

**《基于AJAX技术的多层架构软件开发》**

**项目设计**

|  |  |
| --- | --- |
| **题目：** | **在线考试系统** |

|  |  |
| --- | --- |
| **专业班级:** | **2020级软件工程2班** |
| **学生姓名:** | **钟 涛** |
| **学生学号:** | **202002150402** |
| **指导教师:** | **张 淑 坤** |
| **所属学院:** | **大数据与计算机学院** |

**2023年 5 月**

目录

[一、在线考试系统的需求分析 3](#_Toc136174222)

[(一)、系统设计 3](#_Toc136174223)

[1. 使用技术 3](#_Toc136174224)

[2. 项目结构 3](#_Toc136174225)

[3. 设计思想 5](#_Toc136174226)

[4. 在线考试系统的主要实现功能要求 6](#_Toc136174227)

[(二)、数据库的设计 11](#_Toc136174228)

[1. 概念设计 11](#_Toc136174229)

[2. 逻辑结构设计 12](#_Toc136174230)

[3. 系统数据库表的实现 16](#_Toc136174231)

[(三)、建立功能模型 20](#_Toc136174232)

[二、功能的实现 21](#_Toc136174233)

[（一） 项目介绍 21](#_Toc136174234)

[（二） 登录系统的功能 22](#_Toc136174235)

[（三） 管理员模块 24](#_Toc136174236)

[（四）教师模块 38](#_Toc136174237)

[（五）学生模块 39](#_Toc136174238)

[三、项目总结 44](#_Toc136174239)

[四、参考文献 45](#_Toc136174240)

# 在线考试系统的需求分析

在线考试系统主要包括系统的注册、在线的考试系统、评分系统、在线的管理系统和在自动阅卷系统等。

首先是注册功能，虽然在本系统中没有复杂的注册功能，也不需要用户的详细信息，但

是一定的注册功能对系统的安全还是必要的。

其次是在线的考试系统，这里需要考虑一下的主要是怎样出题、怎样收集答案、怎样保存，怎样判分这些问题。

在线的账号管理系统比较简单，就是要对数据库中的系统用户和考试用户进行管理，而

管理的项目则包括浏览整个系统的用户，添加、删除用户，以及更改用户的密码。

在线的后台系统是对在线考试系统的补充，因为我们的在线系统是自动改卷判分的，

但是为了判分的准确性，以及对用户的答题情况进行详细地了解，对用户的试卷应该进行判定。教师可通过图表展示学生考试分数折线图。

## (一)、系统设计

### 1. 使用技术

开发工具：IDEA2023.1、VSCode、Navicat16、Postman

开发环境：Java（JDK8）、Mysql（8.0.33）

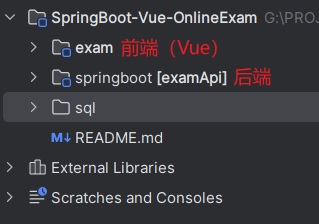
使用技术:

管理：Git（分布式管理）、Maven（后端版本管理）、webpack(前端工程化)

后端框架：SpringBoot2，Spring、SpringMVC、Mybatis、MybatisPlus、Druid

前端：Vue2、Vuex(3)、Element-UI、axios、echarts（可视化图表）

### 2. 项目结构



#### 后端项目结构：

后端采用Springboot+MybatisPlus快速开发接口，不必写繁琐Sql语句以及配置文件。后端开发没有一点技术难度，重点是前端的开发，前端的渲染以及接口处理。

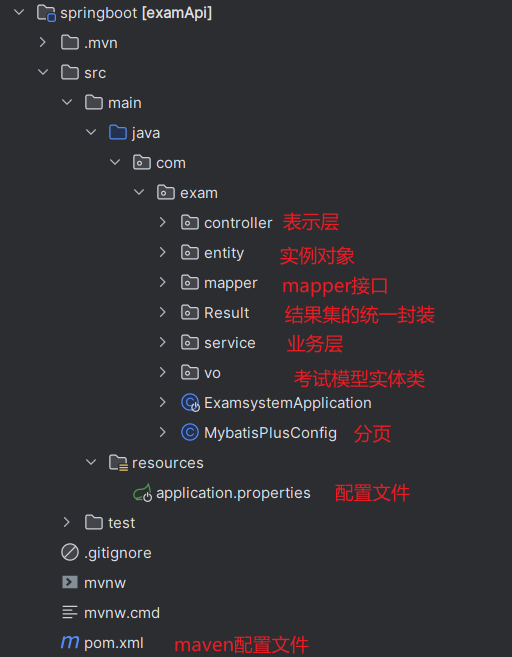
前端采用Element-UI进行页面快速开发。Vue进行页面操作。

包的编写，Mapper层只完成了针对表的相关操作包括写了接口方法和sql语句，并没有编写逻辑的代码，例如对多个Mapper层方法的拼接，逻辑都需要在Service层完成。Mapper就是数据访问的缩写，它只进行数据的访问操作。

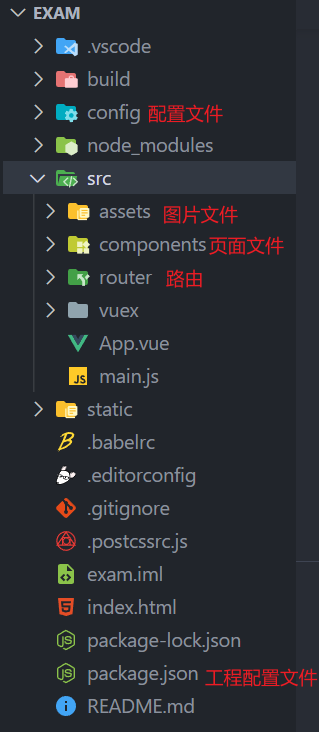
这里有几个常用的包，controller（控制器）能够接受http请求然后在java程序中处理。然后是exception（异常处理包）当用户处理程序中报错时，就会调用异常去处理，跳转到异常界面。Result（结果集包）对结果的统一封装，与前端统一好数据结构，然后是entity实体类。VO里面有个试卷模型类，与题目模型类，service就是处理程序的接口service接口设计。

业务接口:站在使用者(程序员)的角度设计接口。三个方面:1.方法定义粒度，方法定义的要非常清楚；2.参数，要越简练越好；3.返回类型(return 类型一定要友好/或者return异常，我们允许的异常)。Service可以调用Mapper层，控制事务，实现业务。

然后resources资源文件夹放置配置好的propertis文件，这些文件都是起到中间枢纽的作用，调用接口与后台程序的作用，mybatis调用sql 语句，log4j起到单元测试的作用。Mysql就是数据库连接。如下就是包：



#### 前端项目结构



### 3. 设计思想

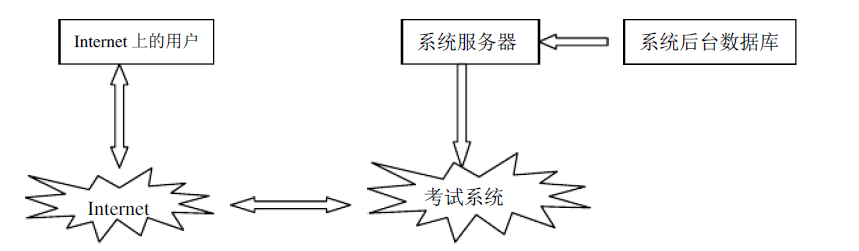
在线考试系统充分利用了Brower/Server 结构的特点，实现了将考试系统移植到Internet的功能，该系统的优势在于极大的简化了老师出题的工作和学生考试的试题，做到了出题的多样性。

目前比较流行的两种网络开发模式分别是Client/Server 模式与Brower/Server 模式。

Client/Server 模式主要是在以局域网为基础的环境下展开应用的，它受到地域的限制。而

Brower/Server 模式样通过Internte 进行通信，可以不受地域的限制，但是它不能够进行联机事务处理，并县城在大量数据处理的情况下，速度较慢。从目前的开发技术来看，Brower 端作为信息收集源，特别是大量的数据录入工作还不能完全取代Client 端的用户界面。在这方面，Client 端的各种开发工具的功能更加强大也更加灵活，而Browser 端却由于其功能结构的限定过于“瘦小”。

对于本在线考试系统，其功能目标是实现将现有的纸张答题式考试向基于Internet 的无纸张考试系统的转变，所以它必须实现整个考试流程的进行，实现对系统的账户不同管理以及完成阅卷、评分这些环节的处理。由于系统数据收集的实时性要求并不高，因此采用Browser/Server 结构来进行设计是合理可行的。图3-1 是本系统的网络应用原理示意图。



**图1-1**

### 4. 在线考试系统的主要实现功能要求

在线考试系统提高了考试的可靠性 、有效性和工作 效率 ，降低了考试成 本 ,顺 应了社 会 的 网络化 趋 势 ,必 将 成 为 一种 不 可 或 缺 的 考 试 方 式 。 本文研究的目的是 开 发 一 个 易于管理和维护的面向教学的考试系统，具有一定的通用性，能够满足多门课程的测试与考核要求。为教师开展平时考核及期末考核提供一个考核平台。

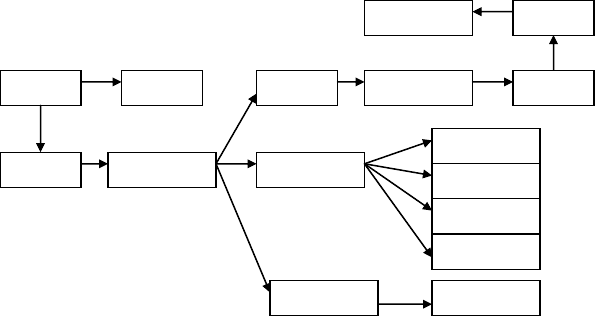
为了满足系统对先进性、安全性、跨平台性、可扩展性、可移植性、分布式等方面的要求，系统总体架构设计采用先进的基于 B/S的三层体系结构：用户层、业务逻辑层、数据库层。要求建立后台数据库，内含一个试题样库，教师可以在题库中添加、编辑和删除试题；教师可以增加新的考试科目；可以充分利用题库中的试题由系统随机组卷，并能在指定时间内发布试卷；学生参加完考试能立即得到答案，并分析考试中答错的题目；教师能得到学生考试后传送到系统的反馈信息，可方便的对成绩进行查询和分析，同时对学生考试中出现的错误进行统计以便今后的教学。此外，还应具有良好的用户界面，操作方便，尽量减少误操作率，具有安全保密机制。

在网络技术逐渐渗入社会生活各个层面的今天随着科学技术的不断发展，考试的手段和媒介也在发生着革命性的变化，从传统的纸笔考试到计算机辅助考试，其实现方法由单机形式逐步向不同时间、不同地点的网络考试方式发展，基于网络的计算机辅助考试系统以其特有的优势在教育中扮演着十分重要的角色在线考试系统是利用计算机技术，对试题库进行存储管理，并由计算机完成智能组卷、标准化测试、试卷评阅等工作，以取代传统的基于纸和笔的考试方式主要体现在：出卷方便快捷、阅卷准确快速、成绩分析统计科学直观；通过题库系统，只要输入考卷的难度级别、待测试知识点及其它相关条件，教师就可以很容易地（自动或半自动）组织出标准考卷；通过计算机阅卷（包括操作过程的自动跟踪、分析和评测），可以大大减轻教师阅卷的工作量，同时大幅度提高阅卷的准确性；成绩分析统计由计算机自动统计由计算机自动完成，可以方便地进行学生成绩查询、打印、统计和分析等处理统计由计算机自动完成，可以方便地进行学生成绩查询、打印、统计和分析等处理。

在线考试可实现真正意义上的无纸化考试，不仅是建设节约社会的需

要，同时在线考试还具有快捷、方便、高效、安全等许多优点，考试

的信度、效度都可大大提高。



成绩查询页面

判分

页面

欢迎页面

注册页面

学生试题选择

断线记录页面

考试页面

浏览用户页面

登录页面

用户登录页面

管理登录页面

添加用户页面

删除用户页面

更改密码页面

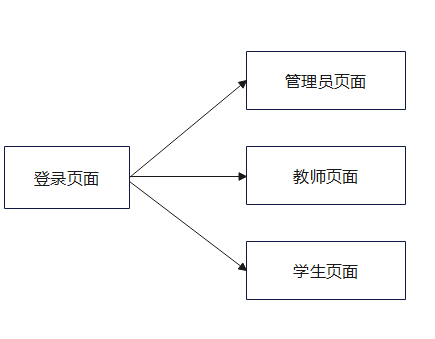
教师页面

试卷管理页面

**图3-1 系统的页面逻辑结构示意图**

#### 4.1. 登录系统的功能

* 欢迎页面
* 用户登录页面
* 不同用户的登录及密码验证



**图2.1**

#### 4.2 管理员模块

**（1）考试管理**

介绍：老师发布了考试,学生才可以在主页面看到相应的考试信息。有考试安排表以后,才能给该次考试添加题目,对应数据表是exammanage。 该表保存该次考试,课程名称,考试时间,所属专业,学院等等信息。

考试查询：可查询所有考试的信息，以及编辑试卷的信息，可删除、修改、通过考试名称进行模糊查询，精确查询

添加考试：填入试卷名称、介绍、所属学院、所属专业、年纪、考试日期、考试时长、总分，考试类型等等，可以添加考试信息。

（2）题库管理

介绍：题库表设计和普通数据表设计有所区别。 分为了三张表,分别是选择题题库表,填空题题库表,判断题题库表, 每个表保存相应类型的题库,通过一张中间表,将题库和试题关联起来。 这样就组成了一张完整的试卷。

题库查询：可查询所有题目的信息，不可删除、修改，由教师端进行删除、修改题目信息。

添加题库（考试题目信息）：可添加每科考试的题目。

（3）成绩查询

学生成绩查询：可查询所有学生的分数信息，学生姓名进行模糊查询，精确查询

成绩分段查询：可查询各科考试的总体分数情况，可通过考试名称进行模糊查询，精确查询。

（4）学生管理

学生信息管理：可查询所有学生的信息，以及编辑学生的信息，可删除、修改、通过学生的名字进行模糊查询，精确查询

添加学生：填入学生姓名、性别、学院、所在专业、年级班级、电话号码、身份证、邮箱等等。可添加学生信息。

（5）教师管理

教师信息管理：可查询所有教师的信息，以及编辑教师的信息，可删除、修改、通过教师的名字进行模糊查询，精确查询

添加教师：填入教师姓名、性别、学院、所在专业、年级班级、电话号码、身份证、邮箱等等。可添加教师信息。

#### 4.3 教师模块

教师模块与管理员模块类似，但是管理员有能编辑教师信息教师模块没有，并且有最高权限对各类信息进行操作。教师只能对自己教授的课程进行考试操作。

（1）考试管理

介绍：老师发布了考试,学生才可以在主页面看到相应的考试信息。有考试安排表以后,才能给该次考试添加题目,对应数据表是exammanage。 该表保存该次考试,课程名称,考试时间,所属专业,学院等等信息。

试卷查询：可查询所有考试的信息，以及编辑试卷的信息，可删除、修改、通过考试名称进行模糊查询，精确查询

添加考试：填入试卷名称、介绍、所属学院、所属专业、年纪、考试日期、考试时长、总分，考试类型等等，可以添加考试信息。

（2）题库管理

介绍：题库表设计和普通数据表设计有所区别。 分为了三张表,分别是选择题题库表,填空题题库表,判断题题库表, 每个表保存相应类型的题库,通过一张中间表,将题库和试题关联起来。 这样就组成了一张完整的试卷。

