基于Web的在线考试系统的设计与实现

【原文对照报告-大学生版】

报告编号：7d0e2dcae4281fa7　　检测时间：2024-05-22 14:51:51　　检测字符数：43329

作者姓名：袁嘉飞　　所属单位：昆明理工大学

检测结论：

全文总相似比 **17.84%** = 复写率 15.97% + 他引率 1.87% + 自引率 0.0% + 专业术语 0.0%

其他指标：

自写率：82.16%

高频词：学生,考试,教师,系统,试卷

典型相似文章：　无

疑似文字图片：0

指标说明：

复写率：相似或疑似重复内容占全文的比重　　他引率：引用他人的部分占全文的比重

自引率：引用自己已发表部分占全文的比重　　自写率：原创内容占全文的比重

典型相似性：相似或疑似重复内容占全文总相似比超过30%　　专业术语：公式定理、法律条文、行业用语等占全文的比重

相似片段：

总相似片段 231

期刊：23　　博硕：65　　综合：0

外文：0　　自建库：30　　互联网：113

检测范围：

中文科技期刊论文全文数据库　　中文主要报纸全文数据库　　中国专利特色数据库

博士/硕士学位论文全文数据库　　中国主要会议论文特色数据库　　港澳台文献资源

外文特色文献数据全库　　维普优先出版论文全文数据库　　互联网数据资源/互联网文档资源

高校自建资源库　　图书资源　　古籍文献资源

个人自建资源库　　年鉴资源　　IPUB原创作品

时间范围：1989-01-01至2024-05-22

原文对照

颜色标注说明：

- 自写片段
- 复写片段（相似或疑似重复）
- 引用片段（引用）
- 专业术语（公式定理、法律条文、行业用语等）

学位论文原创性声明

本人郑重声明：所呈交的论文是本人在导师的指导下独立进行研究所取得的研究成果。除了文中特别加以标注引用的内容外，本论文不包括任何其他个人或集体已经发表或撰写的成果作品。本人完全意识到本声明的法律后果由本人承担。

作者签名：

2024年 5 月 5 日

昆 明 理 工 大 学

毕业设计（论文）任务书

信息工程与自动化 学院 计算机科学与技术 专业 2020 年级

学生姓名： 袁嘉飞

毕业设计（论文）题目： 基于Web的在线考试系统的设计与实现

毕业设计（论文）内容：

该网站基于Spring Boot后端web框架与Vue2.js前端框架搭建，挂载到服务器上，可通过IP及域名访问。

具体实现内容：

1、前端交互界面

1.1 前端界面分为学生界面和教师界面；

1.2 教师首页显示管理的班级，发布的试卷以及题目；

1.3 学生首页显示需要答题的试卷，以及考试记录；

1.4 教师登录后可以拉学生到某个班级，或者学生自行加入班级；

1.5 教师可以管理题库，发布选择题，填空题，简单题，可组卷发布给学生；

2、后台数据库以及数据接口

2.1 后台创建相应的数据接口提供给前端界面访问：如学生数据等；

2.2 设计合理的数据库用于存储用户信息以及试卷内容。

专题（子课题）题目：

专题（子课题）内容：

毕业设计（论文）指导教师（签字）：

主 管 教 学 院 （部） 长（签字）：

2023年 12月 26 日

毕业设计

题目： 基于 Web 的在线考试系统的设计与实现

学校： 昆明理工大学

学院： 信息工程与自动化学院

专业： 计算机科学与技术
班级： 2020级1班
学号： 202010302302
学生姓名： 袁嘉飞
指导教师： 刘英莉
教师职称（务）： 副教授
日期： 2024-5-12
Graduation Design
Title: Design and implementation of online examination system based on Web
School: Kunming University of Science and Technology
Institute: Institute of Information Engineering and Automation
Major: Computer Science and Technology
Class: Grade 2020 Class 1
Student ID: 202010302302
Student Name: Yuan Jiafei
Instructor: Liu Yingli
Teachers titles (Works): Associate professor
Date: May 12, 2024
摘要
随着计算机技术和网络技术的飞速发展，在线考试作为一种创新的考试方式，正逐渐成为教育工作者关注的焦点。这种考试方式利用了现代信息技术的优势，使得考试可以不受地点限制，随时随地进行。此外，通过在线考试系统，结合数据库技术，传统的考试流程得到了极大的简化。学生可以在任何地方，如宿舍或家中进行考试，而教师也能在家中监控考试，这一模式打破了学生必须到校参加考试的常规，极大地便利了学生和教师的考试过程。
本系统是基于 B/S 架构的在线考试系统，系统采用了前后端分离的设计理念，后端基于Spring Boot框架进行构建，前端则使用Vue.js前端框架。为了高效的管理和处理数据，系统还集成MySQL和Redis数据库技术。系统服务于学生用户和教师用户，提供不同的功能。学生登录后可以查看未完成试卷、人脸识别进行在线考试、查看教师已经批改的试卷结果、收藏题目、加入和退出班级。教师可以管理自己创建的班级、组建题库、预发布试卷、试卷重复率查询、随机生成一定数量题目、在线监控学生考试行为、系统自动批改客观题目、班级成绩分析。通过这些功能，本系统为用户提供了一个高效率、易用且公正的考试平台，显著提升教学水平和管理工作效率。
本文主要论述了在线考试系统从需求分析、系统设计，到系统实现、系统测试的整个过程。经过最后的系统测试证实，本系统运行稳定，实现了无纸化考试，并有效地减轻了教师出题、组卷和批改试卷等繁琐任务的工作负担，使得教师能够将更多的时间和精力投入到教学质量的提升和学生的个性化指导中。
关键词：在线考试系统；人脸识别；在线监控；自动批改
Abstract
With the rapid development of computer technology and network technology, as an innovative examination method, online examination is gradually becoming the focus of educators. This way of examination makes use of the advantages of modern information technology, so that the examination can be carried out at any time and anywhere without restrictions on location. In addition, through the online examination system, combined with the database technology, the traditional examination process has been greatly simplified. Students can take the test anywhere, such as the dormitory or at home, and teachers can monitor the test from home, which breaks the routine that students have to go to school to take the test, greatly facilitating the examination process for students and teachers.
This system is an online examination system based on B/S architecture. The system adopts the design concept of separating the front and back ends. The back end is built based on Spring Boot framework, and the front end uses Vue.js front-end framework. In order to efficiently manage and process data, the system also integrates MySQL and Redis database technology. The system serves both

student users and teacher users, providing different functions. After logging in, students can view unfinished papers, conduct online tests with face recognition, view the results of the papers that have been corrected by the teacher, collect questions, join and exit the class. Teachers can manage classes created by themselves, set up question banks, pre-publish test papers, query the repetition rate of test papers, generate a certain number of questions randomly, monitor students' test behavior online, automatically correct objective questions in the system, and analyze class results. Through these functions, the system provides users with an efficient, easy-to-use and fair examination platform, which significantly improves the teaching level and management efficiency.

This dissertation mainly discusses the whole process of online examination system from demand analysis, system design, system realization and system testing. The final system test proves that the system runs stably, realizes paperless examination, and effectively reduces the burden of tedious tasks such as question setting, paper grouping and paper grading, so that teachers can devote more time and energy to the improvement of teaching quality and personalized guidance of students.

Key words: Online examination system; Face recognition; Online monitoring; Automatic correction

目录

绪论

随着信息技术的高速发展，在线考试正逐渐取代传统的纸质线下考试，成为教育领域的主流趋势。传统考试的时空限制一直是其明显劣势，要求学生在特定时间和地点参加考试，这限制了学习的弹性和便利性。而在线考试的灵活性为学生提供了更大的自主选择权，他们可以随时随地通过互联网参加考试，从而极大地提高了学习的便捷性和适应性。在当前环境下，“停课不停教、停课不停学” 的策略[1]不仅深刻影响了中国的教育体系，也对全球的教育模式和教学理念带来了显著变化。这可以被视为一场全方位的“学习革命”，它改变了教学的方式、学习的方式、管理的模式，以及教育的整体形态。简而言之，这场革命重新定义了教育的传统框架，使之更加灵活、多元和适应现代社会的需求。

众所周知，传统的纸笔考试方式对资源的消耗较大。这种方式不仅需要大量的纸张来印刷试卷，教师还需要投入大量的时间来批改试卷和进行成绩分析。随着考试类型的增加，例如各类培训、竞赛和问卷调查，所需的人力物力也在不断增加。因此，开发一个在线考试系统变得尤为关键。在线考试系统的优势在于学生可以在网上答题，从而节约纸张资源；同时，计算机能够自动评估主观题，有效减轻教师的工作负担[2]。

在线考试系统的设计与实现方面，传统考试系统大部分采用C/S架构，他们实现出来的考试系统功能强大，安全性高，但是存在开发成本高，需要安装客户端，维护困难等问题。而现代大部分考试系统采用B/S架构，相较于C/S架构更简单、实用、高效[3]。技术人员也更易开发和维护系统,而现在的在线考试系统也存在一定的弊端，就是教师难以判断学生是否存在作弊行为，对此本在线考试系统也应该着重研究这点。

因此本次设计的在线考试系统的整体架构采用B/S模型，涵盖了两种身份：学生和教师。用户需进行登录后方可访问考试系统。对于学生录后，可以选择为未完成的试卷进行考试，考试期间学生不得操作考试界面以外的东西，并进行了视频监视，有效保证了考试的公平性。而对于教师，登录后则可以查看学生的考试记录，执行一系列操作，如向指定班级发布试卷、批改试卷、视频监控在线考试学生、创建班级、管理班级、学生管理以及添加题目等功能。希望系统为学生和教师提供了一个方便快捷、公平的在线考试平台。

第一章 系统调研及主要内容

1.1研究现状

在“互联网+教育”政策的指导下，加之前些年疫情的影响，在线考试系统的发展和普及进程得到了加快[4]。当谈及在线考试系统的研究和发展时，国内学者们展开了广泛的探讨和实践。在窦营山的研究中，通过元分析办法对在线与传统考试成绩的等效性进行了深入研究，结果显示成绩受科目、场景等多种因素的综合影响[5]。与此同时，张健[6]自主开发了一款性能优越的自动组卷在线考试系统，但该系统尚缺乏实证研究来验证其有效性。另一方面，彭湘华[7]运用人脸识别技术实现了在线监考。在通用考试系统的设计方面，陈海霞[8]提出了采用B/S架构的解决方案。此外，徐福江[9]利用遗传算法找出了组卷的最优方案。这些研究为在线考试系统的发展提供了宝贵的经验和启示，也为未来的研究和实践指明了方向。

在国际上，一些发达国家为满足现实需求，逐步开发了权威的网络认证考试系统及相关课程考试软件。微软推出了诸如微软认证解决方案开发专家考试（MCSD）、ORACLE认证专家考试（OCP）[10]、西班牙国立大学入学语言考试[11]等。思科也开发了一系列职业认证考试和技术专家考试。此后，美国陆续推出了计算机技能测试，如SUN开发的Java技能认证考试和数据库基础测试。Sylvan Learning System Inc目前是美国主流的计算机考试软件[12]，被广泛应用于各种行业和考试机构，大大降低了用人成本。

总的来说，全球范围内在线考试系统的发展和普及正在取得显著进展。国内学者们在此领域进行了广泛的研究和实践，从在线与传统考试成绩等效性、自动组卷系统、在线监考技术到通用考试系统设计等方面进行了深入探讨和探索。同时，一些发达国家也在满足现实需求的背景下积极开发了权威的网络认证考试系统和相关软件。这些努力不仅为在线教育和考试提供了技术支持，也为本文设计的在线考试系统提供了宝贵的思路。

1.2 主要研究内容

本次设计的Web在线考试系统主要利用现有技术将教师创建试题、学生在线考试、教师在线监控整合，为教师和学生提供一个便捷、高效的网络平台，实现教学评估的自动化和数字化。系统最主要的模块包括：用户注册登录模块、教师题目管理模块、试卷设计管理模块、学生班级管理模块、个人信息管理模块、权限管理模块、教师试卷批改管理模块、教师在线监控模块、数据统计成绩分析管理模块、学生在线考试管理模块、学生人脸识别管理模块、学生收藏管理模块。

这次设计的Web在线考试系统采用了模块化和前后端分离的设计理念，这样的不仅使系统编码更加简洁明了，还降低了模块之间的耦合度。通过将系统拆分为不同的模块，每个模块专注于特定的功能或任务，开发人员可以更加集中精力进行开发，提高了个人开发效率。同时，前后端分离的设计使得前端和后端可以独立开发、测试和部署，加速了整个开发周期[13]。

模块功能和特点：

（1）用户注册登录模块：教师和学生填写个人信息，包括手机号码、密码、工号或者学号、学生还应该上传人脸，系统验证信息合法性后完成注册。已注册用户使用手机号和密码登录系统，系统验证后权限管理模块判断，判断学生还是教师跳转不同页面。

（2）教师题目管理模块：教师可以创建各种类型的试题，包括选择题、填空题、判断题、简答题，填写题目内容、选项和标准答案。

（3）试卷设计管理模块：教师根据教学内容和要求，从题库中选择试题组成试卷，设置试卷信息如考试时间、是否可以查看、是否立即发布给学生、支持教师随机题目、支持创建试卷时检测重复度。

（4）学生班级管理模块：教师可以创建班级，设置班级名称，系统自动分配班级代码，然后可以根据学生学号导入到指定班级。

（5）学生管理模块：教师可以对学生进行管理，包括添加、删除学生是否在某个班级，查看学生试卷的考试成绩等。

（6）个人信息管理模块：教师和学生可以查看和修改个人信息，包括昵称、真实姓名、手机号码、邮箱、头像等，保持信息的准确性。学生可以在个人信息管理页面加入或者推出某个班级。

（7）权限管理模块：系统为不同角色的用户分配不同的权限，如教师拥有试题管理权限，学生只有考试权限等。

（8）教师试卷批改管理模块：教师可以对学生提交的试卷进行批改和评分，系统后台自动评判选择题和判断题，填空和简答题需要教师评判。教师批改完成系统记录学生的考试成绩和答题情况。

（9）教师在线监控模块：教师通过获取学生的在线实时考试视频，监督学生考试行为，确保考试的公平性。

（10）数据统计成绩分析管理模块：系统会对学生的考试数据进行统计分析，生成成绩报表和图表，帮助教师了解学生的学习情况。

（11）成绩分析：系统会根据学生的考试成绩，对整个班级计算平均分，为教师管理班级提供参考依据。

（12）学生在线考试管理模块：系统根据教师设计的试卷安排在线考试，学生选择试卷进行人脸识别，人脸识别后加入考试，打开摄像头在线监控，考试期间不允许退出全屏，学生在规定的时间内完成在线考试，答题并提交试卷。

（13）学生收藏管理模块：学生可以将自己感兴趣或觉得重要的试题收藏起来，便于复习和查看。

第二章 主要相关技术介绍

2.1 Spring Boot

Spring Boot 是一个为简化 Spring 应用程序开发而设计的开源框架，它隶属于 Spring 家族，由 Pivotal 团队推出并由 VMware 维护。此框架的核心价值在于自动化配置、快速启动以及最少的前期配置，它让开发者能够快速搭建独立运行、生产就绪的 Spring 应用，极大地减少了以往繁杂的 XML 配置工作[14]。Spring Boot 通过引入起步依赖（Starter Dependencies）的概念，自动集成了项目所需的库，同时内建了如 Tomcat、Jetty 的服务器，使得应用可以直接被打包成一个可执行的 JAR 文件运行，无需额外部署到应用服务器。此外，它还提供了 Actuator 组件用于应用监控和管理，以及丰富的命令行工具来辅助开发，支持多种外部配置方式，确保了应用在不同环境下的灵活性。总之，Spring Boot 凭借其“约定优于配置”的设计理念[15]，成为了现代快速应用开发和微服务架构的理想选择，无论是在初创项目还是复杂的企业级解决方案中都能发挥巨大作用。

2.2 Vue.js

Vue的核心概念围绕着组件化编程，它鼓励开发者将用户界面拆分成多个可复用的组件，每个组件都拥有自己的视图模板、数据逻辑和方法。这样的设计不仅提升了代码的组织结构，也极大地促进了团队协作，使得维护和扩展

变得容易。Vue的模板语法简洁直观，通过指令（Directives）和插值表达式，可以轻松地将数据绑定到DOM上，实现数据与视图的双向绑定，大大简化了状态管理的复杂度[16]。

Vue.js的一大亮点是其响应式系统。当数据模型发生变化时，Vue能够智能地计算出最小化的DOM更新操作，这一切得益于其高效的虚拟DOM（Virtual DOM）实现。虚拟DOM是一种轻量级的内存中DOM树表示，Vue通过比较虚拟DOM的差异来决定实际DOM需要进行哪些最小化变更，从而避免了直接操作真实DOM的高昂成本，确保了应用的高性能运行，即使在数据频繁变动的场景下也能保持流畅的用户体验。

Vue生态系统的完善也是其受欢迎的原因之一。Vue CLI（命令行工具）为快速搭建项目、配置开发环境提供了便利；Vue Router用于实现SPA（单页面应用）的路由管理，使得页面间的跳转和状态管理变得简单；Vuex则是Vue提供的集中式状态管理模式，帮助开发者管理组件间共享的状态，使得状态流清晰可控。此外，Vue还拥有丰富的第三方插件和UI库，如Element UI、Vuetify等，为开发者提供了多样化的UI组件选择，进一步加速了开发进程。Vue的另一个显著特点是其优秀的文档和社区支持。Vue的官方文档条理清晰，示例丰富，即便是初学者也能快速上手。活跃的社区论坛和GitHub项目，意味着开发者遇到问题时，总能及时获得帮助和解决方案。

2.3 MySQL

MySQL采用C和C++编写，确保了其源代码的高效性和可移植性，能够无缝运行在包括Windows、Linux、macOS在内的多种操作系统平台上。它支持多线程并发处理，充分利用CPU资源，尤其适合高并发访问的应用场景。MySQL对SQL标准有着良好的支持，同时也针对Web应用进行了优化，提供快速的查询处理能力。

该数据库系统支持大型数据库，能够处理数百万乃至数十亿条记录的数据仓库，且在64位系统中支持极高的表文件大小，满足了大数据处理的需求。MySQL提供了丰富的API，便于与各种编程语言集成，如C、C++、Java、Python、PHP、Ruby等，这使得开发者可以灵活地在各种项目中使用MySQL。

MySQL的架构设计分为客户端、服务器层以及存储引擎层。其中，服务器层实现了SQL解析、查询优化、事务处理等核心功能；存储引擎层支持多种存储机制，如InnoDB（支持事务处理、行级锁和外键约束）和MyISAM（适合读取密集型应用），用户可以根据应用需求选择合适的存储引擎。MySQL 数据库以其卓越的速度性能、高度的可靠性和强大的适应性著称，充分满足了考试系统日常运作的各项严苛要求[17]。

2.4 Redis

Redis，全称为Remote Dictionary Server，是一个开源的、高性能的键值（Key-Value）存储系统。它使用ANSI C语言编写，支持网络、基于内存存储，同时也支持数据的持久化到磁盘。Redis因其极高的数据读写速度和丰富的数据结构支持，在众多场景中扮演着至关重要的角色，尤其是那些需要快速读写和低延迟操作的场合。

Redis不仅提供基本的字符串（string）类型存储，还支持更为复杂的数据结构，包括列表（list）、集合（set）、有序集合（sorted set）和哈希表（hash），这些特性使其成为处理复杂数据模型和实现多种功能的强大且方便工具[18]。例如，列表可用于消息队列，有序集合能高效地实现排行榜系统，而哈希表则适合存储对象字段。

Redis通过单线程模型和I/O多路复用技术实现高性能操作，尽管是单线程，但得益于其内存操作的高效性，Redis能够处理每秒高达数十万次的读写请求。对于需要高并发访问的场景，Redis可以通过部署集群来实现水平扩展，提高系统的处理能力和可用性。

2.5 Mybatis

MyBatis是一个轻量级且流行的Java持久层框架，专门设计用于简化数据库访问并提供高度灵活的SQL映射功能。相较于全自动化ORM工具如Hibernate，MyBatis更加注重SQL的直接控制与优化潜力，允许开发者手写SQL语句并通过XML映射文件或注解来界定这些SQL如何与Java对象相互作用，完成数据的查询、插入、更新及删除操作，以及对象关系到数据库记录的映射反向过程。此框架支持动态SQL[19]构建，可根据输入参数动态调整查询逻辑，实现诸如条件筛选、分页等复杂需求，同时内置事务管理和缓存机制，前者确保数据操作的原子性和一致性，后者通过存储查询结果来提升应用性能，减少数据库负载。总之，MyBatis以其在SQL定制性与操作简便性间的平衡，成为处理Java应用数据库交互的优选工具之一。

2.6 WebSocket

WebSocket是一种在客户端与服务器之间建立长连接的协议，它打破了传统HTTP协议只能由客户端发起请求的限制，实现了真正的双向实时通信。WebSocket协议建立在TCP之上，起初握手通过HTTP进行，随后协议升级，转换为WebSocket特有的帧格式进行数据传输，这一转换使得连接双方能够在单个TCP连接上进行全双工的通信[20]。

与轮询技术相比，WebSocket显著降低了延迟，提高了数据交换的效率和实时性，因为它允许服务器主动推送数据至客户端，而不需要客户端不断地发起请求来检查是否有新数据。这一特性使得WebSocket成为实时应用领域的理

想选择，广泛应用于在线聊天、协同编辑、实时交易系统、在线游戏、实时数据分析和物联网(IoT)设备通信等多种场景。

WebSocket协议在2011年被IETF（Internet Engineering Task Force）标准化为RFC 6455，并由后续的RFC7936进行补充规范。与此同时，W3C为浏览器环境定义了WebSocket的API标准，确保了现代浏览器对WebSocket的支持，使得开发者能够方便地在网页应用中集成这一实时通信能力。

