**郑州西亚斯学院**

本科毕业论文（设计）

|  |  |
| --- | --- |
| 题 目 | 基于Spring Boot框架实现高校自习室资源管理系统 |
| 学生姓名 | 王啸龙 |
| 学 号 | 2021105420440 |
| 专 业 | 数据科学与大数据技术 |
| 班 级 | 4班 |
| 学 院 | 计算机与软件工程学院 |
| 完成时间 | 2025年5月 |

基于Spring Boot框架实现高校自习室资源管理系统

摘 要

本论文设计并实现了基于Spring Boot架构的高校自习室管理系统，旨在通过智能化手段提升自习室资源利用效率，解决高校自习室资源分配与动态调度中的技术瓶颈。系统结合物联网与移动互联网技术，构建实时化、可视化的管理平台，为师生提供精准化、便捷化的座位服务体验。本系统基于Spring Boot框架实现核心功能，覆盖自习室预约管理、座位分配等模块。系统界面简洁直观，操作流程便捷，支持学生在线查看座位状态、实时预约，交互响应高效流畅。有效解决高校自习室资源调度难题，提升使用效率与管理透明度。该系统顺应校园服务数字化趋势，为智慧校园建设提供标准化解决方案，未来可通过功能扩展持续优化资源管理模式，具有广泛的应用前景。

关键词 自习室管理系统；Spring Boot；Java；互联网

**Implementing a university study room resource management system based on the Spring bootframework**

**Abstract**

This paper designs and implements a college study room management system based on the Spring bootarchitecture, aiming to improve the efficiency of study room resource utilization through intelligent means and solve the technical bottlenecks in resource allocation and dynamic scheduling of college study rooms. The system combines the Internet of Things and mobile Internet technologies to build a real-time and visual management platform to provide teachers and students with precise and convenient seat service experience. This system implements core functions based on the Spring bootframework, covering modules such as study room reservation management and seat allocation. The system interface is simple and intuitive, the operation process is convenient, and it supports students to check seat status online, make reservations in real time, and the interactive response is efficient and smooth. It effectively solves the resource scheduling problem of university study rooms and improves usage efficiency and management transparency. The system conforms to the trend of digitalization of campus services and provides standardized solutions for the construction of smart campuses. In the future, it can continuously optimize the resource management model through functional expansion, and has broad application prospects.

**Keywords**; Study room management system; Spring Boot; Java; Internet

目录

[摘 要 1](#_Toc195684474)

**[Abstract](#_Toc195684475)** [1](#_Toc195684475)

**[1 绪论](#_Toc195684476)** [4](#_Toc195684476)

**[1.1课题背景](#_Toc195684477)** [4](#_Toc195684477)

**[1.2设计目标及意义](#_Toc195684478)** [4](#_Toc195684478)

**[1.3论文结构](#_Toc195684479)** [4](#_Toc195684479)

**[2 相关技术介绍](#_Toc195684480)** [5](#_Toc195684480)

**[2.1 Java语言简介](#_Toc195684481)** [5](#_Toc195684481)

**[2.2 Spring Boot技术简介](#_Toc195684482)** [5](#_Toc195684482)

**[2.3 Mybatis技术简介](#_Toc195684483)** [5](#_Toc195684483)

**[2.4 MyBatis-Plus技术简介](#_Toc195684484)** [6](#_Toc195684484)

**[2.5 MySQL技术简介](#_Toc195684485)** [6](#_Toc195684485)

**[2.6 Redis技术简介](#_Toc195684486)** [6](#_Toc195684486)

**[3 系统需求分析](#_Toc195684487)** [6](#_Toc195684487)

**[3.1经济可行性分析](#_Toc195684488)** [6](#_Toc195684488)

**[3.2技术可行性分析](#_Toc195684489)** [7](#_Toc195684489)

**[3.3操作可行性分析](#_Toc195684490)** [7](#_Toc195684490)

**[3.4功能需求分析](#_Toc195684491)** [7](#_Toc195684491)

**[4 系统设计和实现](#_Toc195684492)** [8](#_Toc195684492)

**[4.1系统架构](#_Toc195684493)** [8](#_Toc195684493)

**[4.2 系统功能结构设计](#_Toc195684494)** [9](#_Toc195684494)

**[4.3系统详细设计](#_Toc195684495)** [12](#_Toc195684495)

**[4.4 数据库设计](#_Toc195684496)** [15](#_Toc195684496)

**[5.系统实现](#_Toc195684497)** [25](#_Toc195684497)

**[5.1 前台用户功能模块实现](#_Toc195684498)** [25](#_Toc195684498)

**[5.2 后台管理员功能模块实现](#_Toc195684499)** [26](#_Toc195684499)

**[5.3 本章小结](#_Toc195684500)** [36](#_Toc195684500)

**[6.系统测试](#_Toc195684501)** [36](#_Toc195684501)

**[6.1 测试目的](#_Toc195684502)** [36](#_Toc195684502)

**[6.2 测试目标](#_Toc195684503)** [36](#_Toc195684503)

**[6.3 测试方法](#_Toc195684504)** [37](#_Toc195684504)

**[6.4 测试用例](#_Toc195684505)** [37](#_Toc195684505)

**[总结](#_Toc195684506)** [38](#_Toc195684506)

**[致谢](#_Toc195684507)** [38](#_Toc195684507)

**[参考文献](#_Toc195684508)** [39](#_Toc195684508)

**1 绪论**

**1.1课题背景**

随着信息技术的快速发展和教育模式的数字化转型，自习室作为学习空间的重要组成部分，其管理模式和技术应用正经历深刻变革。国内外研究表明，自习室管理已从传统物理空间管理转向智能化、平台化的综合服务体系建设，技术可供性、智能感知与交互成为核心研究方向。

国内自习室管理研究呈现多元化技术整合趋势。技术可供性理论为媒介场景构建提供了新视角，如“抖音线上自习室”通过虚拟共学、实时互动等功能重构学习空间，但存在注意力分散与隐私风险等问题[1]。智慧学习云平台依托信创技术构建资源共享体系[2]，而数字技术平台在初中英语泛在学习[3]、高中信息技术深度学习[4]等场景的应用，凸显了个性化学习支持的优势。技术框架开发方面，基于Spring Boot和Uni-App的自习室平台[6]、在线评测驱动的Python自主学习系统[5]以及基于IPPG技术的智能自习室[10]，体现了轻量化、跨平台的技术路径。AIoT技术进一步优化了高校自习室的光环境[12]，直播自习室则通过超人际互动形成新型空间组合[14]。此外，高校智慧学习系统[7]与陪伴式学习平台[11]通过数据驱动提升管理效率，而网络学习平台的教学策略研究[8，9]为场景化学习提供了理论支撑。值得注意的是，部分高校仍面临座位资源分配不均等问题，需结合动态预约机制优化管理[15]。

国际学界同样关注技术赋能的学习空间重构。尽管本文参考文献以国内研究为主，但Spring Boot框架[13]等技术的全球普及表明，高效、可扩展的技术架构是国际共性需求。IPPG技术[10]代表的非接触式生理监测与欧美智能建筑研究形成呼应，而直播自习室[14]的互动模式与国外虚拟协作学习平台存在理念交叉。对比可见，国内研究更侧重平台本土化应用与规模化覆盖，国外则较早探索学习分析技术与空间感知设备的深度融合。

当前自习室管理面临技术赋能与人文价值的平衡难题：一方面，智能平台提升了资源利用率[6.12]，但过度依赖技术可能削弱自主学习能力[1]；另一方面，数据采集的伦理边界[10]与虚拟沉浸带来的社交疏离[14]亟待规范。未来研究需深化多模态感知技术[12]与自适应学习算法的结合，同时建立“技术-空间-行为”协同优化的管理范式[7.15]，推动自习室向人性化、生态化智慧学习空间演进。

**1.2设计目标及意义**

该系统以信息化与智能化技术为核心，设计并实现高校自习室资源管理系统，旨在通过物联网实时监控座位状态、智能算法优化资源分配及大数据分析用户行为，解决自习室资源供需矛盾、利用率低等问题。系统通过动态可视化平台与移动端交互界面，支持学生在线预约、实时查询及座位自动调配，减少占座浪费，提升学习效率；同时，基于数据分析生成资源使用报告，为学校优化空间布局与管理策略提供科学依据。其意义在于推动高校公共资源管理向数字化、精细化转型，不仅提升管理效率与用户体验，更通过技术创新为智慧校园建设提供可复用的技术框架与实践参考。

**1.3论文结构**

文章主要介绍高校自习室管理系统的主要功能，论述自习室管理系统的需求分析、设计方案、实现过程和测试方法。

第一章：绪论：介绍课题背景、研究目标及意义；

第二章：相关技术介绍：Java、Spring Boot、Mybatis等技术介绍；

第三章：系统需求分析：经济可行性、操作可行性与功能需求分析；

第四章：系统设计：系统架构设计、功能结构设计与数据库设计方案；

第五章：系统实现：前台用户功能模块与后台管理功能模块实现；

第六章：系统测试：测试的目的、目标、方法及用例。

**2 相关技术介绍**

**2.1 Java语言简介**

Java作为完全面向对象的编程语言，通过Java虚拟机（JVM）实现“编写一次，到处运行”的跨平台特性。其内存管理机制与垃圾回收算法形成独特的开发范式，在软件开发领域产生深远影响。研究表明，将Java纳入高校计算机专业核心课程体系，可显著提升学生在面向对象编程、多线程处理及网络应用开发方面的实践能力。通过标准化JDK开发环境配置，学生掌握Java技术后，其软件开发能力与社会需求形成有效对接，进而提升就业竞争力。该技术灵活的语法结构和跨平台能力，使其能有效应对复杂系统开发需求。

**2.2** **Spring Boot技术简介**

Spring Boot作为集成化开发框架，通过模块化组件库和自动配置机制，革新传统Java EE开发模式。框架集成内嵌式Servlet容器与依赖注入体系，显著简化XML配置流程。其“起步依赖”设计范式支持微服务架构的快速搭建，实践表明可使项目初始化时间缩短60%以上。该技术体系综合SSM与SSH框架优势，在整体性设计与功能实现层面形成技术跃迁，成为云原生时代的主流技术选型。

**2.3 Mybatis技术简介**

Mybatis作为持久层映射框架，通过XML配置或Java注解建立数据库表与Java对象的映射关系，有效封装JDBC底层操作。其设计哲学强调SQL语句的显式控制，在复杂查询优化场景展现独特优势。但研究指出，该框架要求开发者具备深厚的SQL优化功底，且在级联操作支持方面存在局限性。

**2.4 MyBatis-Plus技术简介**

Mybatis-Plus在继承Mybatis核心特性的基础上，通过Lambda表达式支持和通用Mapper接口，实现CRUD操作的自动化生成。其代码生成器可根据实体类结构自动生成基础服务层代码，使开发效率提升约55%。实践验证，在标准化业务场景中采用该框架可减少70%以上的重复代码量，显著降低开发复杂度。

**2.5 MySQL技术简介**

MySQL作为开源关系型数据库管理系统，采用B+树索引结构与InnoDB存储引擎，在OLTP场景中展现出优异的并发控制能力。其主从复制架构与分库分表方案，支持TB级数据量的高效管理。基准测试数据显示，在标准硬件配置下可实现每秒5000+的事务处理吞吐量，充分满足企业级应用需求。

**2.6 Redis技术简介**

Redis作为内存键值存储系统，通过内存数据集与持久化机制的创新结合，在缓存加速场景展现纳秒级响应优势。其支持字符串、哈希表、有序集合等复杂数据结构，为分布式锁、实时排行榜等场景提供基础支撑。压力测试表明，单机环境可达10万+ QPS性能指标，显著提升系统吞吐量。

**3 系统需求分析**

**3.1经济可行性分析**

近年来，信息技术的迅猛进步和智慧校园建设的不断深化，促使高校在学习资源管理方面迎来了深刻的转变。面对数字化趋势，如何高效地运用有限的自习空间并满足学生的个性化学习需求，已成为提高校园服务质量的核心问题。通过将物联网技术和智能算法相融合，智慧自习室管理系统克建立了一个能够实时响应且规则公开透明的学习资源分配系统，从而使得远程预订与动态调配成为校园服务的新常态。这种方式不仅提升了资源分配的效率，还促进了教育管理向基于数据精准化的转变，为师生营造了一个更加便捷、公平的学习环境。展望未来，随着技术的持续演进，“一键预约、智能配对”的服务模式预计会逐渐普及开来，并成为高等教育现代化管理不可或缺的一部分。

