**上机考试管理系统**

**项目技术文档**

组 长： 康强强

组 员： 王旭辉

## 1.项目介绍

本项目根据前期的问卷调查以及实地考察，作出了适合实际的需求分析，采用了前后端分离的思想，后端使用了SpringBoot和SpringMVC框架，前端使用了Vue.js和ElementUI进行页面的搭建，并且使用了MySQL数据库进行数据交互。整个系统的使用者有3种角色，分别是管理员、教师和学生。

以下是3种不同的角色所分别拥有的功能：

1. 管理员：可进行登录、修改密码、考试管理（增删改查、查看或移除试卷题目功能）、题目管理（增改查、手动新增题目时自动加入题库、设置题目数量自动组卷功能）、查询学生成绩（折线图展现）、查询成绩分段（饼状图展现）、学生管理、教师管理。。
2. 教师：可以登录、修改密码、考试管理（增删改查、查看或移除试卷题目功能）、题目管理（增改查、手动新增题目时自动加入题库、设置题目数量自动组卷功能）、查询学生成绩（折线图展现）、学生管理。
3. 学生：可以登录、修改密码、考试中心（查询考试列表）、考试答题、提交试卷、查看成绩、交流区（发送留言和评论留言）。

# 2.1 相关技术

本课题在Windows系统下使用了JavaEE开发技术、Spring Boot技术、Vue.js技术、Tomcat服务器、MySQL关系型数据库以及ElementUI前端框架技术进行开发。以下是关键技术及开发工具介绍。

## 2.1.1 Java语言

Java是一种面向对象的编程语言，由Sun Microsystems（现在是Oracle公司）于1995年首次发布。它具有可移植性、安全性和跨平台等特性，因此被广泛应用于各个领域的软件开发中。

Java的平台无关性是其最重要的特性之一。通过Java虚拟机（JVM），Java程序可以在不同的操作系统和硬件上运行，而不需要针对特定平台进行修改。这意味着一次编写的Java代码可以在Windows、Linux、Mac等多个平台上运行。

Java是一种面向对象的语言，它支持封装、继承和多态等面向对象的编程概念。这使得Java程序更易于理解、扩展和维护。它还提供了强大的异常处理机制，使开发者能够更好地应对错误和异常情况。

Java还具有丰富的标准库和第三方库，以及一个庞大的开发者社区。这些库提供了许多常用的功能和工具，使得Java的开发变得更加高效和便捷。

总之，Java是一种功能强大、可靠性高、安全性好的编程语言，非常适合开发各种类型的应用程序，包括桌面应用程序、Web应用程序、移动应用程序等。它的广泛应用和稳定性使得Java成为了当今世界上最流行的编程语言之一。

## 2.1.2 HTML、CSS、JavaScript

HTML（超文本标记语言）是一种用于创建和组织网页内容的标记语言。它由一系列标签组成，标签用于标记和描述文档中的不同部分，如标题、段落、链接和图像等。HTML提供了结构化的标记方式，使得浏览器能够正确地解析并显示网页内容。

CSS（层叠样式表）是一种用于描述网页外观和布局的样式语言。通过CSS，可以对HTML文档中的元素进行样式设置，包括字体、颜色、大小、间距、边框和背景等。CSS的设计目标是将样式与内容分离，使得网页的样式可以独立于内容进行修改和调整，从而提高网页的可维护性和灵活性。

JavaScript是一种动态的、基于对象和事件驱动的脚本语言。它广泛用于前端开发，用于为网页增加交互性和动态性。通过JavaScript，可以操作网页的元素、处理用户的输入、响应事件、进行表单验证和数据处理等。它还可以通过AJAX技术与服务器进行数据交互，实现动态加载内容和实时更新。

HTML、CSS和JavaScript三者通常一起使用，被称为前端开发技术。HTML负责网页的内容和结构，CSS负责网页的样式和布局，JavaScript负责网页的交互和动态效果。它们共同作用于网页的不同方面，使网页成为一个功能丰富、外观吸引人且易于使用的用户界面。

总结起来，HTML提供了网页内容的结构和标记，CSS负责网页的样式和布局，JavaScript为网页增加了交互性和动态功能。三者的协同工作使得构建现代化、交互式的网页成为可能。

## 2.1.3 MySQL

MySQL是一种关系型数据库管理系统，是一个多用户，多线程的SQL数据库。MySQL的优点是体积小、速度快、总体拥有成本低，开放源代码等，使得它成为了目前中小企业的最爱。

## 2.1.4 SSM框架

SSM框架是由Spring，SpringMVC和MyBatis三个开源框架整合而成，是当前较为流行稳定的Web项目框架。

Spring是一个开源框架，它是为了解决企业应用开发的复杂性而创建的。Spring的核心是控制反转和面向切面编程。

SpringMVC是Spring框架的组成部分之一，MVC框架提供了模型-视图-控制的体系结构和可以用来开发灵活、松散耦合的Web应用程序的组件。

MyBatis是一个基于Java的持久层框架,它支持定制化SQL、存储过程以及高级映射[[1]](#footnote-0)[3]。该框架解除了SQL与程序代码的耦合，提高了代码的可维护性。同时，MyBatis较为灵活，简单易学以及支持动态SQL语句等优点，让它深受广大编程人员的喜爱。

## 2.1.5 Vue.js

Vue.js是一套用于构建用户界面的渐进式框架[[2]](#footnote-1)[4]。与其它大型框架不同的是，Vue被设计为可以自底向上逐层应用。Vue的核心库只关注视图层，不仅易于上手，还便于与第三方库或既有项目整合。

## 2.1.6 SpringBoot

SpringBoot是由Pivotal团队提供的全新框架，其设计目的是用来简化新Spring应用的初始搭建以及开发过程。使用SpringBoot的最大好处就是简化配置，它实现了自动化配置。

# 2.2 开发环境

以下展示了系统所需的开发环境。

系统所需开发环境表

|  |  |
| --- | --- |
| 操作系统 | Windows 10 |
| 开发工具 | IntelliJ IDEA 2021.03 |
| Java环境 | JDK1.8，Maven 3.8.1 |
| 数据库 | MySQL 8.0.19、Redis 3.0.10 |
| 服务器 | Tomcat 9.0.30 |

## 3.1.1 技术可行性

本系统的主要开发语言为Java，基于目前较为流行的SpringBoot以及Vue.js框架进行编写。本系统采用了MySQL数据库作为数据存储基础阵地，保证了一个安全、稳定的数据使用环境。本系统使用的技术，基本上都是目前较为流行的并且是开源的技术，文档和资料在互联网上都非常健全，且有许多开发成功的项目。因此，本系统开发在技术方面是可行的。

## 3.1.3 运行可行性

本系统基于SpringBoot框架，需要搭建Java，Maven， MySQL等软件环境。在搭建的过程中，有遇到过种种的问题，但通过查询资料和请教导师，都能一一解决，并能熟练掌握。因此，本系统开发在运行方面是可行的。

# 3.2 流程分析

## 3.2.1 流程分析

本系统的使用者一共有管理员、教师和学生这3种角色:

1. 管理员：通过登录系统后，可以进行考试管理和题目管理等功能，包括创建、修改、删除考试和题目，同时可以查询学生成绩并以折线图和饼状图展示，还可以对学生和教师进行管理。
2. 教师：教师通过登录系统后，可以进行考试管理和题目管理等功能，包括创建、修改、删除考试和题目；同时可以查询学生成绩以折线图展示。
3. 学生：登录、修改密码、考试中心（查询考试列表）、考试答题、提交试卷、查看成绩、交流区（发送留言和评论留言）。

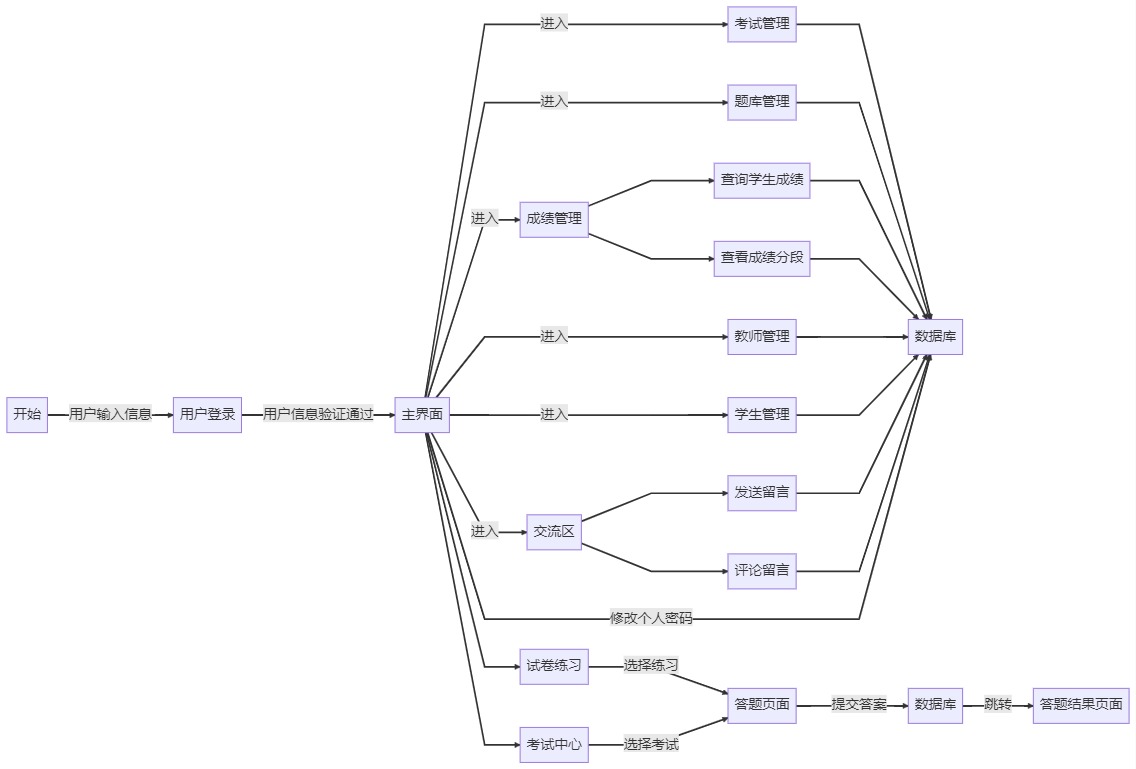
本系统的业务需求主要是提供一个在线考试平台，包括管理员、教师和学生角色，实现考试管理、题目管理、成绩统计和用户管理等功能，以提供便捷、安全、高效的在线考试解决方案。

业务需求表

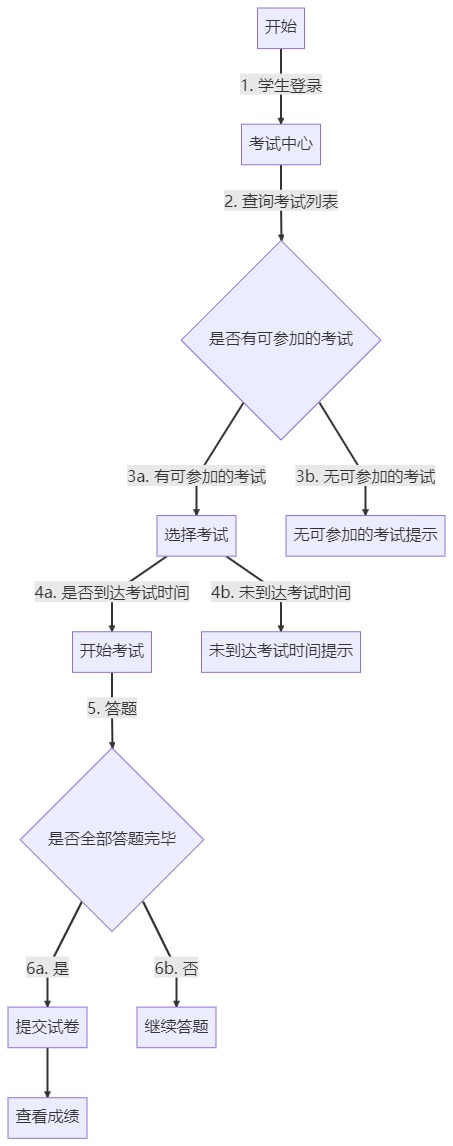
| **编号** | **业务名称** | **操作角色** |
| --- | --- | --- |
| 1 | 用户登录 | 管理员、教师、学生 |
| 2 | 修改密码 | 管理员、学生 |
| 3 | 考试管理 | 管理员、教师 |
| 4 | 题目管理 | 管理员、教师 |
| 5 | 查询学生成绩 | 管理员、教师 |
| 6 | 查询成绩分段 | 管理员、教师 |
| 7 | 学生管理 | 管理员 |
| 8 | 教师管理 | 管理员 |
| 9 | 考试中心 | 学生 |
| 10 | 考试答题 | 学生 |
| 11 | 提交试卷 | 学生 |
| 12 | 查询我的分数列表 | 学生 |
| 13 | 交流区 | 学生 |
| 14 | 发送留言 | 学生 |
| 15 | 评论留言 | 学生 |
| 16 | 查询考试列表 | 学生 |
| 17 | 查看成绩 | 学生 |