2.7 WebRTC

WebRTC，即Web Real-Time Communication，是一项革命性的开源技术，它彻底改变了互联网实时通信的面貌。无需任何插件或专用软件，WebRTC使得现代浏览器之间能够直接进行实时的音频、视频通信及数据共享，为网页和移动应用注入了实时交互的能力[21]。通过集成复杂的音频和视频处理、实时通信信令以及安全的点对点连接技术，WebRTC实现了高清音视频通话、屏幕共享、实时协作等多种功能。这一技术由Google发起，并得到了W3C和IETF的标准化支持，确保了跨平台的兼容性和广泛的适用性。WebRTC的出现，不仅降低了实时通信技术的准入门槛，还促进了视频会议、在线教育、远程医疗、游戏互动等众多领域应用的创新与繁荣，为用户带来无缝、即时的沟通体验。

2.8 JWT

JSON Web Token（JWT）是一种开放标准（RFC 7519），用于在网络应用之间安全地传输信息。JWT以JSON格式表示，由三部分组成：头部（Header）、载荷（Payload）、签名（Signature）[22]。

头部包含了描述该令牌的元数据，例如令牌类型和使用的签名算法。载荷包含了令牌的主要信息，如用户身份、权限等。签名则用于验证令牌的真实性和完整性。

JWT通常用于实现身份认证和授权。用户登录后，服务器生成JWT并发送给客户端。客户端在后续请求中携带JWT，服务器通过验证JWT的签名来确认用户身份和权限，从而允许或拒绝请求。

由于JWT是基于JSON格式的轻量级令牌，因此易于传输和解析，适用于各种网络应用场景，包括单点登录、微服务架构、跨域身份验证等。

JWT的优势在于易扩展、可复用、安全性高、高效率等[23]，同时减少了服务器的负担。然而，需要注意的是，JWT中的信息是以明文形式存储的，因此不适合存储敏感信息。

第三章 系统需求分析

3.1 系统可行性分析

系统可行性分析是项目开发前期的关键步骤，旨在评估项目在技术、经济和操作等方面的可行性，以确保项目的成功实施。对于基于Web的在线考试系统而言，其可行性分析可以从以下几个方面进行深入探讨：

（1）技术可行性

前端技术：系统采用Vue2.js作为前端框架，Vue的组件化开发模式和响应式数据绑定特性，有助于快速构建用户友好的交互界面，符合当前Web开发的主流趋势，技术成熟，社区活跃，易于获取技术支持和资源。

后端技术：Spring Boot框架简化了企业级应用的开发，提供了丰富的开箱即用功能，如自动配置、嵌入式服务器等，极大提升了开发效率。同时，Spring生态丰富，支持多数据库集成，确保了技术方案的可行性。

数据库技术：MySQL作为关系型数据库，广泛应用于互联网项目，具备良好的性能和稳定性，支持高并发访问，满足在线考试系统的大数据处理需求。Redis的引入，作为高速缓存数据库，可以进一步提升系统响应速度，特别是在处理频繁查询和会话管理方面。

通讯技术：WebSocket的使用实现了客户端与服务器的实时双向通信，对于在线考试中实时监控、即时反馈等需求至关重要，技术成熟且得到广泛支持。

（2）经济可行性

成本效益分析：如果只是考虑本地运行的话，不存在什么经济支出，如果要部署在外网上，需要购买云服务器，成本也比较低，所有在经济上完全可行。

（3）操作可行性

用户友好性：系统设计充分考虑了学生和教师的使用习惯，前端界面清晰，操作简单，用户根据界面提示即可快速上手。

综合上述分析，基于Web的在线考试系统在技术、经济和操作等方面均表现出了较高的可行性，为实现系统提供了可靠的支撑。

3.2 系统总体功能

本次设计的Web在线考试系统的总体功能设计得全面且细致，主要服务于教师和学生两大用户群体，确保了考试

流程的完整性、公平性和高效性。

对于本项目主要分为两个角色，教师用户和学生用户：

（一）教师用户：具备题库管理、试卷管理、班级管理、在线考试监控、学生试卷批改等功能。

（二）学生用户：具有在线考试、查看试卷评分、收藏题目、班级操作等功能。本项目的主要角色的用例图如图3.2.1所示。

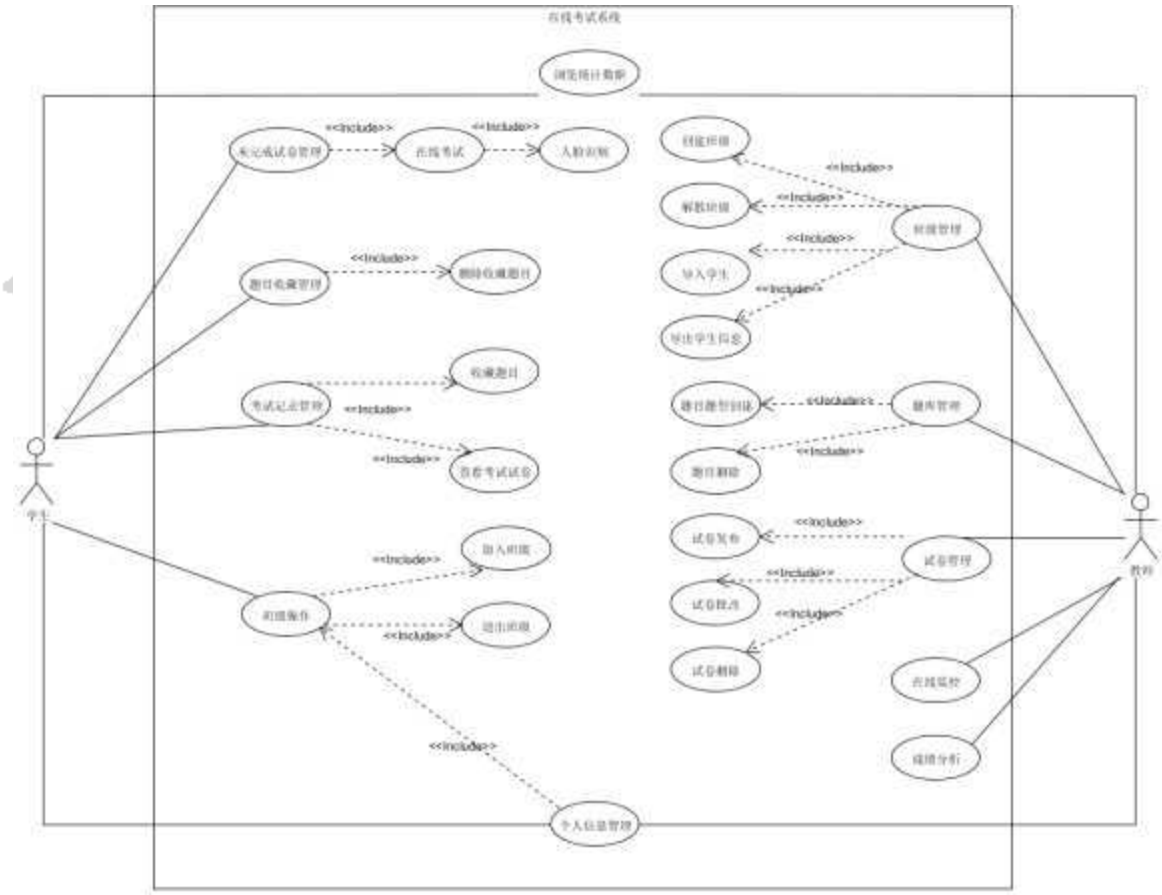


图3.2.1 网站角色用例图

3.3 学生用户功能需求分析

3.3.1 学生注册登录

学生用户的创建与登录机制是系统安全及个性化体验的基础。此功能不仅确保了学生能够访问专属界面，还通过差异化权限划分保护了系统安全，并便于数据统计与管理。具体到学生用户创建流程，它包括以下步骤：

学生在注册时需提供手机号码和基本身份信息。前端首先验证手机号的有效性，用户的唯一性，随后引导学生输入学号及设置密码，并上传个人人脸图像，为未来的在线考试人脸识别验证做准备。

学生注册成功后跳转登录页面，输入手机号和密码，经过服务端验证正确后，返回手机号所属身份为学生，前端判断身份跳转学生的首页。学生注册时序图如图3.3.1所示。

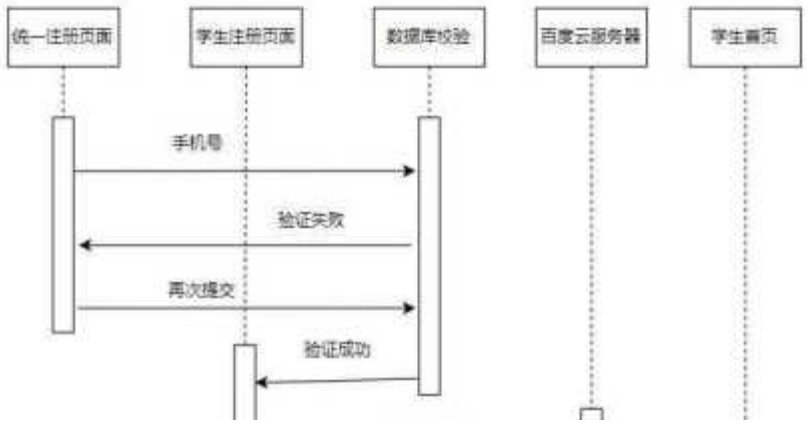


图3.3.2学生注册时序图

3.3.2 学生人脸识别

当学生注册时，他们需要提供自己的人脸照片。注册系统将使用后端的人脸识别功能，以确保所上传的照片中包含人脸。一旦验证通过，系统会将学生的人脸照片以其学号为命名保存在服务器本地，并在数据库中记录照片的存储地址，以便后续的检索和管理。

在考试前，学生需要再次进行人脸识别。这一步是为了确认考试者的身份，确保只有注册时所提供的学生本人才能进入考试系统。考试系统将调用后端的人脸识别接口，验证考试者的人脸与注册时所提供的照片是否匹配。只有验证成功，考试者才能进入考试界面。

这些人脸识别功能的实现依赖于百度云提供的人脸识别API。通过这个API，系统可以实现快速、准确的人脸识别，保障注册和考试过程的安全性和可靠性。

3.3.3 学生在线考试

学生界面设计围绕提升使用便捷性和考试体验展开，具体布局如下：

（1）首页：

•概览面板：简洁直观地展示学生的考试概况，包括“试卷总数”、“未完成试卷”和“已批改试卷”的统计数字，让学生一目了然当前的学习进度和学习状态。

（2）我的试卷：

•试卷入口：每项试卷条目清晰标注考试名称、考试总分及考试总时长，学生点击相应试卷后系统启动人脸识别认证流程，确保考试的公正性与安全性。

（3）考试记录：

•查看权限管理：此部分列出了所有已完成的考试记录，根据教师设定的权限，部分试卷显示“详情”按钮，允许学生查阅成绩详情及标准答案；对于尚未开放查看的试卷，则标记为“待教师批改”或“不允许查看”，增强信息透明度同时维护考试公平。

（4）考试界面优化：

•沉浸式考试环境：进入考试后，界面自动切换至全屏模式，顶部栏显示试卷基本信息，如“试卷名称”、“总分”和动态“倒计时”，营造专注考试氛围。

•多元化题型支持：试卷包含选择题、填空题、判断题和简答题多种题型，每道题目下方提供清晰的作答区域，确保学生能顺畅作答。

•实时监控：考试全程开启摄像头，实施录制并实时传输至教师端，教师可以随时查看学生考试动态，保障考试诚信。

•严格的考试规则：明确告知学生考试期间不可退出全屏或关闭考试页面，违规操作将触发自动提交试卷机制，倒计时归零同样视为考试结束并自动提交，以此强化考试纪律。

学生端在线考试系统最核心功能是考试，包括在线监控、答题和交卷。在考试过程中，系统还可以扩展使用人脸识别和屏幕录制功能。人脸识别通过启用摄像头确保考试的公平性和真实性，屏幕录制通过启用屏幕记录功能上传到服务器，教师端可以拉取服务器视频数据监督学生的行为。这些功能相互配合，确保在线考试的公平和有效。学生端在线考试用例图如图3.3.2所示。

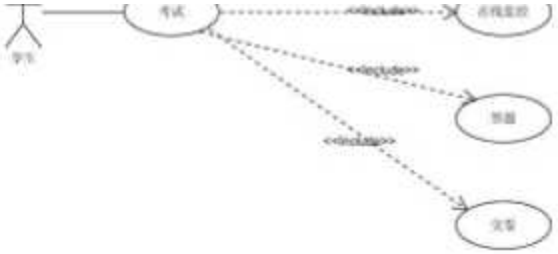


图3.3.2学生在线考试用例图

3.3.4 学生收藏题目

学生在导航页的“考试记录”中可以点击详情，进入试卷界面查看自己的考试题目。每道题目右上角都设有一个星星按钮，点击该按钮可将题目加入收藏列表。一旦收藏，学生可以随时在导航页头像的下拉菜单中找到“我的收藏”，并点击进入查看所有收藏的题目。在“我的收藏”页面，学生同样可以通过点击题目右上角的星星按钮取消收藏。

这种功能的设计有助于学生更方便地管理和回顾自己感兴趣或重要的题目，无需在试卷中来回查找。同时，通过将收藏列表放置在头像的下拉菜单中，使得学生可以在多个页面轻松地访问和管理自己的收藏，提高了系统的可用性和用户体验。

3.4 教师用户功能需求分析

3.4.1 教师端

教师端的核心模块主要是试卷管理、题库管理、班级管理以及在线监控，试卷管理核心模块是组卷，教师组卷需要设置考试试卷的名称，考试的总时长，设置查看权限，添加题目，发布组卷结果。题库管理是对组卷题目的增删查，班级管理则是教师可以对班级创建解散导入学生操作，教师还可以对在线考试的学拉取学生端的视频数据，查看学生考试行为。教师端主要模块用例图如图3.4.1所示。

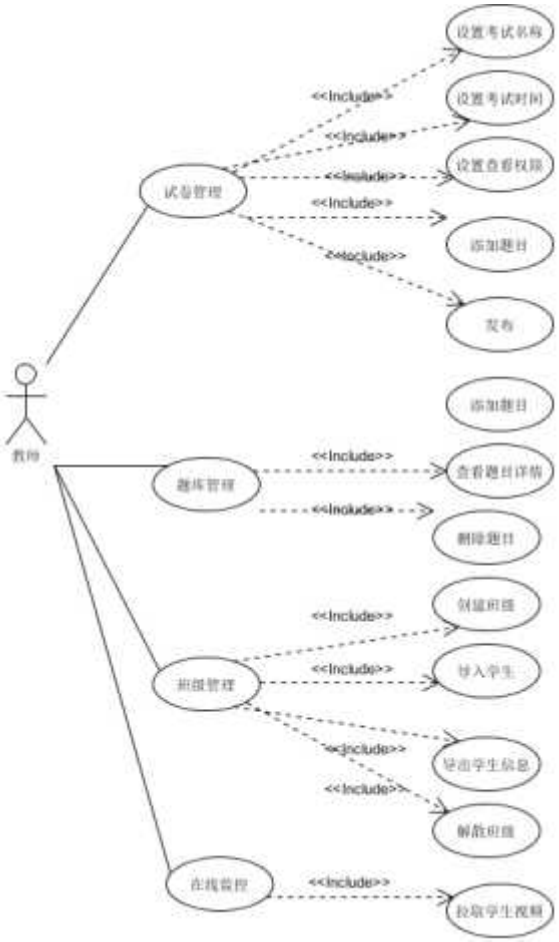


图3.4.1 教师端主要模块用例图

3.4.2 教师注册登录

教师用户的创建与登录同样为核心功能，旨在实现用户角色的精确识别，以展示相应的定制化内容，增强用户体验并维护系统安全性。与学生注册流程相比，教师注册过程相对简化，具体包括：

区别于学生注册，教师仅需填写手机号码、教职工编号及设定密码即可完成注册，无需上传人脸信息。这一设计既满足了教师快速注册登录的需求，又保持了系统的高效与安全，确保每位教师能迅速接入并专注于考试相关功能模块。

教师注册成功后跳转登录页面，输入手机号和密码，经过服务端验证正确后，返回手机号所属身份为教师，前端判断身份跳转教师的首页。教师注册时序图如图3.4.2所示。

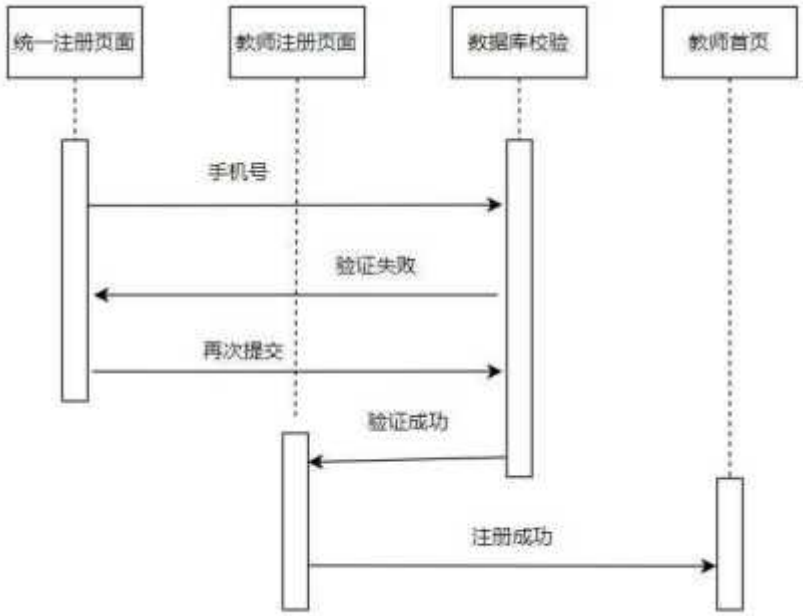


图3.4.2教师注册时序图

3.4.3班级管理

在系统中，教师以班级为管理单位。教师可以在系统左边选项栏的“我的班级”选项中进行班级管理。在这个界面，教师可以点击“创建班级”按钮，输入班级名称即可创建自己管理的班级。下方则展示了教师所管理的所有班级选项卡，包括班级名称、班级代码以及班级人数等相关信息。

在班级管理页面，教师有多种操作选择。首先，他们可以通过操作栏邀请学生加入指定的班级，以便对学生进行管理。其次，教师可以选择导出某个班级的所有学生信息，这对于班级管理和数据分析都十分有用。同时，如果教师不再需要某个班级，也可以选择解散该班级，从而清理无用的数据和资源，保持系统的整洁和高效。
这种设计使教师能够轻松地管理班级和学生，提高了教学和管理的效率，为教学工作提供了便利和支持。

3.4.4 题库管理

在系统中，题库管理是教师必不可少的功能之一。通过题库管理，教师可以轻松地组织和管理自己的题目资源，为教学考试活动提供了重要支持。

在选项栏的“我的题目”中，教师可以方便地查看自己创建的题目列表。这个位置的设置让教师可以快速定位到题目管理功能，节省了操作的时间和精力。在题目列表中，教师可以看到每道题目的详细信息，包括题目名称、类型、分数、创建时间等。页面还设置了题型、题目名称搜索框，方便教师快速定位想要的题目。这样的设计使得教师可以清晰地了解自己所拥有的题目资源，方便随时进行查阅和管理。

同时，系统还提供了丰富的题目创建功能。教师可以根据需要选择不同类型的题目，如单选、多选、判断、填空、简答等。在创建题目时，系统提供了清晰的界面和用户友好的交互，让教师可以轻松填写题目内容、选项、答案等信息，并且通过点击“立即创建”按钮即可完成题目的添加。
总的来说，题库管理功能设计的简洁明了，操作便捷高效，为教师的教学工作提供了强大的支持和便利。

3.4.5 试卷管理

在系统中，试卷管理是教师必不可少的功能之一，它涉及到试卷的创建、发布、管理和学生答卷的批改，为教

学活动提供了重要的支持。

（1）试卷创建和发布：

•在选项栏的“我的试卷”中，教师可以方便地创建和管理试卷。

•在“试卷列表”页面，显示了教师创建的所有试卷信息，包括试卷名称、创建时间、考试时长、发布状态以及发布到的班级。教师可以通过操作栏对试卷进行查看、添加班级、删除等操作，同时还提供了试卷名称搜索框，方便教师快速定位所需试卷。

•在“发布试卷”选项卡中，教师可以填写试卷的相关信息，包括是否允许考试后学生查看试卷、考试时长、发布到的班级等。教师可以从题库中选择题目填充到试卷中，并且可以随机选择题目数量，系统会自动向前端提供指定数量的随机题目。组卷完成后教师可以选择立即发布或预发布到学生端。

（2）学生答卷管理：

•在“试卷批改”页面，教师可以查看需要批改的学生答卷列表。列表显示了学生的姓名、学号、所属班级、试卷信息、考试总分、学生考试时长等。教师可以通过操作栏点击“批改”对学生的答卷进行评分。

•系统会自动批改选择题和判断题，而填空和简答题需要教师手动评分。教师完成评分后，可以提交批改结果。

•批改完成后，教师可以在“完成试卷”选项卡中查看自己的批改结果。

通过以上设计，试卷管理功能实现了教师试卷的创建、发布和管理，同时提供了学生答卷的便捷批改功能，全面满足了教学需求，为教学活动提供了有力支持。

3.4.6 在线监控

在系统中，在线监控是教师不可或缺的功能之一。它为教师提供了实时的学生考试状况，有助于监督学生的行为，确保考试的公平性和规范性。

（1）实时监控功能：

•教师可以通过在线监控功能实时查看学生考试时的情况。

•当学生在考试时打开摄像头，教师端可以通过系统拉取学生端的在线视频数据。如果有多个学生参与考试，教师端将显示多个监控视频块，以便同时监控多名学生的情况。

•这种实时监控功能有助于教师发现和防止考试作弊行为，确保考试的公平性和规范性。

（2）基于WebRTC视频传输技术的实现：

•在线监控功能主要依靠WebRTC视频传输技术实现学生端和教师端的点对点视频传输。

•WebRTC技术能够在不同设备之间直接建立点对点的实时通信连接，有效降低了系统的延迟，确保了监控视频的实时性和稳定性。

•这种技术的应用使得在线监控功能更加高效和可靠，为教师提供了更好的监督和管理手段。

3.4.7 数据统计

数据统计模块的功能旨在对学生和教师的相关数据进行统计，并以可视化的方式展示，从而让用户更直观地了解数据情况。

（1）教师首页数据统计：

•在教师首页，使用 echarts 的饼状图，对教师管理的班级人数分布和班级数据分布进行显示。这些饼状图可以直观地展示不同班级的数据占比情况，帮助教师了解班级的结构和特征。