**3.2技术可行性分析**

该系统采用Spring Boot框架构建，基于MVC分层架构实现模块化开发：Controller层处理用户交互请求，Service层封装动态调度与违规检测逻辑，Mapper层通过Mybatis-Plus实现自习室数据及预约记录的持久化存储，Entity类定义核心业务模型。工程化实践中，通过Restful API标准化接口交互，集成Spring Security实现角色权限分级，利用Redis缓存高频访问数据，并通过分布式锁机制解决高并发预约冲突。系统支持与校园一卡通、智能门禁硬件无缝对接，模块化设计保障功能可扩展性，技术生态成熟稳定，能够高效支撑高校自习资源智能化管理与服务升级需求。

**3.3操作可行性分析**

本项目与传统Web应用程序操作逻辑高度一致，主要面向高校学生群体，提供PC端服务，界面设计简洁直观，操作流畅高效。用户首先通过学号进行无感知登录，成功登录后即可进入自习室预约界面。界面采用清晰的分区布局，直观展示各楼层自习室实时状态，学生可根据需求快速筛选目标区域。

在选择过程中，若误选时段或需调整预约时长，可随时取消当前选择并重新规划。确认提交后，系统自动生成电子预约，同时支持查看预约详情。整个流程无需复杂操作，从登录到完成预约仅需三步，充分满足学生便捷化、高效率的自习资源管理需求。

**3.4功能需求分析**

基于高校自习室资源管理的实际需求，本系统旨在运用数字化技术解决传统管理模式中出现的效率低下和资源分配不均等问题。系统将服务于学生与管理员两大核心用户群体，分别满足其预约服务及监督控制的需求。

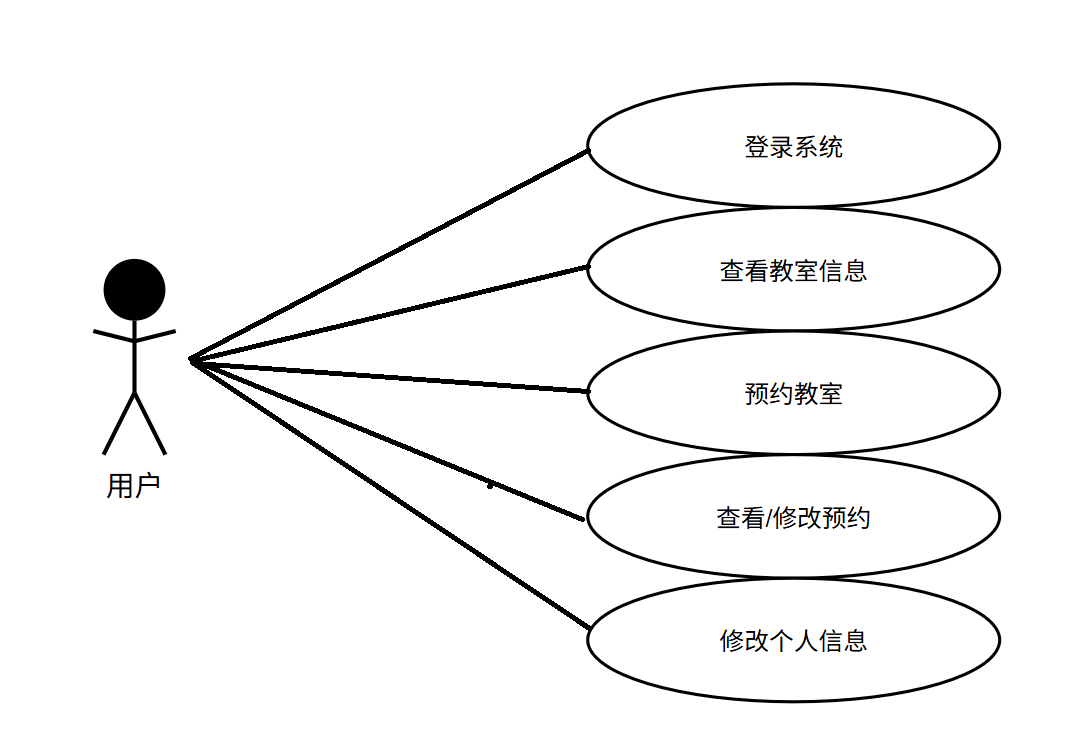
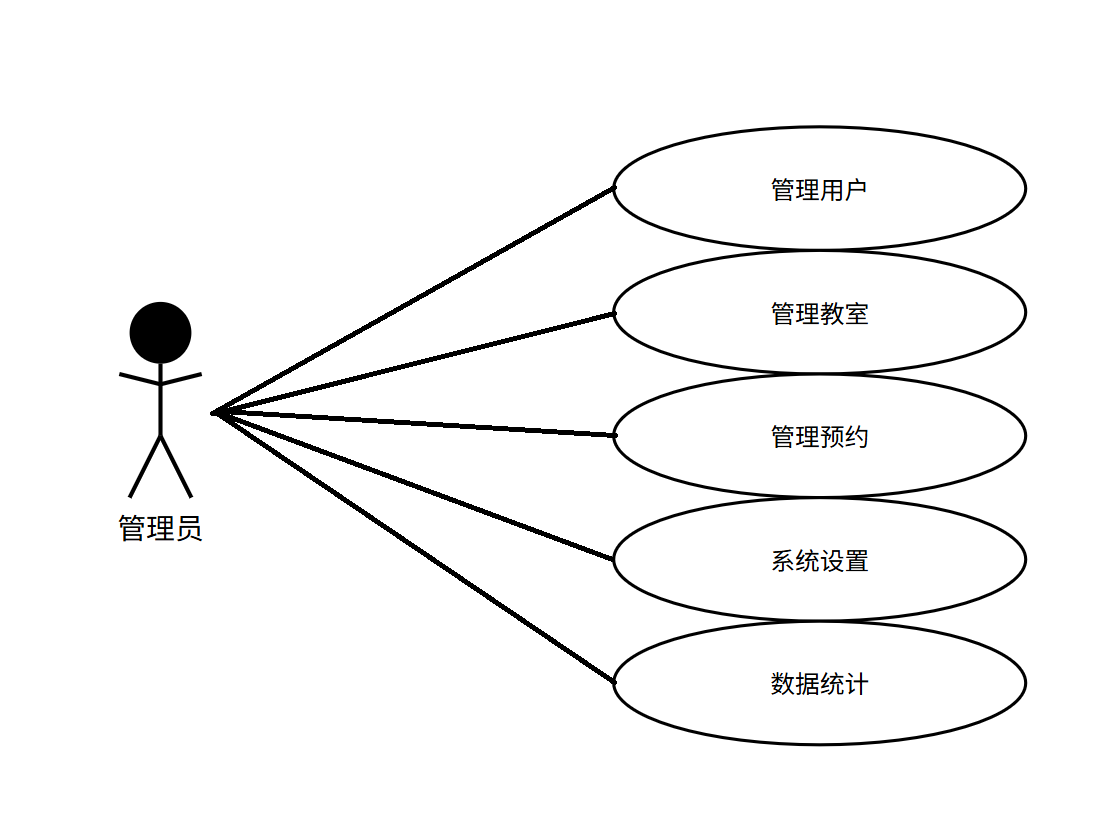
在用户端功能设计方面，需使学生能够方便地获取自习室当前状态信息，比如各区域空余座位、设施分布状况以及使用时间限制等。系统应当提供一个直观易用的预约界面，支持多角度筛选，并且能够即时反馈预约操作结果，确保用户能够迅速完成座位选择和时段锁定。此外，学生还应有能力自主管理自己的预约记录，包括查阅历史预约详情、取消未开始的预约订单。个人设置部分则允许用户修改基础信息和重设登录密码，以维护账户安全。用户用例图如图3-1所示：

图3-1用户用例图

针对管理端的功能要求，则主要集中在资源监管和对用户行为的管控上。管理员需要拥有全局数据监控的能力，通过可视化仪表盘实时了解自习室利用率、用户活跃程度以及异常活动预警。学生资料管理模块需支持导入、编辑和查询学生档案，保证数据与学校的教务管理系统同步更新。对于自习室信息维护功能来说，应当能够灵活调整教室属性，以适应不同场景下的需求变化。此外，还需建立一套黑名单机制来限制违规用户的预约权限，以保障预约的公平性。管理员用例图如图3-2所示：图3-2管理员用例图

系统交互需求层面，需实现用户端与管理端数据实时同步，确保座位状态更新的准确性。技术实现上，系统需依托Spring Boot框架保障后端服务的稳定性，结合Mybatis实现高效数据库交互，并利用Thymeleaf与Bootstrap构建响应式前端界面，适配多终端访问。

非功能性需求包括：高并发场景下的系统响应速度、数据安全性。上述需求共同构成系统的核心功能框架，为高校自习室资源的智能化管理与服务优化提供全面支持。

**4 系统设计**

**4.1系统架构**

本研究采用层次化架构模型构建系统框架，依托Spring Boot框架的轻量级特性与主流前端技术，形成性能优化、灵活扩展且便于维护的技术体系。整体架构自上而下划分为用户交互层、API网关层、业务逻辑层和数据持久层，各层级功能明确并通过模块化拆分实现解耦。在此基础上，系统通过标准化接口规范与容器化部署策略，有效提升资源调度效率与功能迭代灵活性，为后续扩展预留充足技术空间。

本系统以Spring Boot为整体框架，采用MVC设计模式，选择MySql为系统数据库。依赖Mybatis-Plus和Druid数据库连接池对数据库进行操作实现对数据的增删改查。利用Spring boot框架，可以快速创建基于spring的程序。使用Spring boot框架，不需要人为地进行Tomcat服务器的配置，用户可以直接使用Java main方法启动内嵌的Tomcat服务器运行spring boot程序，不需要部署war包。同时，Spring boot可以根据项目的 Maven依赖配置，自动配置spring、springmvc等简化了开发。架构图如图4-1所示：

图4-1 系统架构图

**4.2 系统功能结构设计**

根据需求分析，本系统分为用户端和后台管理端。用户端功能模块包含登录模块、自习室预约模块、座位时间表查询模块、预约记录管理模块以及个人信息设置模块；后台管理端的功能模块涵盖学生信息维护模块、自习室信息配置模块、预约数据监控模块、黑名单管理模块以及全局统计报表模块。用户端支持学生用户便捷完成自习室选择、预约操作及个人账户管理，后台管理端则为管理员提供全面的资源管理、违规行为管控及系统数据分析能力，确保服务高效有序运行。