题库查询：可查询任教题目的信息，可删除、修改，通过试卷名称进行模糊查询、精确查询。

添加题库（考试题目信息）：可添加每科考试的题目。

（3）成绩查询

学生成绩查询：可查询所有学生的分数信息，学生姓名进行模糊查询，精确查询

成绩分段查询：可查询各科考试的总体分数情况，可通过考试名称进行模糊查询，精确查询。

（4）学生管理

学生信息管理：可查询所有学生的信息，以及编辑学生的信息，可删除、修改、通过学生的名字进行模糊查询，精确查询

添加学生：填入学生姓名、性别、学院、所在专业、年级班级、电话号码、身份证、邮箱等等。可添加学生信息。

#### 4.4 学生模块

（1）我的试卷

教师或者管理员发布考试，学生端会在页面展示试卷信息，以及保存历史考试信息。学生可点击相应的试卷进行考试，考试完系统会自动为选择题，判断题、以及有固定答案的填空题进行评分。

（2）我的练习

每次考试过后，会保存试卷信息，学生可通过保存的试卷进行复习以及练习，系统会自动评分，但不保存分数。

（3）我的分数

学生可查看考试分数

（4）我的留言

一个简单的评论区功能，不同的学生用户可以进行留言

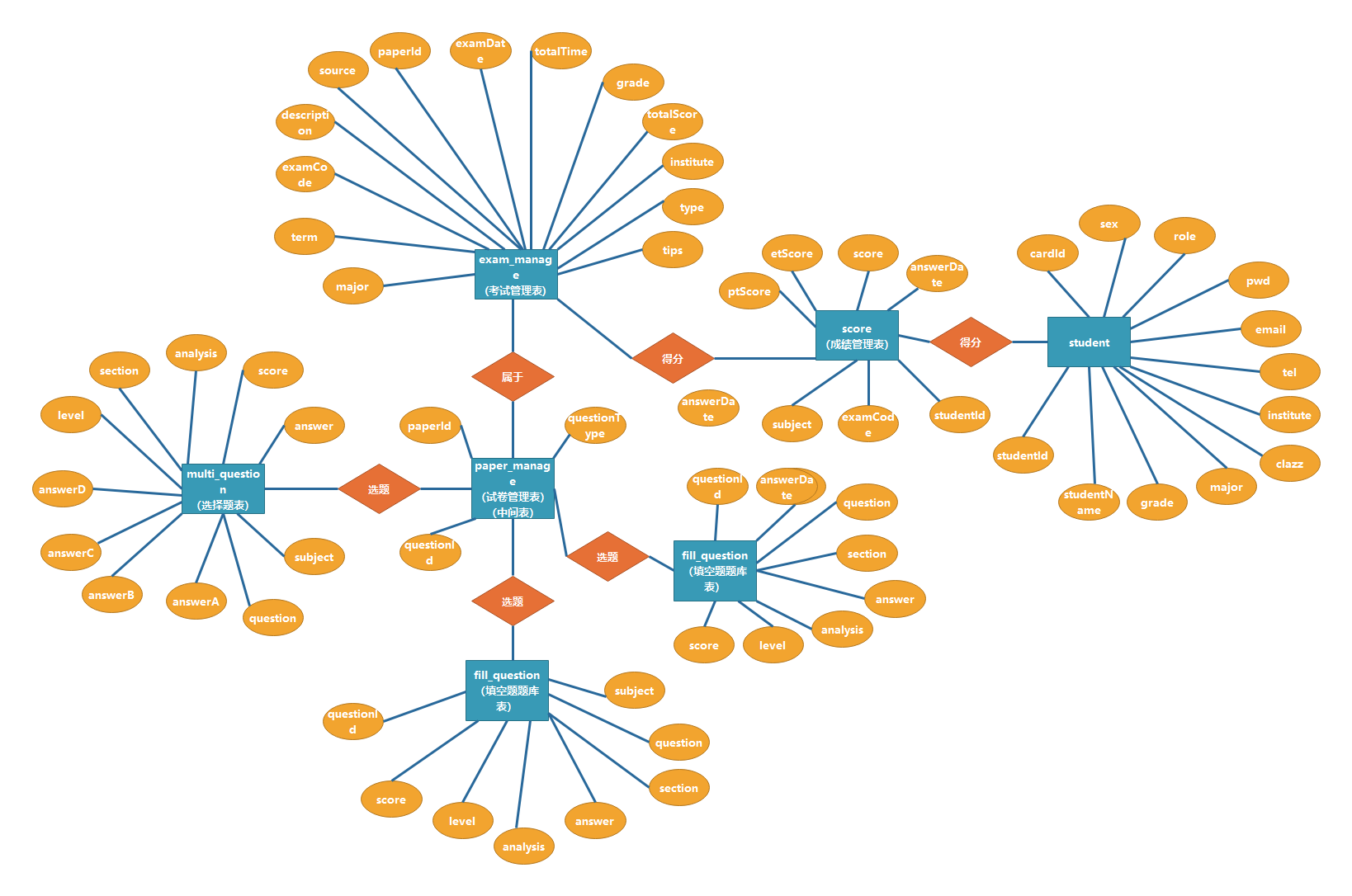
## (二)、数据库的设计

根据系统功能设计的要求以及功能模块的划分，对于在线考试系统的试题库数据库，可以列出以下数据项和数据结构。

### 1. 概念设计

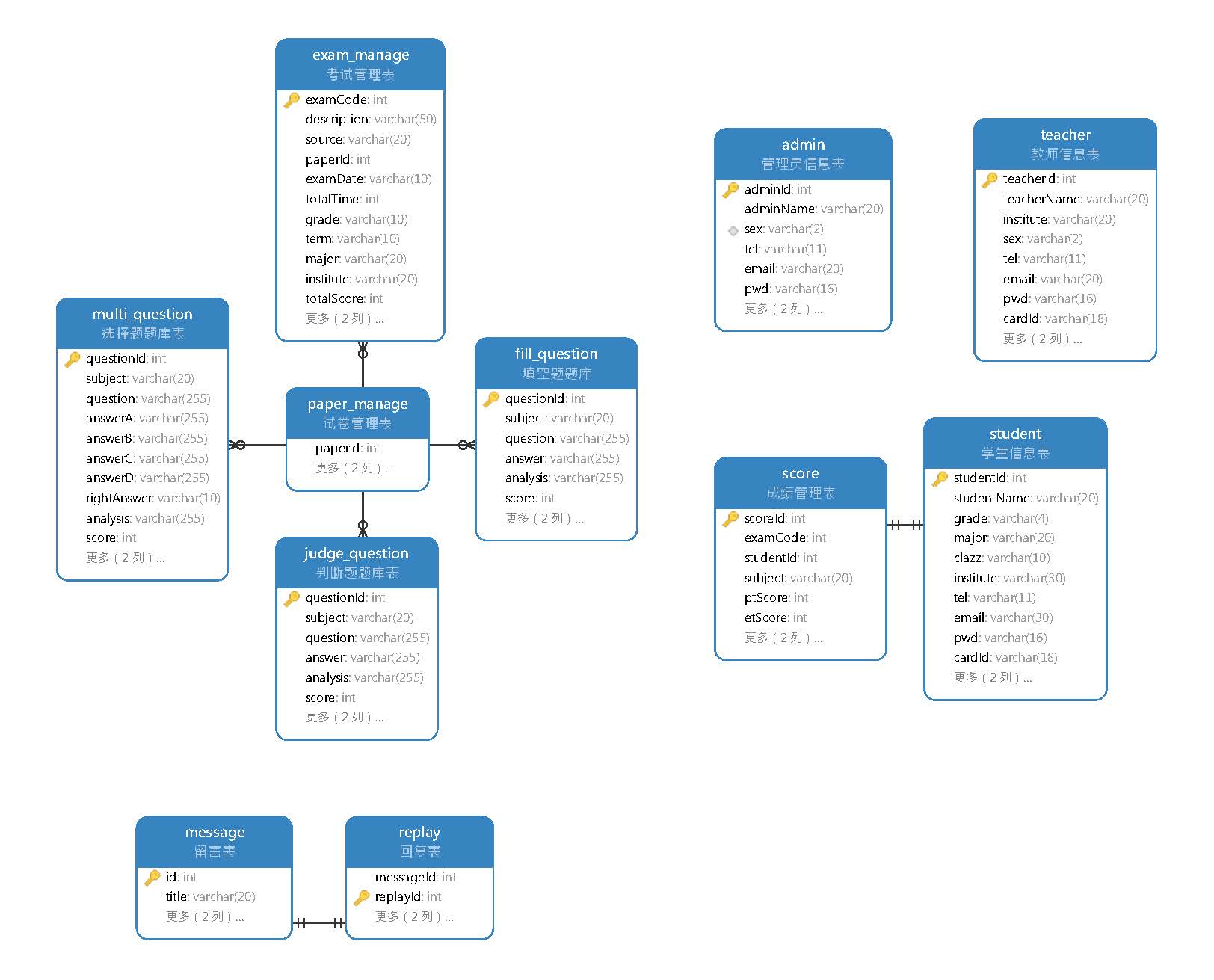
#### 1.1 E-R图

总体E-R图



#### 1.2 物理模型：

整体关系图：



### 2. 逻辑结构设计

对于用户记录数据，主要包括了如下的一些数据项和数据结构：

**管理员信息表（admin）：**主要包括ID号，姓名，性别，电话号码，电子邮箱，密码，身份证号码，角色(0管理员，1教师，2学生)等等。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 主键 | 是否为null | 注释 |
| adminId | int() | 是 | NOT NULL | ID号 |
| adminName | varchar(20) | 否 | NULL | 姓名 |
| sex | varchar(2) | 否 | NULL | 性别 |
| tel | varchar(11) | 否 | NULL | 电话号码 |
| email | varchar(20) | 否 | NULL | 电子邮箱 |
| pwd | varchar(16) | 否 | NULL | 密码 |
| cardId | varchar(18) | 否 | NULL | 身份证号 |
| role | varchar(1) | 否 | NULL | 角色(0管理员，1教师，2学生) |

**学生信息表（student）：**主要包括ID号，姓名，性别，年级，专业，班级，学院，电话号码，电子邮箱，密码，身份证号码，角色(0管理员，1教师，2学生)等等。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 主键 | 是否为null | 注释 |
| studentId | int | 是 | NOT NULL | ID |
| studentName | varchar(20) | 否 | NULL | 姓名 |
| grade | varchar(4) | 否 | NULL | 年级 |
| major | varchar(20) | 否 | NULL | 专业 |
| clazz | varchar(10) | 否 | NULL | 班级 |
| institute | varchar(30) | 否 | NULL | 学院 |
| tel | varchar(11) | 否 | NULL | 电话号码 |
| email | varchar(30) | 否 | NULL | 电子邮件 |
| pwd | varchar(16) | 否 | NULL | 密码 |
| cardId | varchar(18) | 否 | NULL | 身份证号码 |
| sex | varchar(2) | 否 | NULL | 性别 |
| role | varchar(1) | 否 | NULL | 角色(0管理员，1教师，2学生) |

**教师信息表（teacher）**：主要包括ID号，姓名，性别，电话号码，电子邮箱，密码，身份证号码，职称，角色(0管理员，1教师，2学生)等等。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 主键 | 是否为null | 注释 |
| teacherId | int | 是 | NOT NULL | ID |
| teacherName | varchar(20) | 否 | NULL | 姓名 |
| institute | varchar(20) | 否 | NULL | 学院 |
| sex | varchar(2) | 否 | NULL | 性别 |
| tel | varchar(11) | 否 | NULL | 电话号码 |
| email | varchar(20) | 否 | NULL | 邮箱 |
| pwd | varchar(16) | 否 | NULL | 密码 |
| cardId | varchar(18) | 否 | NULL | 身份证号码 |
| type | varchar(20) | 否 | NULL | 职称 |
| role | varchar(1) | 否 | NULL | 角色(0管理员，1教师，2学生) |

**留言表（message）**：留言编号，标题，留言内容，留言时间。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 主键 | 是否为null | 注释 |
| id | int | 是 | NOT NULL | 留言编号 |
| title | varchar(20) | 否 | NULL | 标题 |
| content | varchar(255) | 否 | NULL | 留言内容 |
| time | date | 否 | NULL | 留言时间 |

**回复表：**留言编号，回复编号，内容，回复时间。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 主键 | 是否为null | 注释 |
| messageId | int | 否 | NULL | 留言编号 |
| replayId | int | 是 | NOT NULL | 回复编号 |
| replay | varchar(255) | 否 | NULL | 内容 |
| replayTime | date | 否 | NULL | 回复时间 |

对于在线的考试系统，可以列出以下数据项和数据结构：

值得指出的是试题信息本身包括了3 个表，在本在线考试系统中包括了1个专业的试题表。

**考试管理表（exam\_manage）：**考试编号，考试介绍，课程名称，试卷编号，考试日期，考试时长，年级，学期，专业，学院，总分，考试类型，考试须知，以及一些附加的内容等等。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 主键 | 是否为null | 注释 |
| examCode | int | 是 | NOT NULL | 考试编号 |
| description | varchar(50) | 否 | NULL | 考试介绍 |
| source | varchar(20) | 否 | NULL | 课程名称 |
| paperId | int | 否 | NULL | 考试编号 |
| examDate | varchar(10) | 否 | NULL | 考试日期 |
| totalTime | int | 否 | NULL | 考试时长 |
| grade | varchar(10) | 否 | NULL | 年级 |
| term | varchar(10) | 否 | NULL | 学期 |
| major | varchar(20) | 否 | NULL | 专业 |
| institute | varchar(20) | 否 | NULL | 学院 |
| totalScore | int | 否 | NULL | 总分 |
| type | varchar(255) | 否 | NULL | 考试类型 |
| tips | varchar(255) | 否 | NULL | 考试须知 |

**试卷管理表（paper\_manage）：**主要包括试题的ID、题目编号。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 主键 | 是否为null | 注释 |
| paperId | int | 是 | NOT NULL | 试卷编号 |
| questionType | int | 否 | NULL | 题目类型 |
| questionId | int | 否 | NULL | 题目编号 |

**填空题题库（fill\_question）**：主要包括试题编号、考试科目、试题内容、正确答案、题目解析、分数、难度等级、所属章节等等

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 主键 | 是否为null | 注释 |
| questionId | int | 是 | NOT NULL | 试题编号 |
| subject | varchar(20) | 否 | NULL | 考试科目 |
| question | varchar(255) | 否 | NULL | 考题内容 |
| answer | varchar(255) | 否 | NULL | 正确答案 |
| analysis | varchar(255) | 否 | NULL | 题目解析 |
| score | int | 否 | NULL | 分数 |
| level | varchar(5) | 否 | NULL | 难度等级 |
| section | varchar(20) | 否 | NULL | 所属章节 |

**判断题题库表（judge\_question）：**主要包括试题编号，考试科目，试题内容，正确答案，题目解析，分数，难度等级，所属章节等等。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 主键 | 是否为null | 注释 |
| questionId | int | 是 | NOT NULL | 试题编号 |
| subject | varchar(20) | 否 | NULL | 考试科目 |
| question | varchar(255) | 否 | NULL | 考题内容 |
| answer | varchar(255) | 否 | NULL | 正确答案 |
| analysis | varchar(255) | 否 | NULL | 题目解析 |
| score | int | 否 | NULL | 分数 |
| level | varchar(5) | 否 | NULL | 难度等级 |
| section | varchar(20) | 否 | NULL | 所属章节 |

**选择题题库表（multi\_question）**：主要包括试题编号，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，成绩管理表：主要包括分数编号，考试编号，学号，课程名称，平时成绩，期末成绩，总成绩，答题日期等等

