const degree = req.body.percent;  let sqlStr =  "UPDATE goodsList SET `gname` = ?, `describe` = ?, `unitprice` = ?, `percent` = ?, `type` = ? WHERE `gid` = ?";  let strParams = [goods\_name, goods\_desc, price, degree, category, goods\_id];  mysql.query(sqlStr, strParams, (error, results, fields) => {  if (error) {  res.json({ err\_code: 0, message: "修改失败!" });  } else {  res.json({ success\_code: 200, message: "修改成功!" });  }  });  }); |

管理员修改商品信息界面效果示意图如图6-5所示：

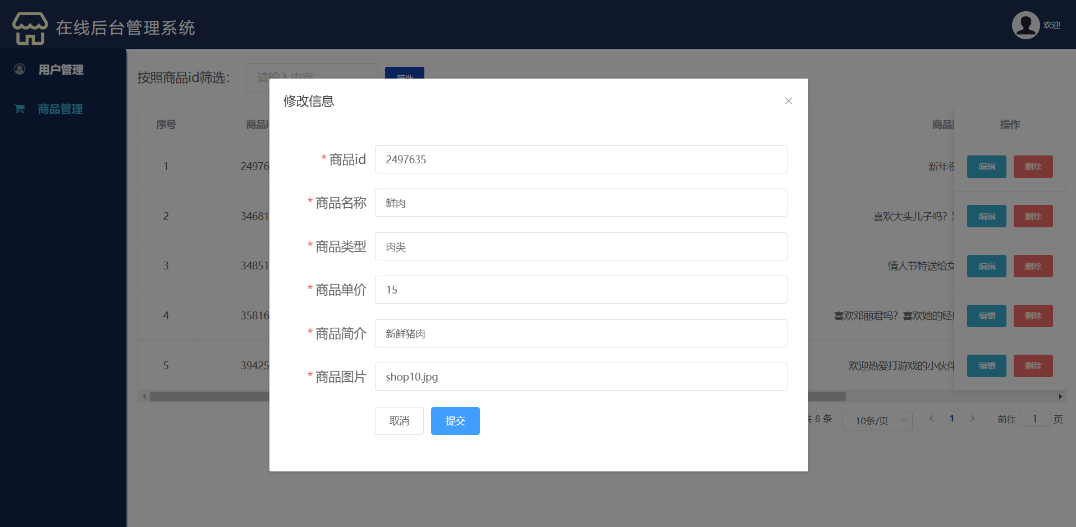


图6-5 管理员修改商品信息图

## 6.6 章节模块开发与实现

### 6.6.1 删除用户信息

管理员删除用户信息接口设计表如表6-12所示：

表6-12 删除用户信息接口设计表

|  |  |  |
| --- | --- | --- |
| 接口名称 | 删除用户信息的接口 | |
| 接口描述 | 管理员删除用户信息时调用该接口，在数据库中删除一条用户信息 | |
| URL | /api/users/delete/${row.id} | |
| Method | POST | |
| 请求参数 | 用户账号：telephone:12345654321 | |
| 返回参数 | {status:200,message:’删除成功’} | {status:0,message:’删除失败’} |

管理员删除用户信息功能后端核心代码如下：

代码6-7 管理员删除用户信息代码

|  |
| --- |
| *router.post(*  *"/delete/:id",*  *passport.authenticate("jwt", { session: false }),*  *(req, res) => {*  *User.destroy({ where: { id: req.params.id } })*  *.then((profile) => {*  *res.json(profile);*  *})*  *.catch((err) => res.status(404).json("删除失败!"));*  *}*  *);* |

### 6.6.2 修改用户信息

管理员用户修改用户信息接口设计表如表6-13所示：

表6-13 修改用户信息接口设计表

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名称 | 修改用户信息的接口 | | |
| 接口描述 | 管理员修改用户信息时调用该接口，在数据库中修改用户信息 | | |
| URL | /api/users/usersList | | |
| Method | POST | | |
| 请求参数 | 用户账号：telephone:12345654321 | 用户名称：name:哈哈哈哈哈 | 身份证号码：cardId:123456789123456789 |
| 返回参数 | {status:200,message:’修改成功’} | | |