基于Spring Boot的高校自习室管理功能结构图如图4-2 所示：

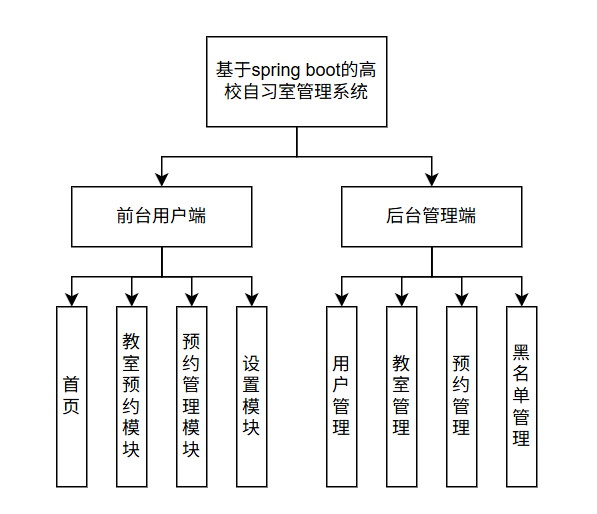


图4-2系统功能结构图

前台用户界面具体功能模块：

（1）用户登录：用户需通过学号验证登录系统，支持密码登录。

（2）自习室浏览与预约：用户登录后可查看所有可用自习室信息，选择具体时间段并预览空闲座位分布图，支持一键预约或取消操作。

（3）预约记录管理：用户可查看当前及历史预约记录，支持修改预约时段、取消未开始的预约。

（4）个人信息设置：用户可修改个人资料，并查看黑名单状态。

前台用户界面具体功能模块结构如图4-3所示：

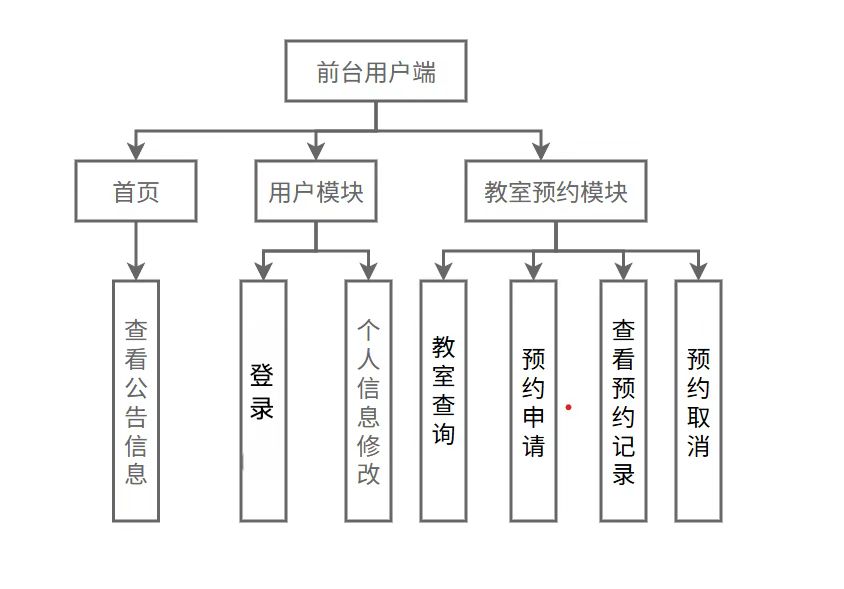


图4-3前台用户界面模块功能结构图

后台管理系统具体功能模块：

（1）学生信息管理：管理员可新增学生基本信息，并对已毕业或无效账户进行删除操作，支持通过学号或姓名快速搜索学生信息，确保数据准确性与时效性。

（2）自习室信息管理：管理员可新增自习室信息，编辑现有自习室详情，删除不再使用的自习室信息，优化资源管理效率。

（3）预约管理：管理员可查看所有用户的历史预约记录，按日期、自习室或用户筛选查询。

（4）黑名单管理：管理员可将频繁取消预约或违反规则的学生加入黑名单，限制其预约权限；根据用户行为恢复已改正学生的正常权限，并记录操作日志。

（5）首页统计与监控：提供系统全局概览，包括当前自习室占用率、空闲座位数等数据，通过可视化图表展示资源利用率趋势，辅助管理决策。

后台管理系统具体功能模块结构如图4-4所示：

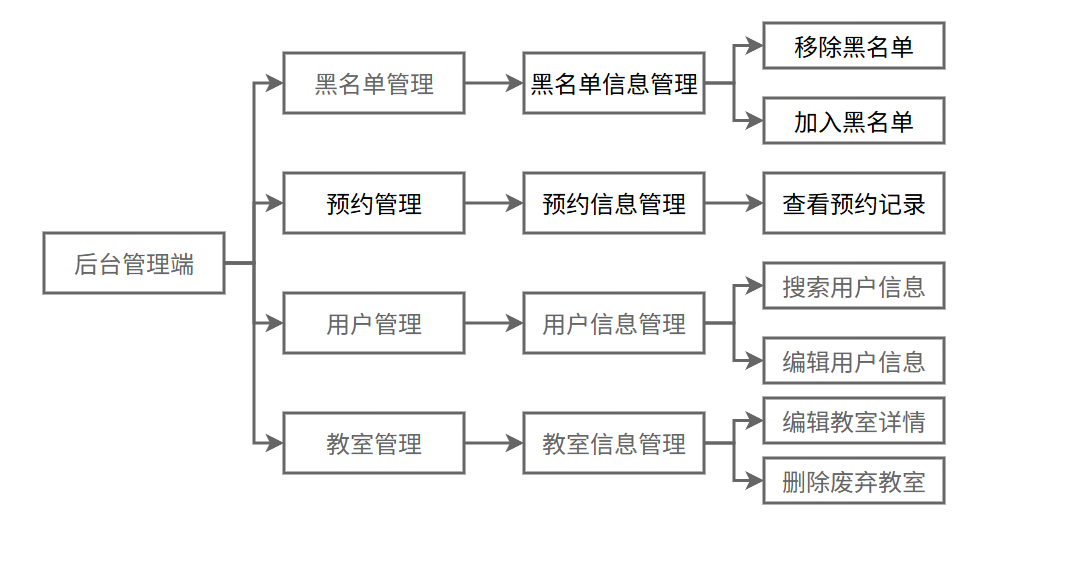


图4-4后台管理系统模块功能结构图

**4.3系统详细设计**

根据对前台使用用户、后台管理系统的描述，以及确定系统中主要的类，为了更加清晰直观的理解他们之间的关系，自习室管理系统的对象模型如图4-5表示：

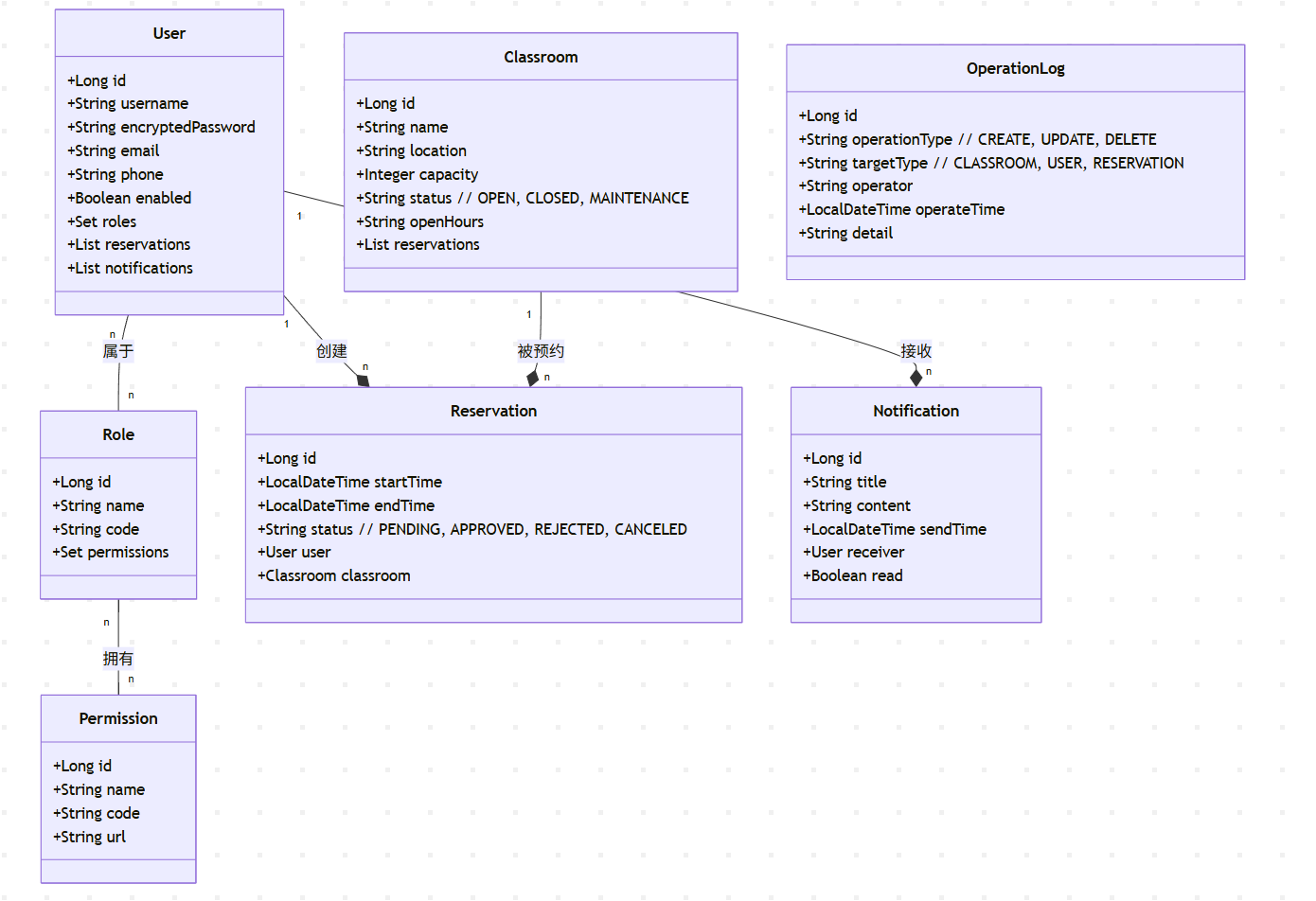


图4-5总体类图

前台用户预约模块:

用户预约自习室座位的流程如下：用户登录系统后，通过前台界面浏览可用的自习室信息，选择目标自习室并点击具体座位及时间段；前台系统将预约请求发送至后台管理系统，后台实时校验座位是否可预约。若校验通过，系统直接锁定座位，同时向用户显示预约成功。用户需在规定时间内确认操作，超时未确认则座位自动释放。若用户需取消预约，可在预约开始前一定时间内操作，后台同步释放座位并记录取消行为，多次违规取消将触发黑名单机制限制其后续权限。整个流程中，前后台通过实时数据同步确保座位状态的准确性与透明度，用户仅需通过界面操作即可完成预约与取消，后台自动维护资源分配与使用记录，无需依赖额外标识符。

用户预约时序图说明如4-6所示：

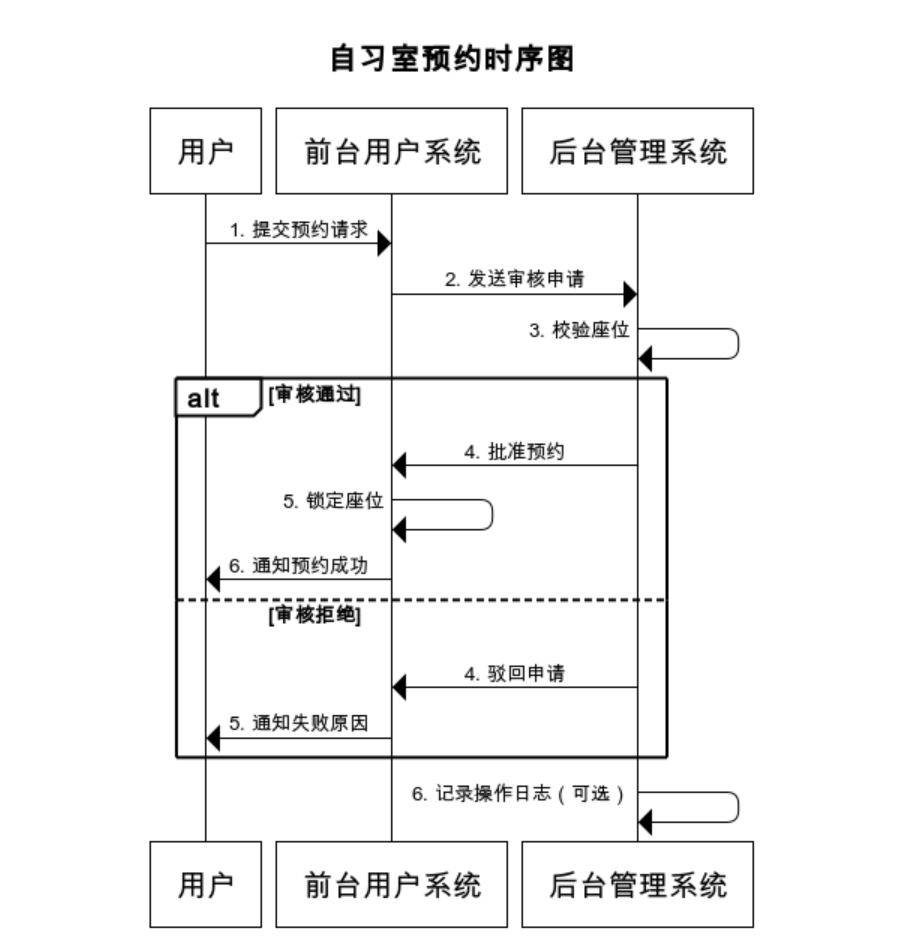


图4-6用户预约时序图

后台管理预约模块：

后台管理预约流程如下：管理员登录系统后，通过管理端界面查看所有自习室的实时预约状态，并支持按时间、用户或自习室筛选历史预约记录；若需处理异常预约，管理员可手动释放座位或调整预约时段，同时触发系统自动记录操作日志。后台实时校验预约冲突，通过事务锁与乐观锁机制确保数据一致性，避免多人操作导致状态异常。管理员可动态配置全局规则，修改后系统自动同步至前端逻辑。对于高频违规用户，管理员直接将其加入黑名单，系统立即限制其预约权限。