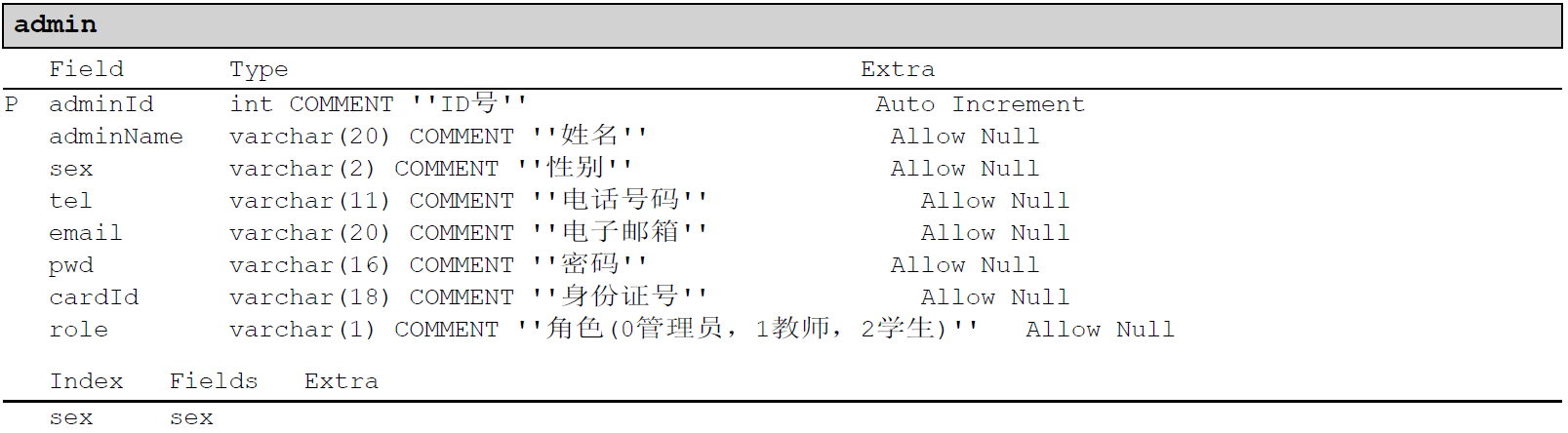
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 主键 | 是否为null | 注释 |
| questionId | int | 是 | NOT NULL | 试题编号 |
| subject | varchar(20) | 否 | NULL | 考试科目 |
| question | varchar(255) | 否 | NULL | 考题内容 |
| answerA | varchar(255) | 否 | NULL | 选项A |
| answerB | varchar(255) | 否 | NULL | 选项B |
| answerC | varchar(255) | 否 | NULL | 选项C |
| answerD | varchar(255) | 否 | NULL | 选项D |
| answer | varchar(255) | 否 | NULL | 正确答案 |
| analysis | varchar(255) | 否 | NULL | 题目解析 |
| score | int | 否 | NULL | 分数 |
| level | varchar(5) | 否 | NULL | 难度等级 |
| section | varchar(20) | 否 | NULL | 所属章节 |

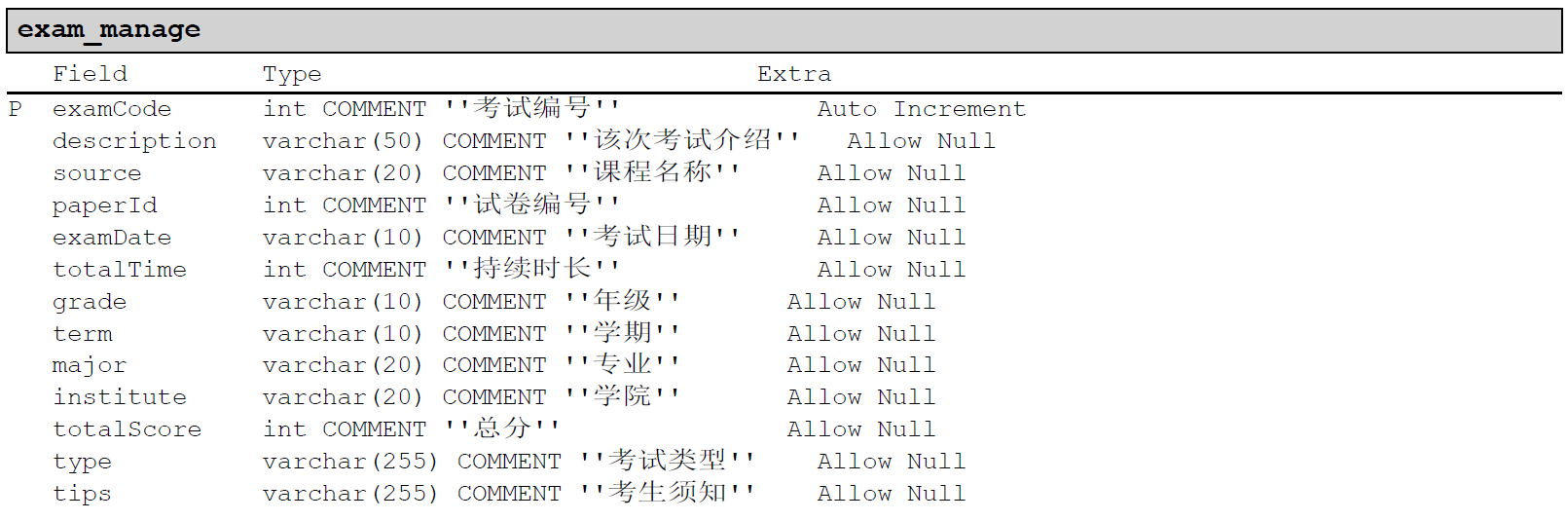
**成绩管理表（score）：**

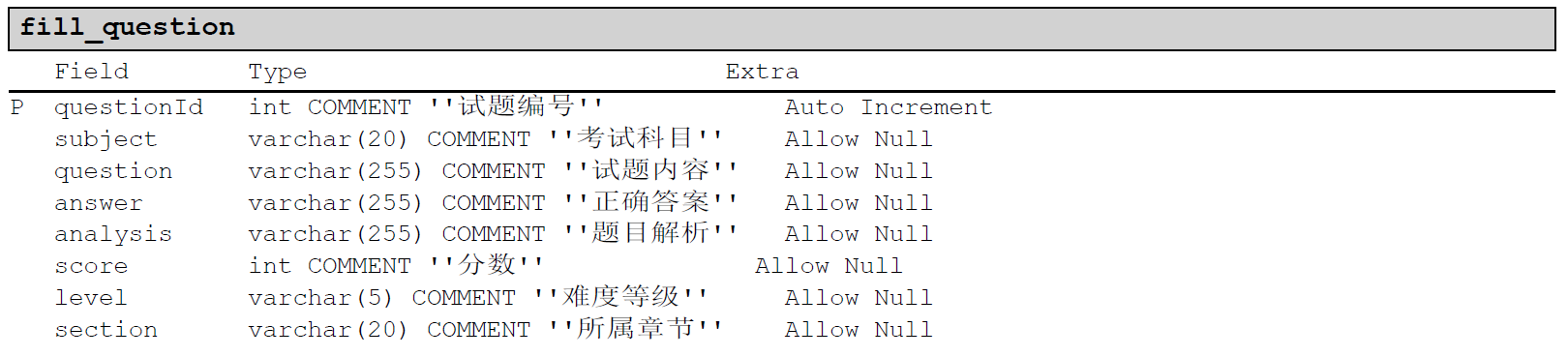
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 主键 | 是否为null | 注释 |
| scoreId | int | 是 | NOT NULL | 分数编号 |
| examCode | int | 否 | NULL | 考试编号 |
| studentId | int | 否 | NULL | 学号 |
| subject | varchar(20) | 否 | NOT NULL | 课程名称 |
| ptScore | int | 否 | NULL | 平时成绩 |
| etScore | int | 否 | NULL | 期末成绩 |
| score | int | 否 | NULL | 总成绩 |
| answerDate | varchar(10) | 否 | NULL | 答题日期 |

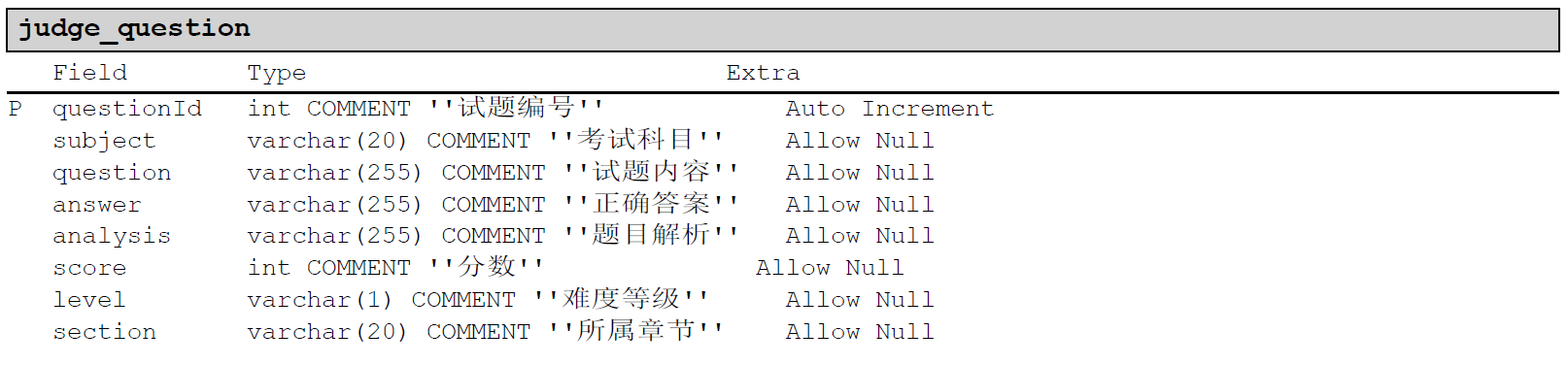
对于在线的试卷管理，因为它主要使用了试卷的一些信息，上面都已经给出了，不再赘述。