管理员修改用户信息功能后端核心代码如下：

代码6-8 管理员修改用户信息代码

|  |
| --- |
| *router.post("/edit/:id", (req, res) => {*  *let profileFields = {};*  *if (req.body.name) profileFields.name = req.body.name;*  *if (req.body.telephone) profileFields.telephone = req.body.telephone;*  *if (req.body.cardId) profileFields.cardId = req.body.cardId;*  *User.update(profileFields, {*  *where: { id: req.params.id },*  *returning: true,*  *}).then((profile) => res.json(profile));*  *});* |

管理员修改用户信息界面效果示意图如图6-6所示：

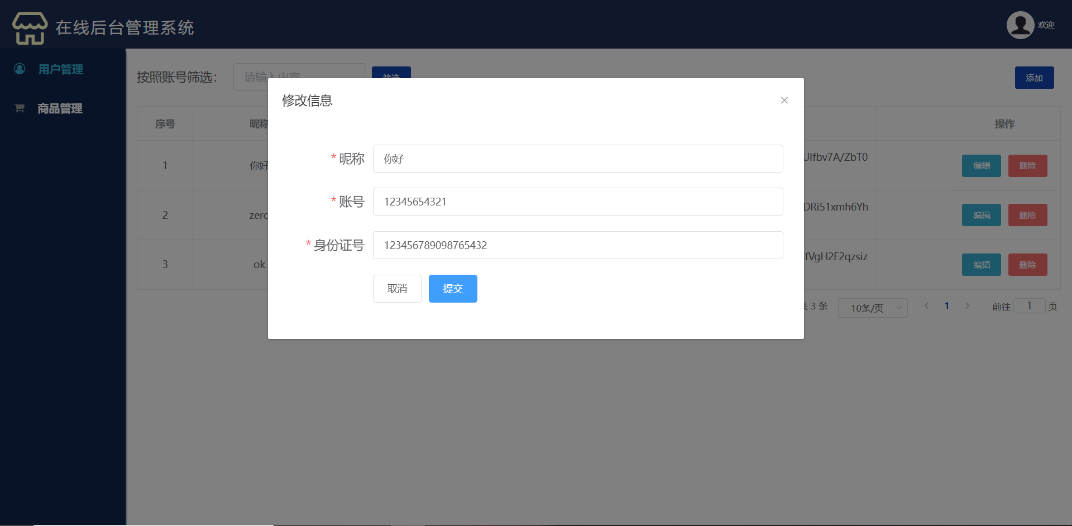


图6-6 管理员修改用户信息图

# 7 系统测试

## 7.1 测试方法说明

功能测试，即检测平台相关功能，例如对数据库进行增删改查能否正常执行，是系统测试的核心内容。在让用户使用平台前，需要检查程序的每个功能模块是否能完成相应功能。软件测试主要分为白盒测试和黑盒测试与白盒测试，白盒测试即为测试程序内部结构是否满足要求，黑盒测试即为软件是否能完成所需功能，将实际结果与预期结果进行比对 [15]。本生鲜交易系统的实现目标是让用户体验相关功能，所以采用黑盒测试。

## 7.2 测试用例及结果分析

### 7.2.1 用户模块

本小节对用户完成注册功能及登录功能进行测试，结果如表7-1所示。

表7-1 用户模块测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例ID | 测试功能 | 测试流程 | 预期输出 | 测试结果 |
| 1 | 注册 | 页面点击注册按钮 | 跳转至注册页 | 通过 |
| 2 | 注册 | 输入正确注册信息 | 成功 | 通过 |
| 3 | 注册 | 输入不合法用户名 | 失败 | 通过 |
| 4 | 注册 | 输入不合法手机号 | 失败 | 通过 |
| 5 | 注册 | 输入不合法身份证号码 | 失败 | 通过 |
| 6 | 注册 | 输入密码非8~12位字母或数字 | 失败 | 通过 |
| 7 | 注册 | 输入确认密码与密码不同 | 失败 | 通过 |
| 8 | 登录 | 页面点击登录按钮 | 跳转至登录页 | 通过 |
| 9 | 登录 | 用户输入正确登录信息 | 成功 | 通过 |
| 10 | 登录 | 用户输入错误账号 | 失败 | 通过 |
| 11 | 登录 | 用户输入错误密码 | 失败 | 通过 |
| 12 | 登录 | 点击“管理员登录”按钮 | 切换登录模块 | 通过 |
| 13 | 登录 | 管理员用户输入正确登录信息 | 成功 | 通过 |
| 14 | 登录 | 管理员用户输入错误账号 | 失败 | 通过 |
| 15 | 登录 | 管理员用户输入错误密码 | 失败 | 通过 |
| 16 | 登录 | 点击“用户登录”按钮 | 切换登录模块 | 通过 |