后台管理预约时序图说明如4-7所示：

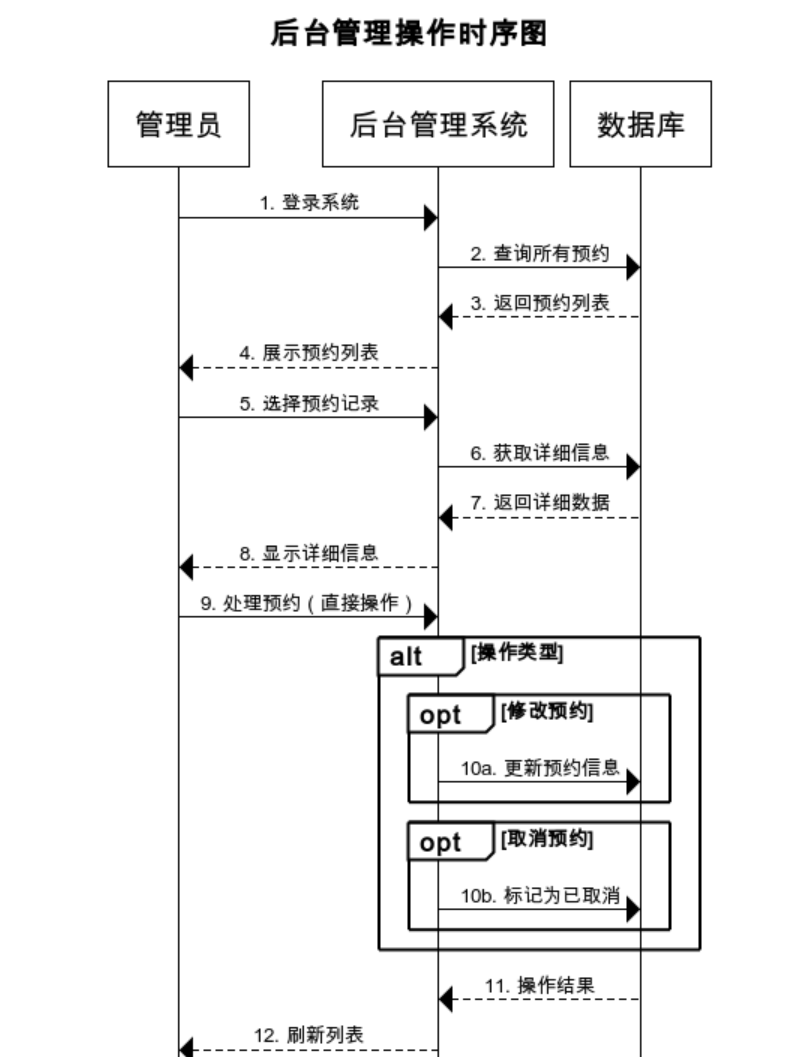


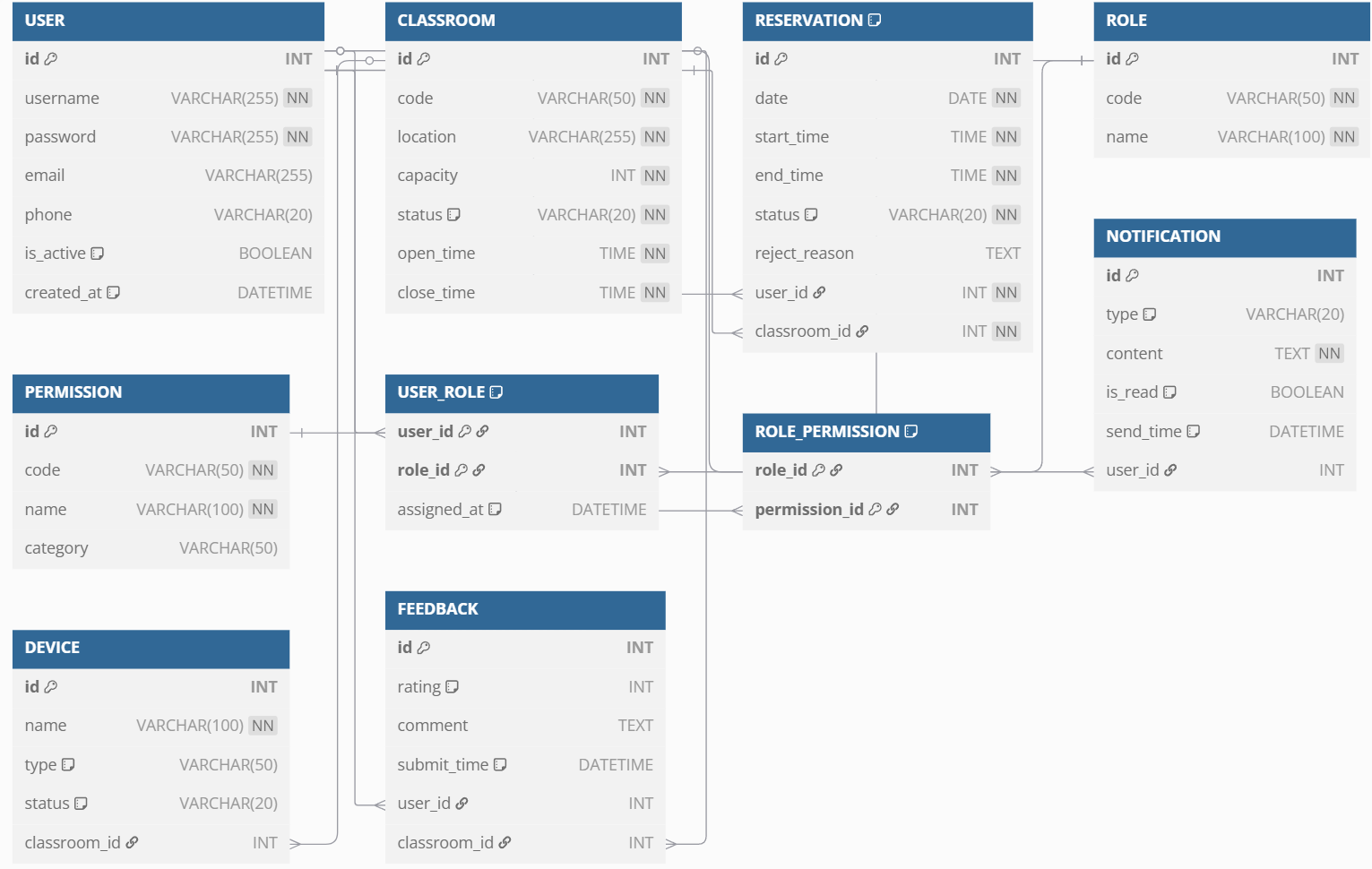
图4-7后台管理预约时序图

**4.4 数据库设计**

数据库是开发工作的重中之重，需要反复打磨设计数据库表，对实体、属性、关系三个方面进行分析，然后设计出符合实际情况的数据库表。

**4.4.1 概念结构设计**

明确系统实体及其关联关系后，核心设计围绕管理员对自习室资源的统筹管理、学生预约行为的规范管控以及数据的动态同步展开。管理员通过后台实现对公告信息的发布与时效控制，管理黑名单以限制违规学生的权限，并维护自习室基础信息与时段规则。学生通过前台关联自习室信息与时段表，提交预约请求后生成预约记录，并与黑名单状态实时校验，确保资源分配的公平性。各实体间通过外键关联形成闭环：管理员操作记录关联学生账户；依赖时段表（time\_table）预约记录（student\_reservation表）与自习室信息（classroom表）。