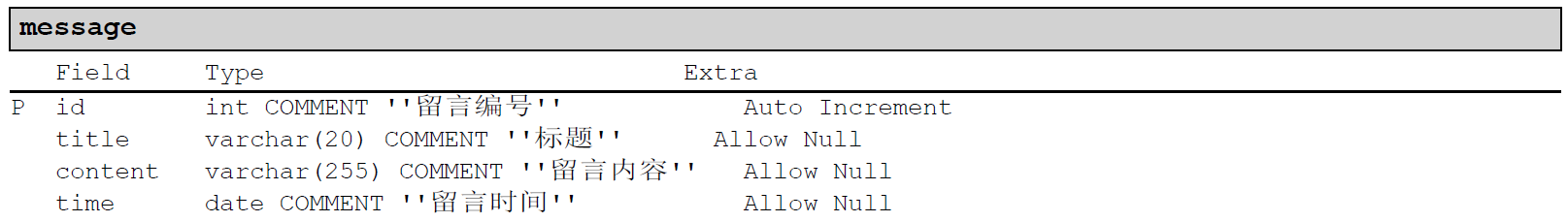
### 3. 系统数据库表的实现

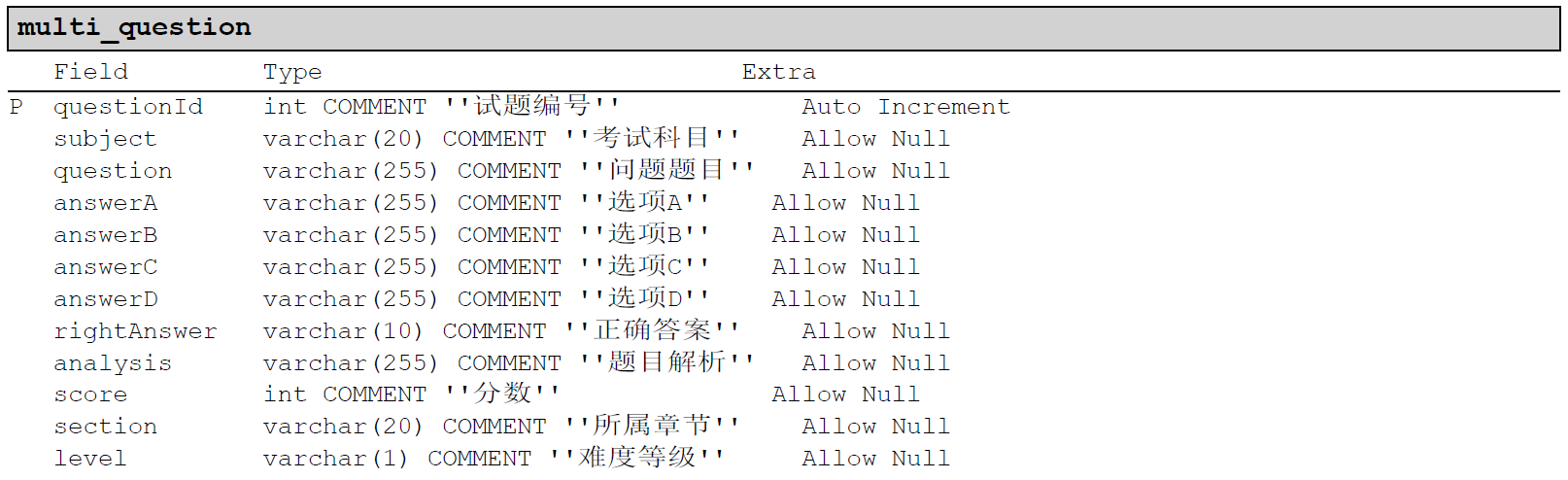




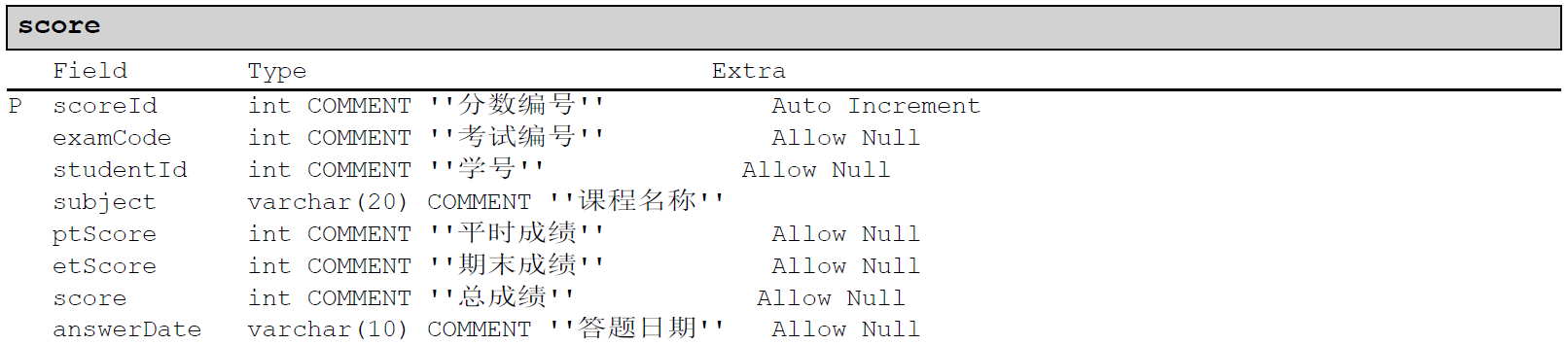


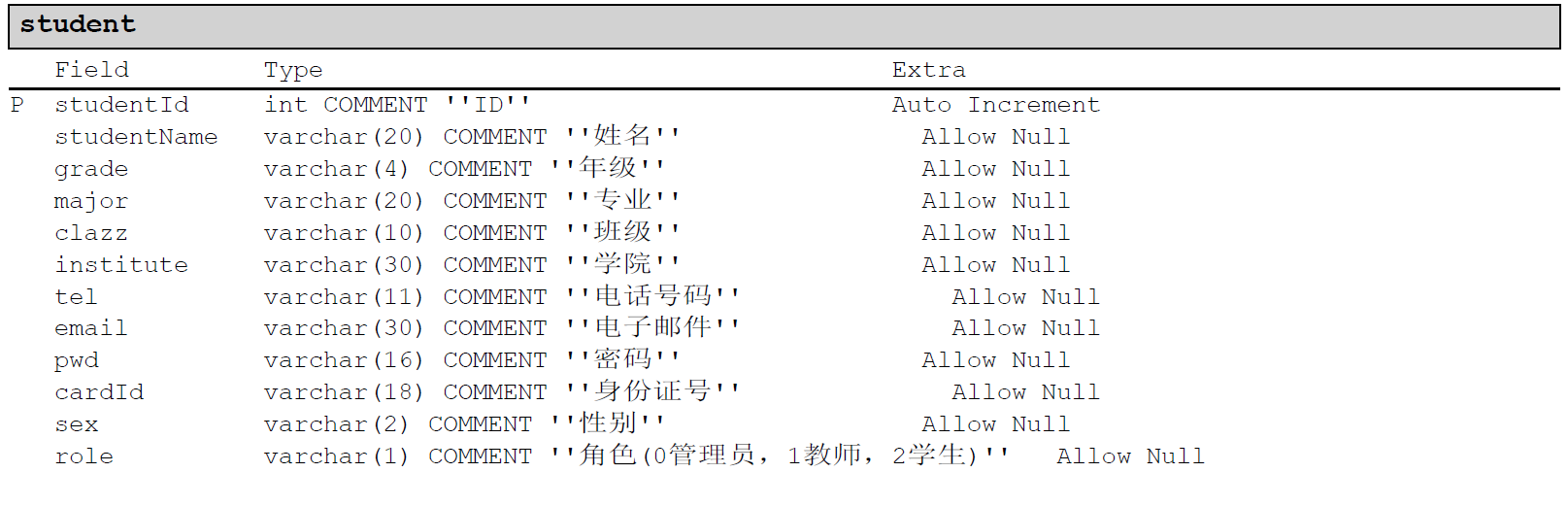














## (三)、建立功能模型

#### 1. 发现参与者

（1）提出问题

谁使用该系统？

谁改变系统的数据？

谁从系统获取信息？

谁需要系统的支持以完成日常工作任务？

谁负责维护、管理并保持系统正常运行？

系统需要应付哪些硬件设备？

系统需要和哪些外部系统交互？

谁对系统运行产生的结果感兴趣？

（2）参与者清单列表

管理员、教师、学生

#### 2. 确定用例

（1）提出问题

参与者需获取何种功能，需要做什么？

参与者需读取、产生、删除、修改或存储系统中什么信息？

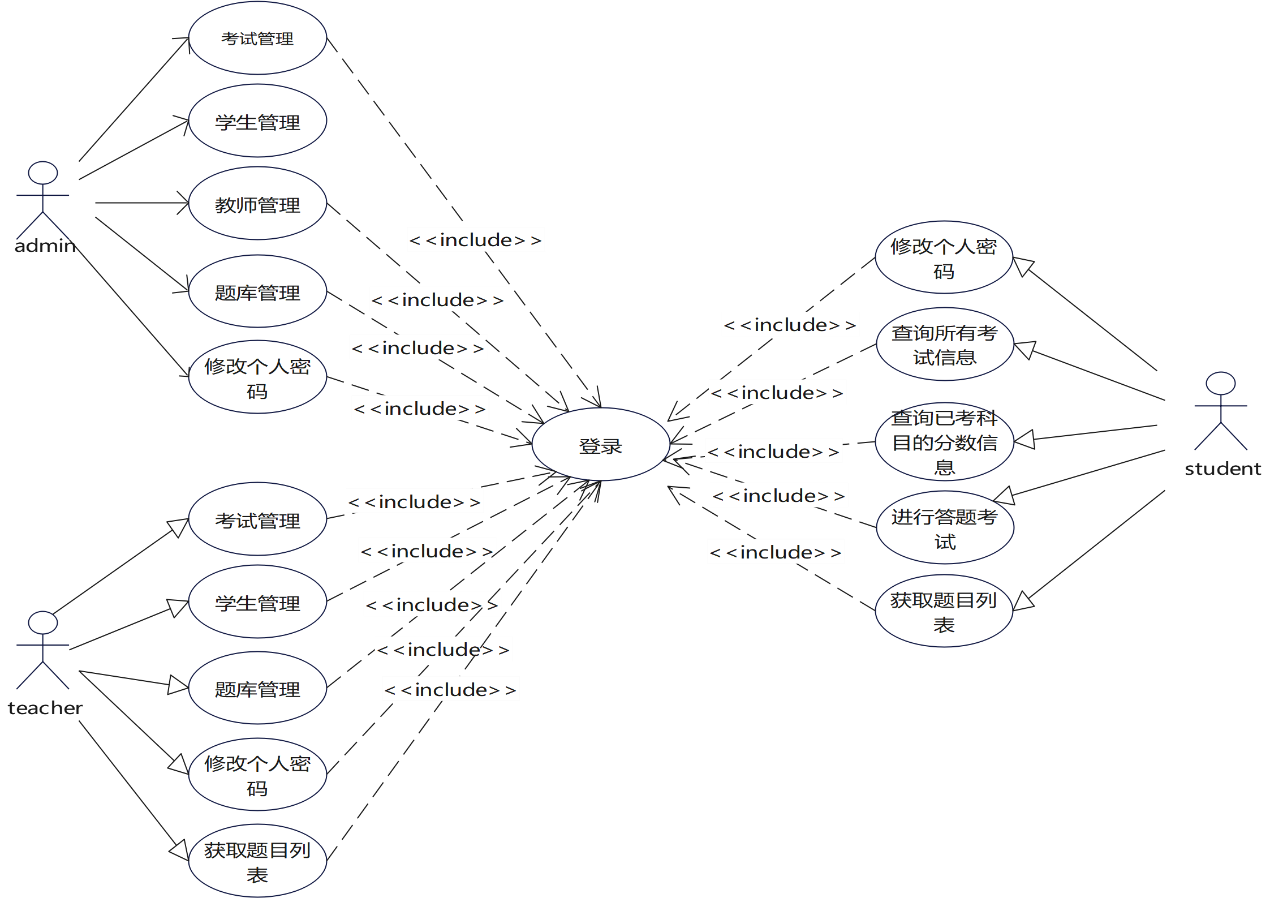
参与者为系统提供什么信息？

系统发生事件和执行者之间是否需要通信？

（2）用例列表

|  |  |  |
| --- | --- | --- |
| 用例名称 | 用例描述 | 预期参与者 |
| 考试管理 | 该用例描述管理员管理考试的信息 | 教师，管理员 |
| 学生管理 | 该用例描述管理员管理学生的信息 | 教师，管理员 |
| 教师管理 | 该用例描述管理员管理教职工的信息 | 管理员 |
| 题库管理 | 该用例描述管理员管理题库的信息 | 教师，管理员 |
| 获取题目列表 | 该用例描述教职工可以通过系统获取题目列表 | 教师，管理员 |
| 查询所有试卷信息 | 该用例描述学生可以通过系统查看试卷的信息 | 学生 |
| 查询已考科目的分数信息 | 该用例描述学生可以通过系统查看已考科目的分数信息 | 学生 |
| 进行答题考试 | 该用例描述学生可以通过系统进行答题考试 | 学生 |
|  |  |  |
| 给学生打分 | 该用例描述可以通过系统自动给学生打分 | 教师 |
| 修改个人密码 | 该用例描述管理员、学生、教职工可以进行修改个人的登录密码 | 管理员、学生、教职工 |

#### 3. 构造总用例图



# 功能的实现

## （一） 项目介绍

后端接口:localhost:8080

前端接口:localhost:8088（访问页面）

前端定义了“api”代替后端访问端口

(Config.index.js)

proxyTable: {

      '/api': {

        target: 'http://localhost:8080',//本地地址

        // target: 'http://gopikachu.top:8080',// 线上部署地址

        changeOrigin: true,

        pathRewrite: {

          '^/api': ''//这里理解成用‘/api’代替target里面的地址，后面组件中我们掉接口时直接用api代替 比如我要调用'http://40.00.100.100:3002/user/add'，直接写‘/api/user/add’即可

        }

      }

    }

项目启动：

后端IDEA：启动boot启动类即可。

前端终端运行:npm i -y （项目下载依赖）npm run start（项目启动）。

主要后端介绍：

类：com.exam.Result.ApiResult<T>——对结果集的统一封装

有三个属性：int code（错误码）、String message（消息）、T data（返回结果的包装）

类com.exam.vo.Item——题目模型

用于创建试卷用的实例对象，这个就是创建考试的类型的关键类之一

private String subject;//考试科目  
private Integer paperId;//考试编号  
private Integer multiNumber;//选择题  
private Integer fillNumber;//填空题  
private Integer judgeNumber;//判断题

类com.exam.vo.AnswerVO——题目模型

用于管理创建题目结构的类

private String question;//问题题目  
private String subject;//考试科目  
private String score;//分数（默认为2）  
private String section;//所属章节  
private String level;//难度等级  
private String type;//类型