## 3.2.2流程图

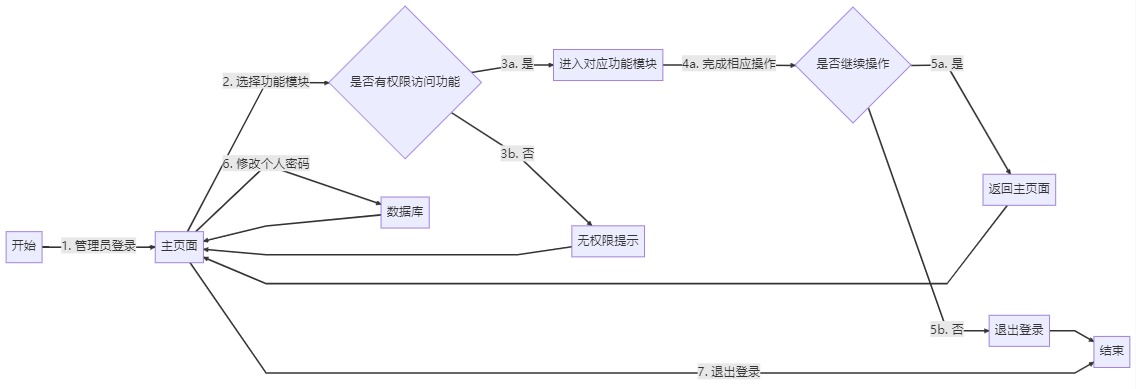
### 3.2.2.1 数据流程图



### 3.2.2.2 业务流程图



学生考试业务流程图



后台管理业务流程图

# 3.3 需求分析

## 3.3.1 需求概述

### 3.3.1.1 管理员角色需求概述

管理员作为在线考试系统的主要管理者，具有全面的权限和责任。管理员需要能够安全登录系统，并具备修改个人密码的能力。在主页面上，管理员需要能够根据自己的需求和权限选择不同的功能模块，包括考试管理、题目管理、成绩分析、教师管理和学生管理等。管理员需要能够创建、修改、删除考试和题目，查询学生成绩并以图表形式展示。同时，管理员还需要有能力管理教师和学生信息，包括添加、修改和删除等操作。管理员需要能够修改个人密码以保证账户安全。总而言之，管理员角色需要具备多种权限和操作能力，以确保在线考试系统的安全性和有效性。

### 3.3.1.2 教师角色需求概述

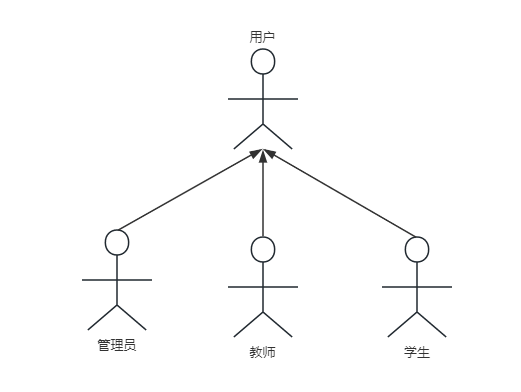
教师在在线考试系统中扮演着重要的角色，旨在为学生提供有效的教学和评估支持。教师需要能够安全登录系统，并具备修改个人密码的能力。在主页面上，教师需要能够选择不同的功能模块，包括考试管理、题目管理、成绩分析等。教师可以创建、修改和删除考试，并设置考试的时间和规则。教师还需要能够管理题目，包括添加、修改和删除等操作，并可以根据需要手动给题库中新增题目。教师可以查询学生的考试成绩情况，提供及时的评估和个别指导。此外，教师也可以修改个人密码以确保账户安全。总之，教师角色需要具备多种功能和权限，以支持学生的学习评估。

### 3.3.1.3 学生角色需求概述

学生在在线考试系统中扮演着重要的角色，旨在参与考试、获取成绩反馈和与教师、学生进行交流。学生需要能够安全登录系统，并具备修改个人密码的能力。在主页面上，学生应能够进入考试中心，查询可参加的考试列表，并根据考试时间参加考试。学生可以选择答题并提交试卷，等待试卷的批改。学生可以随时查看自己的成绩，并获得成绩分析的反馈，以便提高学习效果。同时，学生可以参与交流区的讨论，与教师和其他学生进行互动交流，提出问题并获取指导。此外，学生也可以修改个人密码，确保账户安全。总而言之，学生角色需要能够参与在线考试、获取成绩反馈和参与学习交流，以实现个人的学习和成长。

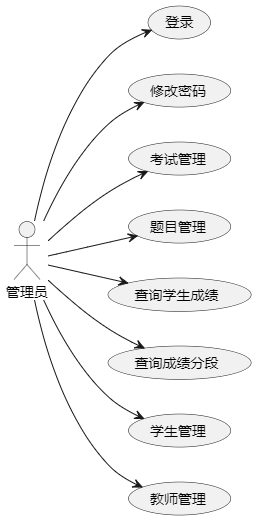
## 3.3.2 用例图

根据本系统的使用者将用户划分为三种角色，分别是管理员，教师和学生。进入系统前需要通过账号以及密码进行登录，患者若无账号可先注册，医生的账号由管理员统一注册，登录成功后则根据他们的各自的角色进入相应的页面。



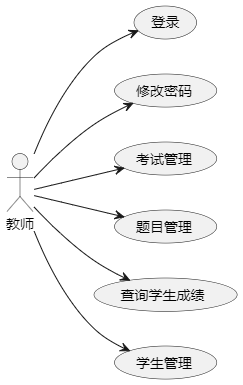
用户关系用例图

管理员进入网站的页面后，可以进行的功能用例如图所示:



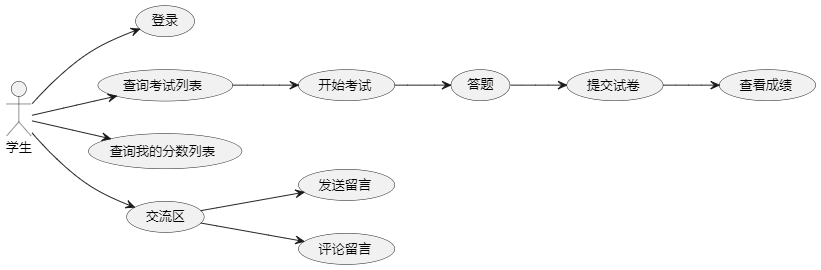
管理员用例图

教师进入网站的页面后，可以进行的功能用例如图所示:



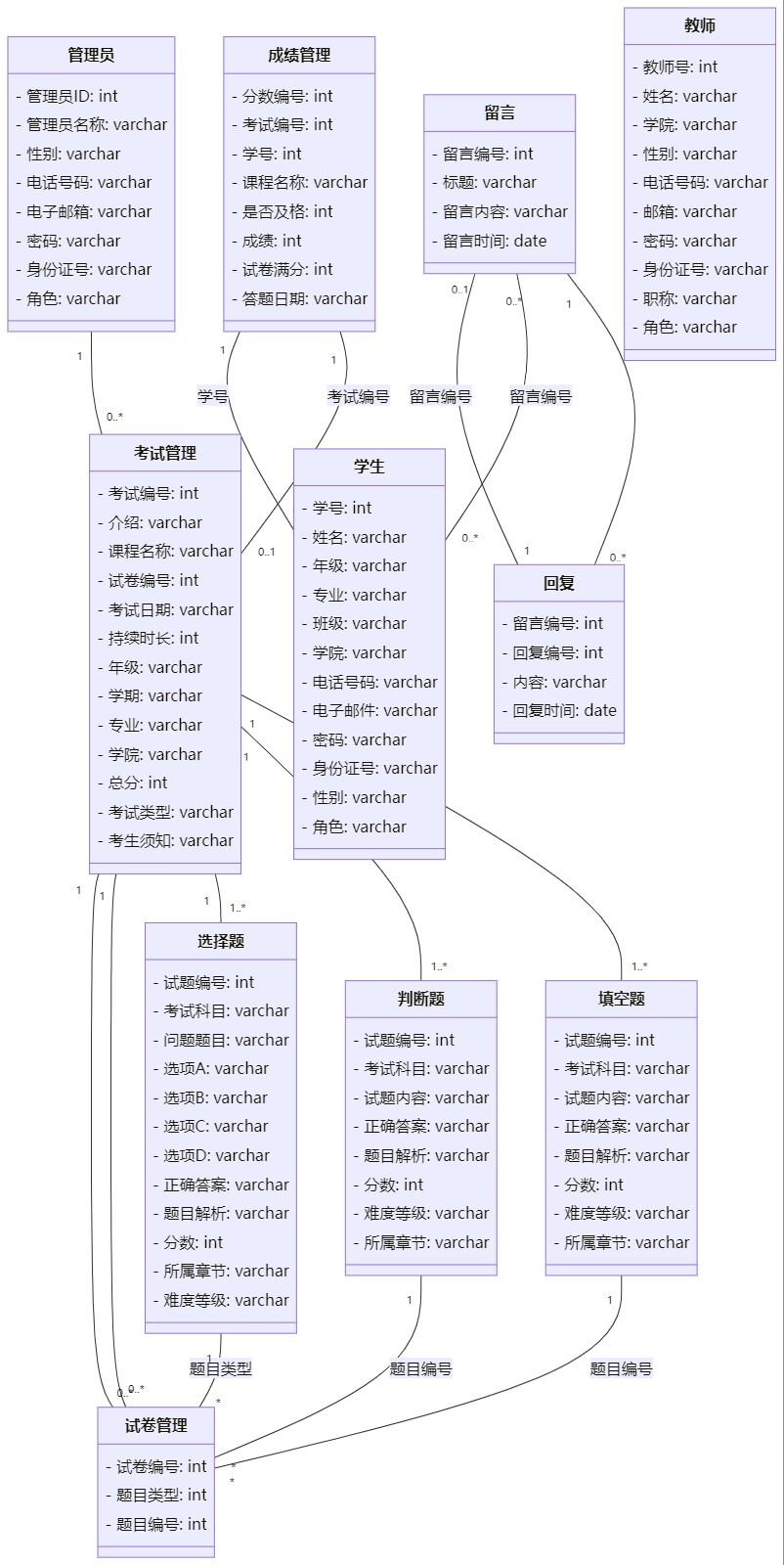
教师用例图

学生进入网站的页面后，可以进行的功能用例如图所示:



学生用例图

## 3.3.3 系统类图



## 3.3.4 数据需求分析

### 3.3.4.1 实体集分析

实体集是具有相同类型及相同属性的实体的集合。本系统主要的实体集包括管理员、考试、题目（包括填空题、判断题、选择题）、留言、回复、成绩、学生、教师。这些实体集记录了系统中的关键信息，如管理员的个人信息、考试的管理信息、各种题目的内容和相关信息、学生和教师的个人信息、交流留言和回复、以及学生的考试成绩。这些实体集之间存在关联关系，构成了在线考试系统中的核心数据模型。

### 3.3.4.2 实体集数据项分析

管理员：管理员ID号、、姓名、性别、电话号码、电子邮箱、密码、身份证号、角色；

考试管理：考试编号、考试介绍、课程名称、试卷编号、考试日期、持续时长、年级、学期、专业、学院、总分、考试类型、考生须知；

填空题：试题编号、考试科目、试题内容、正确答案、题目解析、分数、难度等级、所属章节；

判断题：试题编号、考试科目、试题内容、正确答案、题目解析、分数、难度等级、所属章节；

留言：留言编号、标题、留言内容、留言时间；

选择题：试题编号、考试科目、问题题目、选项A、选项B、选项C、选项D、正确答案、题目解析、分数、所属章节、难度等级；

试卷管理：试卷编号、题目类型、题目编号；

回复：留言编号、回复编号、内容、回复时间；

成绩：分数编号、考试编号、学号、课程名称、是否及格、成绩、试卷满分、答题日期；

学生：学号、姓名、年级、专业、班级、学院、电话号码、电子邮件、密码、身份证号、性别、角色；

教师：教师ID、姓名、学院、性别、电话号码、邮箱、密码、身份证号、职称、角色。

### 3.3.2.3 数据操作分析

管理员的实体操作分析： 管理考试、试卷和题目信息,查询学生成绩和生成成绩分析图表,管理学生和教师信息。

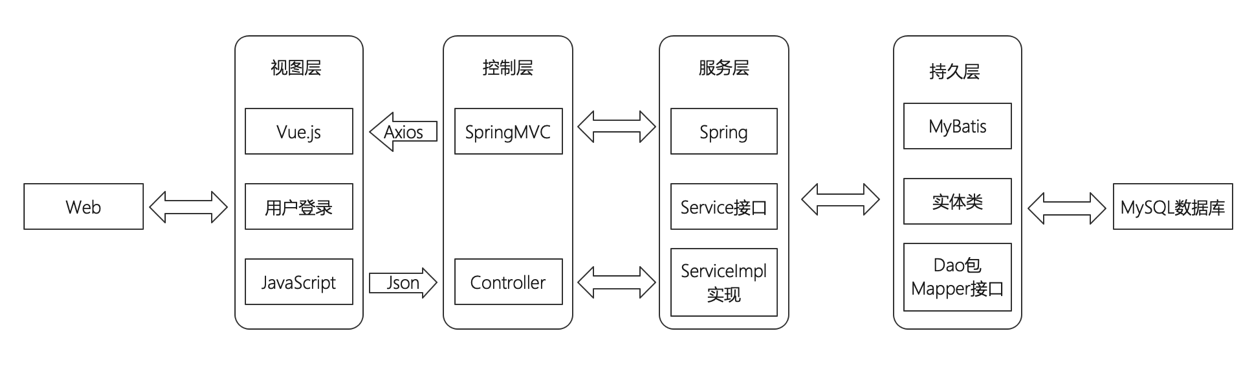
教师的实体操作分析：管理考试、试卷和题目信息,查询学生成绩和生成成绩分析图表,管理学生信息。