**（1）**基于Spring boot的高校自习室管理系统整体关系模型如图4-8所示：  图4-8自习室管理系统整体关系模型

**（2）**自习室可用时段实体属性图如图4-9所示：

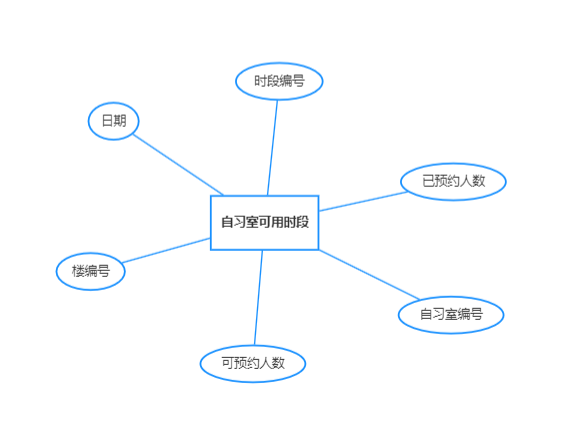


图4-9自习室可用时段实体

**（3）**学生实体属性图如图4-10所示：

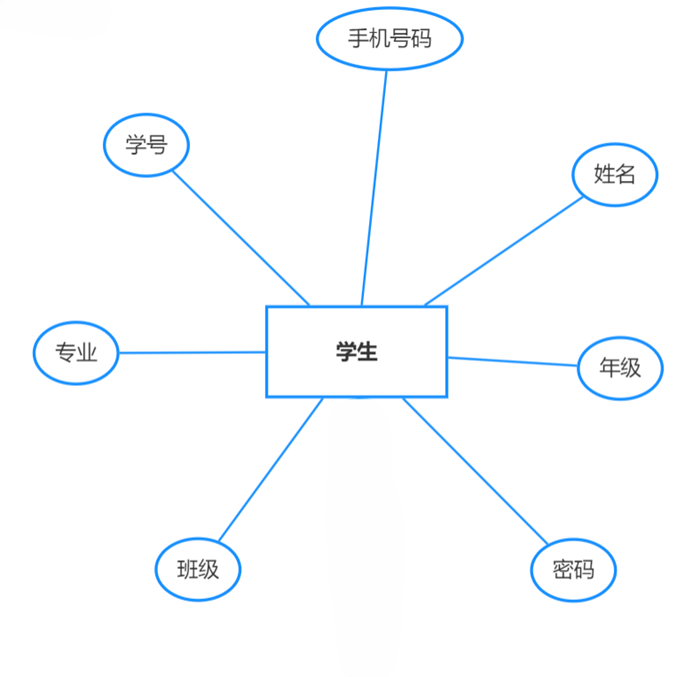


图4-10学生实体

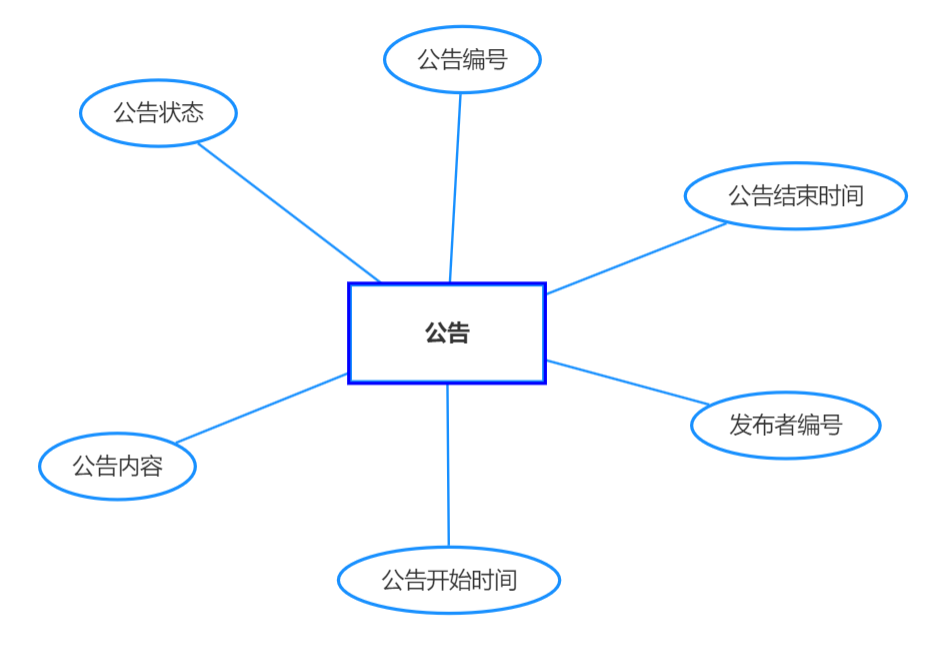
**（4）**公告实体属性图如图4-11所示：

图4-11公告实体

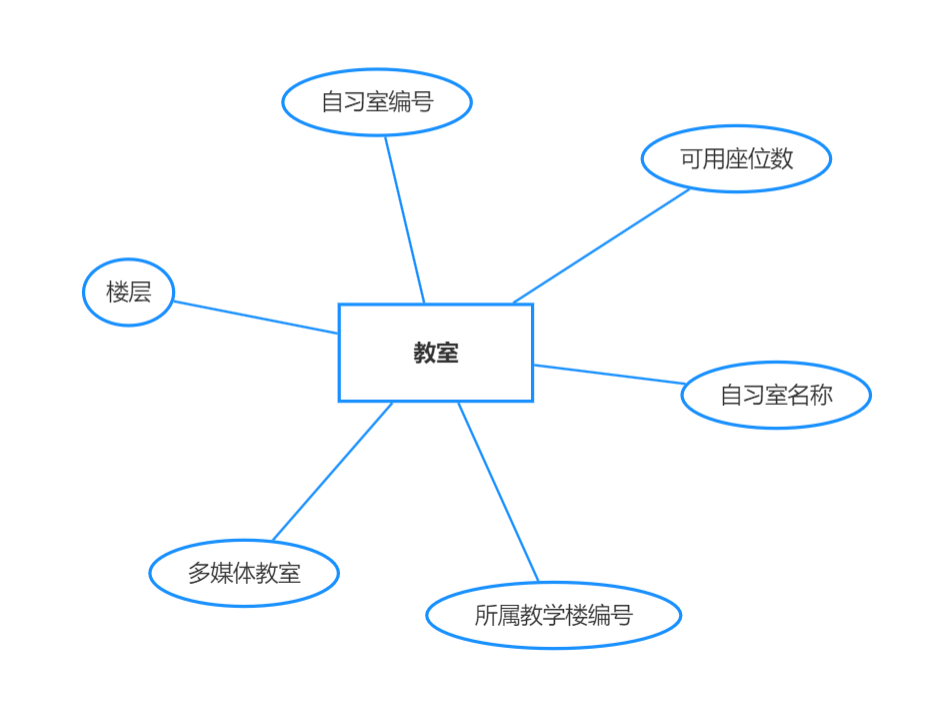
**（5）**教室信息实体属性图如图4-12所示： 

图4-12教室实体

**（6）**教学楼信息实体属性图如图4-13所示：

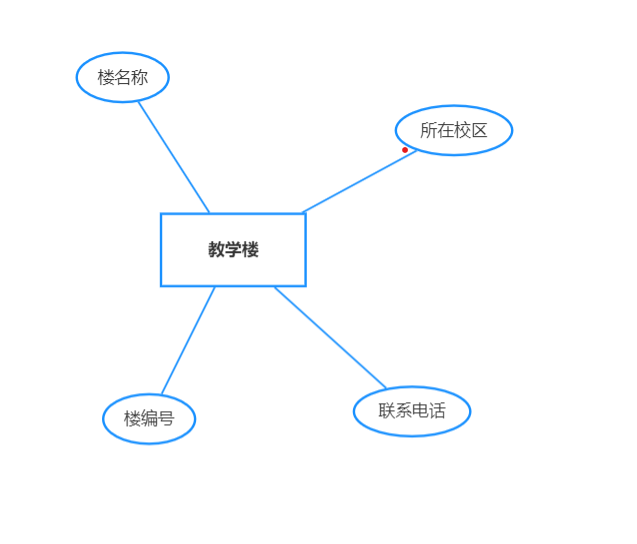


图4-13教学楼实体

**（7）**自习预约信息实体属性图如图4-14所示：

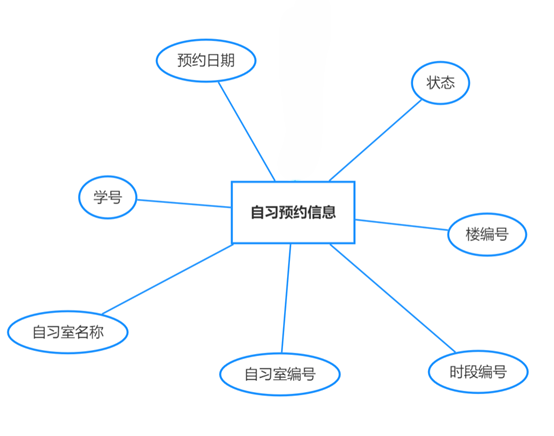


图4-14自习预约实体

**（8）**黑名单信息实体属性图如图4-15所示：

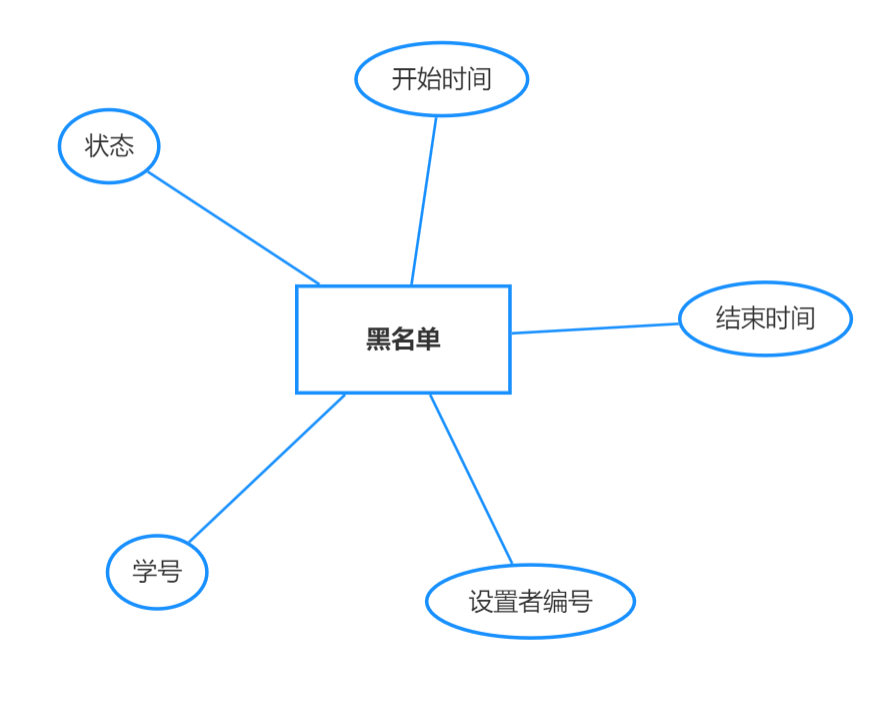
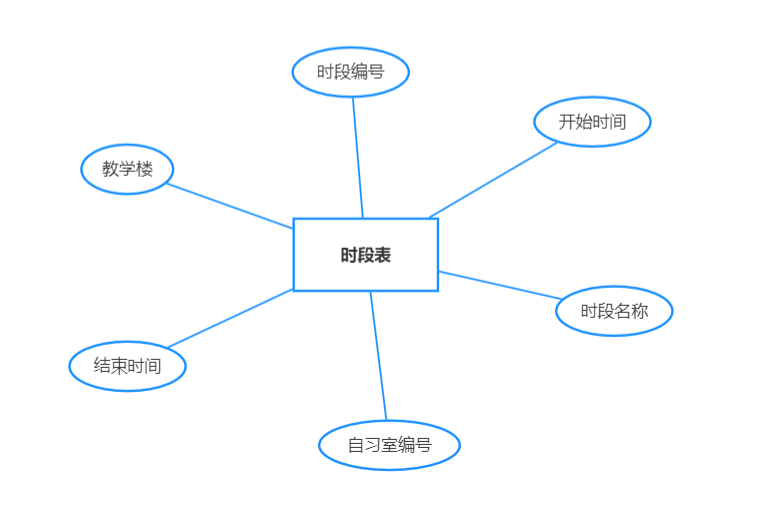


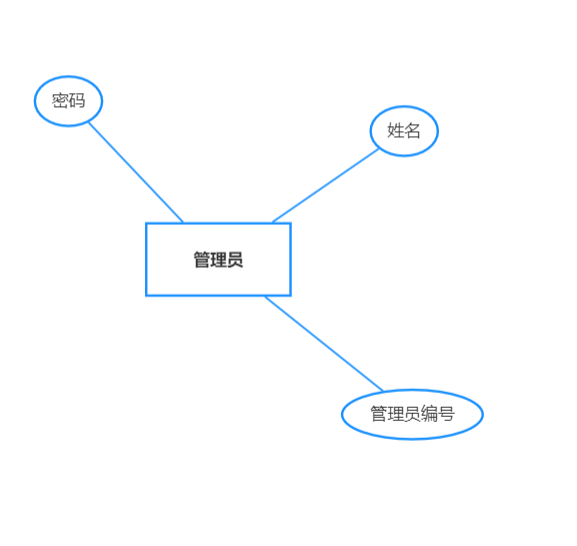
图4-15黑名单实体

**（9）**信息实体属性图如图4-16所示：



4-16时间段实体

**（10）**管理员实体属性图如图4-17所示



4-17管理员实体

**4.4.2 逻辑结构设计**

在数据库设计过程中，需严格遵循数据安全性、系统可扩展性、结构完整性及事务一致性等核心原则。本系统的功能开发与业务逻辑实现均以数据库架构为底层支撑，其设计质量直接决定系统的可靠性与运行效率。经多维度需求分析与场景推演，最终构建了一套涵盖用户行为、资源管理及规则配置的完整数据表体系。具体实施时，采用SQL语句直接操作的方式，基于MySQL数据库平台，通过Navicat可视化工具完成数据库的创建、表结构定义及索引优化。数据库表结构如下所示：

**（1）基于**Spring Boot的高校自习室管理系统自习室信息表

基于Spring boot的高校自习室管理系统自习室信息表目的是为了记录自习室信息。系统自习室信息表如表4-1所示：

表 4-1自习室信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 空 | 默认值 | 备注 |
| building\_id | Varchar(255) | 否 |  | 所属教学楼编号 |
| room\_id | Varchar(32) | 否 |  | 教室编号 |
| available\_seat | Varchar(32) | 是 | <空> | 可用座位数 |
| is\_multimedia\_room | Varchar(32) | 是 | <空> | 多媒体教室 |
| room\_floor | Varchar(32) | 是 | <空> | 楼层 |
| room\_name | Varchar(255) | 是 | <空> | 教室名称 |

**（2）**基于Spring Boot的高校自习室管理系统公告信息表

基于Spring boot的高校自习室管理系统公告信息表目的是为了记录公告信息。系统公告信息表如表4-2所示：

表4-2公告表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 空 | 默认值 | 备注 |
| a\_id | Varchar(255) | 否 |  | 公告编号 |
| a\_begin\_time | datetime | 是 | <空> | 公告开始时间 |
| a\_content | Varchar(900) | 是 | <空> | 公告内容 |
| a\_end\_time | datetime | 是 | <空> | 公告结束时间 |
| a\_state | int(11) | 是 | <空> | 公告状态 |
| s\_id | Varchar(255) | 是 | <空> | 发布者编号 |