•在饼状图上方使用 Element UI 的 el-card，显示教师管理的学生数量、班级数量、题目数量和试卷数量等关键数据。这些数据提供了教师管理情况的概览。

（2）成绩分析功能：

•在“成绩分析”选项卡中，通过柱状图和折线图显示每个答卷班级的平均分。这些图表能够直观地反映出每个班级的学生成绩情况，帮助教师对学生的表现进行分析和评估。

•所有的数据统计和图表展示都是由后端进行统计和处理，然后提供给前端进行显示。后端负责从数据库中获取数据，进行统计计算，并将统计结果以标准格式提供给前端。

第四章 系统设计

4.1 系统总体设计

4.1.1 系统框架设计

本次系统设计采用前后端分离的开发模式，将应用分为前端和后端两大部分。前端应用主要负责展示视图，而后端应用则专注于业务逻辑处理。它们之间通过JSON格式进行数据交互。

目前，主流的实现方案是采用Spring Boot框架作为后端开发工具，而前端则使用Vue框架。Spring Boot提供了快速开发和丰富的功能，而Vue则为前端开发提供了组件化、响应式数据绑定等便利特性。

这种分离模式的优势在于提高了开发的灵活性和可维护性。前后端开发者可以专注于各自的领域，通过定义清晰的接口实现有效的协作[24]。此外，由于采用了JSON格式进行数据交互，系统的可扩展性也得到了增强。

前端和后端分离结构图如4.1.1所示。

图4.1.1 分离结构图

4.1.2 系统模块结构

在前后端分离的架构中，可以将系统划分为多个独立的模块，每个模块负责特定的功能或业务领域，例如用户认证、个人信息管理、题库管理等。每个模块都有自己的接口和数据结构，与其他模块通过清晰的API进行通信。

通过模块化设计，可以更好地管理系统的复杂性，降低各模块之间的耦合度，提高代码的复用性和可测试性。这样的设计不仅使得系统更易于维护和扩展，还有助于提高开发效率和质量。

在线考试系统的功能模块图如图4.1.2所示。

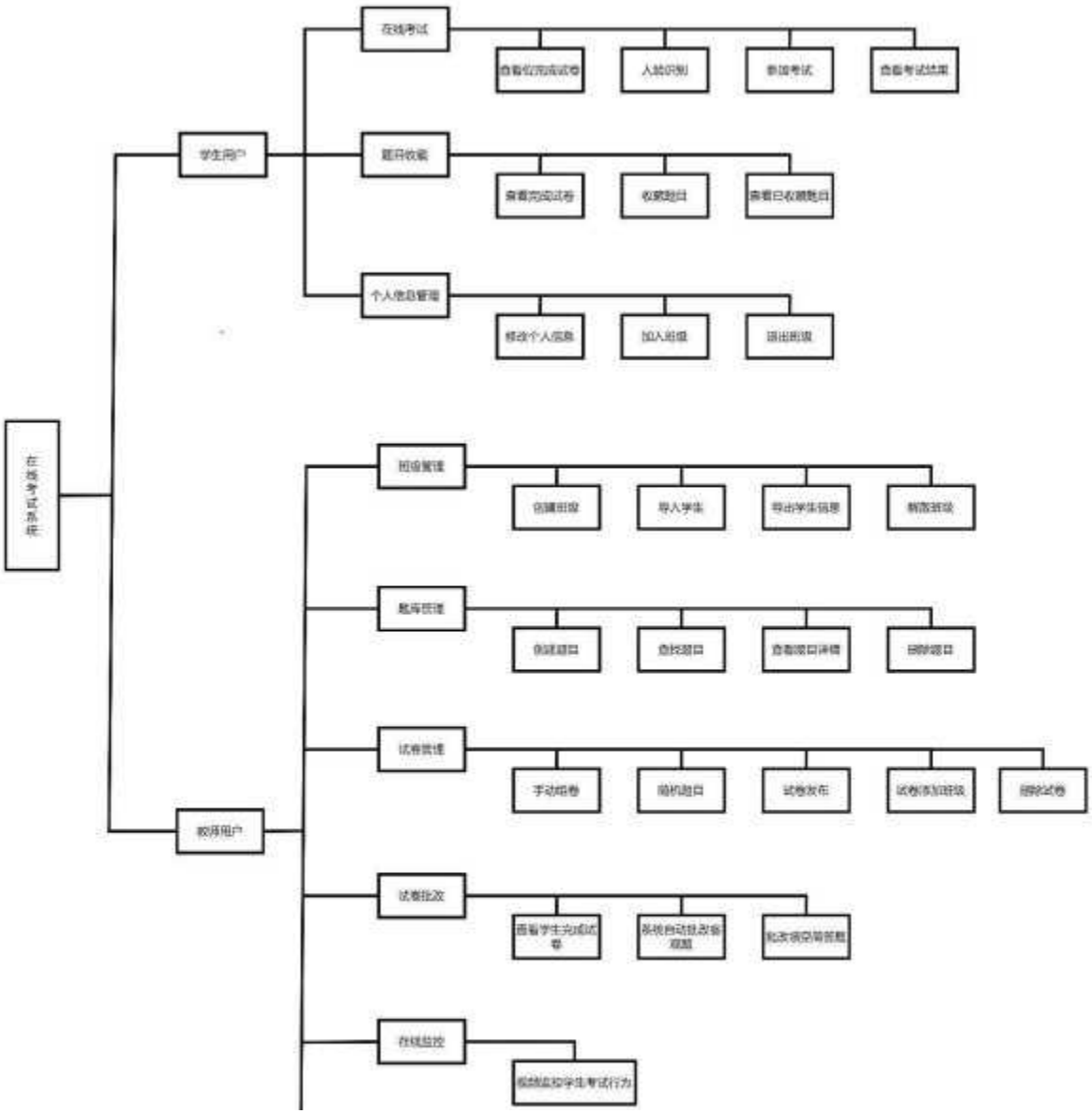


图4.1.2 系统功能模块图

本系统分为学生用户和教师用户两个部分。对于学生用户来说，系统包括在线考试、题目收藏、个人信息管理三个模块。其中，在线考试模块又包括查看未完成试卷、人脸识别、参加考试和查看考试成绩四个子模块；题目收藏模块包括查看完成试卷、收藏题目和查看已收藏题目三个子模块；个人信息管理模块包括修改个人信息、加入班级和退出班级三个子模块；

对于教师用户来说，系统包括班级管理、题库管理、试题管理、试卷管理、在线监控和成绩分析六个模块。班级管理模块包括创建班级、导入学生、导出学生信息和解散班级操作；题库管理包括创建题目、查找题目、查看题目详情、删除题目操作；试卷管理模块包括手动组卷、随机题目、试卷发布、试卷添加班级和删除试卷操作；试卷批改模块包括查看学生完成试卷、系统自动批改客观题和批改填空简答题三个子模块。在线监控模块主要是教师视频监控学生考试行为；成绩分析模块是教师查看试卷班级平均分。

4.2 前端系统详细设计

4.2.1 系统总框架

本系统设计的前端部分采用了现代化的前端开发架构，主要包括以下目录结构：

api：存放与后端API通信的相关文件，包括封装了HTTP请求的函数或者API的配置文件。

assets：存放静态资源文件，例如图片、字体、图标等。

components：存放可复用的Vue组件，包括按钮、表单、模态框等，以提高开发效率。

router：存放Vue Router的路由配置文件，用于管理前端路由，实现页面之间的导航和跳转。

store：存放Vuex状态管理模块，用于管理应用程序的状态，例如用户登录状态、全局数据等。

styles：存放样式文件，包括全局样式和组件样式，以确保整个应用的一致性和美观性。

utils：存放一些工具函数或者辅助方法，用于整个项目的公用功能，例如全局变量、HTTP原始接口请求封装等。

views：存放页面级组件，每个子目录对应一个页面或者一个功能模块，包含了该页面或功能模块所需的组件、样式和逻辑。

404：404页面，用于处理页面未找到的情况。

ForgetPassword：忘记密码页面，用于处理用户忘记密码的情况。

Login：登录页面，用于用户登录操作。

Personal：个人中心页面，用于展示用户的个人信息和管理功能。

Register：注册页面，用于用户注册新账号。

StudentDashboard：学生仪表板页面，用于展示学生相关信息和功能。

TeacherDashboard：老师仪表板页面，用于展示老师相关信息和功能。

以上目录结构设计使得前端开发更加模块化、可维护性更强，有利于项目的持续发展。

在线考试系统的前端目录结构图如图4.2.1所示。

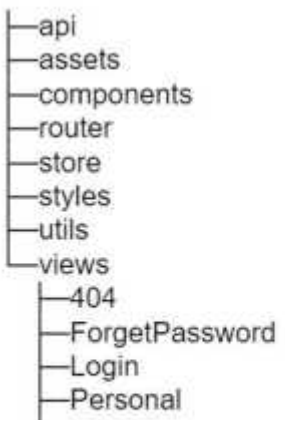


图4.2.1 前端目录结构

4.2.2 注册登录模块

在本系统中，学生用户的注册和教师用户的注册是不一样的，区别在学生用户需要上传人脸照片，而教师不需要，所以前端注册模块需要设置两个Vue界面单独跳转，在跳转到不同界面前，前端需要发送手机号到后端验证是否合法。当选择的身份是学生的时候跳转学生注册界面，教师则跳转教师界面，前端用1和2字符区别跳转界面，1代表学生，2代表教师。学生注册界面和教师注册界面区别在于学生注册界面存在一个拍照选择框，而教师没有。学生注册需要使用一个表单，将人脸图片和注册信息一起上传到服务器。注册信息主要是学号（工号）、密码、手机号。注册相关的信息上传到服务器后，服务器校验信息，主要校验学生的人脸是否存在。注册成功后跳转登录页面。前端注册流程图如图4.2.2所示。

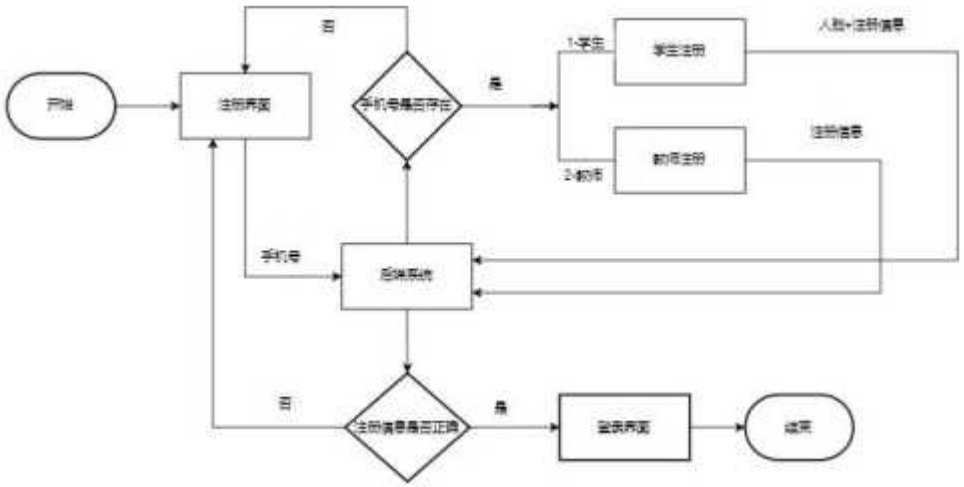


图4.2.2 前端注册流程图

在用户登录界面，无论是学生还是教师，都会看到一个统一的登录入口。用户需要输入自己的手机号和密码，这些信息将被前端系统收集并打包成一个请求，随后发送给后端服务器进行处理。后端服务器接收到前端发送的登录请求后，会对手机号和密码进行验证。服务器会查询数据库，以确认输入的信息是否与数据库中存储的某条记录相匹配。如果用户的信息被验证为正确无误，后端系统将生成一个响应，该响应中包含了一个JSON格式的数据，其中包含了一个名为role的字段。这个role字段是一个关键信息，它指示了用户的身份是学生还是教师。前端系统接收到这个响应后，会解析JSON数据，并根据role字段的值来判断用户的身份。一旦身份被确认，前端系统将根据用户的身份，自动跳转到相应的首页。前端登录流程图如图4.2.3所示。

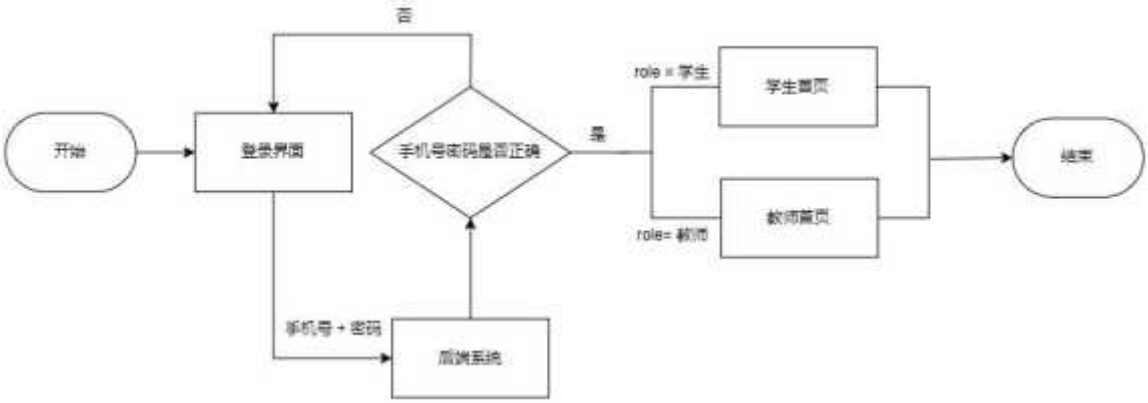


图4.2.3 前端登录流程图

4.2.3 在线监控模块

在线监控模块的主要功能是在学生考试过程中实时监控学生的行为。学生通过打开摄像头，其视频数据可以被教师实时观察。这一功能基于现代技术，利用了WebRTC和WebSocket来实现学生和教师之间的点对点视频传输。

WebRTC用于实现点对点的视频传输，使学生能够将视频流直接传输给教师。而WebSocket则通过服务器作为媒介，协商WebRTC连接的建立参数。例如，教师需要拉取特定学生的监控视频数据，而不是其他学生的视频数据，这需要在服务端设计算法来实现该功能。

在线监控示意图如图4.2.4所示。

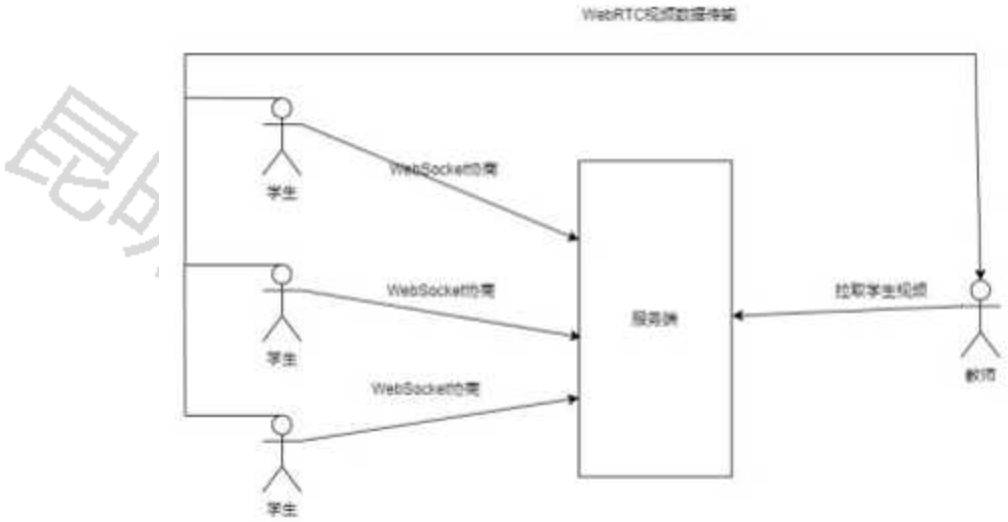


图4.2.4在线监控示意图

4.2.4 在线考试模块

在线考试模块为了确保学生在教师发布试卷后能够顺利参加考试。在考试界面上，学生首先会收到注意事项提示，然后进行人脸识别以验证身份。一旦身份验证成功，考试界面将进行一系列功能的初始化，包括建立WebSocket连接以便与后端进行实时通信，设置WebRTC以实现学生和教师之间的实时视频传输，存储考试倒计时时间以确保学生能够及时提交答卷，打开摄像头以便监控考试过程，以及将浏览器切换到全屏模式以减少外界干扰。初始化完成后，前端会解析后端发送的试卷内容JSON结构字符串，并以清晰易读的格式呈现试卷内容，以便学生可以清晰地理解考试要求。

在线考试流程图如图4.2.5所示。

图4.2.5在线考试流程图

4.2.5 试卷题库班级模块

试卷题库班级模块是系统的核心模块，其中也是涉及内容最多的部分，该模块包含了教师的试卷管理、题库管理、学生的答卷管理、班级管理。

（1）班级管理

首先要做的就是班级管理，因为本系统教师和学生都是以班级作为基本单位进行试卷考试的，教师的班级管理前端文件主要集中在ClassList的Vue文件中，这里面涵盖了班级管理的基本操作，例如创建班级、邀请学生加入班级、导出班级学生信息、班级解散等。学生主要是加入班级、退出班级，学生在个人信息中填写班级名称或者是班级代码即可加入班级，点击退出班级也就退出该班级了。

创建班级：前端通过封装的方法createClassByName将班级名称装载HTTP请求的请求体中发送给后端，让后端创建该名称的班级。

教师班级创建流程图如图4.2.6所示。

否

图4.2.6 教师班级创建流程图

邀请学生加入班级：教师邀请学生可以单独邀请，也可以批量邀请，如果是单独邀请，只需要填写学生的学号即可邀请。如果是批量邀请，需要使用系统固定的邀请模版，模版使用的是Excel文件，第一列填写学生的姓名，第二列填写学生的学号，前端会使用指定的接口把Excel传输到服务端进行解析出学号邀请学生加入班级。

（2）题库管理

题库管理的前端文件主要由以下几个Vue文件组成：TitleList、CreateSingleTitle、CreateMultipleTitle、CreateJudgeTitle、CreateFillTitle、CreateShortTitle。它们分别对应题目列表、创建单选题、创建多选题、创建判断题、创建填空题、创建简答题的功能。每个Vue文件都有不同的网络请求接口来创建题目，但它们传输给后端的数据结构基本相同（除题目列表外）。题目列表式主要使用的是Element UI 的table显示的题目的详情。创建题目的数据主要存储在名为ruleForm的对象中，包含以下字段：题目名称（name）、题目分数（scores）、题目类型（type）、选项（选择题和判断题的选项）、标准答案（answer）、教师ID（teacherId）。这些字段构成的数据结构被封装成JSON格式，然后通过封装好的API发送给后端。后端负责接收这些数据，并将其存储到数据库中。

（3）试卷管理

试卷管理和题库管理类似，但是试卷发布有较大的差别，主要集中在releasePaper这个前端文件中实现。试卷的内容同样存储在一个finalJson对象中，其中也包含试卷名称（paperName）、考试总时间（examTotalTime）、发布的班级ID（selectedClass）、是否允许学生考试后查看（isAllowCheck）、每个子标题下的题目（subheadings）、教师ID、是否立即发布(isReleased)、预发布时间（preReleaseDate）。子标题里面有个对象cardData存储的是多个题目，里面主要是教师选择的题目ID，并不保存题目的真实内容，随机题目就是教师设置一个题目数量给后端让后端随机发送指定题目个数给前端。教师通过发送finalJson对象给后端，后端负责把数据存储到数据库。前端设置了预发布按钮，可以在教师指定的时间内发布给学生端，预发布和立即发布使用不同的接口，但是唯一不同的是只是设置了一个标识isReleased，该标识为0时表预发布，preReleaseDate不为空，为1的时候标识立即发布，该表示主要是告诉后端是否为立即发布。

教师创建试卷流程图如图4.2.7所示。

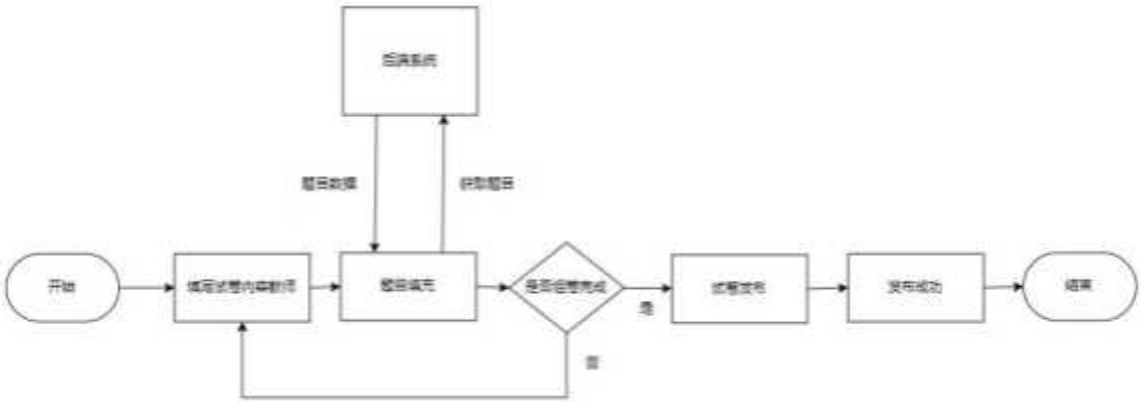


图4.2.7 教师创建试卷流程图

4.2.6 数据统计模块

数据统计模块主要是可视化数据给用户直观的看到自己的相关数据，学生主要是在首页使用三个el-card显示学生的试卷总数、未完成试卷数以及教师已经批改完成的数据,前端主要通过getStudentHomeData接口携带学生ID获取后端统计回来的数据。除此之外，学生首页还使用https://picsum.photos/460/200?random=${index + 1}网站请求随机生成了10张460 × 200大小轮播图显示在三个el-card上方吗，美化学生的首页。