学生的实体操作分析：登录、修改密码,查询考试列表、参加考试、提交试卷、查询成绩,发送留言、评论留言。

# 4.1 系统设计

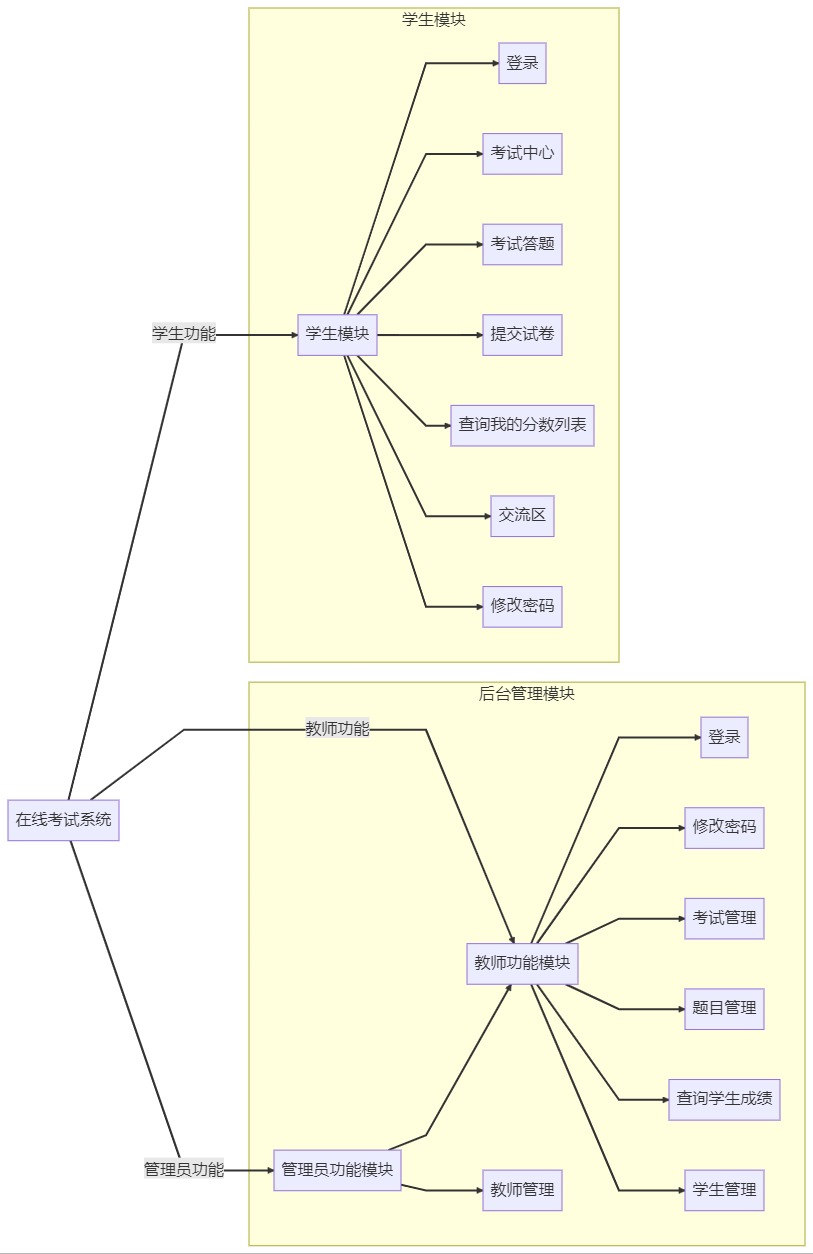
## 4.1.1 系统架构设计

本系统使用SpringBoot、Vue.js等主流开发框架进行开发，遵守MVC模式，同时使用到了Element-UI进行页面的设计。为方便前后端进行整合和日后系统的维护，本系统分为视图层、控制层、业务层和持久层。各层之间相互独立，通过之间的接口进行通信，高内聚，低耦合[[3]](#footnote-2)[6]。