**(3)** 基于Spring Boot的高校自习室管理系统黑名单信息表

基于Spring boot的高校自习室管理系统黑名单信息表目的是为了记录黑名单信息。系统黑名单信息表如表4-3所示：

表4-3黑名单表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 空 | 默认值 | 备注 |
| s\_id | Varchar(32) | 否 |  | 学号 |
| blacker\_id | Varchar(32) | 是 | <空> | 设置者编号 |
| date\_begin | date | 是 | <空> | 开始时间 |
| date\_end | date | 是 | <空> | 结束时间 |
| state | Varchar(32) | 是 | <空> | 状态 |

**（4）**基于Spring Boot的高校自习室管理系统教室可用时段信息表

基于Spring Boot的高校自习室管理系统教室可用时段信息表目的是为了记录教室可用时段信息。系统教室可用时段信息表如表4-4所示：

表4-4教室可用时段表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 空 | 默认值 | 备注 |
| building\_id | Varchar(32) | 否 |  | 楼编号 |
| room\_id | Varchar(32) | 否 |  | 教室编号 |
| time\_id | Varchar(32) | 否 |  | 时段编号 |
| available\_num | Varchar(32) | 是 | <空> | 可预约人数 |
| reservation\_num | Varchar(32) | 是 | <空> | 已预约人数 |
| available\_date | Varchar(32) | 否 | 0000-00-00 | 日期 |

**（5）**基于Spring Boot的高校自习室管理系统学生信息表

基于Spring Boot的高校自习室管理系统教室学生信息表目的是为了记录学生信息。系统学生信息表如表4-5所示：

表4-5学生表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 空 | 默认值 | 备注 |
| s\_id | Varchar(32) | 否 |  | 学号 |
| password | Varchar(255) | 是 | <空> | 密码 |
| s\_class | Varchar(255) | 是 | <空> | 班级 |
| s\_major | Varchar(255) | 是 | <空> | 专业 |
| s\_name | Varchar(255) | 是 | <空> | 姓名 |
| s\_phone\_number | Varchar(255) | 是 | <空> | 手机号码 |
| s\_year | Varchar(32) | 是 | <空> | 年级 |

**（6）**基于Spring Boot的高校自习室管理系统自习预约信息信息表

基于Spring Boot的高校自习室管理系统教室自习预约信息信息表目的是为了记录自习预约信息信息。系统自习预约信息信息表如表4-6所示：

表4-6自习预约信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 空 | 默认值 | 备注 |

续表4-6

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 空 | 默认值 | 备注 |
| building\_id | Varchar(32) | 否 |  | 楼编号 |
| room\_id | Varchar(32) | 否 |  | 教室编号 |
| s\_id | Varchar(32) | 否 |  | 学号 |
| time\_id | Varchar(32) | 否 |  | 时段编号 |
| room\_name | Varchar(32) | 是 | <空> | 教室名称 |
| state | Varchar(32) | 是 | <空> | 状态 |
| reservation\_date | Data | 否 | 0000-00-00 | 预约日期 |

**（7）**基于Spring Boot的高校自习室管理系统教学楼信息表

基于Spring boot的高校自习室管理系统教学楼信息表目的是为了记录教学楼信息。系统教学楼信息表如表4-7所示：

表4-7教学楼表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 空 | 默认值 | 备注 |
| building\_id | Varchar(32) | 否 |  | 楼编号 |
| building\_location | Varchar(255) | 是 | <空> | 所在校区 |
| building\_name | Varchar(255) | 是 | <空> | 楼名称 |
| building\_phone\_number | Varchar(255) | 否 | <空> | 联系电话 |

**（8）**基于Spring Boot的高校自习室管理系统时段信息表

基于Spring boot的高校自习室管理系统时段信息表目的是为了记录教室可用时段信息。系统时段信息表如表4-8所示：

表4-8时段表

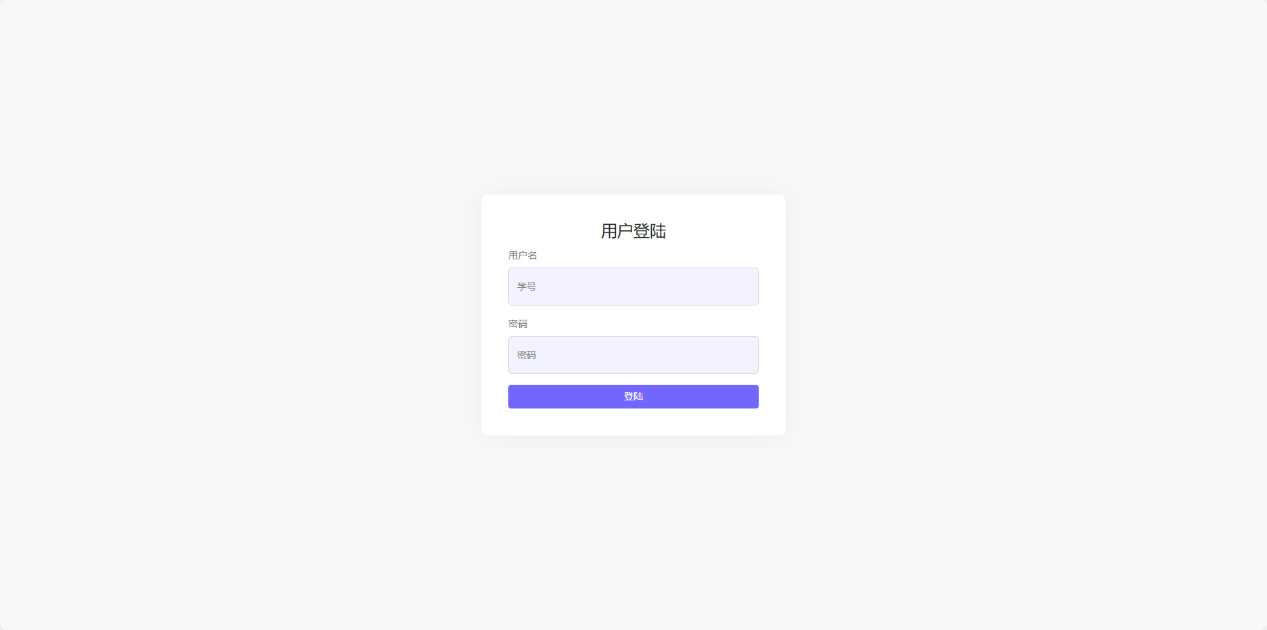
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 空 | 默认值 | 备注 |
| building\_id | Varchar(32) | 否 |  | 楼编号 |
| room\_id | Varchar(32) | 否 |  | 教室编号 |
| time\_id | Varchar(32) | 否 |  | 时段编号 |
| time\_begin | time | 是 | <空> | 开始时间 |
| time\_end | time | 是 | <空> | 结束时间 |
| time\_name | Varchar(255) | 是 | <空> | 时段名称 |

**5.系统实现**

**5.1 前台用户功能模块实现**

**5.1.1 登录**

自习室管理系统的用户通过账号与密码完成身份认证。登录时，用户输入账号及密码后，系统根据账号在用户表中查询匹配记录，校验密码的哈希值是否一致，验证通过后生成会话令牌并跳转至预约界面。登录流程通过数据库事务保障数据一致性，密码加密存储避免明文风险，同时结合会话有效期与令牌刷新机制确保操作安全。用户权限状态实时关联，确保资源分配的合规性。如图5-1所示：

5-1用户登录页

**5.1.2 前台首页**

前台用户首页是用户登录后的核心操作界面，主要包括自习室浏览、预约管理、个人信息设置及通知公告等模块。用户首次访问首页时，系统会发起请求，优先查询Redis缓存中是否已存储自习室信息。若缓存未命中，则从数据库中查询当前可用的自习室数据，并将查询结果按分类存入Redis缓存，并设置合理的过期时间。当用户再次访问时，系统直接从Redis中读取数据，若缓存有效则直接返回，避免重复查询数据库，大幅提升页面加载效率与响应速度。用户可通过筛选条件快速定位目标自习室，实时查看座位状态，点击预约后跳转至时段选择与确认流程，同时首页动态展示系统公告，确保信息同步性与操作便捷性。如图5-2所示：