其余交予service，controller等包类进行处理

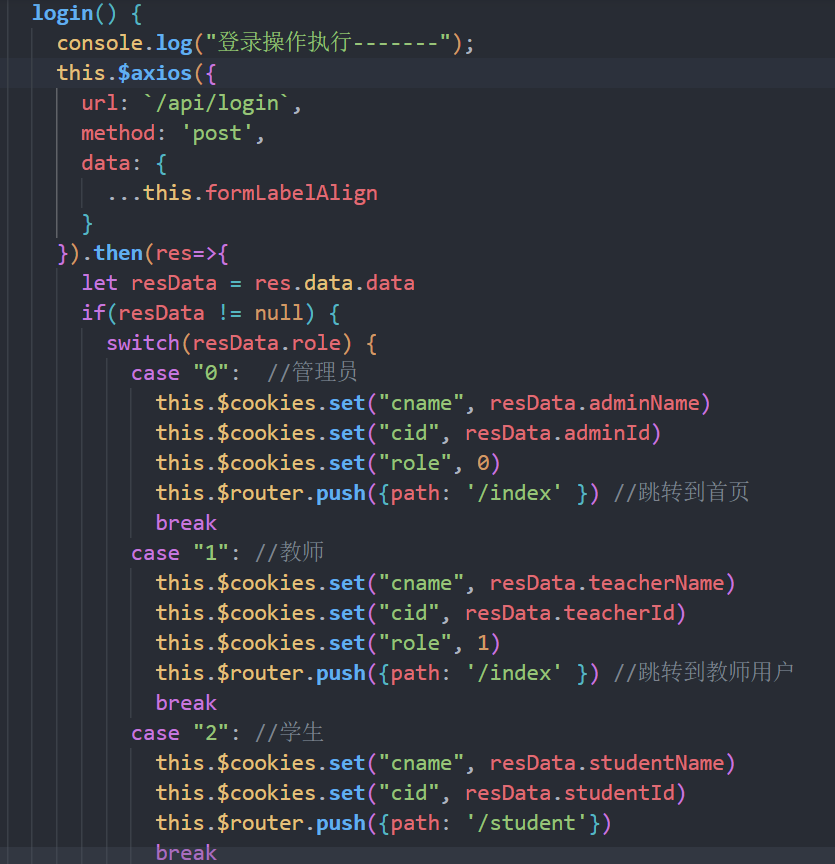
## （二） 登录系统的功能

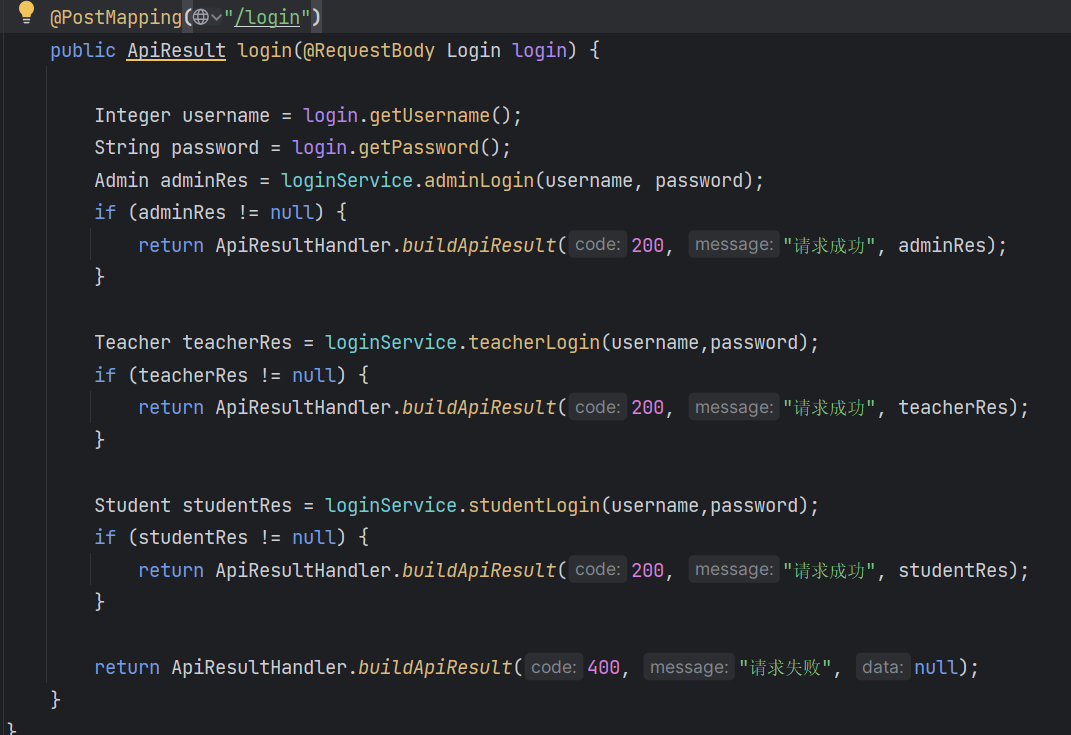
后端接口接口：/api/login

前端访问接口： /#/

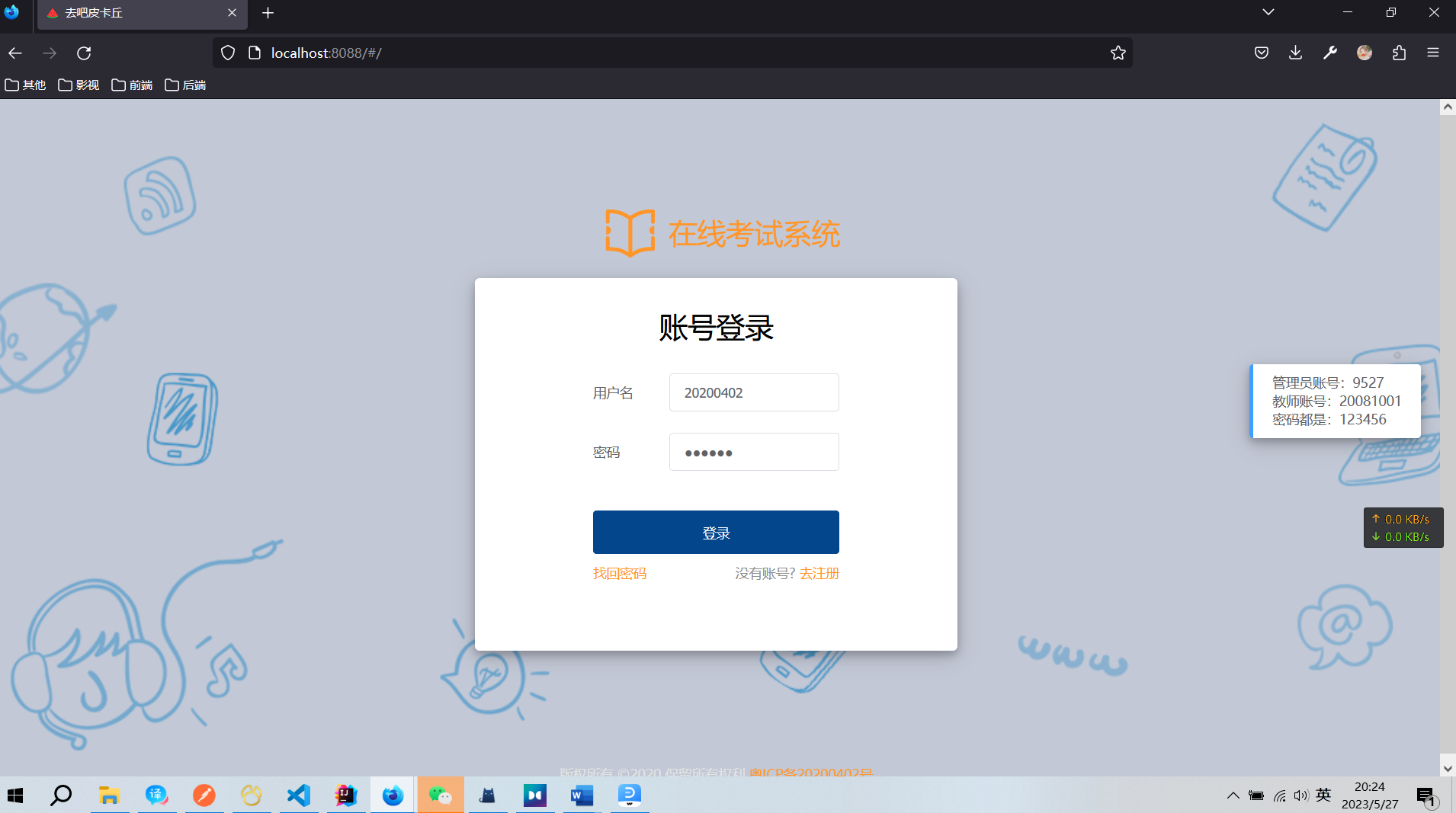
分析：页面未登录的主页，前端进行获取到表单的账号密码信息，交由后端判断输入的信息是否正确，前端进行数据校验。







运行结果：



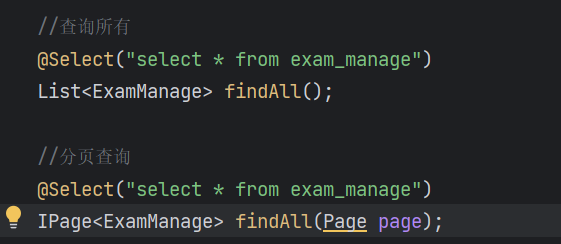
## （三） 管理员模块

#### （1）考试管理

后端接口：/api/exams

查询（精确查询、模糊查询，分页查询）所有考试列表：

用接口查询所有数据返回一个list<>就好了



    //搜索试卷

**search**() {

      this.**$axios**('/api/exams').**then**(res => {

        if(res.data.code == 200) {

          let allExam = res.data.data

          let newPage = allExam.**filter**(item => {

            return item.source.**includes**(this.key)

          })

          this.pagination.records = newPage

        }

      })

    },

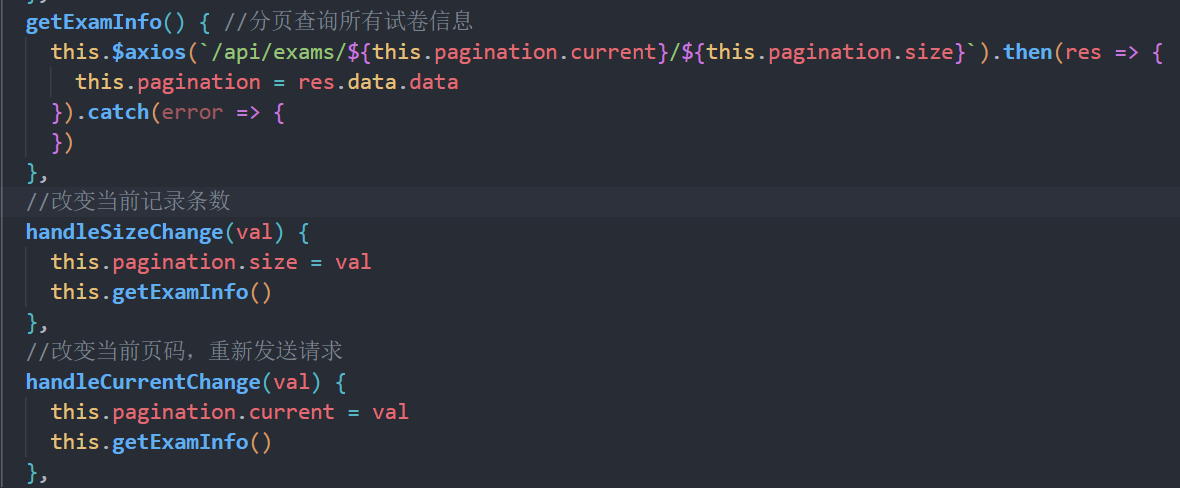
分页查询:

后端接口：/api/exams/{page}/{size}

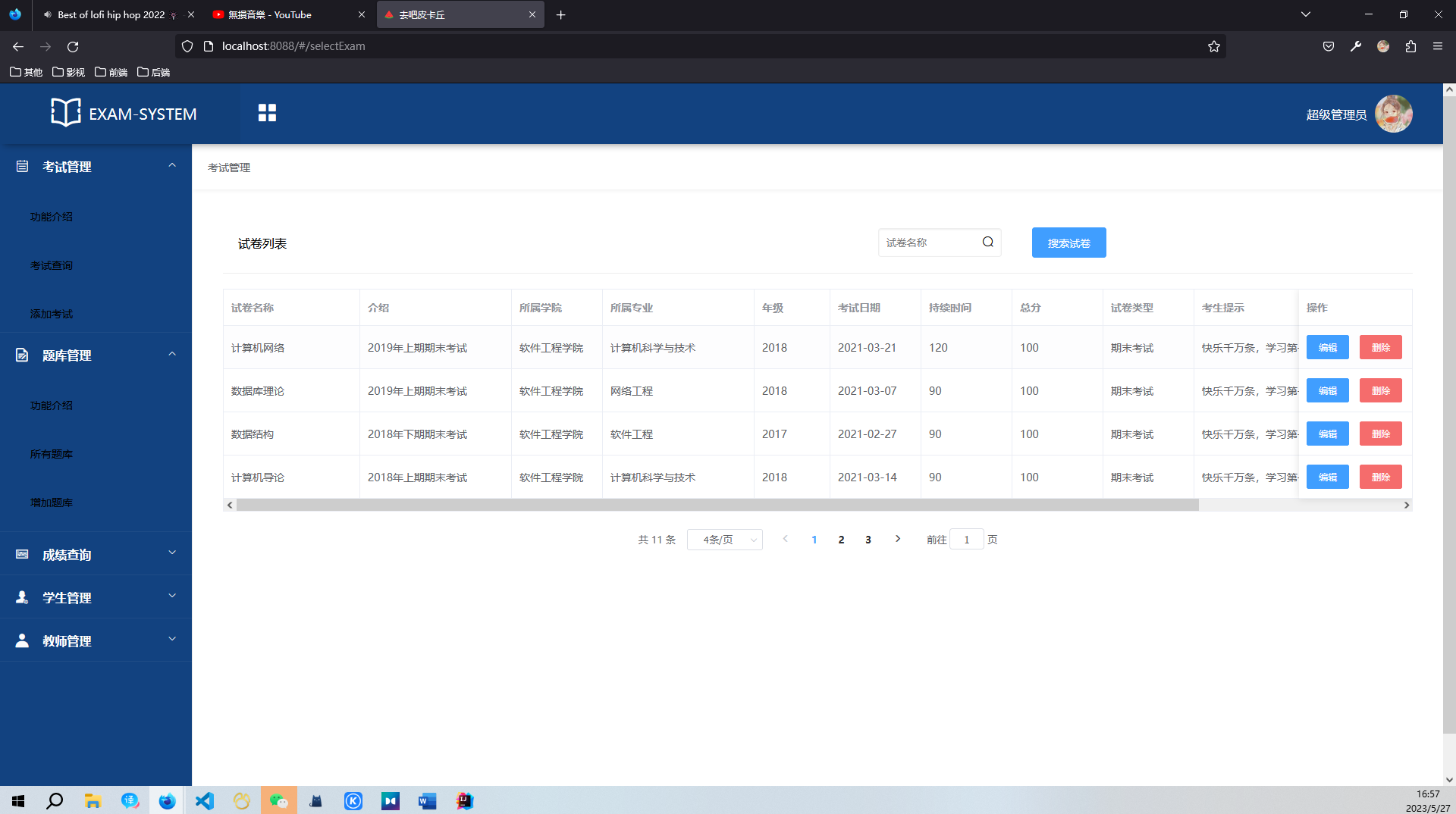
使用MybatisPlus实现

import com.baomidou.mybatisplus.core.metadata.IPage;  
import com.baomidou.mybatisplus.extension.plugins.pagination.Page;

//分页查询  
@Select("select \* from exam\_manage")  
IPage<ExamManage> findAll(Page page);



运行结果：



模糊查询：

使用MybatisPlus实现，直接调用/api/eaxms接口，渲染至页面即可，代码与精确查询类似。



编辑（修改）考试信息：

后端接口：/api/examManagePaperId

返回一个ExamMessage的Jons对象，有输入修改，无输入则不变

@Update("update exam\_manage set description = #{description},source = #{source},paperId = #{paperId}," +  
 "examDate = #{examDate},totalTime = #{totalTime},grade = #{grade},term = #{term}," +  
 "major = #{major},institute = #{institute},totalScore = #{totalScore}," +  
 "type = #{type},tips = #{tips} where examCode = #{examCode}")  
int update(ExamManage exammanage);



运行结果：



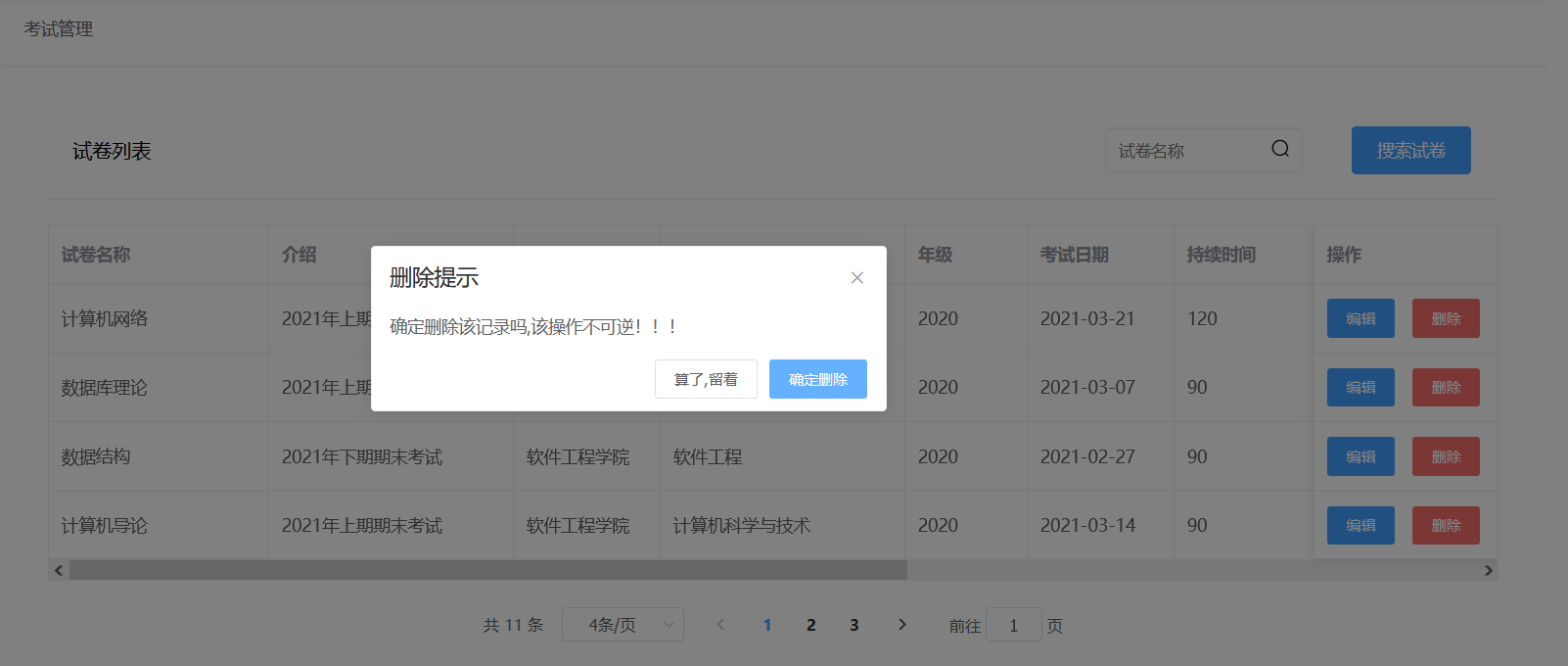
删除考试信息：

后端接口：/api/exams/{examCode}

根据考试的代号（examCode）进行删除整条数据

@Delete("delete from exam\_manage where examCode = #{examCode}")  
int delete(Integer examCode);





添加考试：

后端接口：/api/exams

字段有则加，无则为空

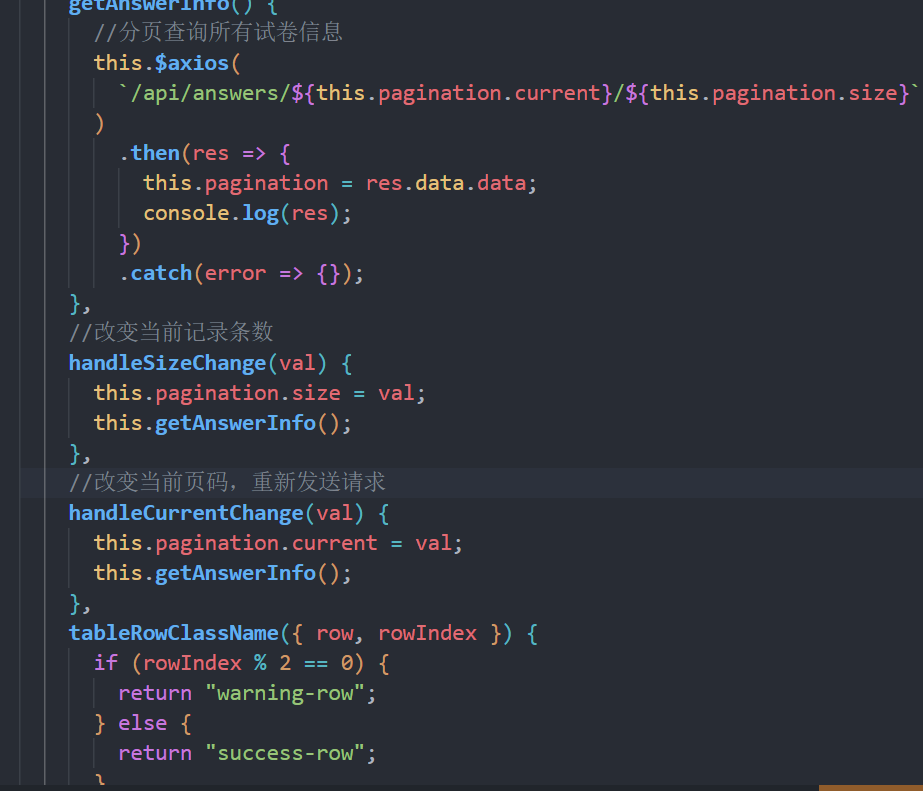
@Options(useGeneratedKeys = true,keyProperty = "examCode")  
@Insert("insert into exam\_manage(description,source,paperId,examDate,totalTime,grade,term,major,institute,totalScore,type,tips)" + "values(#{description},#{source},#{paperId},#{examDate},#{totalTime}"+ "#{grade},#{term},#{major},#{institute},#{totalScore},#{type},#{tips})")  
int add(ExamManage exammanage);