系统架构图

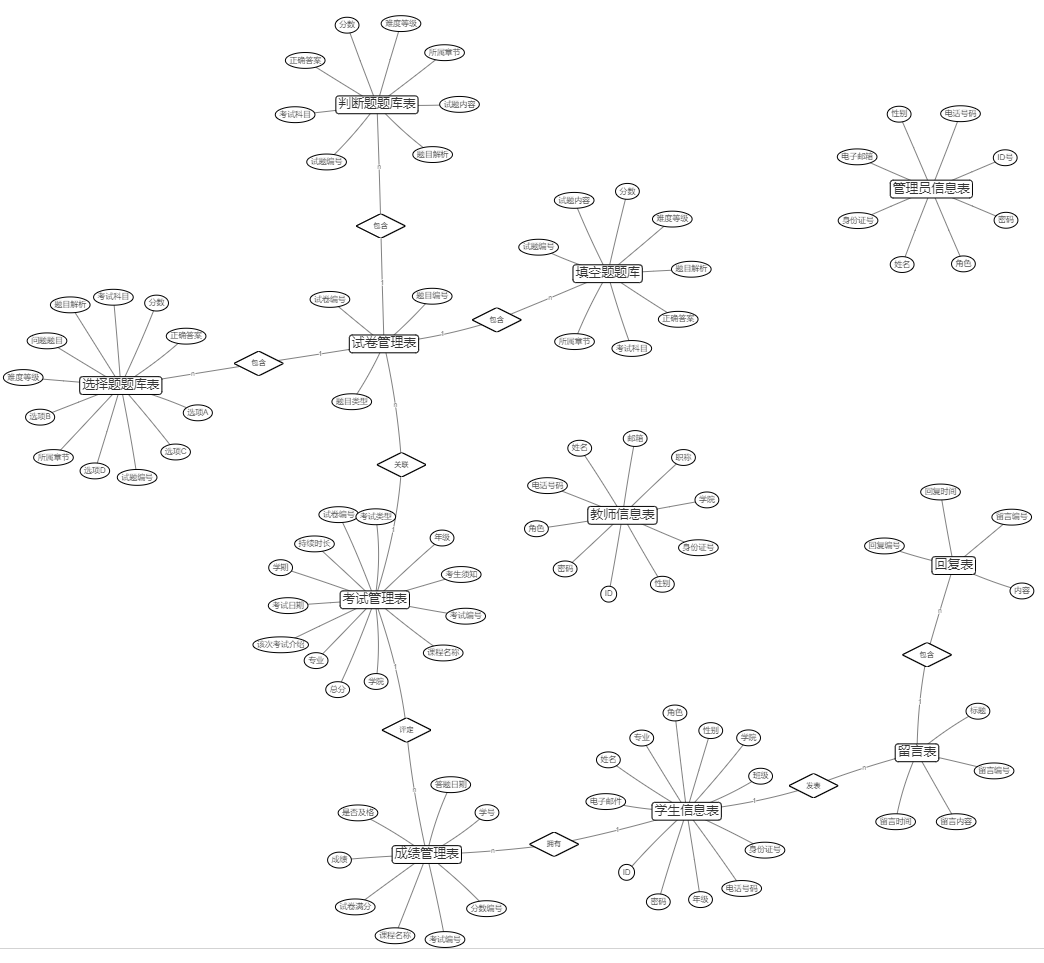
## 4.1.2 系统功能模块图



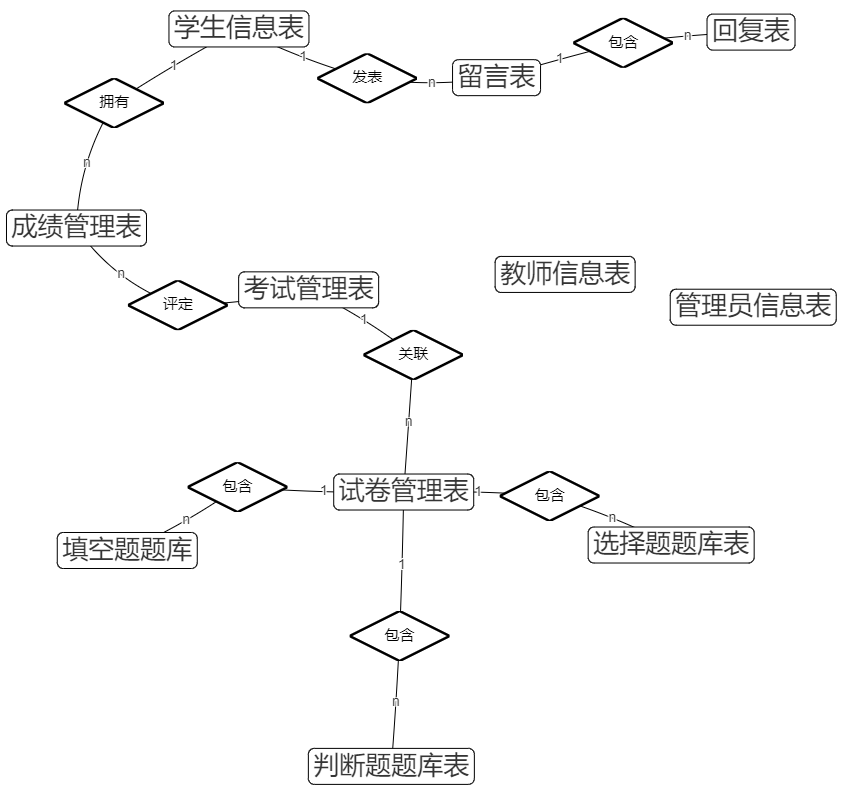
系统功能模块图

# 4.2 数据库概要设计

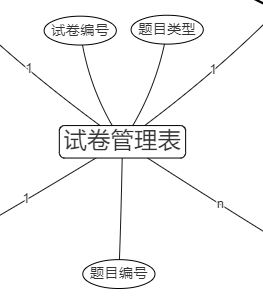
## 4.2.1 ER图



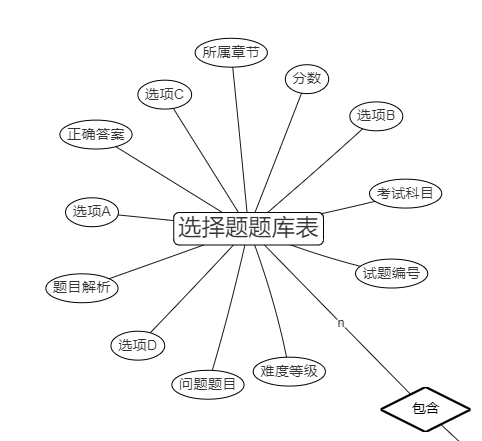
总ER图



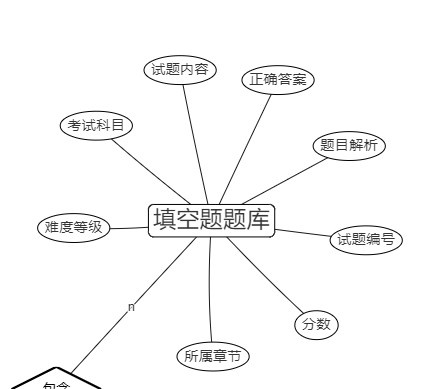
实体关联er图



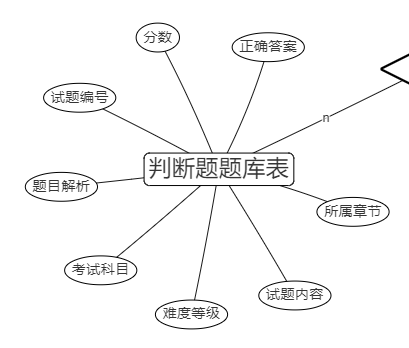
试卷管理表er图



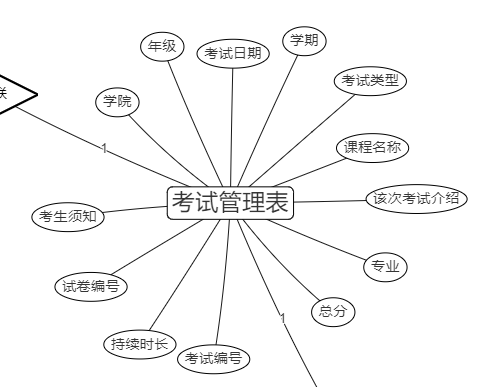
选择题题库表er图



填空题题库表er图



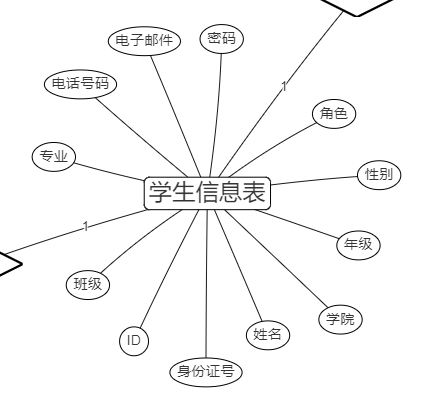
判断题题库表er图



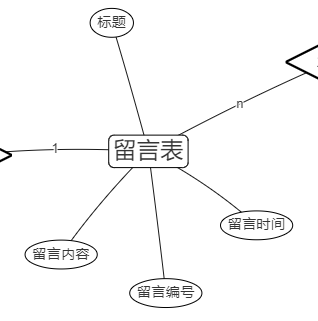
考试管理表er图



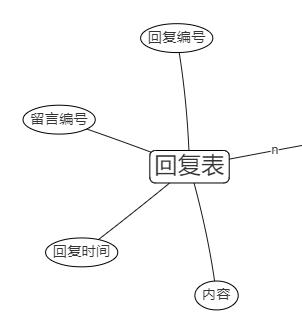
成绩管理表er图



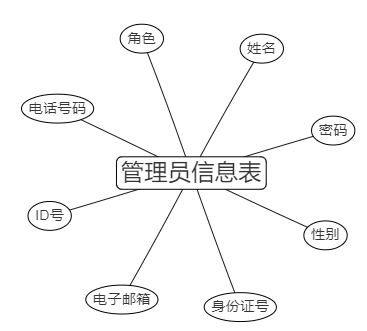
学生信息表er图



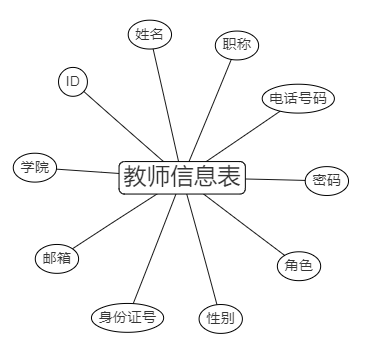
留言表er图



回复表er图



管理员表er图



教师表er图

## 4.2.2 数据库表设计

1、管理员表

| **字段名称** | **字段类型** | **字段描述** |
| --- | --- | --- |
| adminId | int(0) | ID号 |
| adminName | varchar(20) | 姓名 |
| sex | varchar(2) | 性别 |
| tel | varchar(11) | 电话号码 |
| email | varchar(20) | 电子邮箱 |
| pwd | varchar(16) | 密码 |
| cardId | varchar(18) | 身份证号 |
| role | varchar(1) | 角色 |

2、试卷管理表

| **字段名称** | **字段类型** | **字段描述** |
| --- | --- | --- |
| examCode | int(0) | 考试编号 |
| description | varchar(50) | 该次考试介绍 |
| source | varchar(20) | 课程名称 |
| paperId | int(0) | 试卷编号 |
| examDate | varchar(10) | 考试日期 |
| totalTime | int(0) | 持续时长 |
| grade | varchar(10) | 年级 |
| term | varchar(10) | 学期 |
| major | varchar(20) | 专业 |
| institute | varchar(20) | 学院 |
| totalScore | int(0) | 总分 |
| type | varchar(255) | 考试类型 |
| tips | varchar(255) | 考生须知 |

3、填空题题库表

| **字段名称** | **字段类型** | **字段描述** |
| --- | --- | --- |
| questionId | int(0) | 试题编号 |
| subject | varchar(20) | 考试科目 |
| question | varchar(255) | 试题内容 |
| answer | varchar(255) | 正确答案 |
| analysis | varchar(255) | 题目解析 |
| score | int(0) | 分数 |
| level | varchar(5) | 难度等级 |
| section | varchar(20) | 所属章节 |

4、判断题题库表

| **字段名称** | **字段类型** | **字段描述** |
| --- | --- | --- |
| questionId | int(0) | 试题编号 |
| subject | varchar(20) | 考试科目 |
| question | varchar(255) | 试题内容 |
| answer | varchar(255) | 正确答案 |
| analysis | varchar(255) | 题目解析 |
| score | int(0) | 分数 |
| level | varchar(1) | 难度等级 |
| section | varchar(20) | 所属章节 |

5、评论表

| **字段名称** | **字段类型** | **字段描述** |
| --- | --- | --- |
| id | int(0) | 留言编号 |
| title | varchar(20) | 标题 |
| content | varchar(255) | 留言内容 |
| time | date | 留言时间 |

6、选择题题库表

| **字段名称** | **字段类型** | **字段描述** |
| --- | --- | --- |
| questionId | int(0) | 试题编号 |
| subject | varchar(20) | 考试科目 |
| question | varchar(255) | 问题题目 |
| answerA | varchar(255) | 选项A |
| answerB | varchar(255) | 选项B |
| answerC | varchar(255) | 选项C |
| answerD | varchar(255) | 选项D |
| rightAnswer | varchar(10) | 正确答案 |
| analysis | varchar(255) | 题目解析 |
| score | int(0) | 分数 |
| section | varchar(20) | 所属章节 |
| level | varchar(1) | 难度等级 |

7、试卷管理表

| **字段名称** | **字段类型** | **字段描述** |
| --- | --- | --- |
| paperId | int(0) | 试卷编号 |
| questionType | int(0) | 题目类型 |
| questionId | int(0) | 题目编号 |

8、回复表

| **字段名称** | **字段类型** | **字段描述** |
| --- | --- | --- |
| messageId | int(0) | 留言编号 |
| replayId | int(0) | 回复编号 |
| replay | varchar(255) | 内容 |
| replayTime | date | 回复时间 |

9、成绩表

| **字段名称** | **字段类型** | **字段描述** |
| --- | --- | --- |
| scoreId | int(0) | 分数编号 |
| examCode | int(0) | 考试编号 |
| studentId | int(0) | 学号 |
| subject | varchar(20) | 课程名称 |
| ptScore | int(0) | 是否及格 |
| etScore | int(0) | 成绩 |
| score | int(0) | 试卷满分 |
| answerDate | varchar(10) | 答题日期 |

10、学生表

| **字段名称** | **字段类型** | **字段描述** |
| --- | --- | --- |
| studentId | int(0) | ID |
| studentName | varchar(20) | 姓名 |
| grade | varchar(4) | 年级 |
| major | varchar(20) | 专业 |
| clazz | varchar(10) | 班级 |
| institute | varchar(30) | 学院 |
| tel | varchar(11) | 电话号码 |
| email | varchar(30) | 电子邮件 |
| pwd | varchar(16) | 密码 |
| cardId | varchar(18) | 身份证号 |
| sex | varchar(2) | 性别 |
| role | varchar(1) | 角色 |

11、教师表

| **字段名称** | **字段类型** | **字段描述** |
| --- | --- | --- |
| teacherId | int(0) | ID |
| teacherName | varchar(20) | 姓名 |
| institute | varchar(20) | 学院 |
| sex | varchar(2) | 性别 |
| tel | varchar(11) | 电话号码 |
| email | varchar(20) | 邮箱 |
| pwd | varchar(16) | 密码 |
| cardId | varchar(18) | 身份证号 |
| type | varchar(20) | 职称 |
| role | varchar(1) | 角色 |

# 5.1 详细功能模块

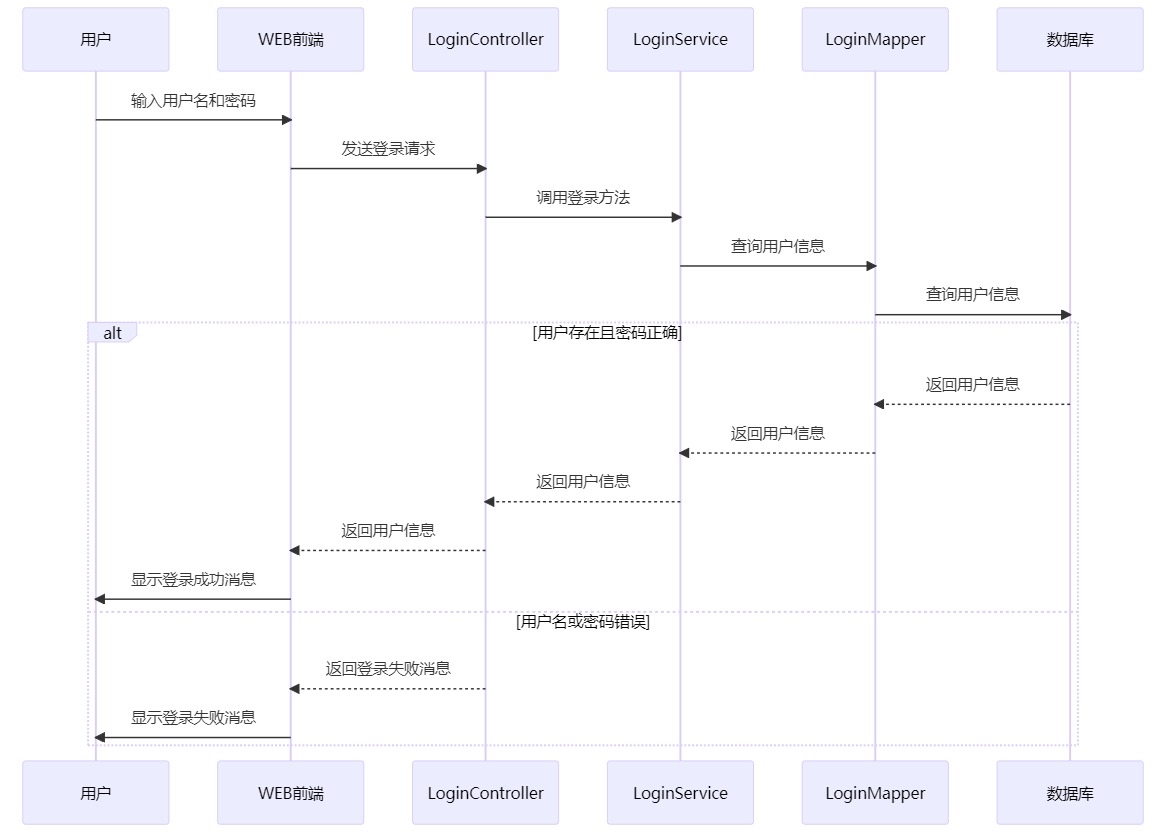
系统功能模块如表所示:

系统功能模块表

| **功能** | **输入** | **处理** | **输出** |
| --- | --- | --- | --- |
| 管理员登录 | 用户名、密码 | 验证用户名密码是否正确，在数据库中查询管理员信息 | 登录成功后显示管理员主页 |
| 修改管理员密码 | 原密码、新密码 | 验证原密码是否正确，更新数据库中的密码 | 密码修改成功提示 |
| 考试管理 | 添加、编辑、删除、查询、移除试卷题目 | 根据操作执行相应的数据库操作：增加、修改、删除、查询、移除试卷题目 | 显示试卷列表、题目管理页面 |
| 题目管理 | 添加、编辑、删除、查询题目 | 根据操作执行相应的数据库操作：增加、修改、删除、查询题目 | 显示题目列表、题目添加成功提示、题目编辑成功提示、题目删除成功提示、题目查询结果 |
| 查询学生成绩 | 根据条件查询学生成绩 | 根据查询条件，在数据库中查询学生成绩 | 显示学生成绩列表 |
| 根据试卷查询成绩分段 | 选择试卷，设置分段标准 | 根据选择的试卷和分段标准，在数据库中查询成绩，并将成绩分段 | 显示成绩分段饼状图 |
| 学生管理 | 添加、编辑、删除、查询学生信息 | 根据操作执行相应的数据库操作：增加、修改、删除、查询学生信息 | 显示学生列表、学生添加成功提示、学生编辑成功提示、学生删除成功提示、学生查询结果 |
| 教师管理 | 添加、编辑、删除、查询教师信息 | 根据操作执行相应的数据库操作：增加、修改、删除、查询教师信息 | 显示教师列表、教师添加成功提示、教师编辑成功提示、教师删除成功提示、教师查询结果 |
| 学生登录 | 学生账号、密码 | 验证学生账号和密码是否正确，在数据库中查询学生信息 | 登录成功后显示学生主页 |
| 考试中心 | 显示可以参加的考试列表 | 查询数据库中的考试信息，过滤出可以参加的考试，并显示在考试中心页面 | 显示考试列表 |
| 考试答题 | 根据考试题目进行答题 | 根据考试题目，在数据库中查询题目信息，并提供答题界面 | 显示考试题目、答题界面 |
| 提交试卷现场出分数 | 提交答案，记录考生成绩 | 将考生提交的答案与正确答案进行对比，计算考生分数，并记录分数 | 显示考生分数，提交成功提示 |
| 查询我的分数列表 | 查询个人成绩列表 | 根据学生账号，查询个人的成绩列表 | 显示个人成绩列表 |
| 交流区 | 发送留言、评论留言 | 将留言或评论信息保存到数据库中 | 显示留言、评论信息 |
| 修改学生密码 | 原密码、新密码 | 验证原密码是否正确，更新数据库中的密码 | 密码修改成功提示 |