教师端在首页使用四个el-card和echart的饼图分别可视化教师的学生数量、班级数量、题目数量、试卷数量、班级人数分布、班级试卷分布，这些数据都是教师首页通过getHomeData接口获取后端统计的而来的。

4.2.7 JWT验证模块

前端网络请求为了后端身份认证需要在Vue的请求拦截器interceptors中往HTTP请求头中加入Authorization：`Bearer ${token}`，token是在登录成功后，后端会签发token和refreshToken的令牌。refreshToken是用来刷新token的，如果token过期了就会使用refreshToken来向后端请求一个新的token。本系统中token设置过期时间为12小时，refreshToken为2个月。如果refreshToken也过期了就需要用户输入手机号和密码重新登录。

JWT认证流程如图4.2.8所示。

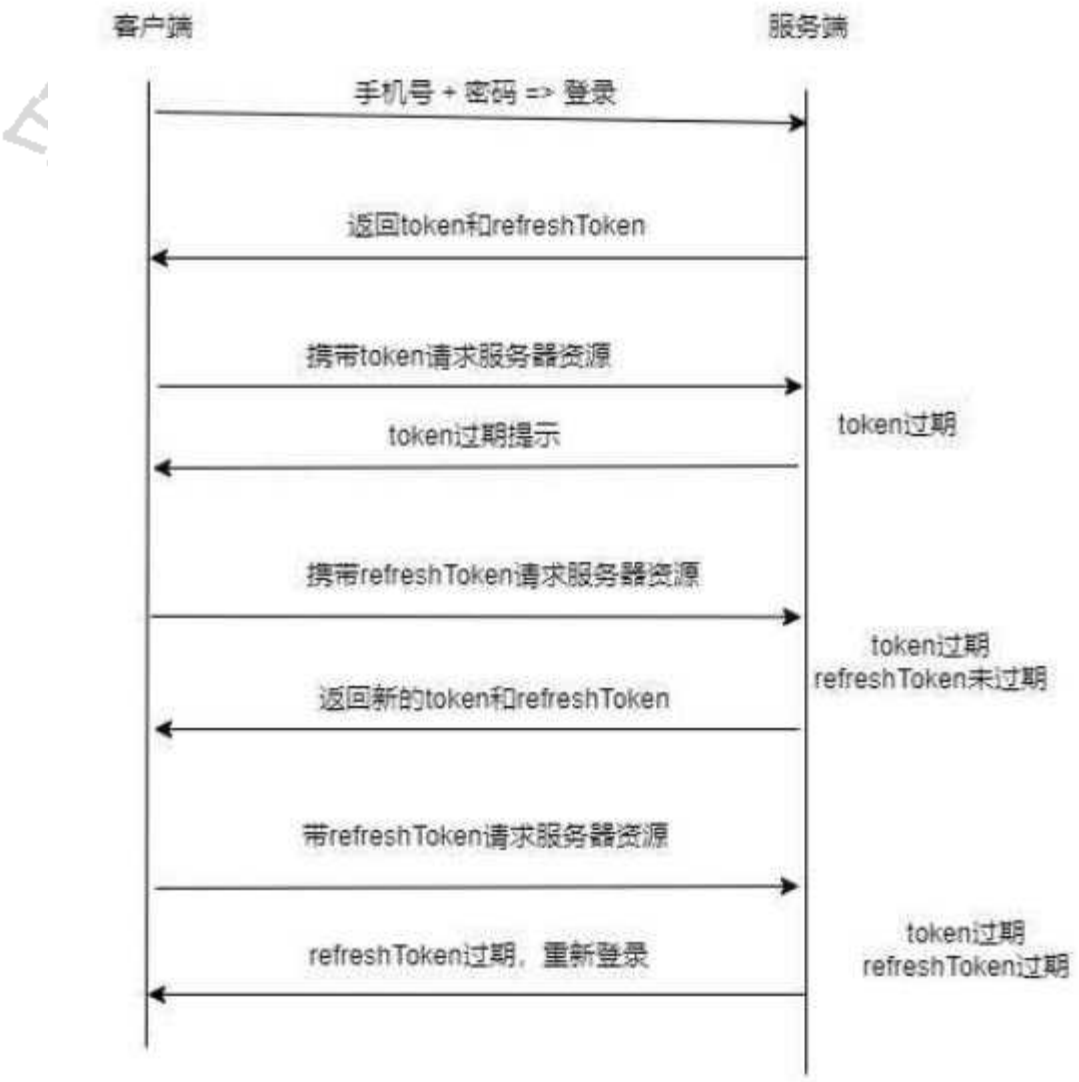


图4.2.8 JWT认证流程

4.3 后端系统详细设计

4.3.1 系统总框架

在后端系统的设计中，采用了一种分层架构，以实现代码的组织和解耦。系统总框架如下：

（1）控制层（Presentation Layer）：

控制层由拦截器组成，具体来说是在请求处理之前进行一些预处理操作，比如这里实现的是JWT（JSON Web Token）的验证逻辑。这个拦截器主要关注于授权和安全性检查，确保每个需要权限保护的API请求都携带了有效的JWT令牌。

（2）表现层（Presentation Layer）：

表现层由控制器（controller）组成，负责处理来自客户端的请求和向客户端发送响应。控制器将请求路由到适当的服务，并将结果返回给客户端。

（3）业务逻辑层（Business Logic Layer）：

业务逻辑层包含服务（service），负责实现应用程序的业务逻辑，Imp服务接口实现层。这些服务处理来自控制器的请求，并协调数据访问层与数据库进行交互，。

（4）数据访问层（Data Access Layer）：

数据访问层由数据访问对象（mapper）组成，负责与数据库进行交互。它们提供了一种将领域对象映射到数据库结构的机制，并执行数据操作，如查询、插入、更新和删除。

（5）实体层（Entity Layer）：

实体层包含领域对象（entity）和数据传输对象（DTO）。领域对象表示业务领域中的实体，而数据传输对象用于在不同层之间传输数据。

（6）配置和工具层（Configuration and Utility Layer）：

这一层包含了配置文件（configration）、拦截器配置（interceptor）、异常处理类（exception）以及通用的工具类（util）。配置文件用于配置系统的行为，而工具类提供了一些通用的功能，如日期处理、Md5码生成等。在线考试系统的后端系统分层架构图如图4.3.1所示。

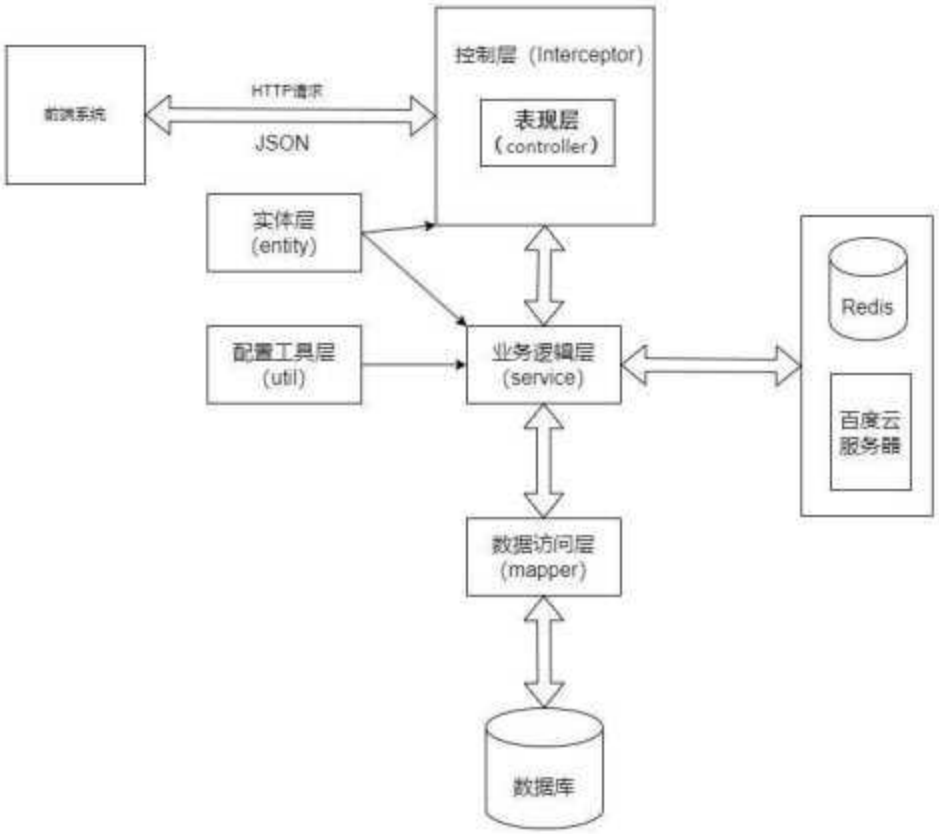


图4.3.1 后端系统分层架构图

4.3.2 注册登录模块

本系统中使用手机号码作为用户的唯一标识，所以在后端中，教师和学生用户注册都要经过uploadRegisterInfo接口检查前端上送的手机号码是否在数据库中存在，存在则说明该用户已经存在数据库中，不允许再次注册，防止数据错乱。

学生注册和教师注册存在比较大的差别，学生注册需要使用uploadFace和studentRegister两个主要接口，前端上传学生的人脸照片和注册信息，uploadFace需要对人脸照片解析识别存储到数据库。

人脸照片解析流程图如图4.3.2所示。

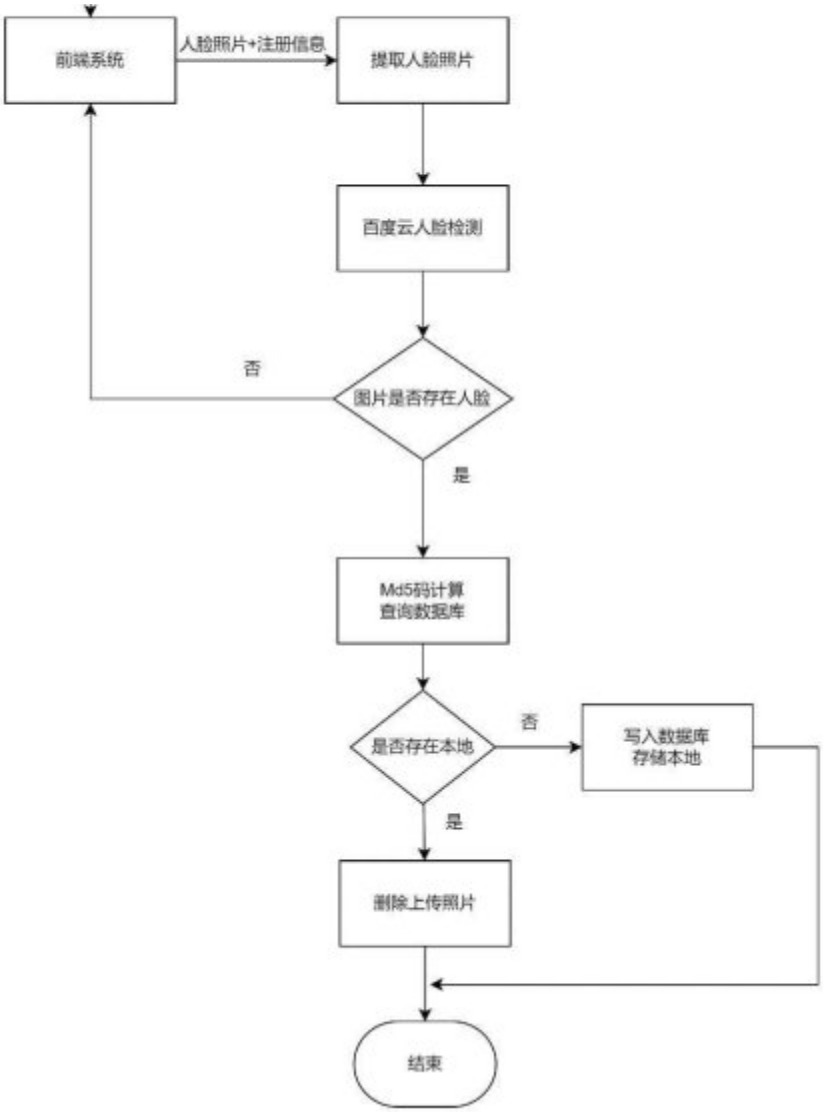
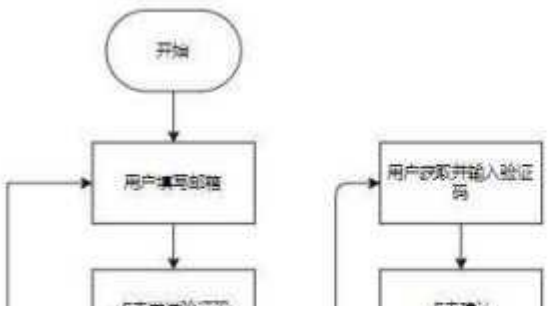


图4.3.2 人脸照片解析流程图

人脸解析完成后studentRegister负责把注册信息存储到数据库，教师注册则是使用teacherRegister接口把注册信息写入数据库，studentRegister和teacherRegister类似，其中用户昵称都是使用随机算法生成8位英文符号。

登录和注册流程类似都是检查用户输入的手机号是否存在验证是否已经注册过了，注册过后就验证上传的密码和数据库的密码是否一致，一致就判断是学生是否还是教师身份返回数据给前端。

用户忘记密码后可以通过邮箱找回密码，该功能通过使用一个主邮箱服务器，发送给需要找回密码的用户邮箱验证码，用户通过获取自己邮箱的收到的验证码设置新密码。具体流程用户首先需要填写自己的邮箱地址，然后点击发送验证码按钮。接着，终端会获取到用户的邮箱号，并生成一个随机验证码存入Redis缓存中，并设置过期时间2分钟。随后，这个验证码会被发送到用户填写的邮箱里。当用户收到验证码后，他们需要将其输入到相应的输入框中，并点击确认按钮。此时，终端会获取到用户输入的验证码和邮箱信息，并从Redis缓存中取出对应的验证码进行比对。如果校验成功，则跳转至设置密码界面；否则不跳转。邮箱找回密码如图4.3.3所示。



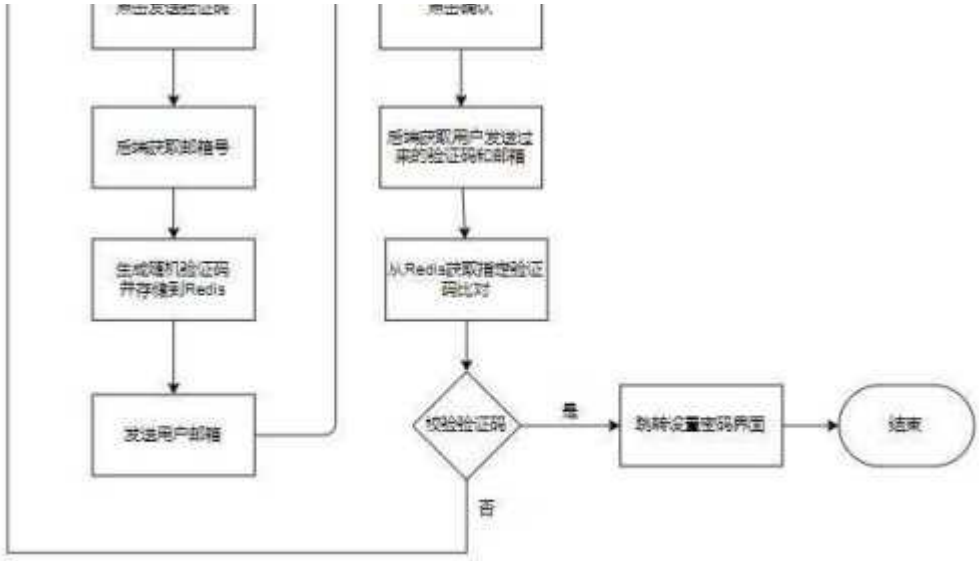


图4.3.3 邮箱找回密码流程图

4.3.3 在线监控模块

后端在线监控模块主要是使用WebSocket协商WebRTC连接的建立并控制识别教师拉取自己学生在线考试的视频流。后端的WebSocket主要使用WebSocketVideoTeacher和WebSocketVideoStudent类和前端建立连接。WebSocketVideoTeacher负责使用教师ID和教师端建立连接，类内有一个teacherSessionsMaps哈希集合，用于映射教师ID和教师的专属连接session，方便在连接断开后清理相关资源。WebSocketVideoStudent负责使用学生ID和学生建立连接，类内使用studentIdToSession来记录学生ID对应的连接session，学生在建立连接后会查询向数据库查询自己属于哪一个教师管辖并使用Redis缓存记录一个键值对，键对应的是教师ID，值对应的是该教师管辖下并在线考试的学生ID集合。当教师要拉取学生的视频数据的时候需要在Redis缓存查询自己ID对应的学生ID集合，并通过studentIdToSession给学生端发送协商信息。

后端在线监控模块示意图如图4.3.4所示。

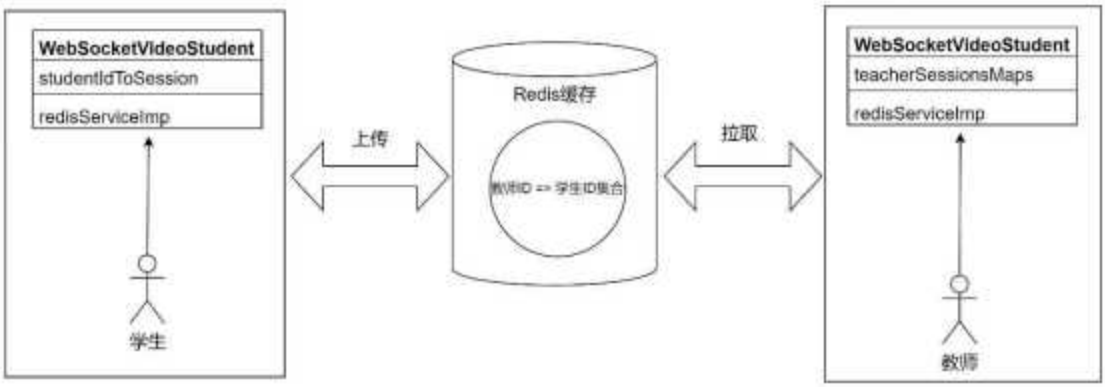


图4.3.4 后端在线监控模块示意图

4.3.4 在线考试模块

后端系统在在线考试模块中扮演着关键角色，主要负责以下几个方面：

系统首先需要集成人脸识别功能，以验证考试前学生的身份。这包括对学生上传的照片进行识别和验证，以确保只有被授权的学生可以参加考试。人脸识别使用的是百度云的人脸对比接口。

其次，后端系统需要与数据库进行交互，根据学生的身份和考试安排，分配适当的试卷给学生。这需要高效地查询数据库，并确保试卷分配的准确性和及时性。

在考试过程中，系统需要定时向前端返回考试倒计时信息，以确保学生在考试期间能够准确地了解剩余时间。为了保证考试时间的准确性，系统需要校准前端倒计时。考试界面初始化的时候前端会和后端WebSocketServerImp建立一个专门发送信息的WebSocket连接，建立连接后后端在onMessage方法中标识前端发送的uploadExamTotalTime

上传考试总时长，服务端则会把时间以到期时间的方式存储在Redis中，利用Redis的自动计时以及java的ScheduledExecutorService计时器每分钟从Redis获取变量过期时间通过uploadRemainingTim推送给前端，这样就可以实现每分钟校准前端的倒计时。
计时器校准流程图如图4.3.5所示。

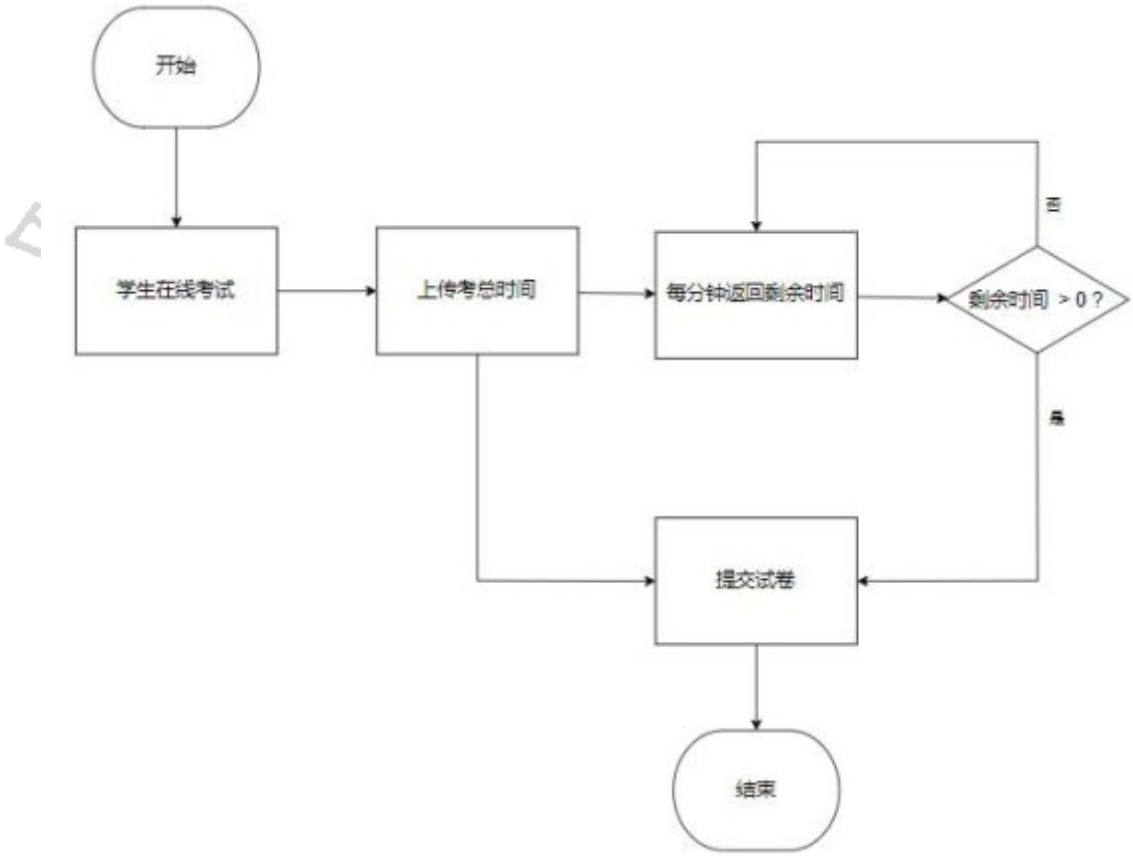


图4.3.5 计时器校准流程图

4.3.5 试卷题库班级模块

后端系统中题库试卷班级模块的业务逻辑主要集中在TitleServiceImp、PaperServiceImp、ClassServiceImp这几个文件中，分别是题库管理、试卷管理、班级管理。