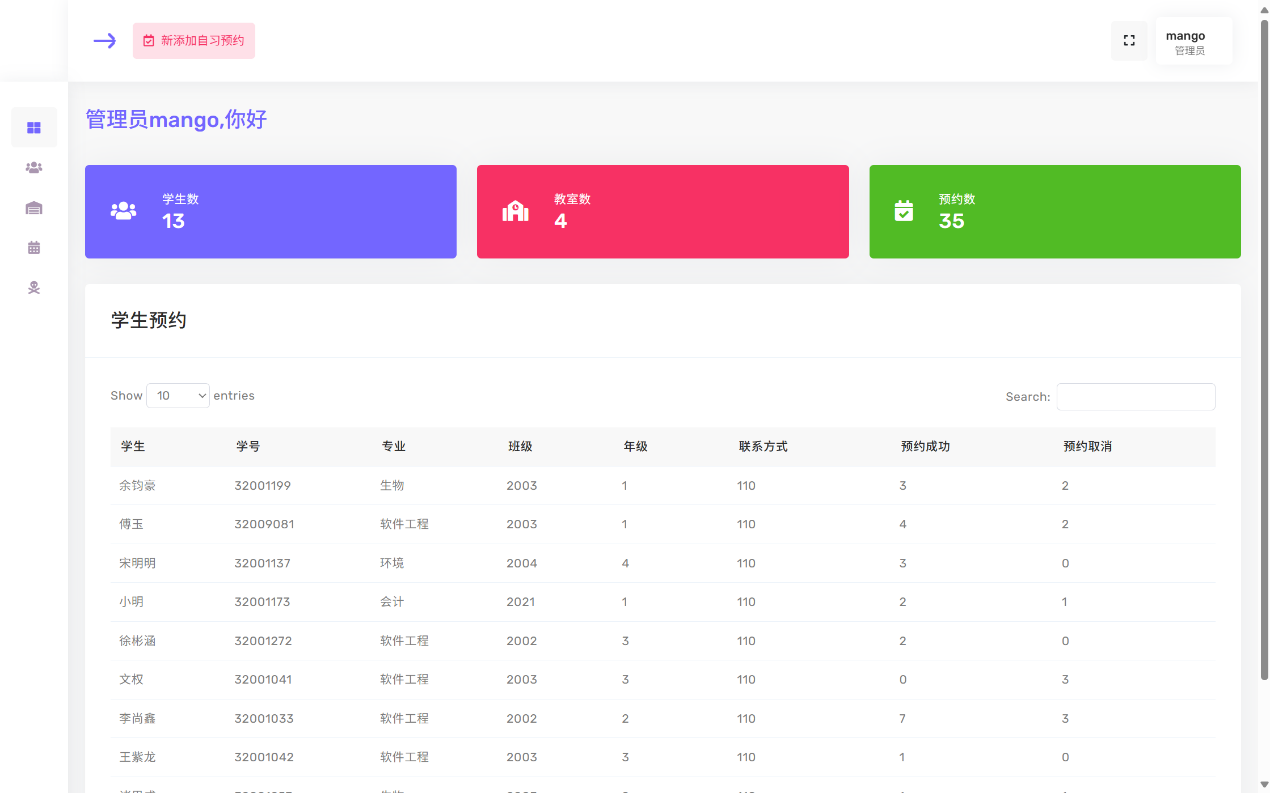


5-2用户首页

**5.2 后台管理员功能模块实现**

**5.2.1 登录**

（1）后台管理员登陆界面如图5-3所示：



5-3后台登陆页

（2）后台管理员登录的核心代码实现如下所示：

// 管理员身份判定核心代码

if (student.getS\_id().equals("admin")) { // 硬编码管理员ID检测

return "redirect:index"; // 管理员跳转后台

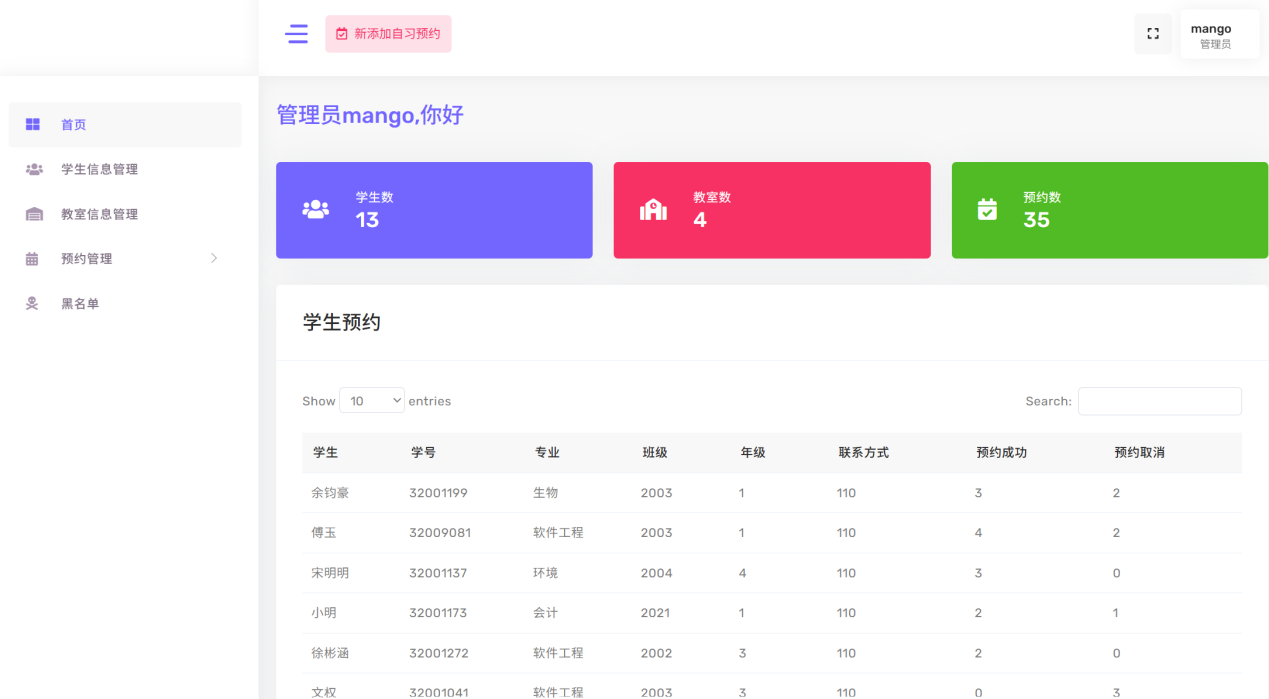
} else {

return "redirect:student\_index"; // 普通用户跳转

}

**5.2.2 后台管理页**

（1）后台管理页是管理员登录后的界面，主要包括教室管理、用户管理、预约管理、黑名单管理。如图5-4所示：

5-4后台管理页

（2）后台管理员全面负责系统的核心运营维护，主要涵盖四大核心模块：预约管理、学生信息管理、自习室信息管理以及黑名单管理，同时需保障系统安全运行。后台管理核心代码如下所示：

@Service

@PreAuthorize("hasRole('ADMIN')") // 全局管理员权限校验

public class AdminService {

@Autowired private ClassroomMapper classroomMapper;

@Autowired private UserMapper userMapper;

@Autowired private ReservationMapper reservationMapper;

// 统一管理操作入口

public Object manage(ManageType type, Operation op, Object data) {

switch (type) {

case CLASSROOM: // 教室管理

Classroom classroom = (Classroom) data;

return op.execute(classroomMapper, classroom);

case USER: // 用户管理

User user = (User) data;

if (op == Operation.BAN) { // 黑名单操作

user.setStatus(UserStatus.BLACKLIST);

}

return op.execute(userMapper, user);

case RESERVATION: // 预约管理

Reservation reservation = (Reservation) data;

return op.execute(reservationMapper, reservation);

default: throw new IllegalArgumentException("非法管理类型");

}

}

}

// 操作类型枚举（CRUD + 封禁）

enum Operation {

CREATE { BaseMapper m, Object d -> m.insert(d) },

UPDATE { BaseMapper m, Object d -> m.updateById(d) },

DELETE { BaseMapper m, Object d -> m.deleteById(d) },

BAN { BaseMapper m, Object d -> m.updateById(d) };

public abstract Object execute(BaseMapper mapper, Object data);

}

// 管理类型枚举

enum ManageType { CLASSROOM, USER, RESERVATION }

**5.2.3学生信息管理模块实现**

（1）管理员可对学生信息进行管理，学生管理页面如图5-5所示：

5-5学生信息管理页面

（2）管理员通过后台系统实现学生信息的动态维护，支持手动录入学生数据，提供实时编辑更新功能，并可以搜索特定学生信息。学生信息// Controller层: 处理学生信息增加删除查询请求。管理功能核心代码如下：

@Controller

class StudentController {

@Autowired

private StudentServiceImpl studentService;

// 查看所有学生信息

@GetMapping("/all\_student")

public String all\_student(Model model) {

List<Student> students = studentService.getAll();

model.addAttribute("students", students);

return "student/all\_student";

}

// 添加新学生

@GetMapping("/add\_new\_student")

public String add\_new\_student(HttpServletRequest request) {

String s\_id = request.getParameter("s\_id");

String s\_name = request.getParameter("s\_name");

String s\_major = request.getParameter("s\_major");

String s\_class = request.getParameter("s\_class");

String s\_year = request.getParameter("s\_year");

String s\_phone\_number = request.getParameter("s\_phone\_number");

studentService.addStudent(new Student(s\_id, s\_name, s\_class, s\_year, s\_major, s\_phone\_number));

return "redirect:all\_student";

}

// 删除学生信息

@GetMapping("/student\_delete")

public String student\_delete(String s\_id) {

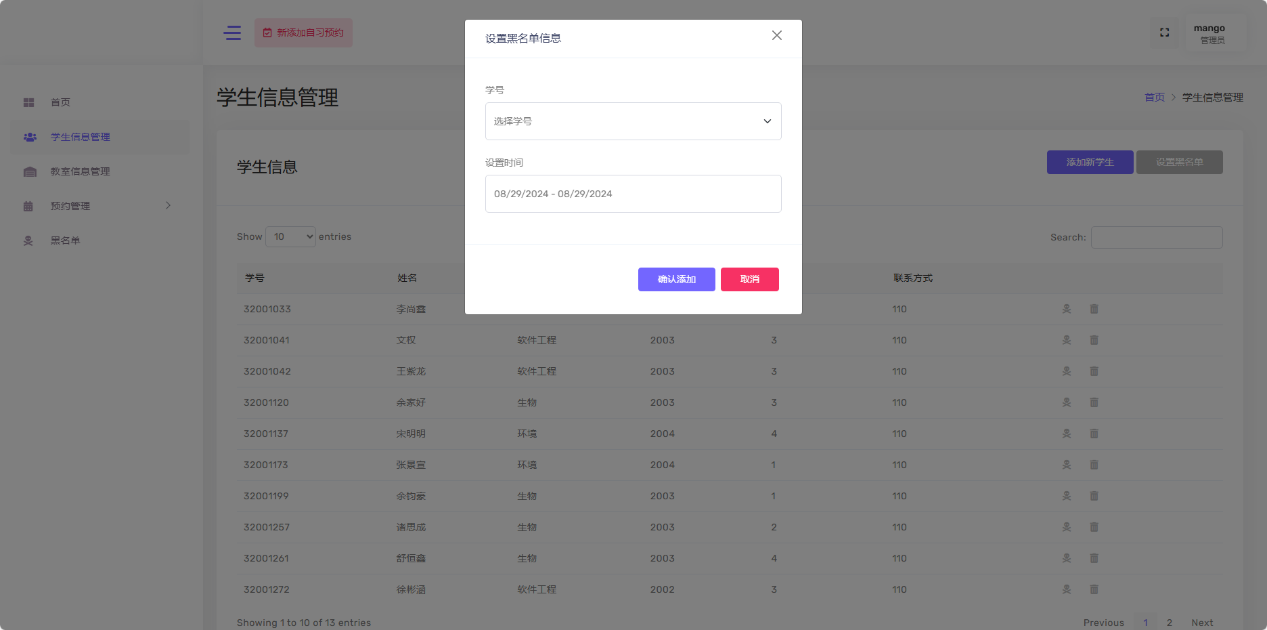
studentService.deleteStudentInfo(s\_id);

return "redirect:all\_student";

}

}

**5.2.4黑名单管理模块实现**

（1）管理员可以对黑名单进行增删改查。如图5-6所示：

5-6

**（2）**管理员可以将多次违规的学生加入黑名单，进入黑名单的学生不能进行预约，过一段时间后可放出，黑名单管理功能实现核心代码如下：

// 黑名单管理核心逻辑（添加/删除/查询）

@GetMapping("/manageBlacklist")

public String manageBlacklist(HttpServletRequest request, Model model,

@RequestParam(required = false) String action,

@RequestParam(required = false) String studentId) {

if ("add".equals(action)) {

// 添加黑名单

BlackList record = new BlackList(

studentId,

parseDate(request.getParameter("beginDate")),

parseDate(request.getParameter("endDate")),

WebConstant.BLACKED\_STATE,

getCurrentAdmin(request)

);

blackListService.addRecord(record);

} else if ("delete".equals(action)) {

// 移除黑名单

blackListService.deleteByStudentId(studentId);

}

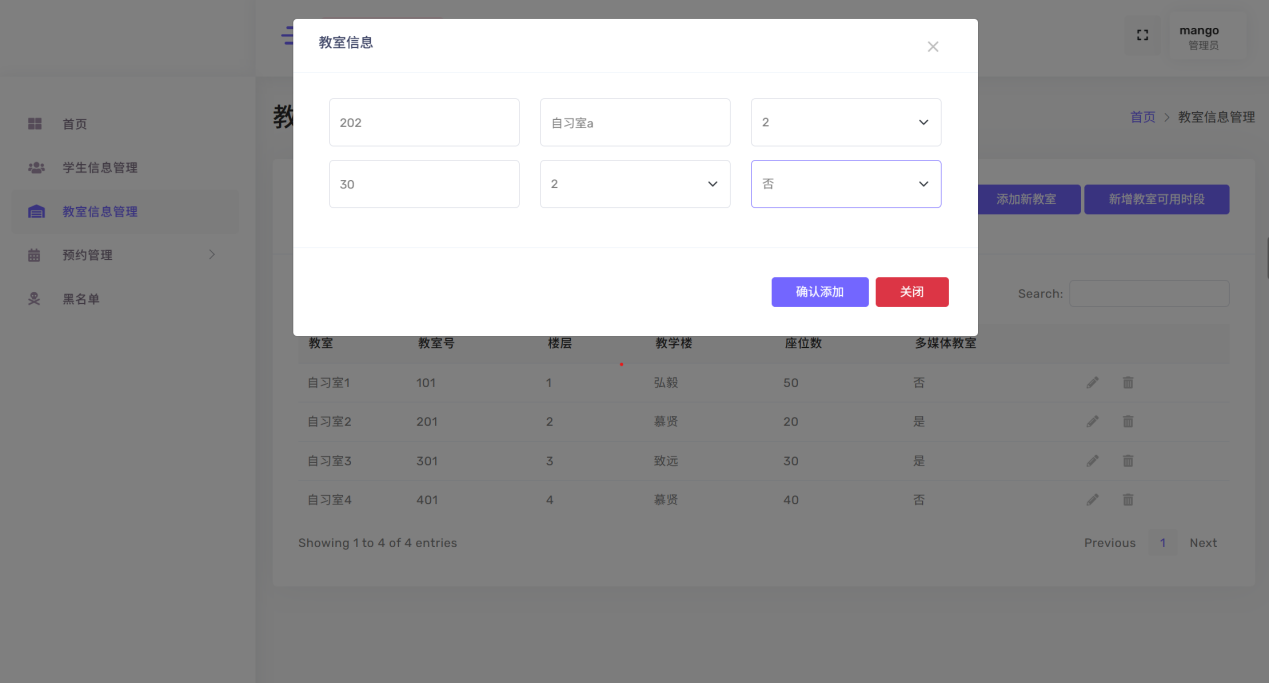
// 查询并返回列表

model.addAttribute("list", blackListService.getAllBlacked());

return "blacklist\_view";