## 5.1.1 主要模块-用户登录模块

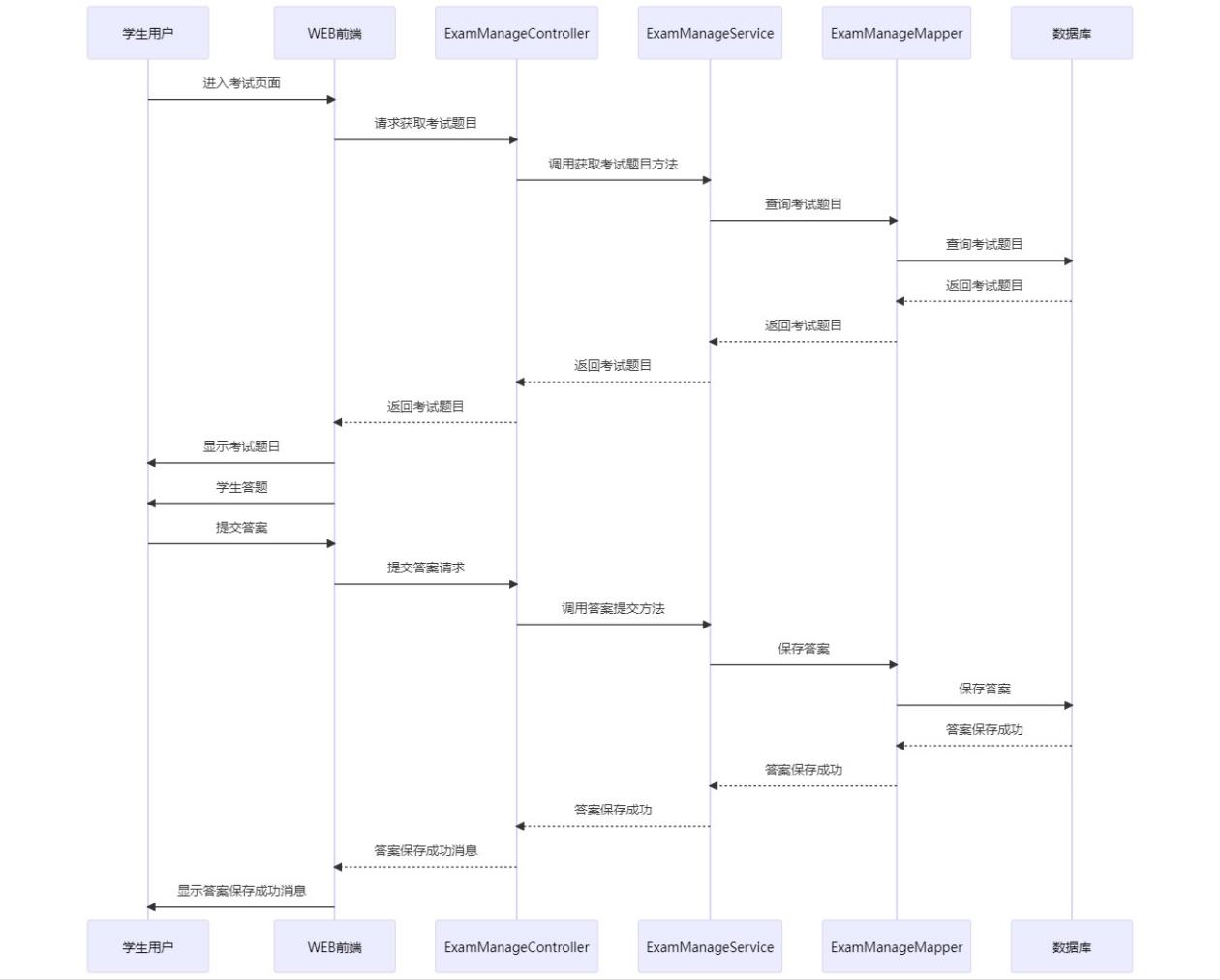
用户（User）在前端（Frontend）输入用户名和密码。前端向LoginController发送登录请求。LoginController调用LoginService的登录方法。LoginService通过LoginMapper查询用户信息。LoginMapper向数据库（Database）查询用户信息。如果用户存在且密码正确，数据库返回用户信息给LoginMapper，依次返回给LoginService、LoginController和前端。前端显示登录成功消息给用户。如果用户不存在或密码错误，LoginController返回登录失败消息给前端。前端显示登录失败消息给用户，如图所示。



用户登录时序图

## 5.1.2 主要模块-学生答题模块

学生（Student）进入考试页面（Frontend）。前端向ExamManageController发送请求获取考试题目。ExamManageController调用ExamManageService的获取考试题目方法。ExamManageService通过ExamManageMapper查询考试题目。ExamManageMapper向数据库（Database）查询考试题目。数据库返回考试题目给ExamManageMapper，依次返回给ExamManageService、ExamManageController和前端。前端显示考试题目给学生。学生答题并在前端提交答案。前端向ExamManageController发送答案提交请求。ExamManageController调用ExamManageService的答案提交方法。ExamManageService通过ExamManageMapper保存答案。ExamManageMapper向数据库保存答案。数据库返回答案保存成功给ExamManageMapper，依次返回给ExamManageService、ExamManageController和前端。前端显示答案保存成功消息给学生，如图所示。



学生答题时序图

# 6.1 后端接口主要代码实现

1、管理员接口

@GetMapping("/admins")

public ApiResult findAll(){

System.*out*.println("查询全部");

return ApiResultHandler.*success*(adminService.findAll());

}

@GetMapping("/admin/{adminId}")

public ApiResult findById(@PathVariable("adminId") Integer adminId){

System.*out*.println("根据ID查找");

return ApiResultHandler.*success*(adminService.findById(adminId));

}

@DeleteMapping("/admin/{adminId}")

public ApiResult deleteById(@PathVariable("adminId") Integer adminId){

adminService.deleteById(adminId);

return ApiResultHandler.*success*();

}

@PutMapping("/admin/{adminId}")

public ApiResult update(@PathVariable("adminId") Integer adminId, Admin admin){

return ApiResultHandler.*success*(adminService.update(admin));

}

@PostMapping("/admin")

public ApiResult add(Admin admin){

return ApiResultHandler.*success*(adminService.add(admin));

}

@GetMapping("/admin/resetPsw/{adminId}/{oldPsw}/{newPsw}")

public ApiResult resetPsw(@PathVariable("adminId") Integer adminId, @PathVariable("newPsw") String newPsw, @PathVariable("oldPsw") String oldPsw) {

return ApiResultHandler.*success*(adminService.resetPsw(adminId, newPsw, oldPsw));

}

2、答题接口

@GetMapping("/answers/{page}/{size}/{subject}/{section}/{question}")

public ApiResult findAllQuestion(

@PathVariable("page") Integer page, @PathVariable("size") Integer size,

@PathVariable("subject") String subject, @PathVariable("section") String section,

@PathVariable("question") String question){

Page<AnswerVO> answerVOPage = new Page<>(page,size);

IPage<AnswerVO> answerVOIPage = answerService.findAll(answerVOPage, subject, section, question);

return ApiResultHandler.*buildApiResult*(200,"查询所有题库", answerVOIPage);

}

*/\*\**

*\* 根据类型和id获取题目*

*\**

*\* @param type 类型*

*\* @param questionId 题目id*

*\* @return 题目信息*

*\*/*

@GetMapping("/answers/{type}/{questionId}")

public ApiResult findByIdAndType(

@PathVariable("type") String type, @PathVariable("questionId") Long questionId

) {

QuestionVO questionVO = answerService.findByIdAndType(type, questionId);

return ApiResultHandler.*buildApiResult*(200, "查询题目", questionVO);

}

3、考试管理接口

@GetMapping("/exams")

public ApiResult findAll(){

System.*out*.println("不分页查询所有试卷");

ApiResult apiResult;

apiResult = ApiResultHandler.*buildApiResult*(200, "请求成功！", examManageService.findAll());

return apiResult;

}

@GetMapping("/exams/{page}/{size}")

public ApiResult findAll(@PathVariable("page") Integer page, @PathVariable("size") Integer size){

System.*out*.println("分页查询所有试卷");

ApiResult apiResult;

Page<ExamManage> examManage = new Page<>(page,size);

IPage<ExamManage> all = examManageService.findAll(examManage);

apiResult = ApiResultHandler.*buildApiResult*(200, "请求成功！", all);

return apiResult;

}

@GetMapping("/exam/{examCode}")

public ApiResult findById(@PathVariable("examCode") Integer examCode){

System.*out*.println("根据ID查找");

ExamManage res = examManageService.findById(examCode);

if(res == null) {

return ApiResultHandler.*buildApiResult*(10000,"考试编号不存在",null);

}

return ApiResultHandler.*buildApiResult*(200,"请求成功！",res);

}

@DeleteMapping("/exam/{examCode}")

public ApiResult deleteById(@PathVariable("examCode") Integer examCode){

int res = examManageService.delete(examCode);

return ApiResultHandler.*buildApiResult*(200,"删除成功",res);

}

@PutMapping("/exam")

public ApiResult update(@RequestBody ExamManage exammanage){

int res = examManageService.update(exammanage);

*// if (res == 0) {*

*// return ApiResultHandler.buildApiResult(20000,"请求参数错误");*

*// }*

System.*out*.print("更新操作执行---");

return ApiResultHandler.*buildApiResult*(200,"更新成功",res);

}

@PostMapping("/exam")

public ApiResult add(@RequestBody ExamManage exammanage){

int res = examManageService.add(exammanage);

if (res ==1) {

return ApiResultHandler.*buildApiResult*(200, "添加成功", res);

} else {

return ApiResultHandler.*buildApiResult*(400,"添加失败",res);

}

}

@GetMapping("/examManagePaperId")

public ApiResult findOnlyPaperId() {

ExamManage res = examManageService.findOnlyPaperId();

if (res != null) {

return ApiResultHandler.*buildApiResult*(200,"请求成功",res);

}

return ApiResultHandler.*buildApiResult*(400,"请求失败",res);

}

4、填空题接口

@PostMapping("/fillQuestion")

public ApiResult add(@RequestBody FillQuestion fillQuestion) {

int res = fillQuestionService.add(fillQuestion);

if (res != 0) {

return ApiResultHandler.*buildApiResult*(200,"添加成功",res);

}

return ApiResultHandler.*buildApiResult*(400,"添加失败",res);

}

@GetMapping("/fillQuestionId")

public ApiResult findOnlyQuestionId() {

FillQuestion res = fillQuestionService.findOnlyQuestionId();

return ApiResultHandler.*buildApiResult*(200,"查询成功",res);

}

@PostMapping("/editFillQuestion")

public ApiResult edit(@RequestBody FillQuestion fillQuestion) {

int res = fillQuestionService.edit(fillQuestion);

if (res != 0) {

return ApiResultHandler.*buildApiResult*(200,"修改成功",res);

}

return ApiResultHandler.*buildApiResult*(400,"修改失败",res);

}

5、组卷接口

@PostMapping("/item")

public ApiResult ItemController(@RequestBody Item item) {

*// 选择题*

Integer changeNumber = item.getChangeNumber();

*// 填空题*

Integer fillNumber = item.getFillNumber();

*// 判断题*

Integer judgeNumber = item.getJudgeNumber();

*//出卷id*

Integer paperId = item.getPaperId();

*// 数据库获取数据*

List<Integer> changeNumbers = multiQuestionService.findBySubject(item.getSubject(), changeNumber);

List<Integer> fills = fillQuestionService.findBySubject(item.getSubject(), fillNumber);

List<Integer> judges = judgeQuestionService.findBySubject(item.getSubject(), judgeNumber);

if(changeNumbers == null || changeNumbers.size() != changeNumber){

return ApiResultHandler.*buildApiResult*(400,"科目【" + item.getSubject() + "】题库【选择题】题目数量不足【" + changeNumber + "】，组卷失败",null);