### 7.2.2 课程模块

本小节对用户完成商品上下架功能和修改商品信息功能进行测试，结果如表7-2所示。

表7-2 商品模块测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例ID | 测试功能 | 测试流程 | 预期输出 | 测试结果 |
| 1 | 商品上架 | 页面点击商品上架按钮 | 跳转至商品上架页 | 通过 |
| 2 | 商品上架 | 正确填写上架信息 | 成功 | 通过 |
| 3 | 商品上架 | 商品名称填写不符合规范 | 失败 | 通过 |
| 4 | 商品上架 | 商品图片未上传 | 失败 | 通过 |
| 5 | 商品上架 | 商品介绍未填写 | 失败 | 通过 |
| 6 | 商品下架 | 点击商品下架按钮 | 成功 | 通过 |
| 7 | 修改商品信息 | 点击修改商品信息按钮 | 跳转至修改商品信息页面 | 通过 |
| 8 | 修改商品信息 | 正确填写商品信息 | 成功 | 通过 |
| 9 | 修改商品信息 | 商品名称填写不符合规范 | 失败 | 通过 |
| 10 | 修改商品信息 | 商品图片未上传 | 失败 | 通过 |
| 11 | 修改商品信息 | 商品介绍未填写 | 失败 | 通过 |

### 7.2.3 博客模块

本小节对用户完成创建订单功能和完成订单功能进行测试，结果如表7-3所示。

表7-3 订单模块测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例ID | 测试功能 | 测试流程 | 预期输出 | 测试结果 |
| 1 | 创建订单 | 页面点击提交订单按钮 | 创建一个新订单 | 通过 |
| 2 | 完成订单 | 页面点击订单管理 | 跳转至订单管理页面 | 通过 |
| 3 | 完成订单 | 点击确认付款 | 订单交易状态改为待收货，交易操作改为确认收货 | 通过 |
| 4 | 完成订单 | 点击确认收货按钮 | 订单交易状态改为已签收，交易操作改为删除订单 | 通过 |
| 5 | 完成订单 | 点击“我购买的”页的删除订单 | 成功 | 通过 |
| 6 | 完成订单 | 点击确认发货 | 订单交易状态改为已售出，交易操作改为删除订单 | 通过 |
| 7 | 完成订单 | 点击“我卖出的”页的删除订单按钮 | 成功 | 通过 |

### 7.2.4 作业模块

本小节对用户完成查看购物车、删除购物车商品、增减商品数量、全选和结算功能进行测试，结果如表7-4所示。

表7-4 购物车模块测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例ID | 测试功能 | 测试流程 | 预期输出 | 测试结果 |
| 1 | 查看购物车 | 页面点击购物车按钮 | 跳转至购物车页 | 通过 |
| 2 | 删除购物车商品 | 点击删除按钮 | 成功 | 通过 |
| 3 | 增加商品数量 | 点击“+” | 商品数量加一 | 通过 |
| 4 | 减少商品数量 | 点击“-” | 商品数量减一 | 通过 |
| 5 | 全选 | 点击全选 | 成功 | 通过 |
| 6 | 结算 | 点击结算按钮 | 跳转至创建订单页面 | 通过 |

### 7.2.5 章节模块

本小节对管理员用户完成下架商品和修改商品信息功能进行测试，结果如表7-5所示。

表7-5 商品审核模块测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例ID | 测试功能 | 测试流程 | 预期输出 | 测试结果 |
| 1 | 下架商品 | 页面点击下架按钮 | 成功 | 通过 |
| 2 | 修改商品信息 | 页面点击修改商品按钮 | 跳转至修改商品页面 | 通过 |
| 3 | 修改商品信息 | 填写正确商品信息 | 成功 | 通过 |
| 4 | 修改商品信息 | 商品名称填写不符合规范 | 失败 | 通过 |
| 5 | 修改商品信息 | 商品图片未上传 | 失败 | 通过 |
| 6 | 修改商品信息 | 商品介绍未填写 | 失败 | 通过 |