}

**5.2.5自习室信息管理模块**

**（1）**管理员可对自习室信息进行管理，可添加新的自习室信息，编辑现有自习室的详情，以及删除不再使用的自习室信息。如图5-7所示：

5-7自习室管理图

（2）管理员可在后台对自习室进行管理，支持新增时录入名称、位置、容量、开放时段、设施设备等详细信息，删除时需二次确认并同步清理关联预约数据，确保信息精准性和操作可追溯性。自习室信息管理核心代码如下：

// 自习室增删改查核心逻辑

@GetMapping("/updateClassroomInfo") // 更新

public String updateClassroom(HttpServletRequest request) {

Map<String,Object> params = extractParams(request, "room\_id", "room\_name", "room\_floor", "available\_seat", "is\_multimedia\_room");

classroomService.updateClassroom(params); // 动态更新非空字段

return "redirect:all\_classroom";

}

@GetMapping("/classroom\_delete") // 删除（含预约检查）

public String deleteClassroom(String room\_id, Model model) {

if (classroomService.getClassroomReserved(room\_id) > 0) { // 存在预约则阻断删除

model.addAttribute("msg", "该教室已被预约，无法删除");

return showAllClassroomsWithError(model);

}

classroomService.deleteClassroomInfo(room\_id); // 级联删除关联表数据

return "redirect:all\_classroom";

}

@GetMapping("/add\_new\_classroom") // 新增

public String addClassroom(HttpServletRequest request) {

Classroom classroom = buildClassroomFromRequest(request); // 从请求参数构建对象

classroomService.addClassroom(classroom);

return "redirect:all\_classroom";

}

**5.2.6预约管理模块**

**（1）**管理员可以对学生的历史预约记录和教室座位剩余情况进行查询，如图5-8所示：5-8预约管理页面

**（2）**管理员可通过后台系统按学号、时间段或自习室名称等条件精准查询学生的历史预约记录，支持分页展示功能。同时可实时查看各教室座位动态数据，并提供按时间段、楼栋区域的多维度筛选展示，辅助管理员进行资源调度优化。预约管理核心代码如下：

// 综合查询模块：学生预约记录与教室座位情况

@GetMapping("/queryReservations")

public String queryReservations(HttpServletRequest request, Model model,

@RequestParam String queryType) { // 查询类型：student/classroom

Map<String, Object> params = new HashMap<>();

// 通用参数处理

String keyword = request.getParameter("keyword"); // 学号/姓名 或 教室ID/名称

String timeRange = request.getParameter("timeRange"); // 时间范围（格式：HH:mm-HH:mm）

String dateRange = request.getParameter("dateRange"); // 日期范围（格式：yyyy-MM-dd yyyy-MM-dd）

// 动态构建查询条件

if ("student".equals(queryType)) {

params.put("s\_id", keyword); // 学生维度：学号/姓名

params.put("year", request.getParameter("year"));

params.put("major", request.getParameter("major"));

model.addAttribute("data", reservationService.getStudentReservations(params));

} else {

params.put("room\_id", keyword); // 教室维度：ID/名称

if (StringUtils.hasText(timeRange)) { // 时段切割

params.put("time\_begin", timeRange.split("-")[0]);

params.put("time\_end", timeRange.split("-")[1]);

}

if (StringUtils.hasText(dateRange)) { // 日期切割

params.put("date\_begin", dateRange.split(" ")[0]);

params.put("date\_end", dateRange.split(" ")[1]);

}

model.addAttribute("data", reservationService.getClassroomSeatStatus(params));

}

return "reservation/result\_view"; // 统一结果视图

}

**5.3 本章小结**

基于Spring Boot的自习室管理系统聚焦高校与公共学习空间，可以省去很不不必要的财力、物力和人力。系统通过智能预约、实时状态监控与自动化管理，显著精简人力投入、提升空间利用率，同时为用户提供便捷查询与快速预约服务，助力构建高效、公平的学习环境。

**6.系统测试**

**6.1 测试目的**

自习室预约系统的稳定性直接影响资源管理效能，软件测试成为关键保障环节。针对高并发预约、时段流量峰值等典型场景，需通过压力测试验证系统承载能力与数据一致性，预防运行时崩溃风险。该质量保障机制可有效降低运维成本，同步优化资源调度算法与用户交互流程，从而提升服务连续性与使用满意度，为智能化管理提供可靠支撑。

**6.2 测试目标**

测试旨在全面验证自习室预约管理系统的功能完整性、性能稳定性及代码质量。通过功能测试覆盖用户端预约流程、管理端学生信息维护及黑名单管控等核心模块，确保逻辑严谨性；性能测试模拟高并发场景以评估系统响应时间与数据库负载能力，保障极端压力下服务不中断；兼容性测试则检查系统在主流浏览器的适配性，并验证异常输入的容错机制。此外，结合代码结构分析，定位潜在缺陷，针对性优化代码，确保系统安全交付并降低后期维护成本。

**6.3 测试方法**

测试从业务逻辑与技术实现双维度展开：业务层面基于真实使用场景设计用例，例如模拟用户频繁取消预约以验证黑名单触发机制，或测试管理员批量导入数据时对异常格式的识别能力；技术层面则结合Spring Boot与Mybatis框架特性，检查接口幂等性、事务回滚逻辑及数据库索引优化效果，避免代码疏漏导致数据异常。测试工具涵盖Postman接口调试、JMeter压力测试、辅以人工黑盒测试模拟用户操作路径，形成多层级验证

**6.4 测试用例**

（1）后台管理员功能测试。系统测试是系统的收尾工作，为了保证后期系统的稳定性，首先从后台管理端进行测试，内容是检查管理员或者用户登录是否正常否可以查看教室信息、进行教室预约、黑名单增加删除等操作，测试结果如下表6-1所示：

表6-1管理员测试表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试编号 | 测试内容 | 预测结果 | 实际结果 | 测试正确性 |
| 1 | 登录 | 登录成功 | 登陆成功 | T |
| 2 | 添加学生 | 添加成功 | 添加成功 | T |
| 3 | 增加教室 | 增加成功 | 增加成功 | T |
| 4 | 加入黑名单 | 加入成功 | 加入成功 | T |
| 5 | 移除黑名单 | 移除成功 | 移除成功 | T |

（2）前台用户功能测试。测试完后台管理端各项功能后，然后对前台用户端的功能进行了详细的测试，前台用户端主要功能是通过学号和密码进行登录、浏览教室信息和预约教室，具体的测试操作如下表6-2所示：

表6-2用户测试表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试编号 | 试内容 | 预测结果 | 实际结果 | 测试正确性 |
| 6 | 登录 | 登录成功 | 登陆成功 | T |
| 7 | 预约教室 | 预约成功 | 预约成功 | T |
| 8 | 取消预约 | 取消成功 | 取消成功 | T |
| 9 | 更换密码 | 更换成功 | 更换成功 | T |

**总结**

本论文设计并实现了基于Spring Boot的高校自习室管理系统。首先阐述了系统的研究背景，针对高校自习室资源供需矛盾与管理效率问题，分析了开发智能化管理平台的理论价值和实践意义。随后通过用户需求调研与场景模拟，采用用例图构建了系统的核心功能框架，并借助活动图细化了预约、管理等模块的业务流程。

在系统架构设计阶段，结合实体关系图构建了用户､座位､预约记录等数据模型，采用 MySQL 数据库 实现数据结构化存储，并通过 Redis 缓存优化高并发场景下的实时座位状态查询｡详细设计重点完成了后台管理模块､学生预约界面和信用积分体系三大核心功能，利用 Spring Security 实现权限控制，通过 Thymeleaf 模板引擎 构建服务端渲染页面，并基于 Spring MVC 实现请求处理与业务逻辑分离｡

系统当前已实现自习室资源可视化、预约智能化、使用数据可追溯等基础功能，但仍存在不足：高峰期瞬时并发请求的负载均衡机制有待优化。后续计划引入分布式架构提升系统吞吐量，并拓展数据分析模块，通过机器学习算法预测座位使用热点，为高校空间资源配置提供决策支持。未来还可对接校园一卡通系统，构建更完善的智慧校园生态体系。

**致谢**

时光如白驹过隙，提笔至此，心中千言竟难成句。二十余载春秋，寒窗苦读、跌撞成长，幸有恩情如灯，照我前行长夜。此间种种，唯有以诚致谢，聊表寸心。

桃李春风，师恩难忘。首谢恩师李慧老师，自选题至定稿，悉心指点、倾囊相授。每遇困顿，您以渊博学识拨云见日；每至迷途，您以严谨治学引我归正。论文一字一句，皆凝您心血。亦谢传道授业之众师，授我以术，教我以德，此间教诲，终身铭怀。唯愿师长康健顺遂，桃李满园。

椿萱之恩，山海难量。再谢父母至亲。儿时顽劣，少时轻狂，然双亲从未弃我分毫。父亲寡言，却以脊梁撑起我凌云之志；母亲絮叨，总以温粥熨帖我漂泊之心。离家求学日，母亲窗前目送，一句“累了便回家”藏尽牵挂；父亲短信寥寥，却总在深夜转账时附言“多吃点，别省”。养育之恩，此生难报，唯愿二老慢行岁月，平安长健。

山河不足重，重在遇知己。感谢室友的陪伴与照顾，为忙碌的学习生活增添了无 尽欢乐，每一个共同成长的日日夜夜都弥足珍贵。还有那些未提及的朋友和人海里的 善意，感谢所有相遇。

人间烟火，诗意栖居。最后，感谢文字与理想。迷茫时，是诗词予我豁达；困顿时，是文学赋我铠甲。愿此生永怀少年赤诚，以笔墨写尽人间清欢。

此去山高水远，愿携诸君所赠之光，行至天光乍破，步履不停。

**参考文献**

[1] 赵宇, 路建宁. 技术可供性视角下媒介场景的构建及其利弊——以"抖音线上自习室"为例[J]. 传媒论坛, 2024, 7(7): 11-14, 23.

[2] 张寒冰. 基于信创技术的智慧学习云平台的构架[J]. 无线互联科技, 2024, 21(22): 13-16.

[3] 徐慧. 基于数字技术平台的初中英语泛在学习实践研究[J]. 中国新通信, 2024, 26(21): 221-223.

[4] 李明松. 基于IT Tools平台的高中信息技术深度学习实践研究[J]. 中国新通信, 2024, 26(19): 109-111.

[5] 胡星彤. 基于在线评测技术的“Python程序设计”课程自主学习平台设计[J]. 无线互联科技, 2024, 21(19): 53-55.

[6] 李金忠, 胡志明, 陈建钢, 陈涵轩, 左左. 基于SpringBoot和Uni-App框架的自习室平台的设计与实现[J]. 河北水利电力学院学报, 2024, 34(2): 77-82.

[7] 李萍, 薛颖. 数字技术赋能下的高校智慧学习系统建设路径研究[J]. 陕西开放大学学报, 2024, 26(3): 5-9, 67.

[8] 张丽芳. 构建智能技术平台，促进学习方式变革[J]. 中小学信息技术教育, 2023(1): 37-39.

[9] 徐振忠. 基于网络学习平台的高中信息技术教学策略[J]. 教育界, 2023(17): 8-10.

[10] LI J X, XIN Y W, ZHAO J Y, et al. Research on intelligent self-studying room based on IPPG technology[C]// [会议名称]. [会议地点]: [主办方], 2022.

[11] 蔡其君. 在线陪伴学习系统的设计与实现[D]. 北京: 北京交通大学, 2022.

[12] 叶智勇, 操思薇, 张楠, 李琦, 郏晨植. 基于AIoT技术的高校自习室智慧光环境营造系统设计与实现[J]. 软件, 2022, 43(12): 183-186.

[13] CHEN G H, XU J M. Design and implementation of efficient learning platform based on Spring bootframework[J]. Journal of Electronics and Information Science, 2020, 6(1):

[14] 范岳亚.“直播自习室”：新媒介技术下的空间组合与超人际互动[J]. 重庆文理学院学报(社会科学版), 2020, 39(6): 86-95, 122.

[15] 魏晓凤. 高校图书馆考研自习室管理的几点建议——以天津职业技术师范大学为例[J]. 办公室业务, 2019(9): 153.