}

if(fills == null || fills.size() != fillNumber) {

return ApiResultHandler.*buildApiResult*(400,"科目【" + item.getSubject() + "】题库【填空题】题目数量不足【" + fillNumber + "】，组卷失败",null);

}

if(judges == null || judges.size() != judgeNumber){

return ApiResultHandler.*buildApiResult*(400,"科目【" + item.getSubject() + "】题库【判断题】题目数量不足【" + judgeNumber + "】，组卷失败",null);

}

*// 符合组题条件，执行组题*

*// 选择题*

for (Integer number : changeNumbers) {

PaperManage paperManage = new PaperManage(paperId,1,number);

int index = paperService.add(paperManage);

if(index==0)

return ApiResultHandler.*buildApiResult*(400,"选择题组卷保存失败",null);

}

*// 填空题*

for (Integer fillNum : fills) {

PaperManage paperManage = new PaperManage(paperId,2,fillNum);

int index = paperService.add(paperManage);

if(index==0)

return ApiResultHandler.*buildApiResult*(400,"填空题题组卷保存失败",null);

}

*// 判断题*

for (Integer judge : judges) {

PaperManage paperManage = new PaperManage(paperId,3,judge);

int index = paperService.add(paperManage);

if(index==0)

return ApiResultHandler.*buildApiResult*(400,"判断题题组卷保存失败",null);

}

return ApiResultHandler.*buildApiResult*(200,"试卷组卷成功",null);

}

6、判断题接口

@PostMapping("/judgeQuestion")

public ApiResult add(@RequestBody JudgeQuestion judgeQuestion) {

int res = judgeQuestionService.add(judgeQuestion);

if (res != 0) {

return ApiResultHandler.*buildApiResult*(200,"添加成功",res);

}

return ApiResultHandler.*buildApiResult*(400,"添加失败",res);

}

@GetMapping("/judgeQuestionId")

public ApiResult findOnlyQuestionId() {

JudgeQuestion res = judgeQuestionService.findOnlyQuestionId();

return ApiResultHandler.*buildApiResult*(200,"查询成功",res);

}

@PostMapping("/editJudgeQuestion")

public ApiResult edit(@RequestBody JudgeQuestion judgeQuestion) {

int res = judgeQuestionService.edit(judgeQuestion);

if (res != 0) {

return ApiResultHandler.*buildApiResult*(200,"修改成功",res);

}

return ApiResultHandler.*buildApiResult*(400,"修改失败",res);

}

7、登录接口

@PostMapping("/login")

public ApiResult login(@RequestBody Login login, HttpServletRequest request, HttpServletResponse response) {

Integer username = login.getUsername();

String password = login.getPassword();

Admin adminRes = loginService.adminLogin(username, password);

if (adminRes != null) {

Cookie token = new Cookie("rb\_token", adminRes.getCardId());

token.setPath("/");

Cookie role = new Cookie("rb\_role", "0");

role.setPath("/");

*//将cookie对象加入response响应*

response.addCookie(token);

response.addCookie(role);

return ApiResultHandler.*buildApiResult*(200, "请求成功", adminRes);

}

Teacher teacherRes = loginService.teacherLogin(username,password);

if (teacherRes != null) {

Cookie token = new Cookie("rb\_token", teacherRes.getCardId());

token.setPath("/");

Cookie role = new Cookie("rb\_role", "1");

role.setPath("/");

response.addCookie(token);

response.addCookie(role);

return ApiResultHandler.*buildApiResult*(200, "请求成功", teacherRes);

}

Student studentRes = loginService.studentLogin(username,password);

if (studentRes != null) {

Cookie token = new Cookie("rb\_token", studentRes.getCardId());

token.setPath("/");

Cookie role = new Cookie("rb\_role", "2");

role.setPath("/");

response.addCookie(token);

response.addCookie(role);

return ApiResultHandler.*buildApiResult*(200, "请求成功", studentRes);

}

return ApiResultHandler.*buildApiResult*(400, "请求失败", null);

}

@PostMapping("/logout")

public void logout(HttpServletRequest request, HttpServletResponse response) {

Cookie token = new Cookie("rb\_token", null);

token.setPath("/");

token.setMaxAge(0);

Cookie role = new Cookie("rb\_role", null);

role.setPath("/");

role.setMaxAge(0);

response.addCookie(token);

response.addCookie(role);

}

8、交流评论接口

@GetMapping("/messages/{page}/{size}")

public ApiResult<Message> findAll(@PathVariable("page") Integer page, @PathVariable("size") Integer size) {

Page<Message> messagePage = new Page<>(page,size);

IPage<Message> all = messageService.findAll(messagePage);

return ApiResultHandler.*buildApiResult*(200,"查询所有留言",all);

}

@GetMapping("/message/{id}")

public ApiResult findById(@PathVariable("id") Integer id) {

Message res = messageService.findById(id);

return ApiResultHandler.*buildApiResult*(200,"根据Id查询",res);

}

@DeleteMapping("/message/{id}")

public int delete(@PathVariable("id") Integer id) {

int res = messageService.delete(id);

return res;

}

@PostMapping("/message")

public ApiResult add(@RequestBody Message message) {

int res = messageService.add(message);

if (res == 0) {

return ApiResultHandler.*buildApiResult*(400,"添加失败",res);

} else {

return ApiResultHandler.*buildApiResult*(200,"添加成功",res);

}

}

9、选择题接口

@GetMapping("/multiQuestionId")

public ApiResult findOnlyQuestion() {

MultiQuestion res = multiQuestionService.findOnlyQuestionId();

return ApiResultHandler.*buildApiResult*(200,"查询成功",res);

}

@PostMapping("/MultiQuestion")

public ApiResult add(@RequestBody MultiQuestion multiQuestion) {

int res = multiQuestionService.add(multiQuestion);

if (res != 0) {

return ApiResultHandler.*buildApiResult*(200,"添加成功",res);

}

return ApiResultHandler.*buildApiResult*(400,"添加失败",res);

}

@PostMapping("/editMultiQuestion")

public ApiResult edit(@RequestBody MultiQuestion multiQuestion) {

int res = multiQuestionService.edit(multiQuestion);

if (res != 0) {

return ApiResultHandler.*buildApiResult*(200,"修改成功",res);

}

return ApiResultHandler.*buildApiResult*(400,"修改失败",res);

}

10、试卷接口

@GetMapping("/papers")

public ApiResult<PaperManage> findAll() {

ApiResult res = ApiResultHandler.*buildApiResult*(200,"请求成功",paperService.findAll());

return res;

}

@GetMapping("/paper/{paperId}")

public Map<Integer, List<?>> findById(@PathVariable("paperId") Integer paperId) {

List<MultiQuestion> multiQuestionRes = multiQuestionService.findByIdAndType(paperId); *//选择题题库 1*

List<FillQuestion> fillQuestionsRes = fillQuestionService.findByIdAndType(paperId); *//填空题题库 2*

List<JudgeQuestion> judgeQuestionRes = judgeQuestionService.findByIdAndType(paperId); *//判断题题库 3*

Map<Integer, List<?>> map = new HashMap<>();

map.put(1,multiQuestionRes);

map.put(2,fillQuestionsRes);

map.put(3,judgeQuestionRes);

return map;

}

@PostMapping("/paperManage")

public ApiResult add(@RequestBody PaperManage paperManage) {

int res = paperService.add(paperManage);

if (res != 0) {

return ApiResultHandler.*buildApiResult*(200,"添加成功",res);

}

return ApiResultHandler.*buildApiResult*(400,"添加失败",res);

}

*/\*\**

*\* 删除试卷中的某条试题*

*\**

*\* @param paperId 试卷id*

*\* @param type 题目类型。1选择，2填空，3判断*

*\* @param questionId 题目id*

*\*/*

@GetMapping("/paper/delete/{paperId}/{type}/{questionId}")

public ApiResult delete(

@PathVariable("paperId") String paperId,

@PathVariable("type") String type,

@PathVariable("questionId") String questionId

) {

paperService.delete(paperId, type, questionId);

return ApiResultHandler.*buildApiResult*(200,"删除成功", null);

}

11、回复接口

@PostMapping("/replay")

public ApiResult add(@RequestBody Replay replay) {

int data = replayService.add(replay);

if (data != 0) {

return ApiResultHandler.*buildApiResult*(200,"添加成功！",data);

} else {

return ApiResultHandler.*buildApiResult*(400,"添加失败！",null);

}

}

@GetMapping("/replay/{messageId}")

public ApiResult findAllById(@PathVariable("messageId") Integer messageId) {

List<Replay> res = replayService.findAllById(messageId);

return ApiResultHandler.*buildApiResult*(200,"根据messageId查询",res);

}

12、成绩管理接口

@GetMapping("/scores")

public ApiResult findAll() {

List<Score> res = scoreService.findAll();

return ApiResultHandler.*buildApiResult*(200,"查询所有学生成绩",res);

}

*// 分页*

@GetMapping("/score/{page}/{size}/{studentId}")

public ApiResult findById(@PathVariable("page") Integer page, @PathVariable("size") Integer size, @PathVariable("studentId") Integer studentId) {

Page<Score> scorePage = new Page<>(page, size);

IPage<Score> res = scoreService.findById(scorePage, studentId);

return ApiResultHandler.*buildApiResult*(200, "根据ID查询成绩", res);

}

*// 不分页*

@GetMapping("/score/{studentId}")

public ApiResult findById(@PathVariable("studentId") Integer studentId) {

List<Score> res = scoreService.findById(studentId);

if (!res.isEmpty()) {

return ApiResultHandler.*buildApiResult*(200, "根据ID查询成绩", res);

} else {

return ApiResultHandler.*buildApiResult*(400, "ID不存在", res);

}

}

@PostMapping("/score")

public ApiResult add(@RequestBody Score score) {

int res = scoreService.add(score);

if (res == 0) {

return ApiResultHandler.*buildApiResult*(400,"成绩添加失败",res);

}else {

return ApiResultHandler.*buildApiResult*(200,"成绩添加成功",res);

}

}

@GetMapping("/scores/{examCode}")

public ApiResult findByExamCode(@PathVariable("examCode") Integer examCode) {

List<Score> scores = scoreService.findByExamCode(examCode);

return ApiResultHandler.*buildApiResult*(200,"查询成功",scores);

}

13、学生信息接口

@GetMapping("/students/{page}/{size}/{name}/{grade}/{tel}/{institute}/{major}/{clazz}")

public ApiResult findAll(@PathVariable Integer page, @PathVariable Integer size,

@PathVariable String name, @PathVariable String grade,

@PathVariable String tel, @PathVariable String institute,

@PathVariable String major, @PathVariable String clazz) {

Page<Student> studentPage = new Page<>(page,size);

IPage<Student> res = studentService.findAll(

studentPage, name, grade, tel, institute, major, clazz

);

return ApiResultHandler.*buildApiResult*(200,"分页查询所有学生",res);

}

@GetMapping("/student/{studentId}")

public ApiResult findById(@PathVariable("studentId") Integer studentId) {

Student res = studentService.findById(studentId);

if (res != null) {

return ApiResultHandler.*buildApiResult*(200,"请求成功",res);

} else {

return ApiResultHandler.*buildApiResult*(404,"查询的用户不存在",null);

}

}

@DeleteMapping("/student/{studentId}")

public ApiResult deleteById(@PathVariable("studentId") Integer studentId) {

return ApiResultHandler.*buildApiResult*(200,"删除成功",studentService.deleteById(studentId));

}

@PutMapping("/studentPWD")

public ApiResult updatePwd(@RequestBody Student student) {

studentService.updatePwd(student);

return ApiResultHandler.*buildApiResult*(200,"密码更新成功",null);

}

@PutMapping("/student")

public ApiResult update(@RequestBody Student student) {

int res = studentService.update(student);

if (res != 0) {

return ApiResultHandler.*buildApiResult*(200,"更新成功",res);

}

return ApiResultHandler.*buildApiResult*(400,"更新失败",res);

}

@PostMapping("/student")

public ApiResult add(@RequestBody Student student) {

int res = studentService.add(student);

if (res == 1) {

return ApiResultHandler.*buildApiResult*(200,"添加成功",null);

}else {

return ApiResultHandler.*buildApiResult*(400,"添加失败",null);

}

}

14、教师信息接口

@GetMapping("/teachers/{page}/{size}")

public ApiResult findAll(@PathVariable Integer page, @PathVariable Integer size){

Page<Teacher> teacherPage = new Page<>(page,size);

IPage<Teacher> teacherIPage = teacherService.findAll(teacherPage);

return ApiResultHandler.*buildApiResult*(200,"查询所有教师",teacherIPage);

}

@GetMapping("/teacher/{teacherId}")

public ApiResult findById(@PathVariable("teacherId") Integer teacherId){

return ApiResultHandler.*success*(teacherService.findById(teacherId));

}

@DeleteMapping("/teacher/{teacherId}")

public ApiResult deleteById(@PathVariable("teacherId") Integer teacherId){

return ApiResultHandler.*success*(teacherService.deleteById(teacherId));