题库管理中主要功能就是题目创建、删除、随机分配。大多数是对数据库的增删改查，题目内容在数据库中是以JSON类型存储的，使用的JSON解析库是Gson，这样组织的目的更有利于内容的结构化，也方便后端读取解析提供给前端显示。比较独特的功能随机题目分配，后端接受到前端发送过来的随机题目数量后，会先去数据库查询该教师下的所有题目获取一个题目ID数组，使用Collections.shuffle打乱ID顺序，然后subList截取一定数量题目返回给前端，这样就实现了随机分配题目。
题目随机分配流程图如图4.3.6所示。

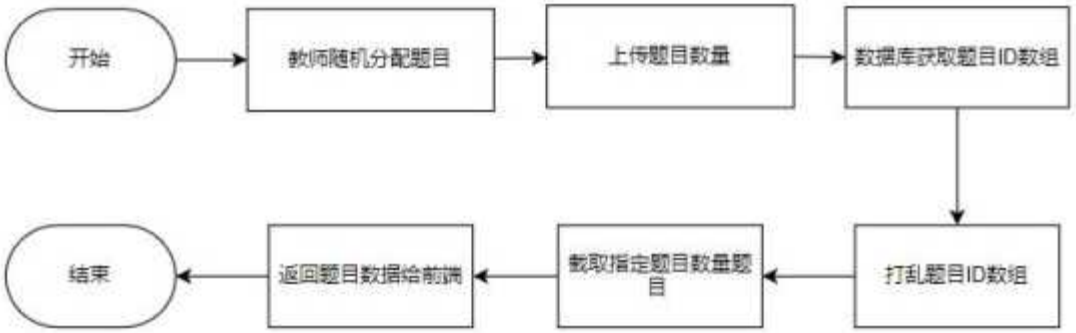


图4.3.6 题目随机分配流程图

试卷管理后端中主要是对前端发送过来的试卷组织存储在数据库中、对数据库中试卷进行查询、删除、重复率查询、预发布、试卷自动批改等操作，其中试卷重复率查询、预发布、自动批改是后端系统中比较核心的功能。

试卷重复率查询是在教师发布试卷的时候进行的，主要是对比该教师数据库内每张试卷内的题目ID和前端发送过来的试卷里面的题目集合交集率是否超过80%，超过则会提示前端，否则默认创建。

交集率计算公式：

$$交集率 = \frac{交集元素个数}{min(列表\ 1\ 元素个数，列表\ 2\ 元素个数)} \times 100 \quad (4\text{-}1)$$

试卷重复率查询算法流程图如图4.3.7所示。

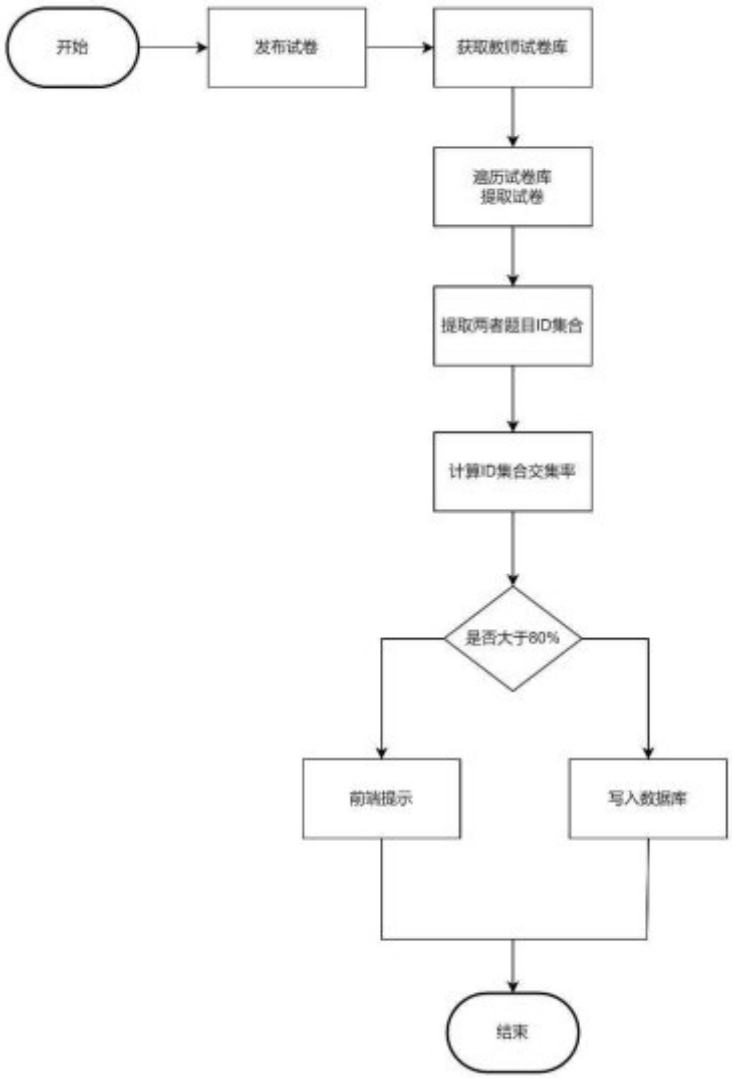
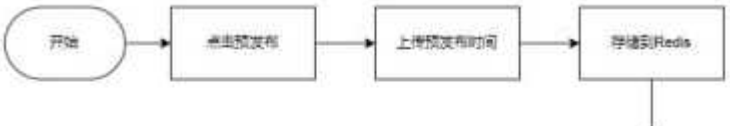


图4.3.7 试卷重复率查询算法流程图

预发布功能本系统是在数据库中用一个字段is_released来表示是否已经发送给学生端了，1表示已经发送给学生端了，0表示还为发送给学生端。预发布需要前端教师选择一个预发布时间发送给后端，后端会把时间以教师ID作为key，预发布时间作为值存储到Redis缓存中，并在本地建立一个计时器，每分钟查询一下Redis缓存中的预发布时间，对比预发布的时间和当前时间，如果预发布时间小于当前时间了，就说明要发布给学生了，就会执行数据库更新操作，把is_released设置为1，否则不进行操作。

预发布功能示意图如图4.3.8所示。



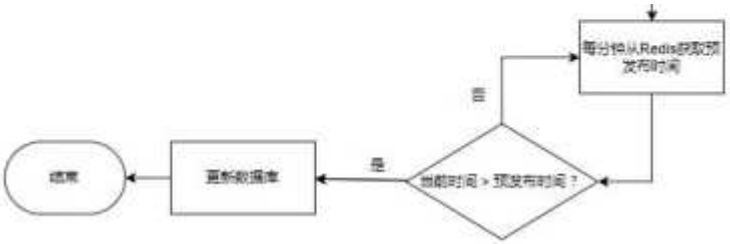


图4.3.8 预发布功能流程图

自动批改的过程包括教师手动批改选择题和判断题，以及后端系统自动批改填空题和简答题。具体流程如下：教师通过系统界面手动批改学生提交的选择题和判断题。这些题目的答案在试卷中已经提供，教师只需对比学生的答案与标准答案是否一致，并给出相应的评分。评分后，教师将分数发送回后端系统，系统将分数写入数据库，更新对应学生的成绩记录。后端系统需要根据学生的答题记录从数据库的 exam_records 表中获取学生的答题记录，以及从 title 表中获取对应题目的标准答案。然后，后端系统使用 singleCompareAnswers、multipleCompareAnswers 和 judgeCompareAnswers 接口，将学生的答案和标准答案进行字符串比对。这些接口使用 Java 的 equals 方法进行比对，如果比对正确，则将相应题目的得分记录到 exam_records 表中；否则，不对数据库进行操作（默认字段值为0）。通过以上流程，后端系统能够自动完成填空题和简答题的批改，减轻了教师的工作负担，并确保了评分的准确性。

班级管理后端主要负责对数据库中的 class 表进行增删改查操作。创建班级时，后端会在 class 表中添加一条记录，并指定对应的老师。解散班级则是从 class 表中删除指定记录。当学生加入班级时，后端会将学生的信息写入 student 表，并在记录中标明所属班级的ID。如果是教师批量邀请学生，后端会解析提取模板文件中的学号，并根据学号逐个向 student 表中写入相应的班级ID。

4.3.6 数据统计模块

数据统计模块的后端主要任务是根据前端的需求查询数据库中的数据。例如，当前端需要获取班级的平均分时，后端会查询数据库以获取该班级所有学生的成绩，并计算平均分。该模块的设计旨在为前端可视化提供支持，使用户能够更直观地了解数据。通过对数据库中的数据进行统计和分析，后端系统能够生成各种图表和报表，帮助用户更全面地了解班级、学生或试卷的情况。这种直观的数据展示方式不仅提高了用户体验，还能帮助用户更有效地进行数据分析和决策。

4.3.7 JWT验证模块

后端的JWT模块主要负责签发和验证加密后的 token 和 refreshToken。签发操作通常在登录接口中进行，当验证用户的手机号和密码正确后，使用 JWT.create() 方法生成 token 和 refreshToken，其中将用户ID作为 JWT 的载荷，并使用 HMAC256 算法进行加密签名。然后将生成的 token 和 refreshToken 返回给前端。

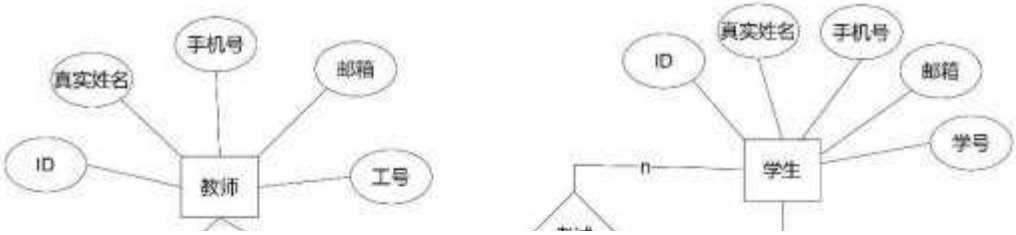
验证操作通常在控制层的前置拦截器中进行，首先会检查 HTTP 请求中的 Authorization 是否存在，若不存在则提示用户登录。若存在，则提取出 token 并进行解码验证，验证 token 是否正确且未过期。如果验证不通过，则拦截请求，不允许访问服务器资源。

通过这种方式，JWT 模块能够有效地保护后端资源，只有经过身份验证和授权的用户才能访问相应的接口和资源，提高了系统的安全性和稳定性。

4.4 数据库设计

4.4.1 数据库概念结构设计

概念结构设计是将用户需求进行抽象化的过程。E-R 图（Entity Relationship Diagram）是一种有效的方式，用来描述概念模型，展现实体之间的联系。在E-R图中，实体指的是具有属性的对象，在数据库中通常对应着表。通过展示实体和属性之间的关系，E-R图展示了数据库的整体逻辑结构。本文采用E-R图来呈现系统的数据库概念模型，具体来说是在线考试系统的E-R图，示例如图4.4.1所示。



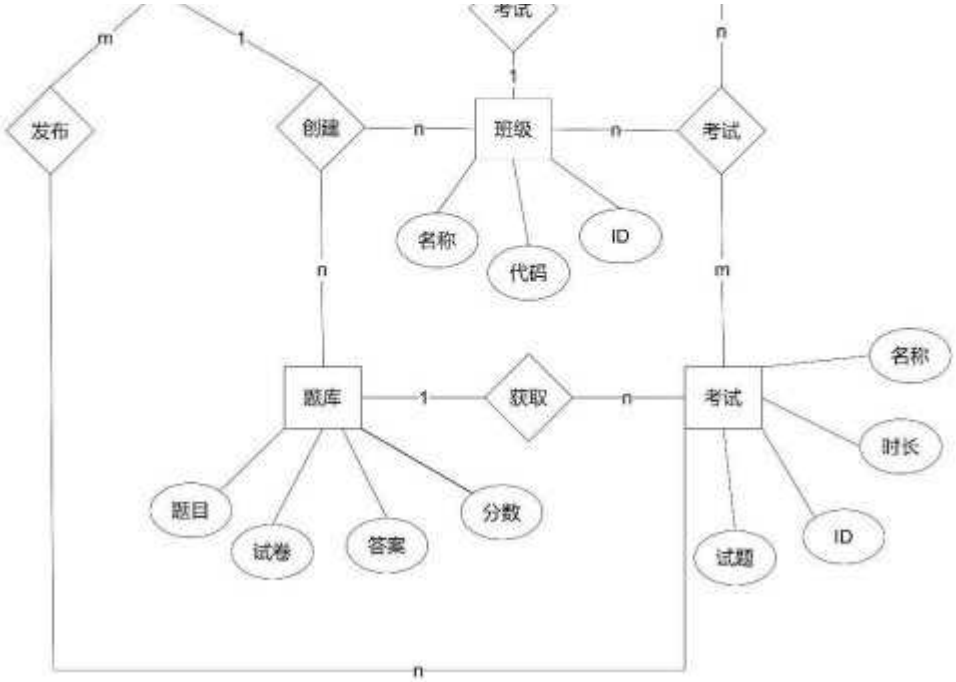


图4.4.1 在线考试系统E-R图

4.4.2 物理结构设计

本系统采用MySQL数据库。系统根据模块划分为用户信息模块、班级模块、题库模块、试卷模块、在线监控模块、在线考试模块、数据统计模块、JWT验证模块。其中最主要的是用户信息模块、班级模块、题库模块、试卷模块四个模块。本系统的数据库表主要根据这四个模块的功能和用户的角色类型进行表的划分和表中字段的设计的。数据库表有用户表（user）、学生表（student）、教师表（teacher）、班级表（class）、题库表（title）、试卷表（paper）、学生试卷表（student_paper）、考试记录表(exam_records)、试卷班级映射表(paper_class)、文件表(file)。这些表都是存储examination_system数据库下面的。

这些表格存储了在线考试系统所需的各种数据。当用户通过后端接口访问这些数据时，后端程序会根据需求调用数据库来执行增加、删除、修改和查询等操作。以下是数据库表格的详细字段说明：

用户信息表user，用来存放用户的信息。详情见表4-1

表4-1 用户信息表

字段名	数据类型	说明	备注
user_id	int	用户ID	主键，自动递增
user_name	varchar(50)	名称	无
user_password	varchar(50)	密码	无
user_realName	varchar(50)	真名	无
user_role	int	角色	无
user_avatar_id	int	头像文件ID	无
user_gender	char(1)	性别	默认男
user_age	int	年龄	无
user_email	varchar(50)	邮箱	无
user_phone	varchar(50)	手机	无

教师表，用来存储学生信息。详情见表4-2

表4-2 教师信息表

字段名	数据类型	说明	备注
teacher_id	int	教师ID	主键，自动递增
teacher_number	varchar(50)	工号	无
user_id	int	用户ID	外键

学生表，用来存储学生注册信息和人脸信息。详情见表4-3
表4-3 学生信息表

字段名	数据类型	说明	备注
student_id	int	学生ID	主键，自动递增
student_number	varchar(50)	学号	无
student_face_id	int	人脸文件ID	无
user_id	int	用户ID	外键
class_id	int	班级ID	外键

试卷表，存储教师创建的试卷信息。详情见表4-4
表4-4 试卷表

字段名	数据类型	说明	备注
paper_id	int	试卷ID	主键，自动递增
paper_name	varchar(50)	名称	无
paper_total_time	bigint	考试总时长	无
paper_content	json	试卷内容	无
paper_score	int	分数	无
paper_create_stamp	timestamp	创建时间	无
is_allow_check	tinyint	是否考后查看	默认0
is_released	tinyint	是否发布	默认0
teacher_id	int	教师ID	无

班级表，用来存储班级信息。详情见表4-5
表4-5 班级信息表

字段名	数据类型	说明	备注
class_id	int	班级ID	主键，自动递增
class_name	varchar(10)	名称	无
class_code	varchar(10)	代码	无
student_numbers	int	学生学号	无
teacher_id	int	教师ID	无

考试记录表，学生考试后记录学生的答题信息。详情见表4-6
表4-6 考试记录表

字段名	数据类型	说明	备注
exam_id	int	考试记录ID	主键，自动递增
student_id	int	学生ID	无

paper_id	int	试卷ID	外键
title_id	int	题目ID	无
answer	varchar(100)	学生答案	无
is_correct	tinyint	是否已经批改	默认0
scores	int	分数	默认0
is_favorite	tinyint	是否收藏	默认0

学生试卷表，用来记录发布到学生端的试卷信息。详情见表4-7
表4-7 学生试卷表

字段名	数据类型	说明	备注
student_paper_id	int	ID	主键，自动递增
student_id	int	学生ID	外键
paper_id	int	试卷ID	外键
is_finish	tinyint	是否完成	默认0
is_correct	tinyint	是否批改	默认0
scores	int	分数	默认0
spend_time	int	考试花费时间	默认0

题目表，用来记录教师的创建的题目信息。详情见表4-8
表4-8 题目表

字段名	数据类型	说明	备注
title_id	int	题目ID	主键，自动递增
title_type	int	题目类型	无
title_content	json	题目内容	无
title_create_stamp	timestamp	创建时间	无
teacher_id	int	教师ID	无

试卷班级归属表，用来记录教师创建的试卷归属于哪个班级。详情见表4-9
表4-9 试卷班级归属表

字段名	数据类型	说明	备注
paper_class_id	int	试卷归属班级ID	主键，自动递增
paper_id	int	试卷ID	无
class_id	int	班级ID	外键

文件表，用来记录文件在本地的存储信息，详情见表4-10
表4-10文件表

字段名	数据类型	说明	备注
file_id	int	文件ID	主键，自动递增
file_name	varchar(50)	名称	无
file_type	varchar(10)	类型	无
file_size	bigint	大小(kb)	无

file_url	varchar(255)	本地路径	无
file_md5	varchar(32)	文件md5码	无
file_is_delete	tinyint	是否被删除	默认0
file_enable	tinyint	是否可用	默认0

第五章 系统实现

5.1 系统概述

本系统采用Spring Boot +Vue2框架、Element UI、JWT、WebRTC、WebSocket等现代技术对该系统进行了编码实现，系统的核心模块包括题库管理、试卷管理、在线监控和考试管理。教师可以轻松创建、删除题库，并根据需要灵活组合题目生成试卷。在线监控功能让教师可以实时了解学生的考试状态，确保考试的公平性。学生在规定的时间内参与在线考试，借助系统的便捷性和稳定性完成考试。

本章将全面展示在线考试系统的功能和效果，主要围绕界面展示、运行流程和功能说明展开介绍。

5.2 运行环境

该项目是在Windows 10操作系统上开发的，采用了B/S架构，并且需要在Java JDK 1.8环境下运行。在开发过程中，使用了IDEA作为代码编译器。为了确保项目的本地运行，首先需要启动Spring Boot和Redis后端服务，以便为前端提供必要的接口。随后，启动前端Vue服务器，以实现前后端之间的有效交互。

为了集成人脸识别功能，项目还采用百度云提供的人脸识别接口。这意味着使用该项目需要注册一个百度云账号，并将相关服务部署在后端系统中。为了提高系统的性能和稳定性，本项目采用MySQL 8.0数据库来存储持久化数据，并利用Redis缓存技术来加快数据访问速度。

5.3 主要实现的功能

5.3.1 注册登录模块

注册分两步完成，一步一个界面，步骤1提示用户输入手机号和选择身份，发送后端验证手机号合法才进入下一个步骤，下一个步骤就会根据步骤1选择的身份跳转不同的注册界面，分别是学生注册界面和教师注册界面。主要区别是学生需要上上传人脸，而教师不需要。步骤1注册界面如图5.3.1所示。

图5.3.1 步骤1注册界面

步骤2学生注册界面如图5.3.2所示。

图5.3.2 步骤2学生注册界面

步骤2教师注册界面如图5.3.3所示。

图5.3.3 步骤2教师注册界面

用户输入自己账户的邮箱点击发送验证码后，发送按钮倒计时说明后端已经成功给用户填写的邮箱发送了随机验证码。用户填写上发送到邮箱的验证码，跳转到重新设置密码页面。用户重新输入两次密码即可重置密码。

图5.3.4 邮箱找回密码界面

5.3.2 试卷班级题库模块

班级、题库和试卷之间是都是息息相关的，教师需要创建班级才能创建试卷，有了题库，教师才能从题库中获取题目组成试卷。学生只有加入班级，才能收到教师发布在班级里的试卷。班级管理界面如图5.3.5所示。

图5.3.5 班级管理界面

教师可以在题目列表中查看自己创建的题目或者删除不需要的题目。题库管理界面如图5.3.6所示。

图5.3.6 题库管理界面

教师可以创建自己的题目。在创建题目界面上输入题目的相应信息点击立即创建即可存储到题库中，可以创建包括单选题、多选题、判断题、填空题以及简答题。创建题目界面如图5.3.7所示。

图5.3.7 创建题目界面

教师可以在试卷管理界面立即发布预发布的试卷，也可以继续添加试卷给班级。试卷管理界面如图5.3.8所示。

图5.3.8 试卷管理界面

教师可以在教师发布界面组卷立即发布或者预发布试卷。教师可以在填写完试卷的相关信息后，点击立即发布或者是预发布，立即发布则学生端可以立即接收到教室创建的试卷，预发布则是教师指定时间发布试卷到学生端。试卷发布界面如图5.3.9所示。