#### （2）题库管理

查询所有题库：

@Select("select question, subject, score, section,level, \"选择题\" as type from multi\_question " +  
 "union select question, subject, score, section,level, \"判断题\" as type from judge\_question " +  
 "union select question, subject, score, section,level, \"填空题\" as type from fill\_question")  
IPage<AnswerVO> findAll(Page page);

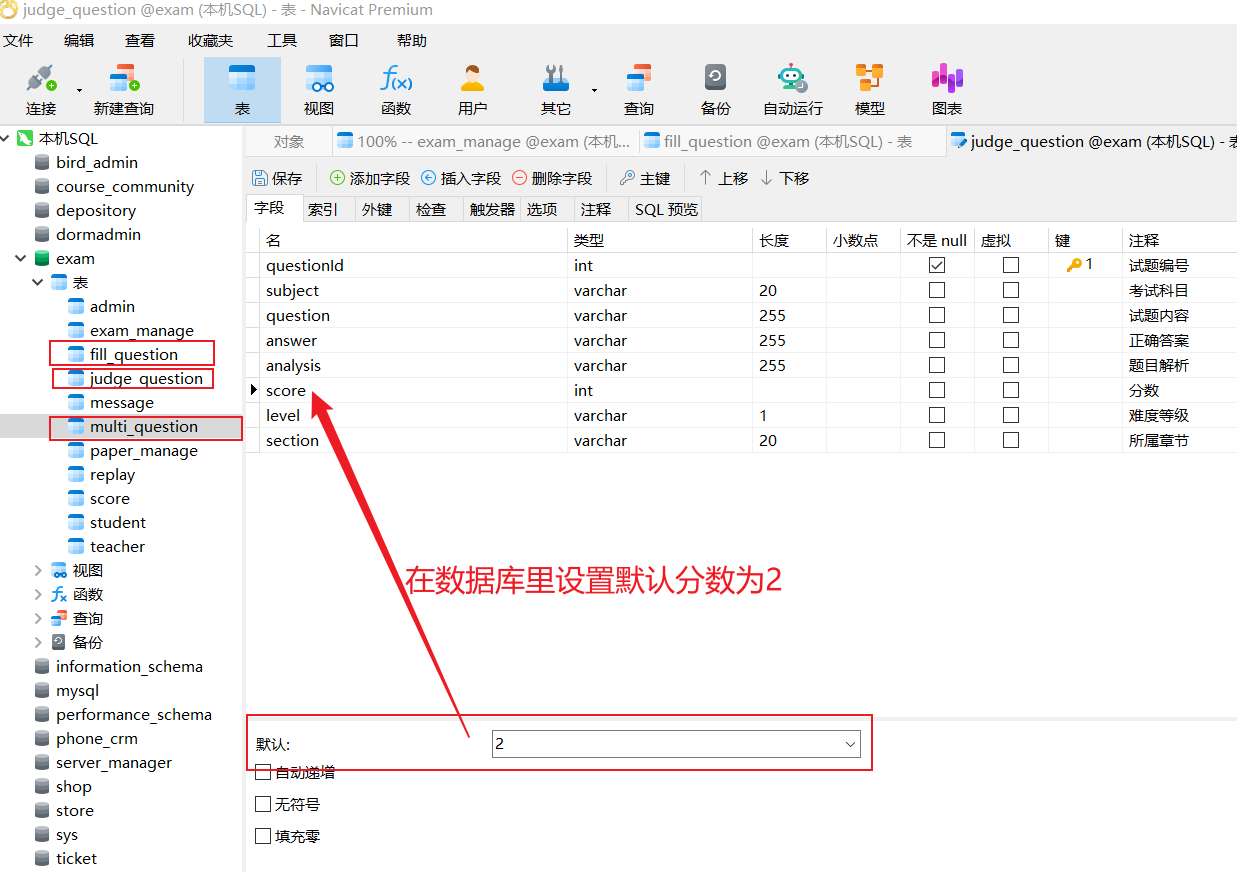


运行结果：



添加题库

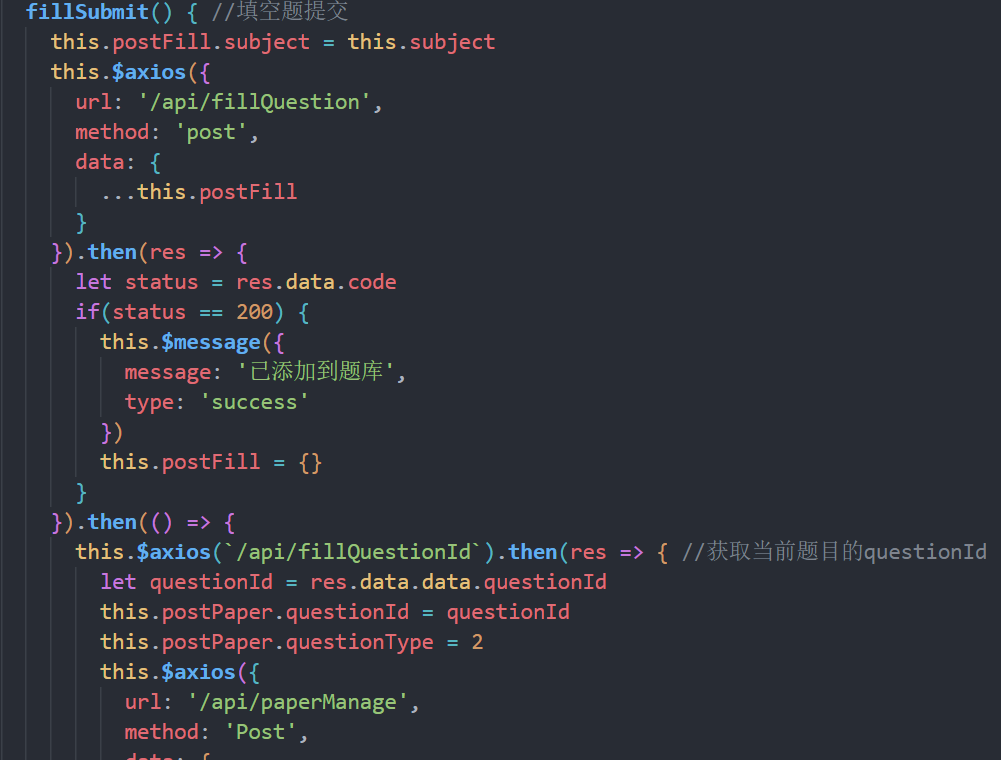
添加新题库在页面上没有开发输入功能，而是在数据库建表的时候加上了一个默认值为“2”为什么呢，因为没时间写还有忽略了这个，我就直接砍了这个功能，以后再加上就好了。系统有存在不合理的地方，以后慢慢完善。



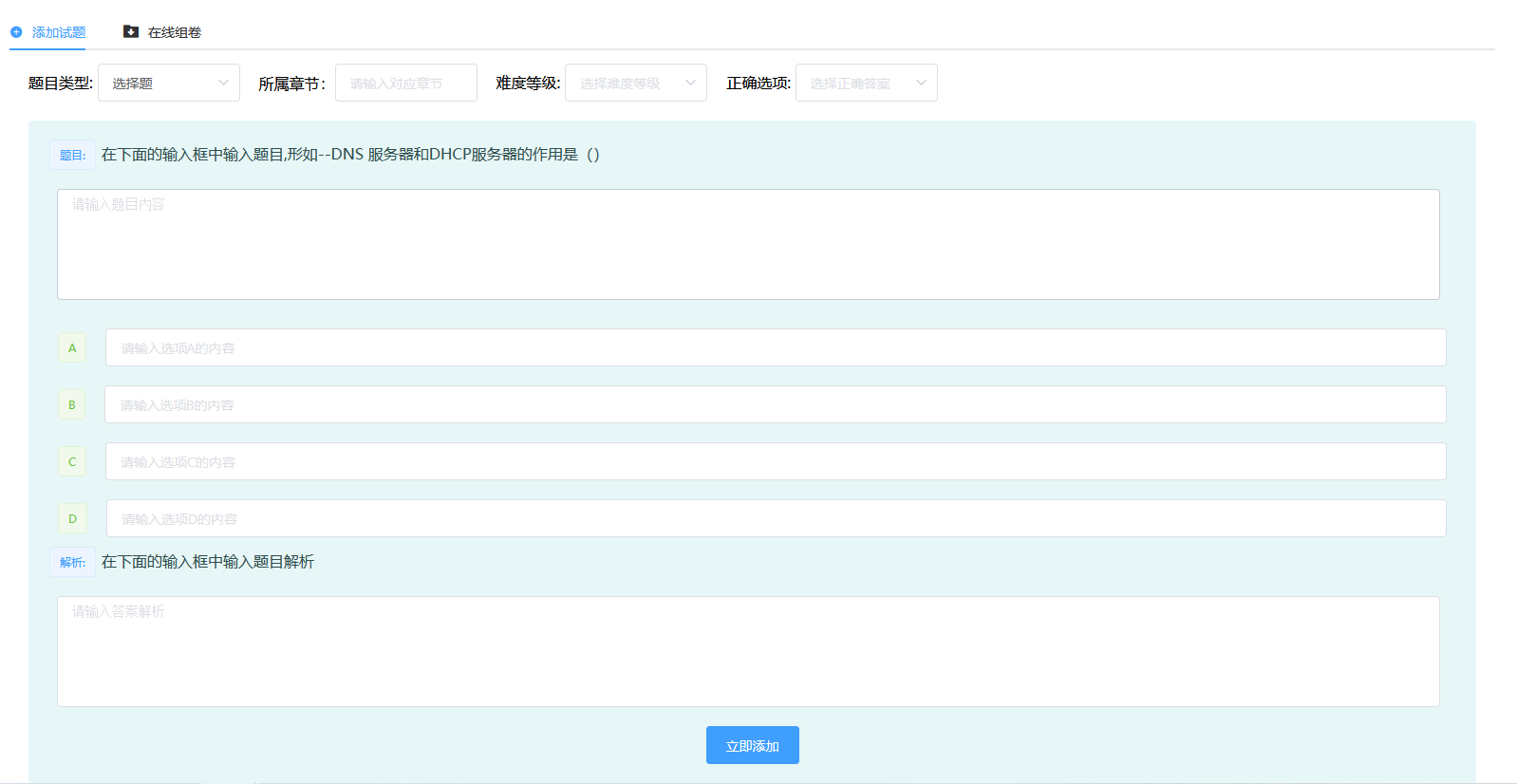
前端校验获取到考试科目，根据下拉列表选着相应的题目类型，前端根据选择的题目类型进行判断调用哪个接口添加哪个题目。



其中一个的具体实现：



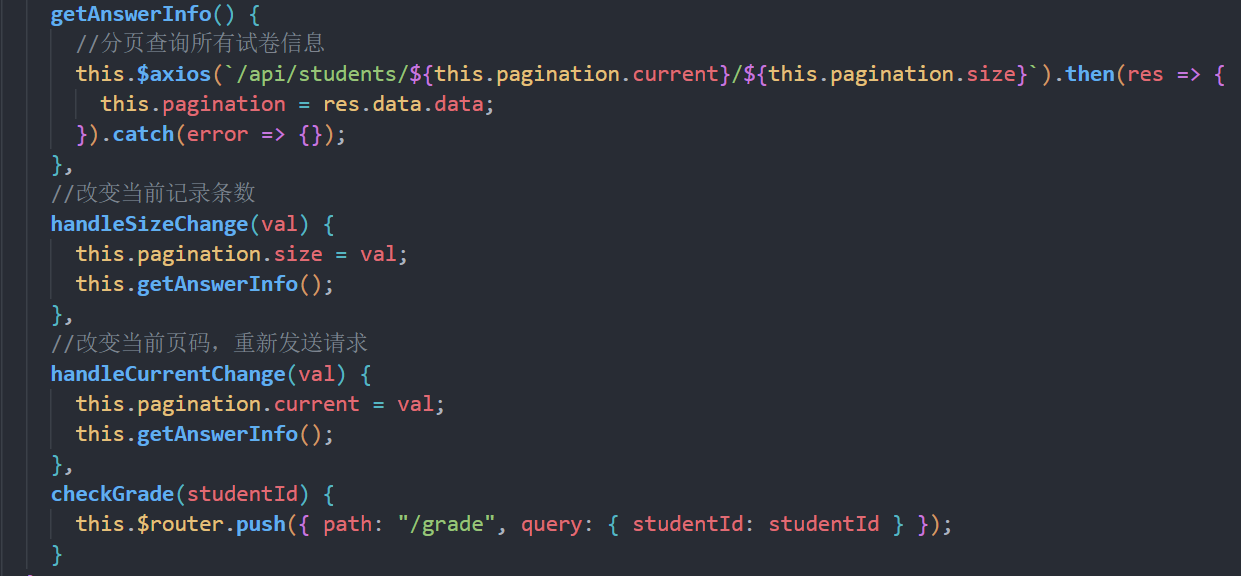
运行结果：



#### （3）成绩管理

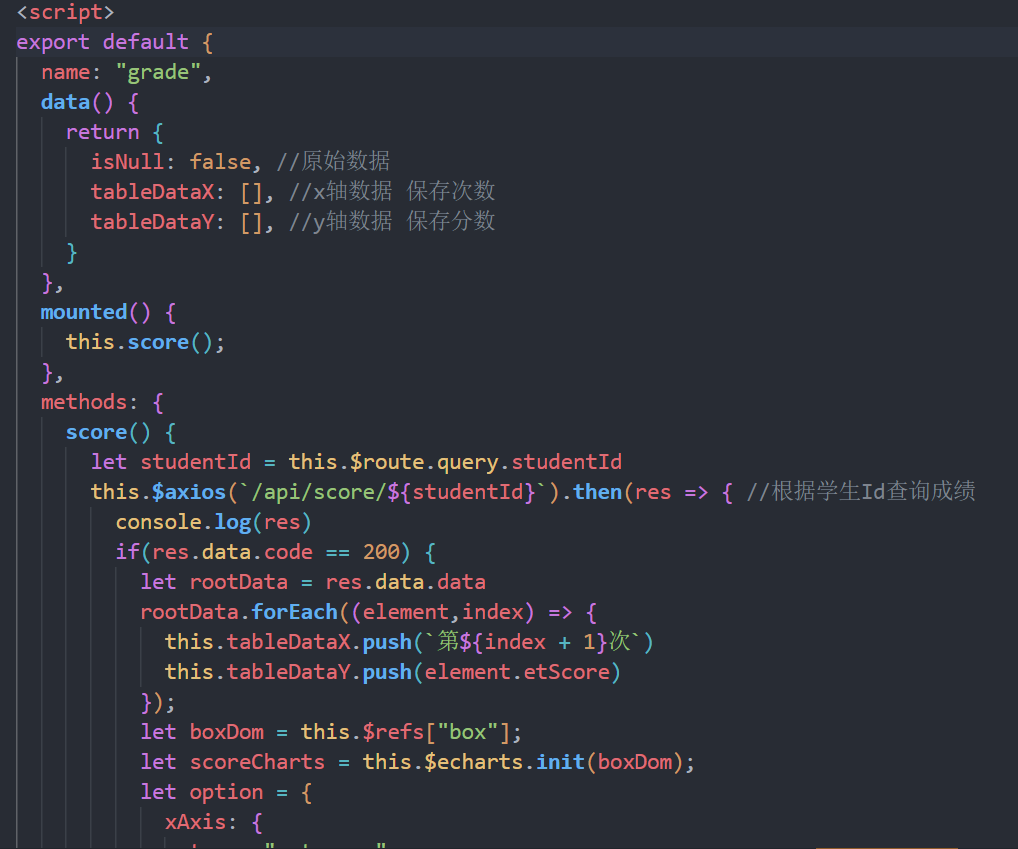
查询所有学生的成绩信息

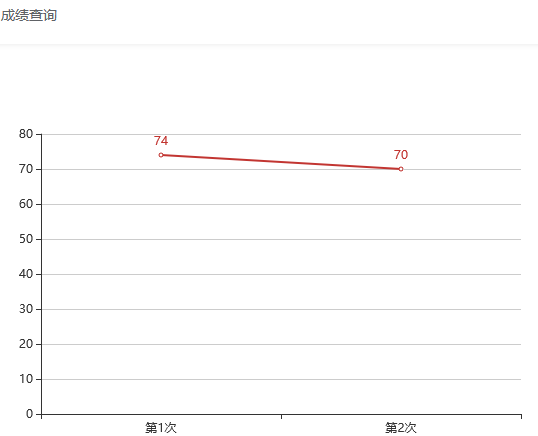
@Select("select \* from student")  
IPage<Student> findAll(Page page);