}

@PutMapping("/teacher")

public ApiResult update(@RequestBody Teacher teacher){

return ApiResultHandler.*success*(teacherService.update(teacher));

}

@PostMapping("/teacher")

public ApiResult add(@RequestBody Teacher teacher){

return ApiResultHandler.*success*(teacherService.add(teacher));

}

# 6.2 前端页面主要代码实现

## 6.2.1 登录注册页

<!-- 用户登录界面 -->

<template>

    <div id="login">

        <div class="bg"></div>

        <el-row class="main-container">

            <el-col :lg="8" :xs="16" :md="10" :span="10">

                <div class="top" style="color: black;font-size: 32px">

                    <i class="iconfont icon-r-team title-icon"></i> 在线考试系统

                    <span class="title"> </span>

                </div>

                <br />

                <div class="bottom">

                    <div class="container">

                        <p class="title">账号登录</p>

                        <el-form

                            :label-position="labelPosition"

                            label-width="80px"

                            :model="formLabelAlign"

                        >

                            <el-form-item label="用户名">

                                <el-input

                                    v-model.number="formLabelAlign.username"

                                    placeholder="请输入用户名"

                                ></el-input>

                            </el-form-item>

                            <el-form-item label="密码">

                                <el-input

                                    v-model="formLabelAlign.password"

                                    placeholder="请输入密码"

                                    type="password"

                                ></el-input>

                            </el-form-item>

                            <div class="submit">

                                <el-button

                                    type="primary"

                                    class="row-login"

                                    @click="login()"

                                    > 登录</el-button

                                >

                            </div>

                        </el-form>

                    </div>

                </div>

            </el-col>

        </el-row>

        <el-row class="footer">

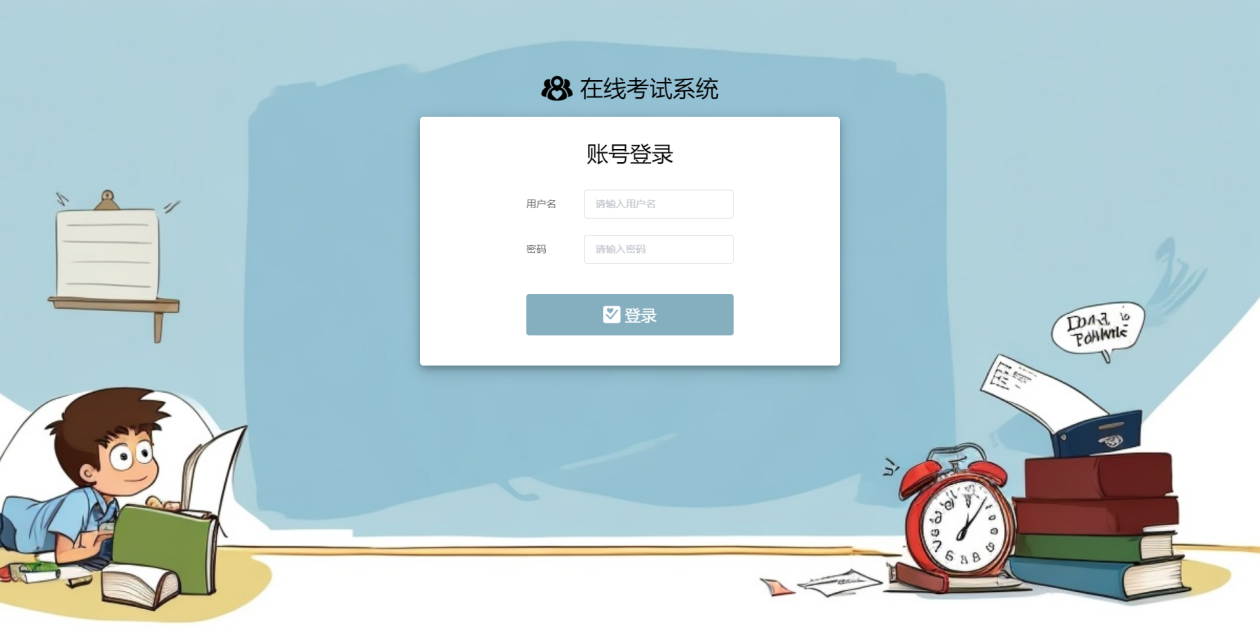
            <el-col>

            </el-col>

        </el-row>

    </div>

</template>



## 6.2.2 学生首页

<!--学生考试首页-->

<template>

    <div id="student">

        <el-row class="padding-50">

            <el-col :span="24">

                <ul class="list">

                    <img

                        src="@/assets/033.png"

                        style="

                            width: 40px;

                            height: 40px;

                            margin: 13px -15px 0 10px;

                            -webkit-user-drag: none;

                            -khtml-user-drag: none;

                            -moz-user-drag: none;

                            user-drag: none;

                        "

                    />

                    <li class="logo"><span>在线考试系统</span></li>

                    <li @click="exam()"><a href="javascript:;">考试中心</a></li>

                    <li @click="practice()">

                        <a href="javascript:;">试卷练习</a>

                    </li>

                    <li>

                        <router-link to="/scoreTable">我的分数</router-link>

                    </li>

                    <li><router-link to="/message">交流区</router-link></li>

                    <li

                        class="right"

                        @mouseenter="flag = !flag"

                        @mouseleave="flag = !flag"

                    >

                        <el-row>

                            <el-col :span="12">

                                <el-avatar

                                    shape="square"

                                    :size="50"

                                    :src="require('@/assets/img/avatar.jpg')"

                                    style="margin: 5px 5px 0 0"

                                ></el-avatar>

                            </el-col>

                            <el-col :span="12">

                                <a

                                    href="javascript:;"

                                    style="font-size: 22px; font-weight: 600"

                                    ><i

                                        class="iconfont icon-Userselect icon"

                                    ></i

                                    >{{ user.userName }}</a

                                >

                                <div class="msg" v-if="flag">

                                    <p @click="manage()">修改密码</p>

                                    <p class="exit" @click="exit()">退出</p>

                                </div>

                            </el-col>

                        </el-row>

                    </li>

                </ul>

            </el-col>

        </el-row>

        <!--路由区域-->

        <div class="main">

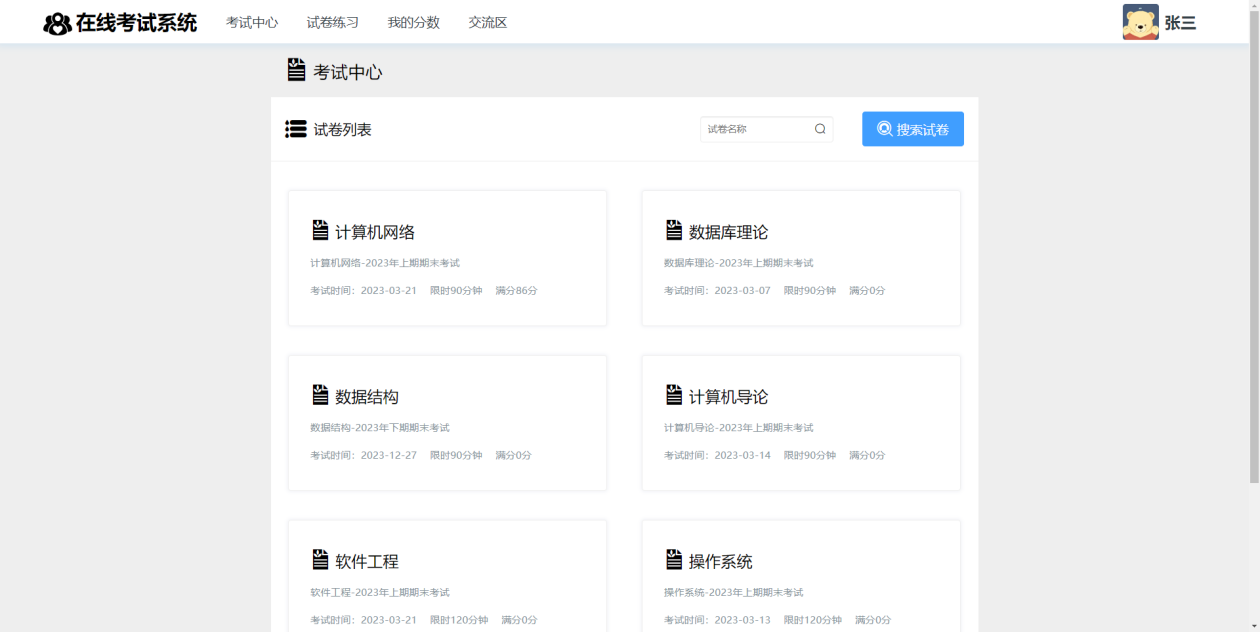
            <router-view></router-view>

        </div>

        <v-footer></v-footer>

    </div>

</template>



学生首页

## 6.2.3 答题页

<!--考生答题界面-->

<template>

    <div id="answer">

        <!--顶部信息栏-->

        <div class="top">

            <ul class="item">

                <li class="item-top">

                    <i class="iconfont icon-menufold icon20" ref="toggle" @click="slider\_flag = !slider\_flag"></i>

                    <i class="iconfont icon-r-paper" style="font-size: 32px;">

                    </i>

                    {{ examData.type }}-{{ examData.source }}

                </li>

                <li @click="flag = !flag" style="cursor: pointer">

                    <i class="iconfont icon-user icon20"></i>

                    <div class="msg" v-if="flag" @click="flag = !flag">

                        <p>姓名：{{ userInfo.name }}</p>

                        <p>准考证号: {{ userInfo.id }}</p>

                    </div>

                </li>

                <li><i class="iconfont icon-arrLeft icon20"></i></li>

            </ul>

        </div>

        <div class="flexarea">

            <!--左边题目编号区-->

            <transition name="slider-fade">

                <div class="left" v-if="slider\_flag">

                    <ul class="l-top">

                        <li>

                            <a href="javascript:;"></a>

                            <span>当前</span>

                        </li>

                        <li>

                            <a href="javascript:;"></a>

                            <span>未答</span>

                        </li>

                        <li>

                            <a href="javascript:;"></a>

                            <span>已答</span>

                        </li>

                        <li>

                            <a href="javascript:;"></a>

                            <span>标记</span>

                        </li>

                    </ul>

                    <div class="l-bottom">

                        <div class="item">

                            <p>选择题部分</p>

                            <ul>

                                <li v-for="(list, index1) in topic[1]" :key="index1">

                                    <a href="javascript:;" @click="change(index1)" :class="{

                                        border:

                                            index == index1 &&

                                            currentType == 1,

                                        bg:

                                            bg\_flag &&

                                            topic[1][index1].isClick ==

                                            true,

                                    }">

                                        <span :class="{

                                            mark:

                                                topic[1][index1].isMark ==

                                                true,

                                        }"></span>

                                        {{ index1 + 1 }}

                                    </a>

                                </li>

                            </ul>

                        </div>

                        <div class="item">

                            <p>填空题部分</p>

                            <ul>

                                <li v-for="(list, index2) in topic[2]" :key="index2">

                                    <a href="javascript:;" @click="fill(index2)" :class="{

                                        border:

                                            index == index2 &&

                                            currentType == 2,

                                        bg: fillAnswer[index2][3] == true,

                                    }"><span :class="{

                                        mark:

                                            topic[2][index2].isMark ==

                                            true,

                                    }"></span>{{ topicCount[0] + index2 + 1 }}

                                    </a>

                                </li>

                            </ul>

                        </div>

                        <div class="item">

                            <p>判断题部分</p>

                            <ul>

                                <li v-for="(list, index3) in topic[3]" :key="index3">

                                    <a href="javascript:;" @click="judge(index3)" :class="{

                                        border:

                                            index == index3 &&

                                            currentType == 3,

                                        bg:

                                            bg\_flag &&

                                            topic[3][index3].isClick ==

                                            true,

                                    }"><span :class="{

                                            mark:

                                                topic[3][index3].isMark ==

                                                true,

                                        }"></span>{{

                                            topicCount[0] +

                                            topicCount[1] +

                                            index3 +

                                            1

                                        }}</a>

                                </li>

                            </ul>

                        </div>

                        <el-button type="primary" style="width: 100%;font-size: 22px;margin: 25px 0 0 0;"

                            @click="finishExam()"> 结束考试</el-button>

                    </div>

                </div>

            </transition>

            <!--右边选择答题区-->

            <transition name="slider-fade">

                <div class="right">

                    <div class="title">

                        <p style="font-size: 22px;"><i class="iconfont icon-r-edit" style="font-size: 28px;"></i> {{ title }}</p>

                        <i class="iconfont icon-right auto-right"></i>

                        <span>全卷共{{

                            topicCount[0] + topicCount[1] + topicCount[2]