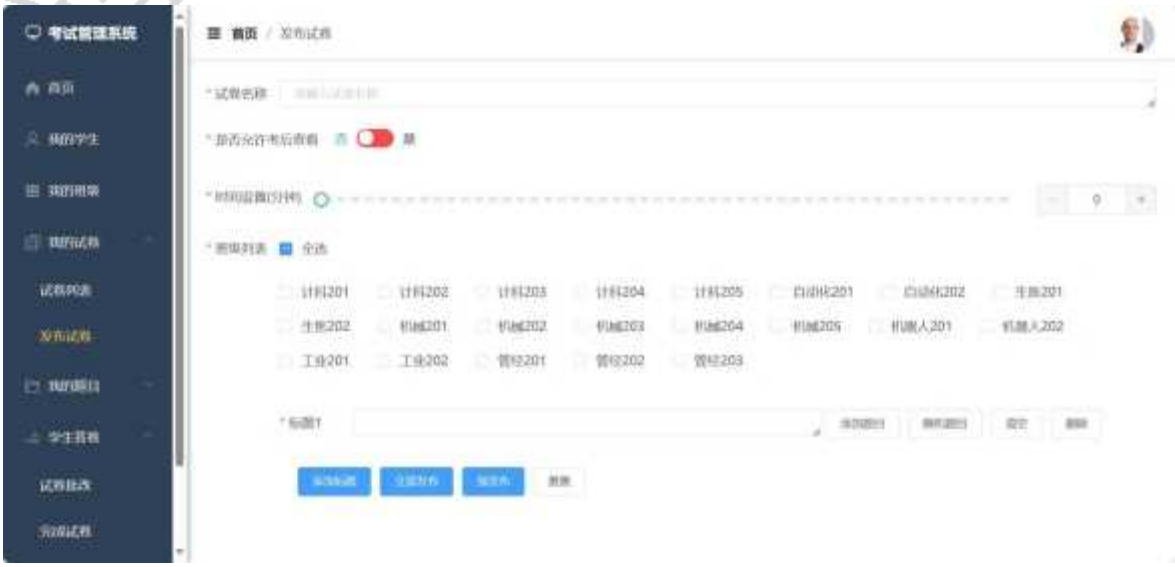


图5.3.9 试卷发布界面

教师需要再学生答卷中批改学生的答卷，选择题和判断题系统会自动批改并提示前端，填空题和简答题则需要教师自行批改，并提交试卷的批改结果。批改界面如图5.3.10所示。

图5.3.10 试卷批改界面

5.3.3 在线考试和在线监控模块

教师发布试卷后可以在“首页”和“我的试卷”查看，首页则是显示10张随机轮播图和3个统计模块卡片。学生首页如图5.3.11所示。

。
图5.3.11 学生首页界面
在“我的试卷”中，学生可以查看自己未完成的试卷，可以在此界面点击试卷卡片项目的开始考试按钮进行考试。如图5.3.12所示。

图5.3.12 我的试卷界面
考试前系统后端会调用百度云的人脸识别模块，确保是账号本人才能进行考试。人脸识别如图5.3.13所示。

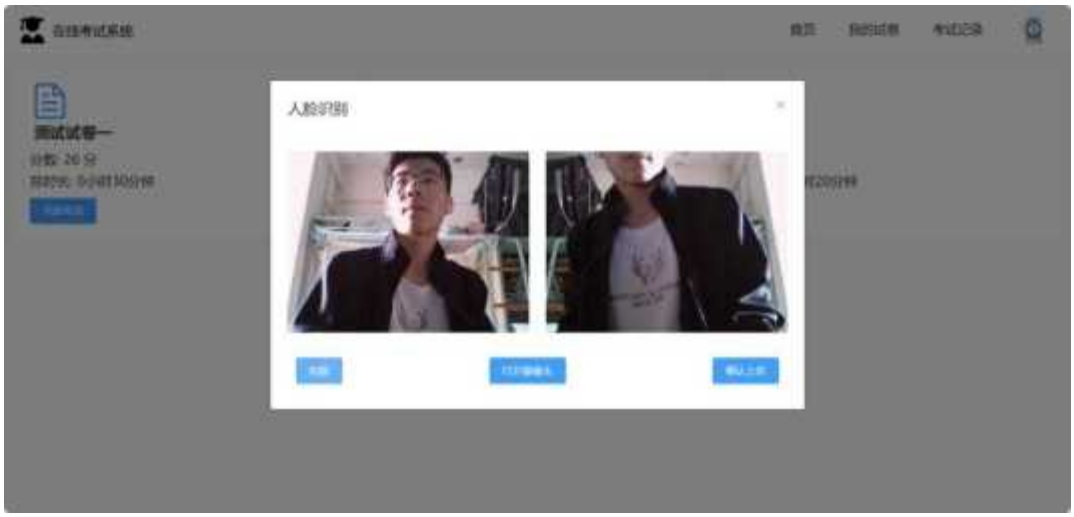


图5.3.13 人脸识别界面
学生的在线考试界面，系统会打开摄像头，在线监控学生的考试行为，学生可以在考试界面按要求填写考试答案，填写完成，到页面末尾提交试卷即可。在线考试和在线监控如图5.3.14所示。

图5.3.14 在线考试界面

教师在“在线监控”选项卡刷新页面，即可查看学生的在线考试行为，监督学生是否存在作弊动作。教师在线监控如图5.3.15所示。

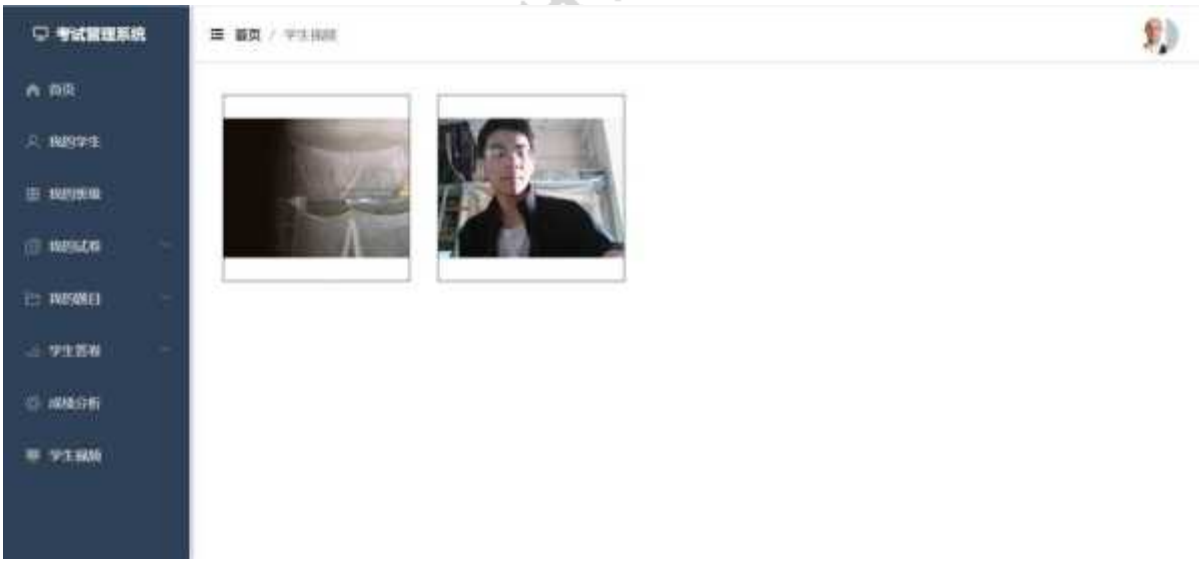


图5.3.15 教师在线监控界面

5.3.4 数据统计模块

学生的统计模块在学生的首页，前面已经展示过，教师统计模块有两个，分别是首页展示基本的信息，然后是成绩分析部分。教师首页统计如图5.3.16所示。

图5.3.16 教师首页统计界面

教师可以在成绩分析选项卡中查看每张试卷每个班级的平均分，系统使用柱状图和折线图展示已经考试了的班级平均分。成绩分析如图5.3.17所示。

图5.3.17 成绩分析界面

5.3.5 个人信息模块

个人信息模块旨在展示用户的基本信息，包括头像、真实姓名、昵称、学号（或工号）、联系电话等。在前端，使用身份鉴别，同一个Vue页面能够展示不同的信息。对于学生用户，页面将额外展示班级信息，而对于教师用户，则不显示该信息。这样的设计可以更好地满足用户的需求，并提升用户体验。学生个人信息界面如图5.3.18所示。

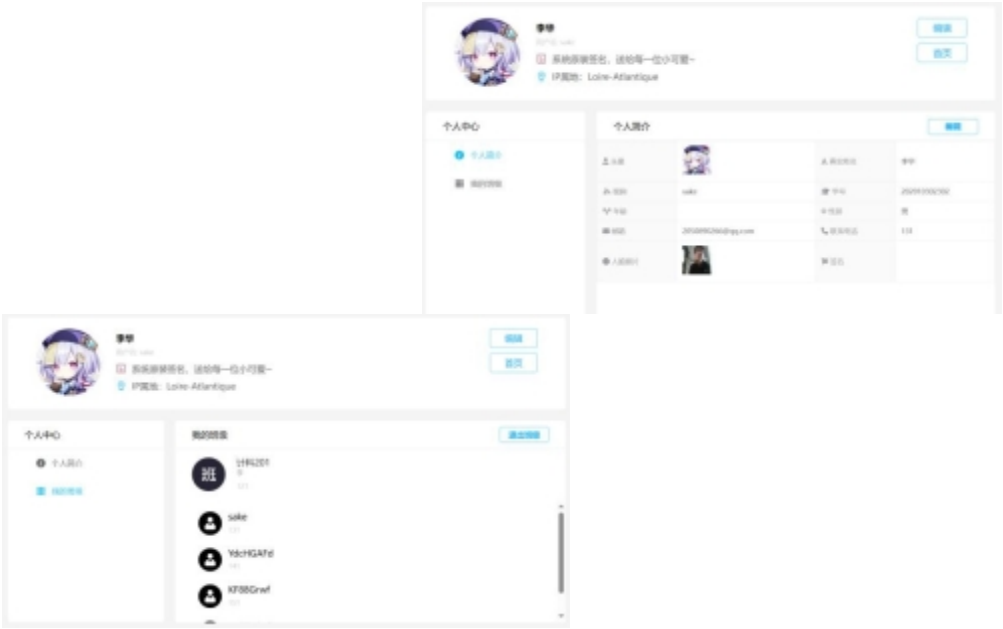


图5.3.18 学生个人信息界面

教师个人信息界面如图5.3.19所示。

图5.3.19 教师个人信息界面

5.3.6 学生题目收藏模块

学生可以在考试记录中选中试卷查看，然后在选择要收藏点击收藏即可。学生收藏页面如图5.3.20所示。

图5.3.20 学生收藏界面

5.4 遇到的问题和解决办法

5.4.1 教师组卷不能跨页题目选择问题

教师在创建试卷选择题库中的试卷时，不可以隔页选择题目，勾选第一页的题目，想要选择第二页的题目的时候，第一页的题目又没有选中了。在处理需求时，发现Element-UI官方没有提供分页跨页多选功能，于是上网查询资料看了许多博客，找到了问题的修复方法。隔页选择如图5.4.1所示。

图5.4.1 隔页选择界面

解决该问题的思路为：在题目选择的el-table标签上添加 :row-key="getRowKey" 属性，getRowKey方法返回题目的ID，行数据的 Key，用来优化 Table 的渲染； 在使用reserve-selection功能与显示树形数据时，该属性是必填的。 类型为 String 时，支持多层访问：user.info.id，但不支持 user.info[0].id，此种情况请使用

Function。最后在选择框那一列 el-table-column上添加:reserve-selection="true"即可解决不能跨页选择题目的问题。如图5.4.2所示。

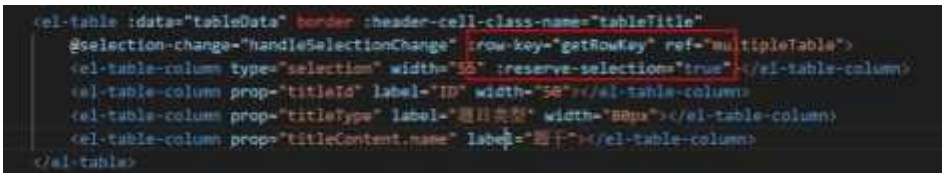


图5.4.2 隔页选择代码修改图

5.4.2浏览器不能打开摄像头问题

其他网络测试（除本地网络）用户在注册或者学生在点击试卷进行考试并进行人脸识别需要打开摄像头的时候，无法打开摄像头，打开浏览器的后台调试界面发现Error in v-on handler: "TypeError: Cannot read properties of undefined (reading 'getUserMedia')"问题。网上查阅资料后是说这个问题是由于浏览器的安全限制引起的。浏览器限制了HTTP协议下的摄像头访问，主要是出于保护用户隐私的考虑。如果任何网站都可以在HTTP协议下随意调用摄像头，用户的隐私将会受到极大的威胁。恶意网站可能会利用这一漏洞进行窃取用户的个人信息和隐私，而用户往往毫无察觉。因此，为了保障用户的隐私安全，现代浏览器采取了严格的安全措施，只有在HTTPS协议下或通过本地IP访问自己本地启动的服务时才能访问摄像头。问题如图5.4.3所示。

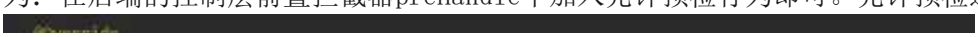
图5.4.3 浏览器摄像头打开失败

解决该问题的思路为：第一种思路就说把http改为https，由于本系统做本地测试，就没有申请公网的https，所以使用的是第二种办法就是在浏览器中访问 chrome://flags/#unsafely-treat-insecure-origin-as-secure，然后在搜索框中输入报错的地址。接着，找到相应Insecure origins treated as secure选项在下面输入框输入vue的首页访问地址，并单击右侧的复选框以启用它。启用这个选项后，浏览器会将该不安全的来源视为安全，允许在HTTP协议下访问摄像头。这个方法仅适用于调试目的，不建议在生产环境中使用，因为会降低用户的安全性和隐私保护水平。如图5.4.4所示。

图5.4.4 允许网络来源通过设置

5.4.3 跨域资源共享中的预检请求问题

在跨域资源共享中，当浏览器发起跨域请求时，即一个源（或域）的 JavaScript 代码在访问另一个源的资源时所发生的情况。会先发送一个OPTIONS方法的预检请求，用于询问服务器是否允许实际的跨域请求。服务器在收到预检请求后，需要进行相应的处理以允许跨域请求。如果服务器不进行处理会导致前端访问后端资源失败的情况。解决该问题的思路为：在后端的控制层前置拦截器preHandle中加入允许预检行为即可。允许预检通过如图5.4.5



```
public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler)
    if (request.getMethod().equals("OPTIONS")) {
        response.setStatus(HttpServletResponse.SC_OK);
        return true;
    }
```

图5.4.5 允许预检通过

第六章系统测试

6.1 测试方案

软件测试是确保软件质量的重要手段之一。它的主要目的在于验证系统是否符合预期需求，并在测试阶段及时发现并解决问题，从而有效降低后期维护成本[25]。在测试在线考试系统时，将采用两种不同的方法来确保系统能够正常运行。首先，会执行正确的操作，检查所有页面和模块是否都能按预期运行。其次，会执行错误的操作，测试系统是否能够提供提示和有效处理错误，以确保系统的稳定性和用户友好性。通过这两种测试方法的结合，可以比较全面地评估在线考试系统的质量和可靠性。

6.2 界面显示测试

在先前的系统实现展示中，通过一系列设计的界面截图，全面展现了系统的外观布局与核心功能模块，确认了包括主页、在线考试、在线监控、用户个人中心等在内的多个关键界面均呈现正常，无明显的视觉错位或功能缺失问题。这些截图不仅体现了UI设计的一致性和美观性，还初步验证了系统在不同界面间切换的流畅度与响应速度。因此本章节不再重复测试相同内容的功能。

6.3 注册登录模块测试

（1）注册测试

在注册界面输入一个已被使用的手机号，然后填写符合系统要求的密码，并点击注册按钮的情况。预期结果应该是立即于页面顶部显眼位置弹出一个醒目的红色信息提示框，其内容清晰地告知用户“该账号已经存在”，以此有效防止了重复注册行为的发生。登录和注册判断手机号是否已经存在后端使用的是同一个接口，在这里就不做登录同样的测试。测试结果如图6.3.1所示。

图6.3.1 重复手机号测试

6.4 人脸识别模块测试

在学生注册界面尝试上传一个没有人脸的照片进行测试，在点击确认上传后，会弹出无法检测到人脸提示，符合系统预期。学生在线考试前的人脸识别调用的是百度云的人脸对比接口，注册调用的是人脸识别接口，两个类似的效果，在这里也不展示在线考试前的人脸识别。无人脸测试如图6.4.1所示。

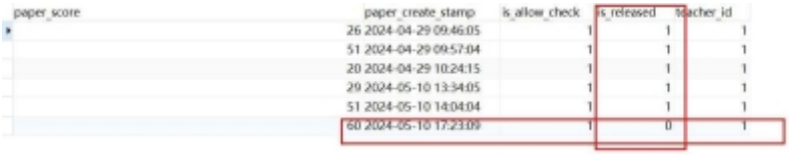
图6.4.1 无人脸测试

6.5 试卷模块测试

（1）预发布测试

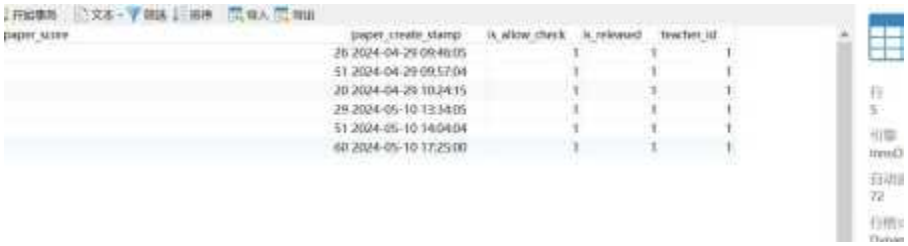
在组卷后点击预发布按钮，选择在2024-5-10 17:24:57时间确定预发布，预计在系统时间2024-5-10 17:25会发布到学生端，符合系统预期。如图6.5.1所示，可以看到预发布后数据库字段的is_released为0，当系统时间到达后，如图6.5.2所示，当2024-5-10 17:25可以看到数据库字段的is_released为1，如图6.5.3所示，以及查看后端控制台该试卷已经发布，如图6.5.4所示，测试结果符合系统预期。

图6.5.1 试卷预发布



paper_score	paper_create_stamp	is_allow_check	is_released	teacher_id
26	2024-04-29 09:46:05	1	1	1
51	2024-04-29 09:57:04	1	1	1
20	2024-04-29 10:24:15	1	1	1
29	2024-05-10 13:34:05	1	1	1
51	2024-05-10 14:04:04	1	1	1
60	2024-05-10 17:23:09	1	0	1

图6.5.2 预发布后数据库



paper_score	paper_create_stamp	is_allow_check	is_released	teacher_id
26	2024-04-29 09:46:05	1	1	1
51	2024-04-29 09:57:04	1	1	1
20	2024-04-29 10:24:15	1	1	1
29	2024-05-10 13:34:05	1	1	1
51	2024-05-10 14:04:04	1	1	1
60	2024-05-10 17:25:00	1	1	1

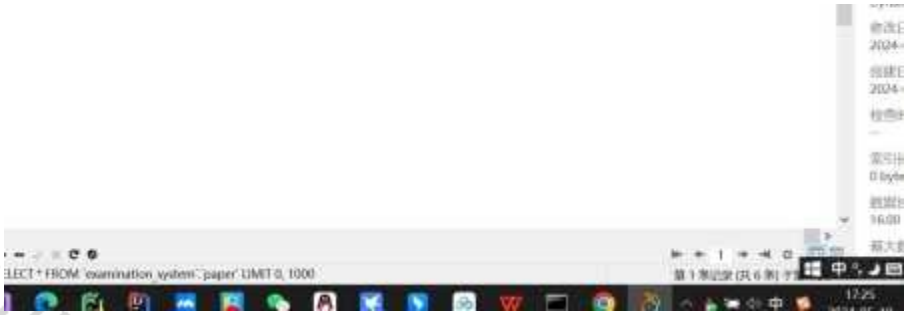


图6.5.3 发布后数据库

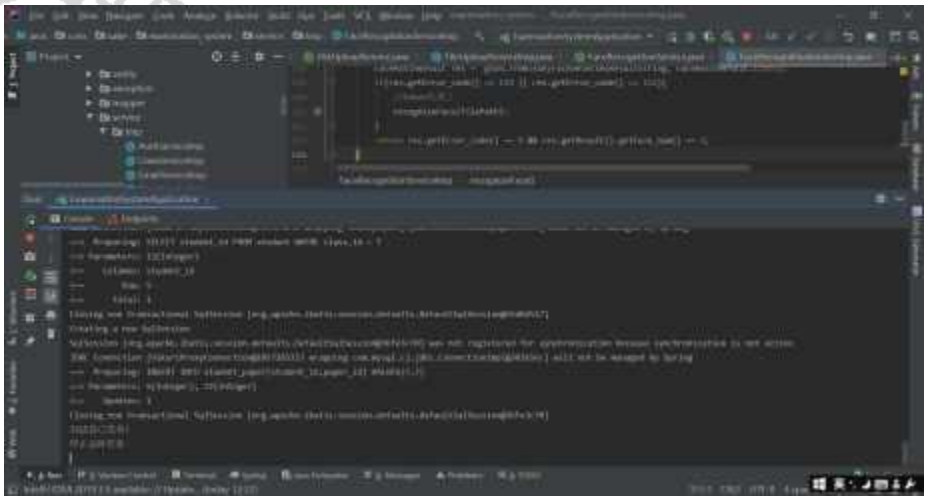


图6.5.4 后端控制台显示

（2）试卷重复率测试

数据库中已经存储了许多试卷，本次测试组卷将选中一些前面试卷选择比较多的题目组一张试卷，测试一下试卷的重复率大于80%时，在发布试卷选项卡中选择发布试卷，填写好相应的内容，选择重复高题目，看系统能否弹窗提示教师。组卷点击立即发布后，系统弹窗提示发现重复率超过80%，系统询问是否确认发布，符合预期效果。试卷重复率测试如图6.5.5所示。