查看分数：

使用echarts（可视化图表）实现





#### （4）学生管理

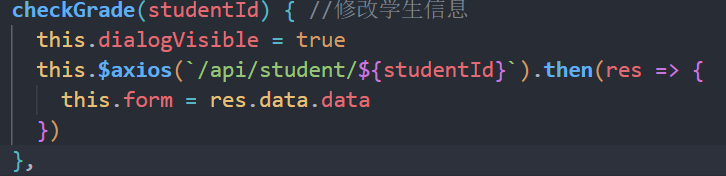
查询所有学生：



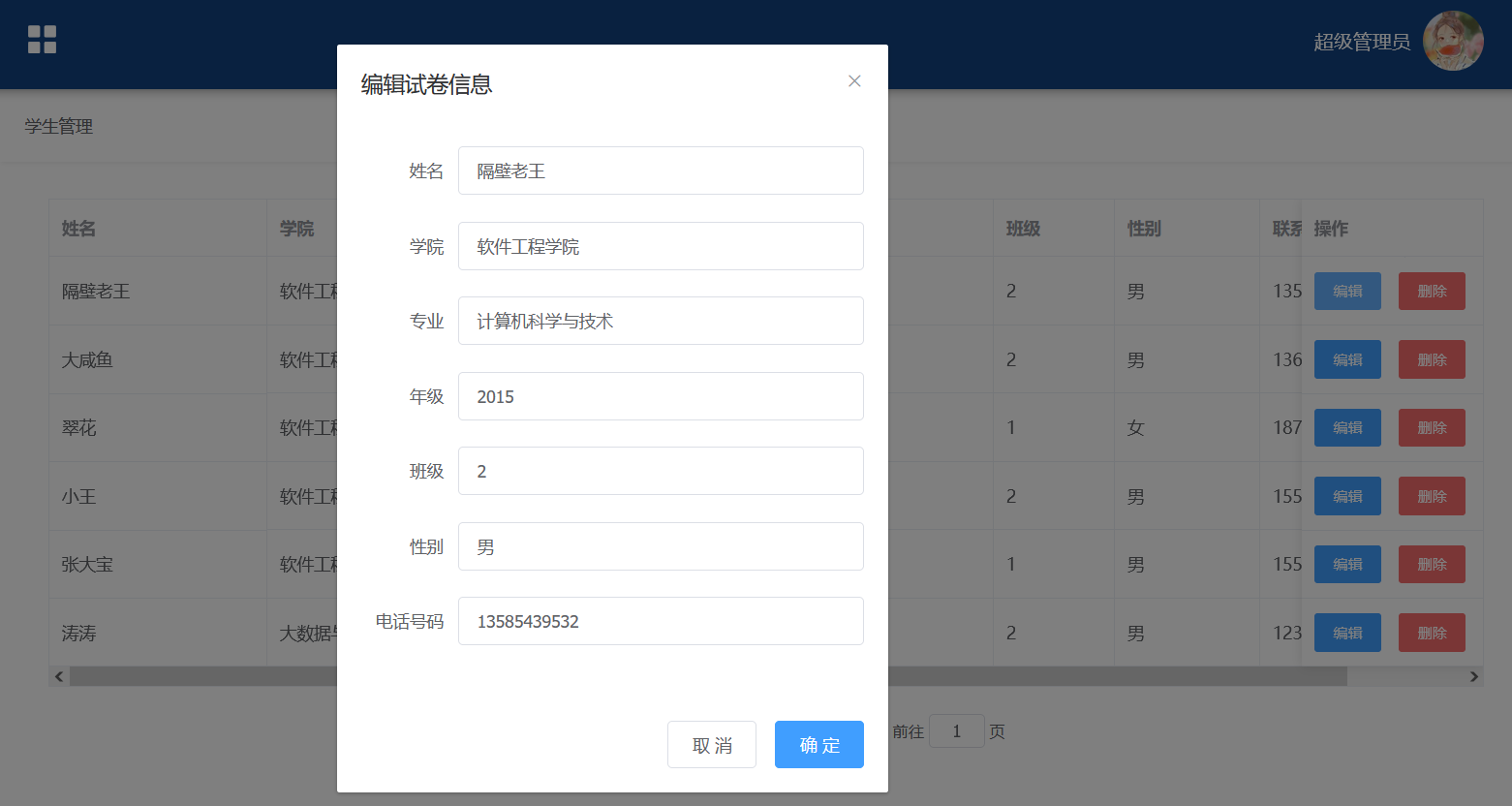


编辑（修改）学生信息：

@Update("update student set studentName = #{studentName},grade = #{grade},major = #{major},clazz = #{clazz}," +  
 "institute = #{institute},tel = #{tel},email = #{email},pwd = #{pwd},cardId = #{cardId},sex = #{sex},role = #{role} " +  
 "where studentId = #{studentId}")  
int update(Student student);

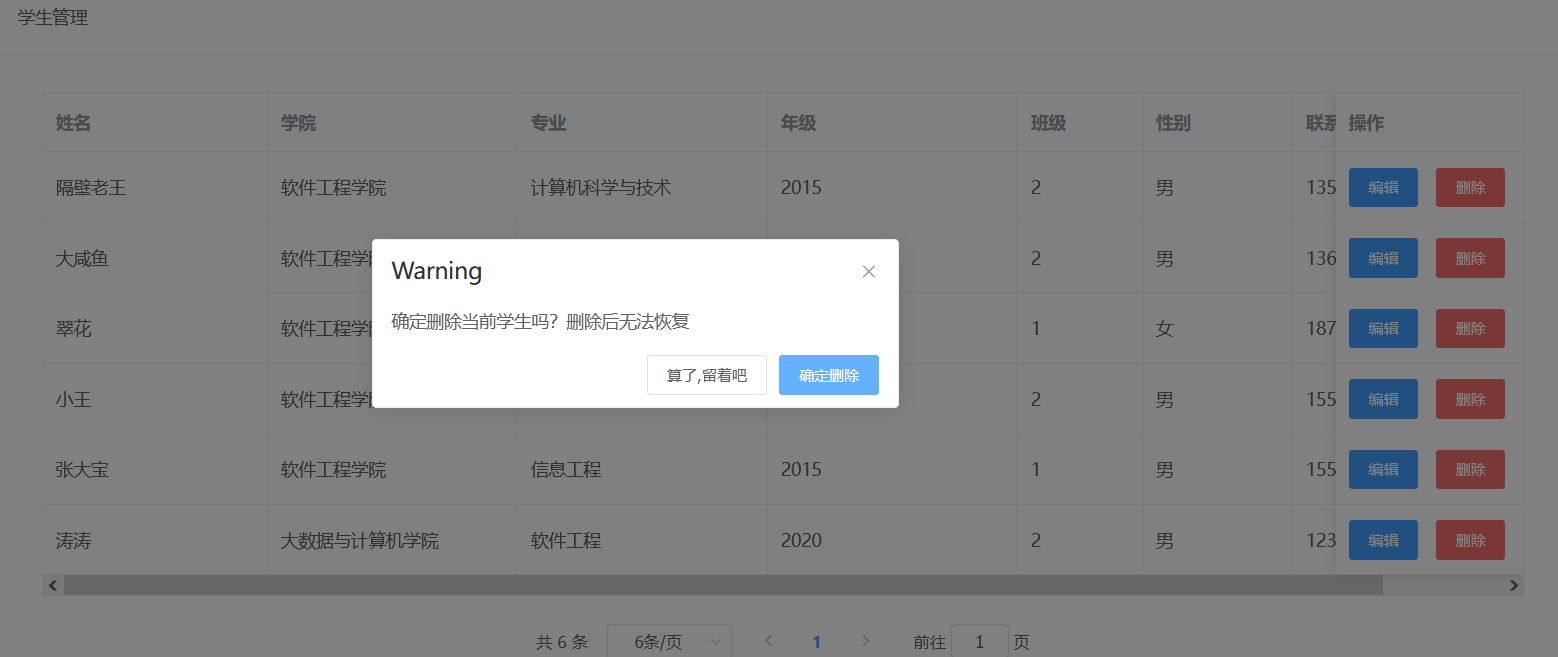


运行结果：



删除学生信息：



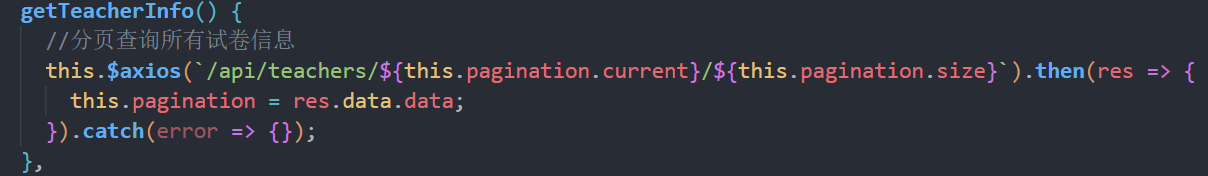


添加学生信息：

@Options(useGeneratedKeys = true,keyProperty = "studentId")  
@Insert("insert into student(studentName,grade,major,clazz,institute,tel,email,pwd,cardId,sex,role) values " +  
 "(#{studentName},#{grade},#{major},#{clazz},#{institute},#{tel},#{email},#{pwd},#{cardId},#{sex},#{role})")  
int add(Student student);



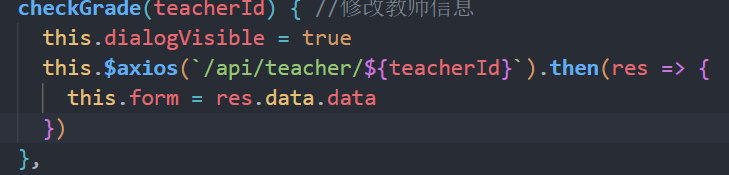
#### （5）教师管理





编辑（修改）教师信息：

@Update("update teacher set teacherName = #{teacherName},sex = #{sex}," +  
 "tel = #{tel}, email = #{email},pwd = #{pwd},cardId = #{cardId}," +  
 "role = #{role},institute = #{institute},type = #{type} where teacherId = #{teacherId}")  
public int update(Teacher teacher);





添加教师：

@Options(useGeneratedKeys = true,keyProperty = "teacherId")  
@Insert("insert into teacher(teacherName,sex,tel,email,pwd,cardId,role,type,institute) " +  
 "values(#{teacherName},#{sex},#{tel},#{email},#{pwd},#{cardId},#{role},#{type},#{institute})")  
public int add(Teacher teacher);