### 7.2.6 评论模块

本小节对管理员用户删除和编辑用户信息功能进行测试，结果如表7-6所示。

**表7-6 用户管理模块测试用例表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例ID | 测试功能 | 测试流程 | 预期输出 | 测试结果 |
| 1 | 删除用户信息 | 页面点击用户模块按钮 | 跳转至用户模块页 | 通过 |
| 2 | 删除用户信息 | 点击删除按钮 | 成功 | 通过 |
| 3 | 编辑用户信息 | 更改用户名称 | 成功 | 通过 |

# 8 总结与展望

设计并实现友邻生鲜交易系统的工作总结如下：

首先，分析了现目前国内外生鲜交易平台的情况，从中总结值得借鉴的地方，汲取经验，然后了解到很少有平台能够让高风险地区的用户与用户之间实现生鲜资源交易。之后决定采用Vue、element-UI、express、node.js等技术进行设计和实现上述功能，用MySql存储数据。从用户的角度出发，本文对该平台的功能性需求和非功能性需求进行了详细的分析。接下来，以上述需求分析为基础，将该平台分为了一下几个部分：账号模块、商品模块、订单模块、购物车模块、商品审核模块和用户管理模块，其中，前四个模块为用户使用的模块，后两个模块为管理员用户使用的模块。然后进行系统框架的设计，采用MVVM的架构，实现前后端分离，并详细设计了数据库。最后测试平台各个功能并进行记录。

由于时间关系，本论文实现的平台的细节仍需加以打磨。部分页面的美观程度有待提升；首页对用户进行商品资源的推荐是随机的，应该使用其他技术对用户的喜好进行分析并推荐可能与用户喜好相关的商品。日后，我需不断学习技术，完善系统要求。

# 参考文献

1. 任翔.新形势下我国电商平台与消费需求的相互影响与作用[J].黄冈职业技术学院学报,2022,24(04):79-82.
2. 陈传明.基于第四次零售革命的零售业创新发展趋势研究[J].商业经济研究,2020(08):44-46.
3. 李宇,何跃,吴中霞,唐琪雨,郑灏键.新冠疫情下生鲜电商的交易情况预测分析[J].商场现代化,2020(16):1-3.DOI:10.14013/j.cnki.scxdh.2020.16.001.
4. 弓序.新零售背景下生鲜电商前置仓模式的研究——以朴朴超市为例[J].全国流通经济,2022(23):39-41.DOI:10.16834/j.cnki.issn1009-5292.2022.23.039.
5. 庞东升.新冠肺炎疫情下生鲜类社区电商运营发展策略[J].中国市场,2022(18):160-162.DOI:10.13939/j.cnki.zgsc.2022.18.160.
6. 赵文硕,王玉杰,白丽.我国生鲜电商平台比较及选择研究[J].蔬菜,2022(06):32-38.
7. 伏红勇,雷瑶,王磊.基于价值创造的“上门”型生鲜产品O2O模式研究[J].物流研究,2020(02):44-51.
8. 张栋,林靖涵,毛益琳,洪琳.新零售背景下消费者网购生鲜农产品意愿影响因素浅析[J].中国集体经济,2021(12):85-87.
9. 何煜. 基于MVC架构的生鲜电子商务交易系统的设计与实现[D].大连交通大学,2019.DOI:10.26990/d.cnki.gsltc.2019.000531.
10. 臧怡. 中国生鲜电商平台经营者监管建议—中法比较法视野[D].华东政法大学,2021.DOI:10.27150/d.cnki.ghdzc.2021.001058.
11. 齐本荣,侯启缘.中美电商平台规制体系比较——新经济规制体系的适应性、可流动性与可预期性[J].中国流通经济,2022,36(10):58-68.DOI:10.14089/j.cnki.cn11-3664/f.2022.10.006.
12. 王悦. 社区团购平台国内及海外发展中的营销策略研究[D].山东财经大学,2022.DOI:10.27274/d.cnki.gsdjc.2022.000770.
13. Chitra Suresh. Research on Programming Language of Web Site Front-end Development[J]. Computer Informatization and Mechanical System,2020,3(1).
14. Paul Krill. Vue.js 3.0 brings more speed, more TypeScript[J]. InfoWorld.com,2020.
15. 妥泽花.基于黑盒测试与白盒测试的比较探究[J].电子世界,2021(11):55-56.DOI:10.19353/j.cnki.dzsj.2021.11.027.

# 致谢