图6.5.5 试卷重复率测试

（3）试卷删除测试

点击试卷列表中的删除按钮，系统会在条目右下角提示用户是否要删除，如图6.5.6所示，防止用户的误操作，

点击确认后，页面立马更新了数据，如图6.5.7所示，符合系统预期。其他删除操作接口类似，本章就不做同样类型测试。

图6.5.6 防误删除提示

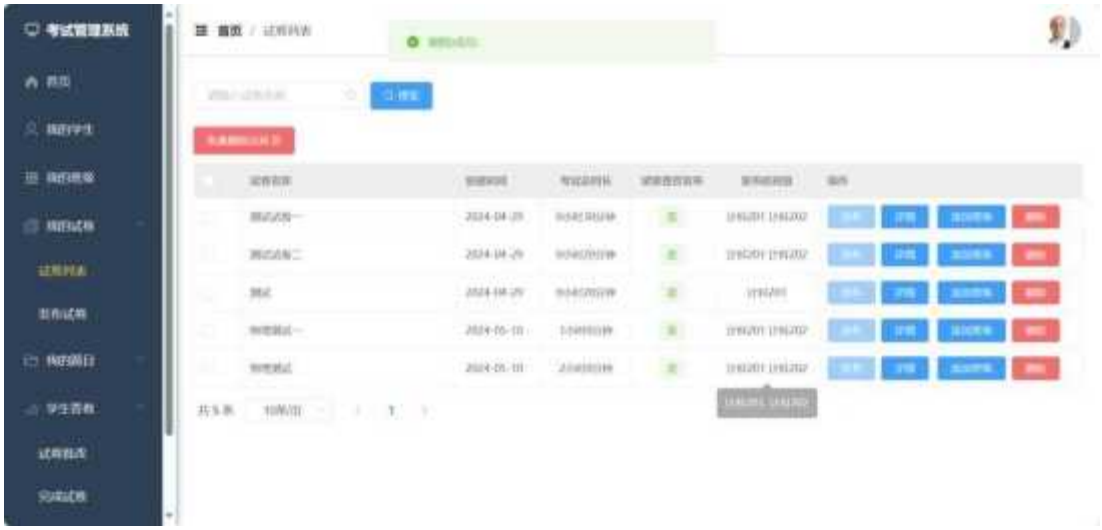


图6.5.7 删除成功

6.6 个人信息模块测试

个人信息模块主要测试修改个人信息，学生加入没有创建的班级和退出班级。学生尝试修改个人信息，更换了头像，改真实姓名为王五，昵称为yuan，年龄输入20后点击提交，系统刷新页面，个人信息修改成功，符合系统预期。如图6.6.1所示。

图6.6.1 修改个人信息

学生个人信息修改成功后如图6.6.2所示。

图6.6.2 修改个人信息成功

学生退出班级测试，在个人信息界面的我的班级选项卡中，点击退出班级后，系统提示退出成功，符合系统预期。如图6.6.3所示。

图6.6.3 退出班级成功

学生加入一个没有创建的班级尝试，在加入班级弹窗中输入测试作为班级名称，系统提示加入失败，班级不存在，符合系统预期。如图6.6.4所示。

图6.6.4 加入未创建班级

6.7 JWT模块测试

为了可以成功测试JWT的token过期，把后端token的过期时间调整为了5分钟，登录成功后，经过5分钟没有请求后端，再次请求服务器资源后显示token过期，错误码为4003；而后使用了refreshToken刷新token成功，再次请求服务器资源，可以看到没有再报4003错误，表明成功请求服务器资源，符合系统预期。token过期成功刷新如图6.7.1所示。

图6.7.1 token过期成功刷新

结论

本文深入探讨了网络交流平台的基础研究领域，细致阐述了应用于本次毕业设计的关键技术要点，并全面地从系统需求的精细剖析、系统架构的精心设计、实际开发的周密实施，到系统全面测试的严格流程，对在线考试系统的实现路径进行了详实论述。

经过全面开发和集成，系统成功涵盖了题库维护、试卷编排、在线考试和在线监控等四大核心模块。教师可以轻松注册并登录系统，在任何时间、任何地点创建班级、管理题库，灵活地组卷。系统能够根据教师设定的条件，随机或手动生成试卷，并自动与题库比对，确保试卷的独特性和题目的多样性。学生登录后即可即时获取教师发布的试卷，通过人脸识别确认身份后开始考试。友好的考试界面提供了试卷名称、总分和倒计时等基本信息，全程打开摄像头确保考试的公正性，并通过在线监控学生的考试行为。为减轻教师的负担，系统还提供了客观题的自动评分功能。所有这些功能的实现效果与我们的预期目标完全一致，为教师和学生提供了极大的便利和保障。

尽管取得显著成效，系统在某些维度上仍有待优化：一方面，在线监控功能偶现视频拉取失败的情况，需学生端刷新后教师端方能接收到考试视频流。这提示系统需进一步提升数据传输的稳定性和即时性，以确保监考过程的无缝衔接。另一方面，关于系统功能性，当前教师在发布试卷时缺乏直接查询试卷重复度的自定义选项。未来可考虑通过前端传递重复率筛选条件至后端，利用已有的API接口智能匹配，从而精准控制和优化试卷内容的原创性。

总结与体会

通过亲手搭建一个Web在线考试平台，让我对Spring Boot和Vue.js框架的理解更加深入。从对网站构建一窍不通，到逐步掌握项目搭建的要领，直至最终能够对网站进行有效优化，这一过程不仅考验了我的学习能力和适应力，还促使我涉猎了大量课外知识，这些宝贵经验无疑为我的职业生涯铺垫了坚实的基础。

毕业设计不仅是对校内所学知识的实践验证，更是一个强化个人技能集的绝佳机会。在此前，我有过运用Python结合Bootstrap进行开发的经历，那是一种前后端未完全分离的模式。而这次毕业设计采纳了现代的前后端分离架构，前端专注于用户界面展示，后端则专司数据处理，两者界限清晰，互不干扰。即使前端出现故障，也不会波及后端服务，这种设计在实践中让我深刻体会到其带来的灵活性与稳定性优势。

在前端界面的创作中，我选择了Element UI这一高效Vue UI组件库，它极大地加速了我的开发进程，减少了重复劳动，让我能更专注于创意与逻辑实现。此外，此项目让我在Spring Boot框架的运用、Vue的安全机制、尤其是JSON Web Token（JWT）的应用，以及数据库的开发与维护技巧上均有颇多斩获，这些技能的精进无疑将成为我职场生涯中的重要助力，提升未来工作的效率与质量。

谢辞

在这匆匆四载的大学时光里，从青涩步入成熟的门槛，每一步都镌刻着成长的印记。随着毕业论文的尘埃落定，心中满溢的不仅是成就感，更多的是对一路相伴人们的深深感激。

首先，我要由衷地感谢我的导师刘英莉副教授。您真的是一个非常温柔体贴的人。感谢您在我的毕业论文中提

出了许多宝贵的建议，帮助我完善了论文。同时，我也要感谢求学路上给予我指点和帮助的所有老师。祝愿您们幸福安康。

其次，对我的家人表达无尽的感激。无论是在寂静深夜里的一盏灯火，还是在面临挑战时那温暖的话语，都是我坚持下去的力量源泉。家人的理解、支持与无私奉献，是我能够安心求学、勇攀学术高峰的坚强后盾。

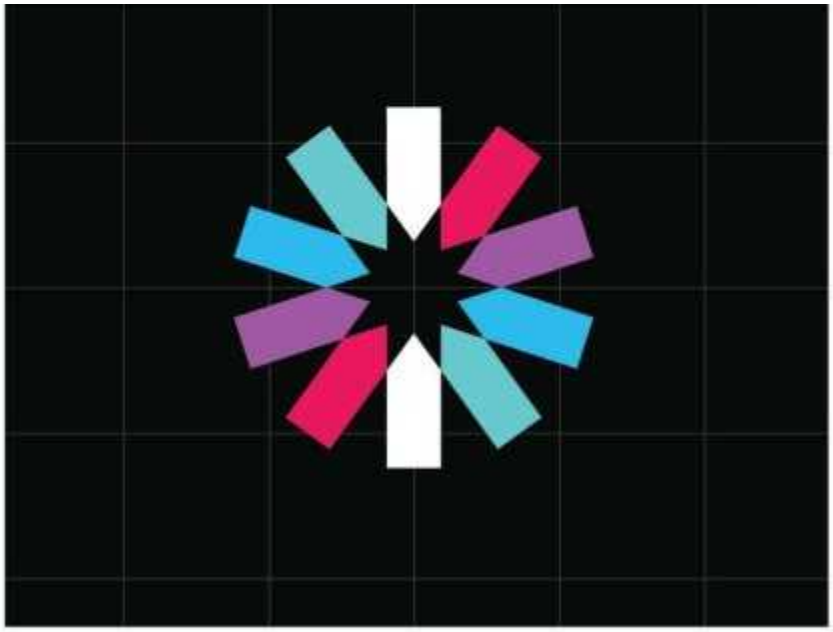
最后，我要感谢大学的同窗们。四年的并肩作战，从教室到图书馆，从理论探讨到实践探索，是你们的陪伴与讨论，让学习之路不再孤单。每一次的思想碰撞，都为我的论文增添了不同的色彩，也为我的大学生活留下了难忘的记忆。感谢你们的鼓励、竞争与合作，共同塑造了这段宝贵的青春时光。

虽然毕业论文的完成标志着一段学习旅程的结束，但它同时也是新旅程的起点。我将带着这份感恩与所学，继续前行，希望能在未来的学习与工作中回馈社会，不负这段美好的岁月。

参考文献

[1]侯平甫,吴长宇,汤莉莉,等. 在线考试模式的改革实践与思考[J]. 中国继续医学教育,2022,14(20):1-4.
[2]尹逸铎,陈科. 基于B/S架构的网络考试系统的设计与实现[J]. 现代信息科技,2021,5(14):21-23,28.
[3]王霏儿. 基于Spring Boot的在线考试系统设计与实现[D]. 江西:江西师范大学,2023.1-2
[4]陈小姣,曾彩霞. 无纸化考试系统质量评价体系的构建与应用[J]. 湖南邮电职业技术学院学报,2022,21(3):103-105,116.
[5]窦营山. 在线考试与传统考试成绩等效性研究——基于2000—2020年国际实证研究的元分析[J]. 中国远程教育，2022(1)：73—84.
[6]张健. 基于混合算法的自动组卷在线考试系统[D]. 扬州：扬州大学，2021.42-48
[7]彭湘华. 人脸识别技术在高职院校课程在线考试中的应用研究[J]. 高教学刊，2020(1)：185—187.
[8]陈海霞. 一种通用在线考试系统的设计[J]. 电脑编程技巧与维护，2021(5)：19—20，44.
[9]徐福江. 基于云计算技术的英语在线考试系统设计[J]. 微型电脑应用，2021，37(3)：27—30.
[10]Azim, N., Naqvi, I., & Rehman, K. U. (2009). Online Examination System and Assessment of Subjective Expression.
[11]Gimeno-Sanz, A., & Siqueira, J. M. D. (2012). Implementing online language exams within the Spanish National University Entrance Examination: The PAULEX Universitas Project. Procedia-Social and Behavioral Sciences, 34(3), 68-72.
[12]Holland, J. (1975). Adaptation in natural and artificial systems: an introductory analysis with application to biology. Control & Artificial Intelligence.
[13]王建,罗政,张希,等. Web项目前后端分离的设计与实现[J]. 软件工程,2020,23(4):22-24.
[14]宋文泽. 企业线上培训系统的设计与实现[D]. 北京:北京交通大学,2022.8-9
[15]王琦. 基于Spring Boot的Java编程作业混合测评系统的设计与实现[D]. 北京:首都经济贸易大学,2022.7-9
[16]翟文辉. 注册电气工程师考试软件系统的研究与设计[D]. 上海:东华大学,2020.8-12
[17]蔡劲. 丹阳市人事考试管理系统的设计与实现[D]. 江苏:江苏科技大学,2021.7-13
[18]王原昭,卢春雨,蒲鹏. 面向高并发在线考试系统的性能优化[J]. 软件,2024,45(2):14-18.
[19]荣艳冬.关于Mybatis持久层框架的应用研究[J].信息安全与技术,2015,6(12):86-88.
[20]叶忠文,黄鹏,施金金. 基于WebSocket的Web实时通信系统[J]. 火力与指挥控制,2014(z1):181-183.
[21]丁常坤,夏兵,王江淮,等. 基于客服呼叫平台和WebRTC的实时视频接入与排队技术[J]. 现代计算机,2023,29(7):107-111.
[22]Peyrott, S. E. The JWT Handbook[M]. Auth0 Inc,2018:18-29.
[23]范展源,罗福强. JWT认证技术及其在WEB中的应用[J]. 数字技术与应用,2016(2):114-114.
[24]王建,罗政,张希,等. Web项目前后端分离的设计与实现[J]. 软件工程,2020,23(4):22-24.
[25]黎君霞. 基于遗传算法的在线考试管理系统的设计与实现[D]. 湖北:华中科技大学,2022.54-57

附录 外文参考文献及翻译



JWT HANDBOOK

By Sebastián Peyrott

Auth0

The JWT Handbook

Sebastián E. Peyrott, Auth0 Inc.

Version 0.14.1, 2016-2018

Contents

2.2.1 Access and Refresh Tokens

Access and refresh tokens are two types of tokens you will see a lot when analyzing different federated identity solutions. We will briefly explain what they are and how they help in the context of authentication and authorization.

Both concepts are usually implemented in the context of the OAuth2 specification[10]. The OAuth2 spec defines a series of steps necessary to provide access to resources by separating access from ownership (in other words, it allows several parties with different access levels to access the same resource). Several parts of these steps are *implementation defined*. That is, competing OAuth2 implementations may not be interoperable. For instance, the actual binary format of the tokens is *not specified*. Their purpose and functionality is.

Access tokens are tokens that give those who have them access to protected resources. These tokens are usually short-lived and may have an expiration date embedded in them. They may also carry or be associated with additional information (for instance, an access token may carry the IP address from which requests are allowed). This additional data is implementation defined.

Refresh tokens, on the other hand, allow clients to request new access tokens. For instance, after an access token has expired, a client may perform a request for a new access token to the authorization server. For this request to be satisfied, a refresh token is required. In contrast to access tokens, refresh tokens are usually long-lived.

[10] https://tools.ietf.org/html/rfc6749#section-1.4

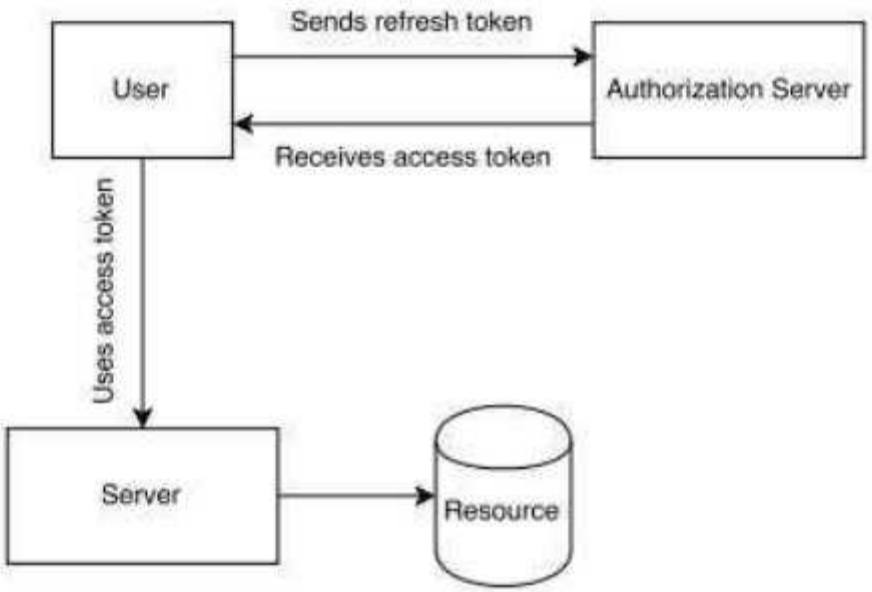


Figure 2.7: Refresh and access tokens

The key aspect of the separation between access and refresh tokens lies in the possibility of making access tokens easy to validate. An access token that carries a signature (such as a signed JWT) may be validated by the resource server on its own. There is no need to contact the authorization server for this purpose.

Refresh tokens, on the other hand, require access to the authorization server. By keeping validation separate from queries to the authorization server, better latency and less complex access patterns are possible. Appropriate security in case of token leaks is achieved by making access tokens as short-lived as possible and embedding additional checks (such as client checks) into them.

Refresh tokens, by virtue of being long-lived, must be protected from leaks. In the event of a leak, blacklisting may be necessary in the server (short-lived access tokens force refresh tokens to be used eventually, thus protecting the resource after it gets blacklisted and all access tokens are expired).

> Note: the concepts of access token and refresh token were introduced in OAuth2. OAuth 1.0 and 1.0a use the word *token* differently.

Chapter 3

JSON Web Tokens in Detail

As described in chapter 1, all JWTs are constructed from three different elements: the header, the payload, and the signature/encryption data. The first two elements are JSON objects of a certain structure. The third is dependent on the algorithm used for signing or encryption, and, in the case of *unencrypted* JWTs it is omitted. JWTs can be encoded in a *compact representation* known as *JWS/JWE Compact Serialization*.

> The JWS and JWE specifications define a third serialization format known as *JSON Serialization*, a non-compact representation that allows for multiple signatures or recipients in the same JWT. Is is explained in detail in chapters 4 and 5.

The compact serialization is a Base64[1] URL-safe encoding of the UTF-8[2] bytes of the first two JSON elements (the header and the payload) and the data, as required, for signing or encryption (which is not a JSON object itself). This data is Base64-URL encoded as well. These three elements are separated by dots (".").

> JWT uses a variant of Base64 encoding that is safe for URLs. This encoding basically substitutes the "+" and "/" characters for the "-" and "_" characters, respectively. Padding is removed as well. This variant is known as base64url[3]. Note that all references to Base64 encoding in this document refer to this variant.

The resulting sequence is a printable string like the following (newlines inserted for readability):

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWV9.
TJVA95OrM7E2cBab30RMHrHDcEfxjoYZgeFONFh7HgQ
```

Notice the dots separating the three elements of the JWT (in order: the header, the payload, and the signature).

In this example the decoded header is:

[1] https://en.wikipedia.org/wiki/Base64
[2] https://en.wikipedia.org/wiki/UTF-8
[3] https://tools.ietf.org/html/rfc4648#section-5

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

The decoded payload is:

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "admin": true
}
```

And the secret required for verifying the signature is `secret`.

> JWT.io[4] is an interactive playground for learning more about JWTs. Copy the token from above and see what happens when you edit it.

3.1 The Header

Every JWT carries a header (also known as the *JOSE header*) with claims about itself. These claims establish the algorithms used, whether the JWT is signed or encrypted, and in general, *how* to parse the rest of the JWT.

According to the type of JWT in question, more fields may be mandatory in the header. For instance, encrypted JWTs carry information about the cryptographic algorithms used for key encryption and content encryption. These fields are not present for unencrypted JWTs.

The only mandatory claim for an *unencrypted* JWT header is the `alg` claim:

- `alg`: the main algorithm in use for signing and/or decrypting this JWT.

For unencrypted JWTs this claim must be set to the value `none`.

Optional header claims include the `typ` and `cty` claims:

- `typ`: the media type[5] of the JWT itself. This parameter is only meant to be used as a help for uses where JWTs may be mixed with other objects carrying a JOSE header. In practice, this rarely happens. When present, this claim should be set to the value `JWT`.
- `cty`: the content type. Most JWTs carry specific claims plus arbitrary data as part of their payload. For this case, the content type claim *must not* be set. For instances where the payload is a JWT itself (a nested JWT), this claim *must* be present and carry the value `JWT`. This tells the implementation that further processing of the nested JWT is required. Nested JWTs are rare, so the `cty` claim is rarely present in headers.

So, for unencrypted JWTs, the header is simply:

[4] https://jwt.io
[5] http://www.iana.org/assignments/media-types/media-types.xhtml

```
{
    "alg": "none"
}
```

which gets encoded to:

```
eyJhbGciOiJub25lIn0
```

It is possible to add additional, user-defined claims to the header. This is generally of limited use, unless certain user-specific metadata is required in the case of encrypted JWTs before decryption.