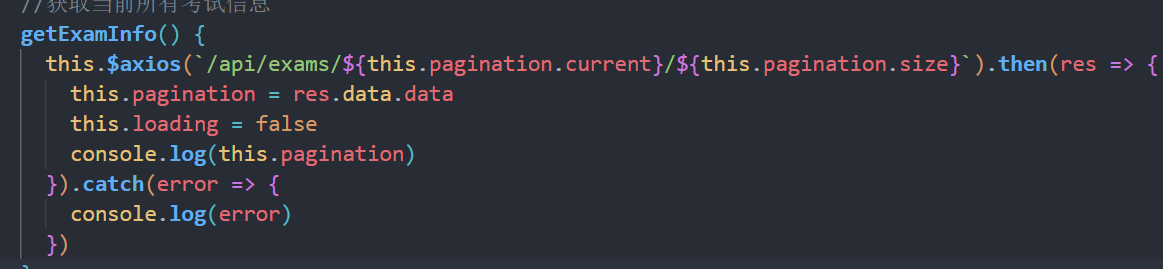
## （四）教师模块

与管理员模块类似，管理员模块多了个教师管理，教师模块没有，其他基本一样，就不过多说明了。

## （五）学生模块

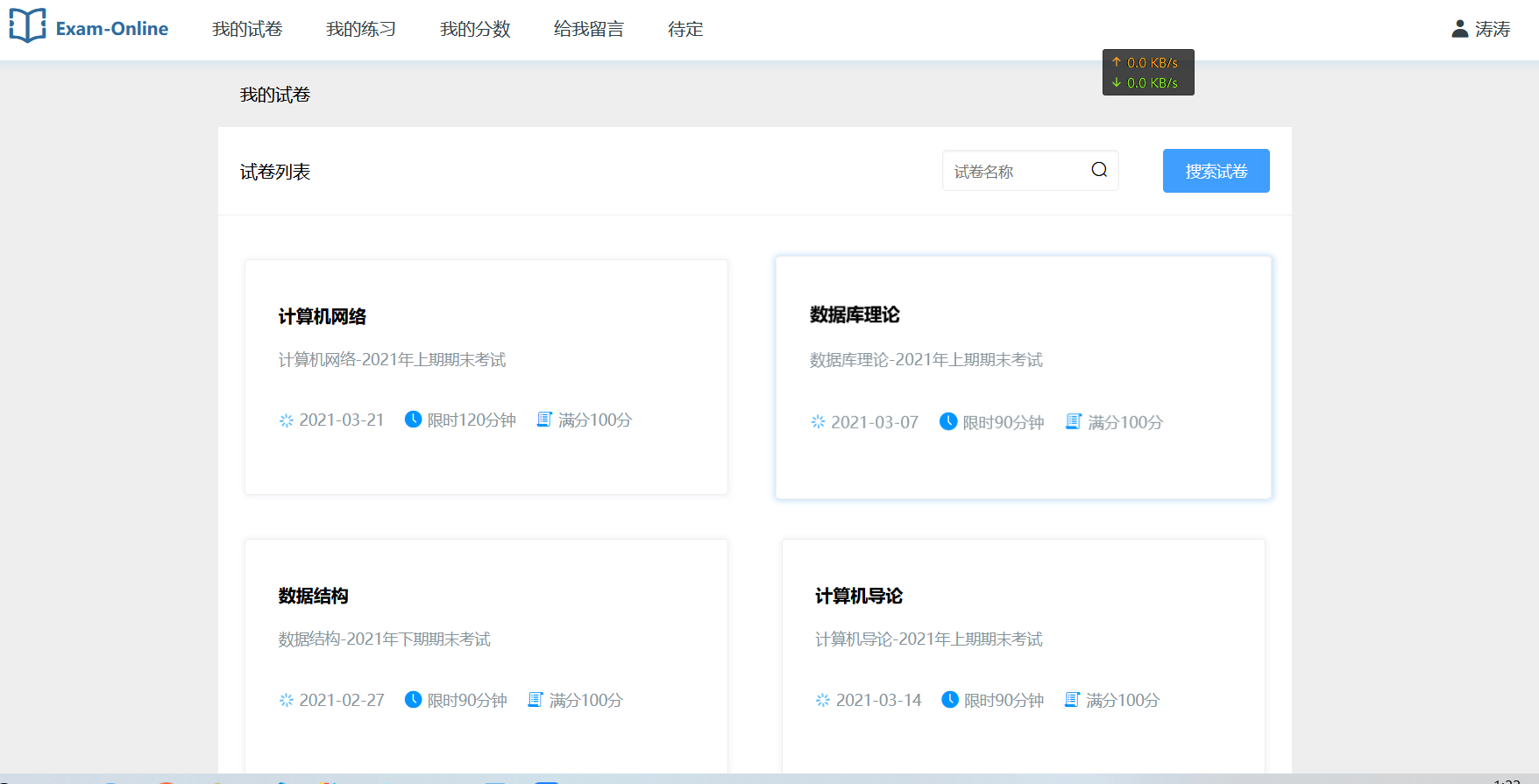
由于题库太少，只设置了《计算机网络》有答题功能

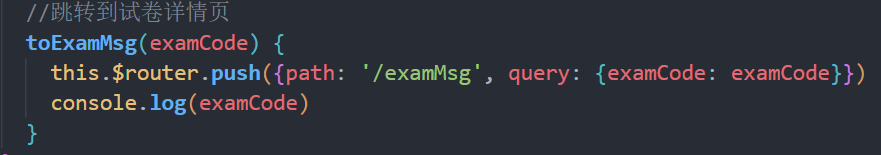
查询试卷信息（模糊查询、精确查询、分页查询）：





运行结果：





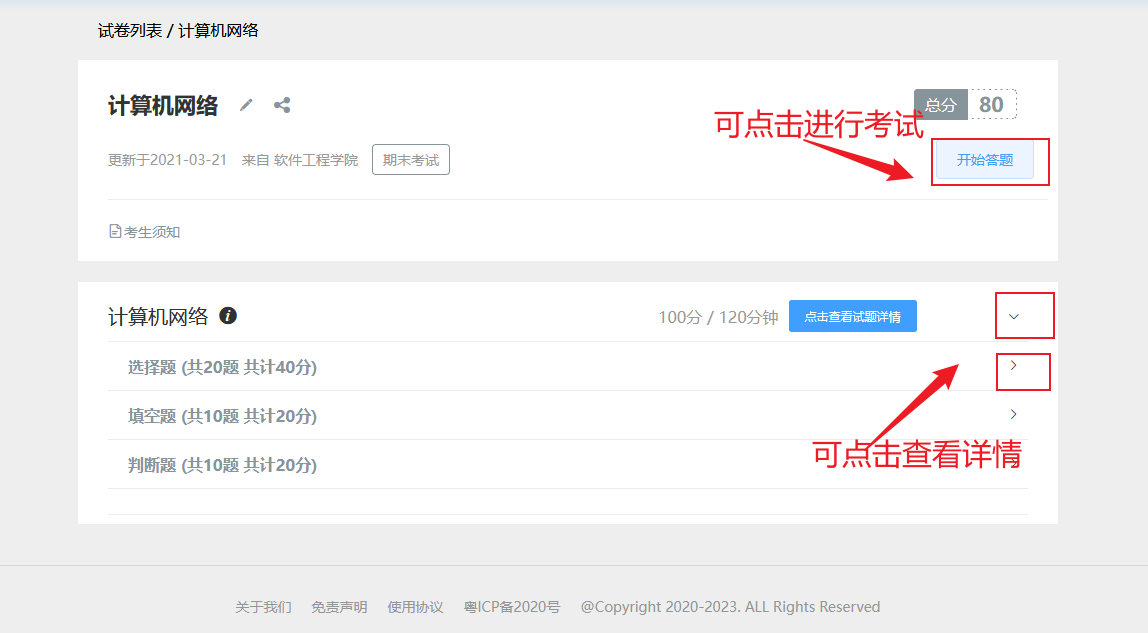


跳转到考试页面

**toAnswer**(id) {

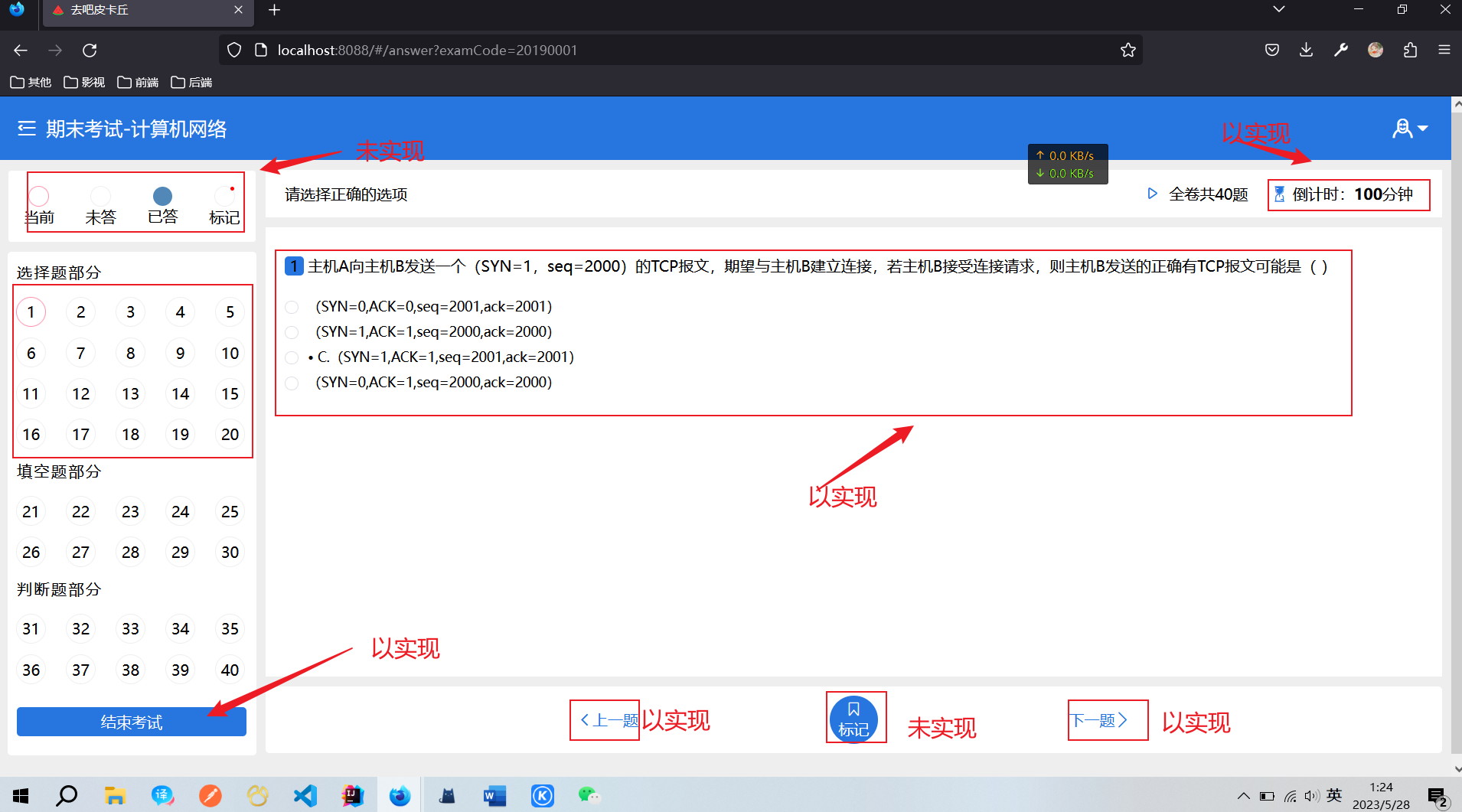
      this.$router.**push**({path:"/answer",query:{examCode: id}})

    },



this.$axios(`/api/exam/${examCode}`).then(res => {})获取路由传递过来的试卷编号，通过examCode请求试卷详细信息， this.examData = { ...res.data.data} 获取考试详情， this.$axios(`/api/paper/${paperId}`).then(res => { })通过paperId获取试题题目信息,对象转数组.循环每种题型,计算出总分,把每种题型总分存入score。

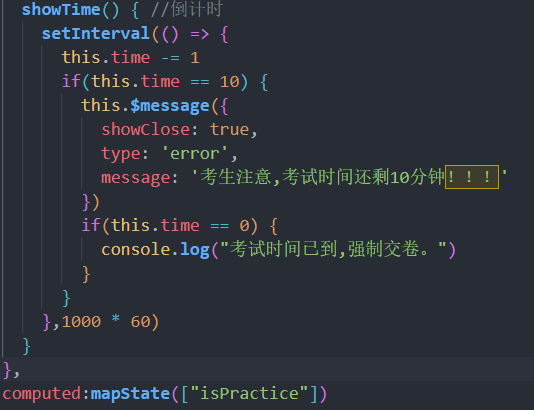


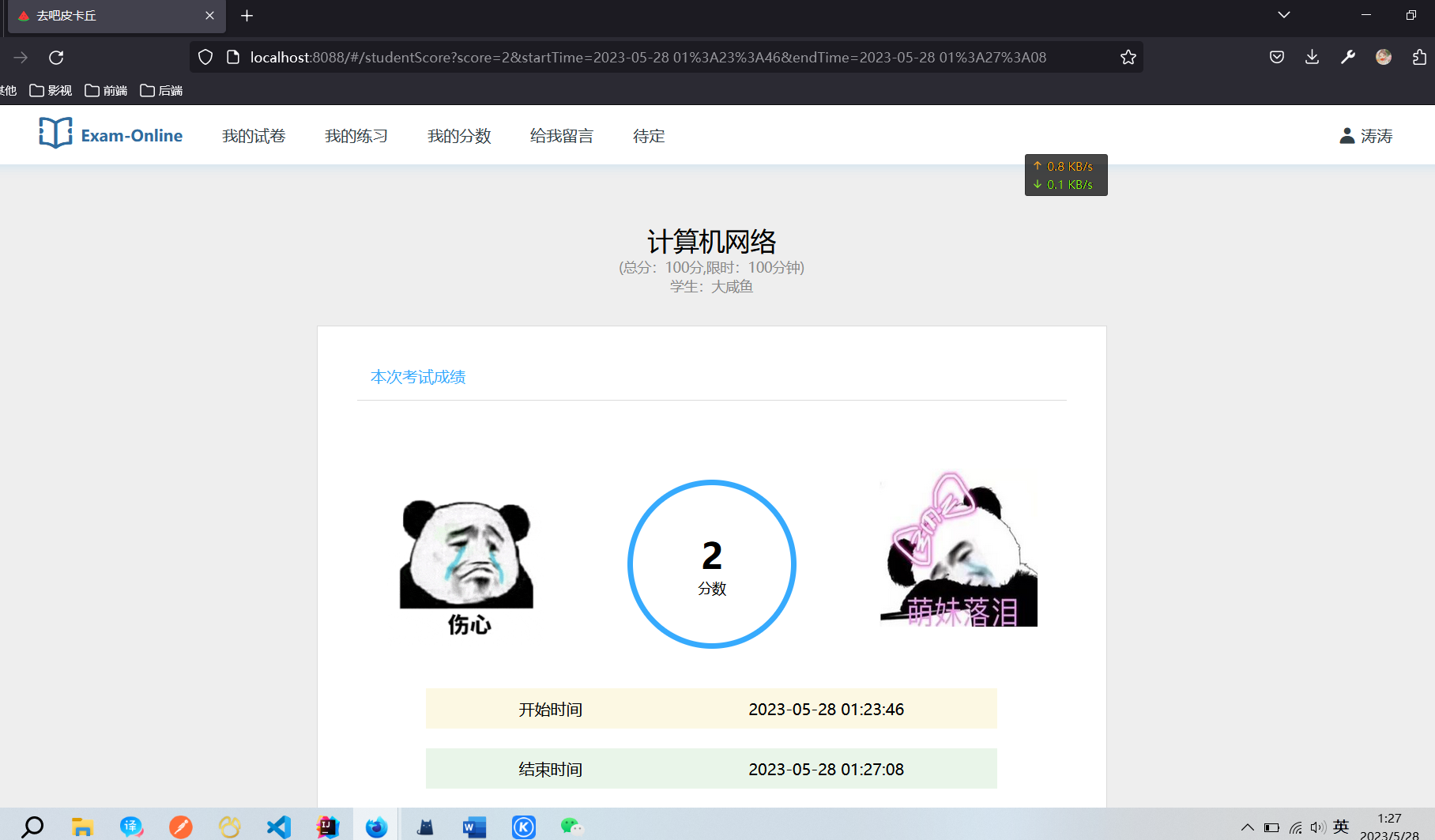
 

答题完成上传考试信息



倒计时功能





# 项目总结

基于Springboot+Vue的在线考试系统的的设计与实现经过系统分析、需求分析、概

要设计、详细设计等阶段基本了解了该系统的功能要求及设计的可行性。在线考试系统的基本功能，包括系统用户的添加、考试试卷的添加、试题的添加和在线考试以及分数计算等。最后将所有模块组装成一个完整的系统。经过测试，系统实现了需求分析中的要求,运行正常，操作界面简洁。由于时间和技术水平的缘故，本系统还存在着许多方面的工作尚未完善。系统的各项功能基本都能正常运行，达到了初步设计的要求，但是还不够完善。比如插入试题模块功能还不完善，系统安全性有待提高。另外在试题的插入时，只能由管理员逐题进行录入，还没有实现试题的成套录入。界面的美化还得进一步细化，各功能模块也有待进行整改和完善。一个好的系统需要一段时间的跟踪及长期的维护，也许后期还有可能进行功能上的增加或改进，这些都需要在系统设计时考虑好，才能使一个系统被应用的时间更长。自己有待进一步提高，相信在以后的学习及生活中会将本设计修改的更加完善。

# 参考文献

[1]霍金明，孙滨，周贤来.Java Web程序设计[M].成都：电子科技大学出版社，2019.

[2]胡静，常瑞，张青，郭纯一.XML基础教程[M].北京：清华大学出版社，2015.

[3]孙滨，李恋，陈强.Java EE入门到实战[M].哈尔滨：哈尔滨工程大学出版社，2019.

[4]马生骏. 基于AJAX技术在线远程考试系统的设计与实现[J]. 甘肃科技纵横,2018,47(01):1-3.

[5]金誉华,周蕾. 基于可变比例的模块化试题生成的在线考试系统的设计与实现[J]. 安徽电子信息职业技术学院学报,2018,17(02):14-17.

[6]郭子文,刘平. 基于MVC模式的在线考试系统设计与实现[J]. 电脑知识与技术,2018,14(05):71-72.

[7]王保银,雷新华. 中职JAVA在线考试系统的研究[J]. 电脑知识与技术,2018,14(07):75-77.

[8]吴光成. 基于.NET的在线考试系统的设计与实现[J]. 教育教学论坛,2018(22):94-95.