                        }}题

                            <span v-if="!isPractice">

                                <i class="iconfont icon-time"></i>倒计时：<b>{{

                                    time

                                }}</b>分钟</span>

                        </span>

                    </div>

                    <div class="content">

                        <p class="topic">

                            <span class="number">{{ number }}</span>{{ showQuestion }}

                        </p>

                        <div v-if="currentType == 1">

                            <el-radio-group v-model="radio[index]" @change="getChangeLabel">

                                <el-radio :label="1">{{

                                    showAnswer.answerA

                                }}</el-radio>

                                <el-radio :label="2">{{

                                    showAnswer.answerB

                                }}</el-radio>

                                <el-radio :label="3">{{

                                    showAnswer.answerC

                                }}</el-radio>

                                <el-radio :label="4">{{

                                    showAnswer.answerD

                                }}</el-radio>

                            </el-radio-group>

                            <div class="analysis" v-if="isPractice && radio[index] != undefined">

                                <ul>

                                    <li>

                                        <el-tag type="success">正确答案：</el-tag><span class="right">{{

                                            reduceAnswer.rightAnswer

                                        }}</span>

                                    </li>

                                    <li><el-tag>题目解析：</el-tag></li>

                                    <li>

                                        {{

                                            reduceAnswer.analysis == null

                                            ? "暂无解析"

                                            : reduceAnswer.analysis

                                        }}

                                    </li>

                                </ul>

                            </div>

                        </div>

                        <div class="fill" v-if="currentType == 2">

                            <div v-for="(item, currentIndex) in part" :key="currentIndex">

                                <el-input placeholder="请填在此处" v-model="fillAnswer[index][currentIndex]" clearable

                                    @blur="fillBG">

                                </el-input>

                            </div>

                            <div class="analysis" v-if="

                                isPractice &&

                                fillAnswer[index][0] != undefined

                            ">

                                <ul>

                                    <li>

                                        <el-tag type="success">正确答案：</el-tag><span class="right">{{

                                            topic[2][index].answer

                                        }}</span>

                                    </li>

                                    <li><el-tag>题目解析：</el-tag></li>

                                    <li>

                                        {{

                                            topic[2][index].analysis == null

                                            ? "暂无解析"

                                            : topic[2][index].analysis

                                        }}

                                    </li>

                                </ul>

                            </div>

                        </div>

                        <div class="judge" v-if="currentType == 3">

                            <el-radio-group v-model="judgeAnswer[index]" @change="getJudgeLabel" v-if="currentType == 3">

                                <el-radio :label="1">正确</el-radio>

                                <el-radio :label="2">错误</el-radio>

                            </el-radio-group>

                            <div class="analysis" v-if="

                                isPractice &&

                                judgeAnswer[index] != undefined

                            ">

                                <ul>

                                    <li>

                                        <el-tag type="success">正确答案：</el-tag><span class="right">{{

                                            topic[3][index].answer == "T"

                                            ? "正确"

                                            : "错误"

                                        }}</span>

                                    </li>

                                    <li><el-tag>题目解析：</el-tag></li>

                                    <li>

                                        {{

                                            topic[3][index].analysis == null

                                            ? "暂无解析"

                                            : topic[3][index].analysis

                                        }}

                                    </li>

                                </ul>

                            </div>

                        </div>

                    </div>

                    <div class="operation">

                        <ul class="end">

                            <li @click="previous()" style="font-size: 24px;">

                                <i class="iconfont icon-r-left" style="font-size: 32px;margin-right: 5px;"></i><span>

                                    上一题</span>

                            </li>

                            <li @click="mark()">

                                <span>标记</span>

                            </li>

                            <li @click="next()" style="font-size: 24px;">

                                <span>下一题 </span><i class="iconfont icon-r-right"

                                    style="font-size: 32px;margin-left: 5px;"></i>

                            </li>

                        </ul>

                    </div>

                </div>

            </transition>

        </div>

    </div>

</template>



## 6.2.4 管理员、教师首页

<template>

  <div id="index">

    <header1 class="topbar"></header1>

    <section class="container">

      <div class="left\_side">

        <mainLeft></mainLeft>

      </div>

      <div class="main\_wrapper">

        <navigator class="nav"></navigator>

        <router-view></router-view>

      </div>

    </section>

  </div>

</template>

<template>

    <div class="index">

        <div class="hello">

            <i

                class="iconfont icon-r-user2"

                style="font-size: 34px;"

            ></i>

            <span> 很高兴遇见你, {{ user.userName }}。</span>

        </div>

        <div style="margin-top: 20px; font-size: 24px">

            <h1>欢迎登录在线考试系统后台</h1>

            <br />

            <img

                :src="require('@/assets/img/admin\_bg.jpg')"

                style="margin-top: 10px; width: 60vw"

            />

        </div>

    </div>

</template>



## 6.2.5 考试管理页面

<template>

  <div class="exam">

    <el-table :data="pagination.records" border>

      <el-table-column

        fixed="left"

        prop="source"

        label="试卷名称"

        width="180"

      ></el-table-column>

      <el-table-column

        prop="description"

        label="介绍"

        width="200"

      ></el-table-column>

      <el-table-column

        prop="institute"

        label="所属学院"

        width="120"

      ></el-table-column>

      <el-table-column

        prop="major"

        label="所属专业"

        width="200"

      ></el-table-column>

      <el-table-column prop="grade" label="年级" width="100"></el-table-column>

      <el-table-column

        prop="examDate"

        label="考试日期"

        width="120"

      ></el-table-column>

      <el-table-column

        prop="totalTime"

        label="持续时间"

        width="120"

      ></el-table-column>

      <el-table-column

        prop="totalScore"

        label="总分"

        width="120"

      ></el-table-column>

      <el-table-column

        prop="type"

        label="试卷类型"

        width="120"

      ></el-table-column>

      <el-table-column

        prop="tips"

        label="考生提示"

        width="400"

      ></el-table-column>

      <el-table-column fixed="right" label="操作" width="420">

        <template slot-scope="scope">

          <el-button type="success"  @click="getExamDetail(scope.row.examCode, scope.row.paperId)"

            >

            试题详情</el-button

          >

          <el-button

            @click="edit(scope.row.examCode)"

            type="primary"

            > 编辑</el-button

          >

          <el-button

            @click="deleteRecord(scope.row.examCode)"

            type="danger"

            >删除</el-button

          >

        </template>

      </el-table-column>

    </el-table>

    <el-pagination

      @size-change="handleSizeChange"

      @current-change="handleCurrentChange"

      :current-page="pagination.current"

      :page-sizes="[4, 8, 10]"

      :page-size="pagination.size"

      layout="total, sizes, prev, pager, next, jumper"

      :total="pagination.total"

      class="page"

    >

    </el-pagination>

    <!-- 编辑对话框-->

    <el-dialog

      title="编辑试卷信息"

      :visible.sync="dialogVisible"

      width="30%"

      :before-close="handleClose"

    >

      <section class="update">

        <el-form ref="form" :model="form" label-width="80px">

          <el-form-item label="试卷名称">

            <el-input v-model="form.source"></el-input>

          </el-form-item>

          <el-form-item label="介绍">

            <el-input v-model="form.description"></el-input>

          </el-form-item>

          <el-form-item label="所属学院">

            <el-input v-model="form.institute"></el-input>

          </el-form-item>

          <el-form-item label="所属专业">

            <el-input v-model="form.major"></el-input>

          </el-form-item>

          <el-form-item label="年级">

            <el-input v-model="form.grade"></el-input>

          </el-form-item>

          <el-form-item label="考试日期">

            <el-col :span="11">

              <el-date-picker

                type="date"

                placeholder="选择日期"

                v-model="form.examDate"

                value-format="yyyy-MM-dd"

                style="width: 100%"

              ></el-date-picker>

            </el-col>

          </el-form-item>

          <el-form-item label="持续时间">

            <el-input v-model="form.totalTime"></el-input>

          </el-form-item>

          <!-- <el-form-item label="总分">

            <el-input v-model="form.totalScore"></el-input>

          </el-form-item> -->

          <el-form-item label="试卷类型">

            <el-input v-model="form.type"></el-input>

          </el-form-item>

          <el-form-item label="考生提示">

            <el-input type="textarea" v-model="form.tips"></el-input>

          </el-form-item>

        </el-form>

      </section>

      <span slot="footer" class="dialog-footer">

        <el-button @click="dialogVisible = false" style="font-size: 20px;"> 取 消</el-button>

        <el-button type="primary" @click="submit()"> 确 定</el-button>

      </span>

    </el-dialog>

    <!-- 查询试卷对话框 -->

    <el-dialog title="试卷内容" :visible.sync="showExamDetail" width="60%">

      <span>

        <div class="content">

          <el-collapse v-model="activeName">

            <el-collapse-item class="header" name="0">

              <template slot="title">

                <div class="title">

                  <span>{{ examData.source }}</span

                  ><i class="header-icon el-icon-info"></i>

                  <span class="time"

                    >{{ score[0] + score[1] + score[2] }}分 /

                    {{ examData.totalTime }}分钟</span

                  >

                </div>

              </template>

              <el-collapse class="inner">

                <el-collapse-item>

                  <template slot="title">

                    <div class="titlei">

                      选择题 (共{{ topicCount[0] }}题 共计{{ score[0] }}分)

                    </div>

                  </template>

                  <div class="contenti">

                    <ul

                      class="question"

                      v-for="(list, index) in topic[1]"

                      :key="index"

                    >

                      <li>

                        {{ index + 1 }}. {{ list.question }} {{ list.score }}分

                        <a style="color: red" href="#" @click.prevent="deleteQuestion(1, list.questionId)">删除</a>

                      </li>

                    </ul>

                  </div>

                </el-collapse-item>

                <el-collapse-item>

                  <template slot="title">

                    <div class="titlei">

                      填空题 (共{{ topicCount[1] }}题 共计{{ score[1] }}分)

                    </div>

                  </template>

                  <div class="contenti">

                    <ul

                      class="question"

                      v-for="(list, index) in topic[2]"

                      :key="index"

                    >

                      <li>

                        {{ topicCount[0] + index + 1 }}.{{ list.question }}

                        {{ list.score }}分

                        <a style="color: red" href="#" @click.prevent="deleteQuestion(2, list.questionId)">删除</a>

                      </li>

                    </ul>

                  </div>

                </el-collapse-item>

                <el-collapse-item>

                  <template slot="title">

                    <div class="titlei">

                      判断题 (共{{ topicCount[2] }}题 共计{{ score[2] }}分)

                    </div>

                  </template>

                  <div class="contenti">

                    <ul

                      class="question"

                      v-for="(list, index) in topic[3]"

                      :key="index"

                    >

                      <li>

                        {{ topicCount[0] + topicCount[1] + index + 1 }}.

                        {{ list.question }} {{ list.score }}分

                        <a style="color: red" href="#" @click.prevent="deleteQuestion(3, list.questionId)">删除</a>

                      </li>

                    </ul>

                  </div>

                </el-collapse-item>

              </el-collapse>

            </el-collapse-item>

          </el-collapse>

        </div>

      </span>

      <span slot="footer" class="dialog-footer">

        <el-button @click="showExamDetail = false" style="font-size: 20px;"> 取 消</el-button>

      </span>

    </el-dialog>

  </div>

</template>



## 6.2.6 题库管理页面

<template>

  <div class="exam">

    <el-form :inline="true" class="demo-form-inline">

      <el-form-item label="试卷名称">

        <el-input v-model="subject" placeholder="试卷名称"></el-input>

      </el-form-item>

      <el-form-item label="章节">

        <el-input v-model="section" placeholder="章节"></el-input>

      </el-form-item>

      <el-form-item label="题目内容">

        <el-input v-model="question" placeholder="题目内容"></el-input>

      </el-form-item><br>

      <el-form-item>

        <el-button type="primary" @click="getAnswerInfo"  style="font-size: 20px;"> 查询</el-button>

      </el-form-item>

    </el-form>

    <el-table :data="pagination.records" border :row-class-name="tableRowClassName">

      <el-table-column fixed="left" prop="subject" label="试卷名称" width="180"></el-table-column>

      <el-table-column prop="question" label="题目信息" width="490"></el-table-column>

      <el-table-column prop="section" label="所属章节" width="200"></el-table-column>

      <el-table-column prop="type" label="题目类型" width="200"></el-table-column>

      <el-table-column prop="score" label="试题分数" width="150"></el-table-column>

      <el-table-column prop="level" label="难度等级" width="133"></el-table-column>

      <el-table-column fixed="right" label="操作" width="160">

        <template slot-scope="scope">

          <el-button @click="toEdit(scope.row.type, scope.row.questionId)" type="primary" > 编辑</el-button>

        </template>

      </el-table-column>

    </el-table>

    <el-pagination

      @size-change="handleSizeChange"

      @current-change="handleCurrentChange"

      :current-page="pagination.current"

      :page-sizes="[6, 10]"

      :page-size="pagination.size"

      layout="total, sizes, prev, pager, next, jumper"

      :total="pagination.total"

      class="page"

    ></el-pagination>

  </div>

</template>



1. [↑](#footnote-ref-0)
2. [↑](#footnote-ref-1)
3. [↑](#footnote-ref-2)