3.2 The Payload

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "admin": true
}
```

The payload is the element where all the interesting user data is usually added. In addition, certain claims defined in the spec may also be present. Just like the header, the payload is a JSON object. No claims are mandatory, although specific claims have a definite meaning. The JWT spec specifies that claims that are not understood by an implementation should be ignored. The claims with specific meanings attached to them are known as *registered claims*.

3.2.1 Registered Claims

- iss: from the word *issuer*. A case-sensitive string or URI that uniquely identifies the party that issued the JWT. Its interpretation is application specific (there is no central authority managing issuers).
- sub: from the word *subject*. A case-sensitive string or URI that uniquely identifies the party that this JWT carries information about. In other words, the claims contained in this JWT are statements about this party. The JWT spec specifies that this claim must be unique in the context of the issuer or, in cases where that is not possible, globally unique. Handling of this claim is application specific.
- aud: from the word *audience*. Either a single case-sensitive string or URI or an array of such values that uniquely identify the intended recipients of this JWT. In other words, when this claim is present, the party reading the data in this JWT must find itself in the *aud* claim or disregard the data contained in the JWT. As in the case of the *iss* and *sub* claims, this claim is application specific.
- exp: from the word *expiration* (time). A number representing a specific date and time in the format "seconds since epoch" as defined by POSIX[6]. This claims sets the exact moment from

[6] http://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1_chap04.html#tag_04_15

which this JWT is considered *invalid*. Some implementations may allow for a certain skew between clocks (by considering this JWT to be valid for a few minutes after the expiration date).

- nbf: from *not before* (time). The opposite of the *exp* claim. A number representing a specific date and time in the format "seconds since epoch" as defined by POSIX[7]. This claim sets the exact moment from which this JWT is considered *valid*. The current time and date must be equal to or later than this date and time. Some implementations may allow for a certain skew.
- iat: from *issued at* (time). A number representing a specific date and time (in the same format as *exp* and *nbf*) at which this JWT was issued.
- jti: from *JWT ID*. A string representing a unique identifier for this JWT. This claim may be used to differentiate JWTs with other similar content (preventing replays, for instance). It is up to the implementation to guarantee uniqueness.

As you may have noticed, all names are short. This complies with one of the design requirements: to keep JWTs as small as possible.

String or URI: according to the JWT spec, a URI is interpreted as any string containing a : character. It is up to the implementation to provide valid values.

3.2.2 Public and Private Claims

All claims that are not part of the *registered claims* section are either private or public claims.

- Private claims: are those that are defined by *users* (consumers and producers) of the JWTs. In other words, these are ad hoc claims used for a particular case. As such, care must be taken to prevent collisions.
- Public claims: are claims that are either *registered* with the IANA JSON Web Token Claims registry[8] (a registry where users can register their claims and thus prevent collisions), or named using a collision resistant name (for instance, by prepending a namespace to its name).

In practice, most claims are either registered claims or private claims. In general, most JWTs are issued with a specific purpose and a clear set of potential users in mind. This makes the matter of picking collision resistant names simple.

Just as in the JSON parsing rules, duplicate claims (duplicate JSON keys) are handled by keeping only the last occurrence as the valid one. The JWT spec also makes it possible for implementations to consider JWTs with duplicate claims as *invalid*. In practice, if you are not sure about the implementation that will handle your JWTs, take care to avoid duplicate claims.

[7] http://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1_chap04.html#tag_04_15
[8] https://tools.ietf.org/html/rfc7519#section-10.1

3.3 Unsecured JWTs

With what we have learned so far, it is possible to construct unsecured JWTs. These are the simplest JWTs, formed by a simple (usually static) header:

```
{
    "alg": "none"
}
```

and a user defined payload. For instance:

```
{
  "sub": "user123",
  "session": "ch72gsb320000udocl363eofy",
  "name": "Pretty Name",
  "lastpage": "/views/settings"
}
```

As there is no signature or encryption, this JWT is encoded as simply two elements (newlines inserted for readability):

```
eyJhbGciOiJub25lIn0.
eyJzdWIiOiJ1c2VyMTIzIiwic2Vzc2lvbiI6ImNoNzJnc2IzMjAwMDB1ZG9jbDM2M
2VvZmkiLCJuYW1lIjoiUHJldHR5IE5hbWUiLCJsYXN0cGFnZSI6Ii92aWV3cy9zZXR0aW5ncyJ9.
```

An unsecured JWT like the one shown above may be fit for client-side use. For instance, if the session ID is a hard-to-guess number, and the rest of the data is only used by the client for constructing a view, the use of a signature is superfluous. This data can be used by a single-page web application to construct a view with the "pretty" name for the user without hitting the backend while he gets redirected to his last visited page. Even if a malicious user were to modify this data he or she would gain nothing.

> Note the trailing dot (.) in the compact representation. As there is no signature, it is simply an empty string. The dot is still added, though.

In practice, however, unsecured JWTs are rare.

3.4 Creating an Unsecured JWT

To arrive at the compact representation from the JSON versions of the header and the payload, perform the following steps:

1. Take the header as a byte array of its UTF-8 representation. The JWT spec *does not* require the JSON to be minified or stripped of meaningless characters (such as whitespace) before encoding.
2. Encode the byte array using the Base64-URL algorithm, removing trailing equal signs (=).
3. Take the payload as a byte array of its UTF-8 representation. The JWT spec *does not* require the JSON to be minified or stripped of meaningless characters (such as whitespace) before encoding.

4. Encode the byte array using the Base64-URL algorithm, removing trailing equal signs (=).
5. Concatenate the resulting strings, putting first the header, followed by a "." character, followed by the payload.

Validation of both the header and the payload (with respect to the presence of required claims and the correct use of each claim) must be performed before encoding.

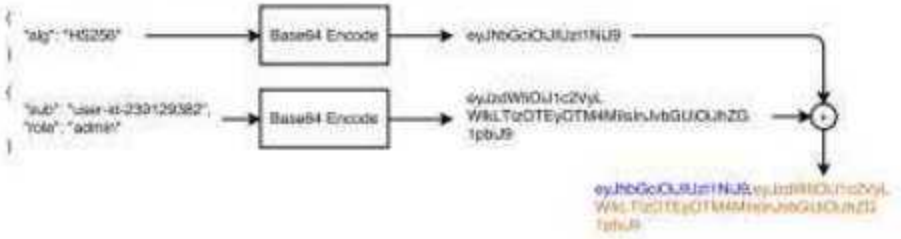


Figure 3.1: Compact Unsecured JWT Generation

3.4.1 Sample Code

```
// URL-safe variant of Base64
function b64(str) {
    return new Buffer(str).toString('base64')
                          .replace(/=/g, '')
                          .replace(/\+/g, '-')
                          .replace(/\//g, '_');
}

function encode(h, p) {
    const headerEnc = b64(JSON.stringify(h));
    const payloadEnc = b64(JSON.stringify(p));
    return `${headerEnc}.${payloadEnc}`;
}
```

The full example is in file `coding.js` of the accompanying sample code.

3.5 Parsing an Unsecured JWT

To arrive at the JSON representation from the compact serialization form, perform the following steps:

1. Find the first period "." character. Take the string before it (not including it.)

2. Decode the string using the Base64-URL algorithm. The result is the JWT header.
3. Take the string after the period from step 1.
4. Decode the string using the Base64-URL algorithm. The result is the JWT payload.

The resulting JSON strings may be "prettified" by adding whitespace as necessary.

3.5.1 Sample Code

```
function decode(jwt) {
    const [headerB64, payloadB64] = jwt.split('.');
    // These supports parsing the URL safe variant of Base64 as well.
    const headerStr = new Buffer(headerB64, 'base64').toString();
    const payloadStr = new Buffer(payloadB64, 'base64').toString();
    return {
        header: JSON.parse(headerStr),
        payload: JSON.parse(payloadStr)
    };
}
```

The full example is in file `coding.js` of the accompanying sample code.

4. Encode the byte array using the Base64-URL algorithm, removing trailing equal signs (=).
5. Concatenate the resulting strings, putting first the header, followed by a "." character, followed by the payload.

Validation of both the header and the payload (with respect to the presence of required claims and the correct use of each claim) must be performed before encoding.

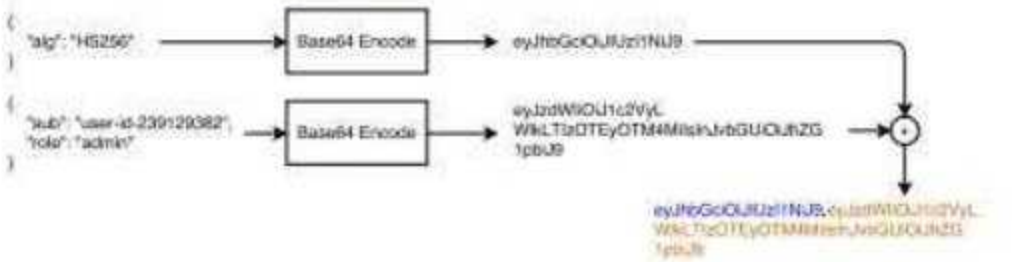


Figure 3.1: Compact Unsecured JWT Generation

3.4.1 Sample Code

```
// URL-safe variant of Base64
function b64(str) {
    return new Buffer(str).toString('base64')
                          .replace(/=/g, '')
                          .replace(/\+/g, '-')
                          .replace(/\//g, '_');
}

function encode(h, p) {
    const headerEnc = b64(JSON.stringify(h));
    const payloadEnc = b64(JSON.stringify(p));
    return `${headerEnc}.${payloadEnc}`;
}
```

The full example is in file `coding.js` of the accompanying sample code.

3.5 Parsing an Unsecured JWT

To arrive at the JSON representation from the compact serialization form, perform the following steps:

1. Find the first period "." character. Take the string before it (not including it.)

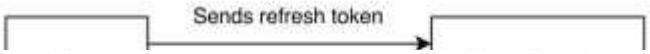
2.2.1访问和刷新令牌

访问令牌和刷新令牌是在分析不同联合身份解决方案时经常会遇到的两种令牌类型。我们将简要解释它们是什么以及它们如何在身份验证和授权上下文中提供帮助。

这两个概念通常在OAuth2规范的背景下实施。OAuth2规范定义了一系列必要的步骤，通过将访问与所有权分离来提供对资源的访问(换句话说，它允许具有不同访问级别的多个方访问相同的资源)。这些步骤的几个部分是由实现定义的。也就是说，相互竞争的OAuth2实现可能无法互操作。例如，令牌的实际二进制格式未指定，但其目的和功能已被定义为：

访问令牌是允许拥有它们的人访问受保护资源的令牌。这些令牌通常具有短暂的生命周期，并且可能会嵌入一个截止日期。它们还可能携带或与其他信息相关联(例如，访问令牌可能携带允许请求的IP地址)。这些额外的数据是由具体实现定义的。

另一方面，刷新令牌允许客户端请求新的访问令牌。例如，在访问令牌过期后，客户端可以向授权服务器请求新的访问令牌。为了满足这个请求，需要使用刷新令牌。与访问令牌不同，刷新令牌通常具有较长的生命周期。



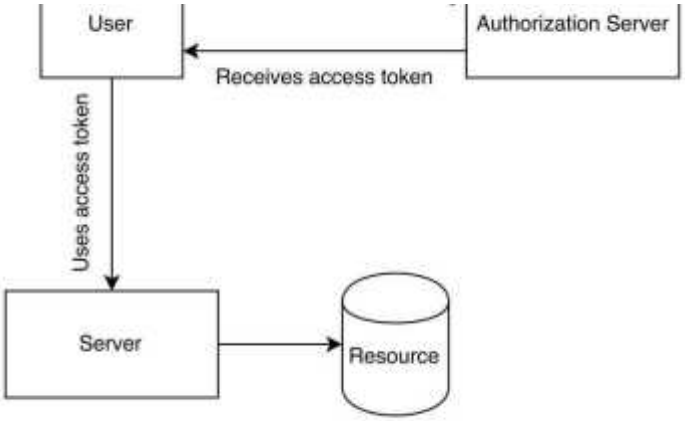


Figure 2.7: Refresh and access tokens

分离访问令牌和刷新令牌的关键在于使访问令牌易于验证。带有签名(例如签名的JWT)的访问令牌可以由资源服务器自己进行验证。无需为此目的联系授权服务器。

另一方面，刷新令牌需要访问授权服务器。通过将验证与对授权服务器的查询分开，可以实现更好的延迟和更简单的访问模式。在令牌泄漏的情况下，通过使访问令牌尽可能短暂并在其中嵌入额外的检查(如客户端检查)来实现适当的安全性。

由于刷新令牌的长寿命，必须保护其不被泄漏。在发生泄漏的情况下，可能需要在服务器上进行黑名单处理（短寿命的访问令牌最终会强制使用刷新令牌，因此在对资源进行黑名单处理并且所有访问令牌都过期后，资源得到保护）。

注意：访问令牌和刷新令牌的概念是在OAuth2中引入的。OAuth 1.0和1.0a使用 "token" 这个词的含义有所不同。

第三章 JSON Web令牌详解

所有JWT都由三个不同的元素构成：头部（header）、载荷（payload）和签名/加密数据。前两个元素是特定结构的JSON对象。第三个元素取决于用于签名或加密的算法，在未加密的JWT中可能会被省略。JWT可以以一种称为JWS/JWE紧凑序列化的方式进行编码。

JWS（JSON Web Signature）和JWE（JSON Web Encryption）规范定义了第三种序列化格式，称为JSON序列化，这是一种非紧凑表示法，允许在同一个JWT中存在多个签名或接收者。这在规范的第4章和第5章中有详细说明。

紧凑的序列化是对第一个两个JSON元素（头部和载荷）以及用于签名或加密的数据（该数据本身不是JSON对象）的UTF-8字节的Base64 URL安全编码，如有需要。这些数据也进行了Base64-URL编码。这三个元素之间用点号（“.”）分隔。

JWT使用一种适用于URL的Base64编码的变体。该编码基本上用 "-" 和 "_" 字符替换了 "+" 和 "/" 字符，而且移除了填充。这个变体被称为 base64url。请注意，在本文档中所有关于Base64编码的引用都指的是这个变体。

得到的序列是一个可打印的字符串，如下所示：

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.

eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWV9.

TJVA95OrM7E2cBab30RMHrHDcEfxjoYZgeFONFh7HgQ

请注意，点号将JWT的三个元素分隔开来（顺序为：头部、载荷和签名）。

在这个例子中，解码后的头部是：

```
{
"alg": "HS256",
"typ": "JWT"
}
```

解码后的有效载荷是：

```
{
"sub": "1234567890",
"name": "John Doe",
"admin": true
```

}

而用于验证签名的密钥是 "secret"。JWT.io 是一个交互式的平台，可用于深入了解JWT。复制上面的令牌，然后尝试进行编辑，看看会发生什么。

3.1 头部

每个JWT都携带一个头部（也称为JOSE头部），其中包含有关自身的声明。这些声明确定了所使用的算法、JWT是否已签名或加密，以及通常如何解析JWT的其余部分。

根据所涉及的JWT类型，头部中可能会有更多的字段是强制性的。例如，加密的JWT携带有关用于密钥加密和内容加密的密码算法的信息。这些字段在未加密的JWT中是不存在的。

对于未加密的JWT头，唯一的强制声明是alg声明:

•alg：用于签名和/或解密此JWT的主要算法

对于未加密的JWT，此声明必须设置为值none

可选的头部声明包括 typ 和 cty 声明：

•typ：JWT本身的媒体类型。此参数仅意为在JWT可能与携带JOSE头部的其他对象混合的情况下使用。实际上，这种情况很少发生。当存在时，此声明应设置为值 JWT。

•cty：内容类型。大多数JWT携带特定的声明以及作为其载荷一部分的任意数据。对于这种情况，不得设置内容类型声明。对于载荷本身是JWT（嵌套JWT）的情况，此声明必须存在并携带值 JWT。这告诉实现需要进一步处理嵌套JWT。嵌套JWT很少见，因此 cty 声明在头部中很少出现。

因此，对于未加密的JWT，头部简单地为：

{
"alg": "none"
}

它被编码为:

eyJhbGciOiJub25lIn0

可以向头部添加额外的用户定义的声明。通常情况下，除非在解密之前需要特定于用户的元数据，否则这样做的用途有限。

3.2载荷

{
"sub": "1234567890",
"name": "John Doe",
"admin": true
}

载荷是通常添加所有有趣用户数据的元素。此外，规范中定义的某些声明也可能存在。与头部一样，载荷是一个JSON对象。没有强制性的声明，尽管特定的声明具有明确的含义。JWT规范规定，实现不理解的声明应该被忽略。具有特定含义的声明称为注册声明。

3.2.1 注册声明

•iss：来自单词 "issuer"。一个区分大小写的字符串或URI，用于唯一标识发出JWT的一方。其解释是特定于应用程序的（没有中央管理发行方的权威机构）。

•sub：来自单词 "subject"。一个区分大小写的字符串或URI，用于唯一标识此JWT携带信息的一方。换句话说，此JWT中包含的声明是关于此一方的陈述。JWT规范规定，该声明在发行方的上下文中必须是唯一的，或者在不可能的情况下，全局唯一。此声明的处理方式是特定于应用程序的。

•aud：来自单词 "audience"。可以是单个区分大小写的字符串或URI，也可以是一组这样的值，用于唯一标识此JWT的预期接收方。换句话说，当存在此声明时，读取此JWT中数据的一方必须在 aud 声明中找到自己，或者忽略JWT中包含的数据。与 iss 和 sub 声明一样，此声明是特定于应用程序的。

•exp：来自单词 "expiration"（时间）。一个表示特定日期和时间的数字，格式为 POSIX 规定的“自纪元以来的秒数”。此声明设置了JWT被视为无效的确切时刻。一些实现可能允许在时钟之间存在一定偏差（即在到期日期后的几分钟内将此JWT视为有效）。

•nbf：来自 "not before"（时间）的缩写。与 exp 声明相反。一个表示特定日期和时间的数字，格式为

POSIX 规定的“自纪元以来的秒数”。此声明设置了JWT被视为有效的确切时刻。当前时间和日期必须等于或晚于此日期和时间。一些实现可能允许存在一定的偏差。
•iat：来自 "issued at"（时间）。一个表示特定日期和时间的数字（与 exp 和 nbf 具有相同的格式），表示此JWT的发行时间。
•jti：来自 "JWT ID"。一个字符串，表示此JWT的唯一标识符。此声明可用于区分具有类似内容的JWT（例如，防止重播攻击）。确保唯一性是由实现负责的。
正如您可能已经注意到的那样，所有的名称都很短。这符合设计要求之一：使JWT尽可能小。
字符串或URI：根据JWT规范，URI被解释为包含冒号（:）字符的任何字符串。由实现提供有效的值。
3.2.2 公共和私有声明
所有不属于注册声明部分的声明都是私有或公共声明。
•私有声明：由JWT的用户（消费者和生产者）定义的声明。换句话说，这些是为特定情况使用的特设声明。因此，必须注意防止冲突。
•公共声明：是要么在IANA JSON Web Token Claims注册表（用户可以在其中注册其声明以防止冲突的注册表）中注册的声明，要么使用具有防冲突名称的命名（例如，通过在其名称前添加命名空间）。
实际上，大多数声明要么是注册声明，要么是私有声明。一般而言，大多数JWT都是为特定目的发行的，并考虑了明确的一组潜在用户。这使得选择具有防冲突名称的任务变得简单。
3.3 不安全的JWT
根据我们目前学到的知识，可以构建不安全的JWT。这些是最简单的JWT，由一个简单的（通常是静态的）头部组成：
{
"alg": "none"
}
以及用户定义的有效载荷。例如：
{
"sub": "user123",
"session": "ch72gsb320000udoc1363eofy",
"name": "Pretty Name",
"lastpage": "/views/settings"
}
由于没有签名或加密，这个JWT被编码为简单的两个元素（为了可读性插入了换行符）：
eyJhbGciOiJub25lIn0.
eyJzdWIiOiJ1c2VyMTIzIiwic2Vzc2lvbiI6ImNoNzJnc2IzMjAwMDB1ZG9jbDM2M
2VvZnkiLCJuYW1lIjoiUHJldHR5IE5hbWUiLCJsYXN0cGFnZSI6Ii92aWV3cy9zZXR0aW5ncyJ9.
像上面展示的不安全的JWT可能适用于客户端使用。例如，如果会话ID是一个难以猜测的数字，并且其余数据仅由客户端用于构建视图，那么使用签名就是多余的。这些数据可以被单页Web应用程序用于构建带有用户的“漂亮”名称的视图，而无需在用户被重定向到其最后访问的页面时与后端交互。即使恶意用户尝试修改这些数据，也不会获得任何好处。
请注意紧凑表示中的尾随点号（.）。由于没有签名，它只是一个空字符串。尽管如此，点号仍然被添加。
然而，在实际应用中，不安全的JWT是比较罕见的。
3.4 创建不安全的JWT
为了从头部和载荷的JSON版本得到紧凑表示，执行以下步骤：
1.将头部作为其UTF-8表示的字节数组。JWT规范不要求在编码之前将JSON最小化或剥离无意义的字符（如空格）。
2.使用Base64-URL算法对字节数组进行编码，去除尾随的等号（=）。
3.将载荷作为其UTF-8表示的字节数组。JWT规范不要求在编码之前将JSON最小化或剥离无意义的字符（如空格）。
4.使用Base64-URL算法对字节数组进行编码，去除尾随的等号（=）。
5.连接生成的字符串，将头部放在首位，后跟一个“.”字符，然后是载荷。
在进行编码之前，必须对头部和载荷进行验证（验证是否存在所需的声明以及每个声明的正确使用）。

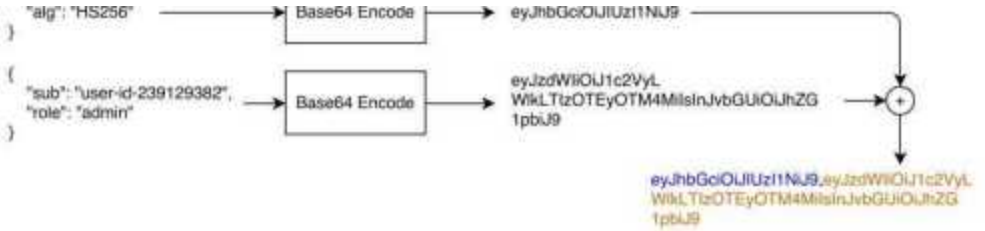


Figure 3.1: Compact Unsecured JWT Generation

3.4.1样例代码

```
// URL安全的Base64变体
function b64(str) {
return new Buffer(str).toString('base64')
.replace(/=/g, '')
.replace(/\+/g, '-')
.replace(/\//g, '_');
}
function encode(h, p) {
const headerEnc = b64(JSON.stringify(h));
const payloadEnc = b64(JSON.stringify(p));
return `${headerEnc}.${payloadEnc}`;
}
```

完整的示例在附带的样例代码文件coding.js中。

3.5 解析不安全的JWT

为了从紧凑序列化形式得到JSON表示，执行以下步骤：

1.找到第一个点号“.”字符。取该字符之前的字符串（不包括该字符）。

2.使用Base64-URL算法对字符串进行解码。结果是JWT的头部。

3.取步骤1中点号之后的字符串。

4.使用Base64-URL算法对字符串进行解码。结果是JWT的载荷。

生成的JSON字符串可以通过根据需要添加空格来“美化”。

3.5.1样例代码

```
function decode(jwt) {
const [headerB64, payloadB64] = jwt.split('.');
// 这些支持解析Base64的URL安全变体
const headerStr = new Buffer(headerB64, 'base64').toString();
const payloadStr = new Buffer(payloadB64, 'base64').toString();
return {
header: JSON.parse(headerStr),
payload: JSON.parse(payloadStr)
};
}
```

完整的示例在附带的样例代码文件coding.js中。

相似片段说明

相似片段中"综合"包括：《中文主要报纸全文数据库》《中国专利特色数据库》《中国主要会议论文特色数据库》《港澳台文献资源》《图书资源》《维普优先出版论文全文数据库》《年鉴资源》《古籍文献资源》《IPUB原创作品》