# 1 绪论

## 1.1 课题的背景和意义

C语言程序设计是学习计算机的同学必须学习的一门课程,并且一般是学生在程序设计方面的第一门课程，在学习方面有着一定的难度，网络上有着许多视频、文档类型的免费资源，但是这些资源存在着难度不一、质量参差不齐、不能很符合本校学生学习情况的问题，学生在选择时会有一定的难度。同时也有着一些供学生可以进行自我检测的题库，但是也因为不能时常进行更新造成题目难度不一或者针对性不强的问题，导致学生往往花费了大量时间，却收获甚微，呈现出事倍功半的现象。渐渐磨灭了学生的学习热情，成为一个恶性循环。

再者对于老师来说，大部分学校还是采用人工出卷的方式，在筹备练习、组织阅卷和统计学生成绩仍需要大量资源。而人工批改也是一种浪费人力和时间的方法。如果在考生答题结束后，通过系统完成网上评卷，并取得相应的成绩，就可以节省大量的人力、财力、物力和时间资源。当今传统的练习方式正面临变革,先进信息技术的采用有助于加强教学管理、提高教学质量，将练习从传统模式转向具有更高管理效率的网络练习模式［1］。

本课题计划开发一个《C语言程序设计》课程在线题库系统，可以使学生在学习、练习的时候不受地域，时间的限制，学生可以通过在线练习的方式来查看自己的学习成果，并分析出自己有哪些不足。可以在系统中的留言模块寻找自己有疑问的问题的答案，可以在算法游戏模块体验到不一样的学习方式，并且改变了一般的练习系统，评价方式单一的问题。在算法游戏模块增加了小组互评，使得评价更加全面。本系统实现了自动化出题、阅题，从工作量方面来说，一定程度上减轻了教师的负担；系统可以对学生的练习成绩进行必要的统计，帮助教师了解学生对课程中传授的知识具体掌握情况。

## 1.2 国内外研究现状

在线题库系统种类繁多，其基本的模式可以包括 C/S 与 B/S 两种。 由于在线题库系统所表现出的强大的便捷性、超时空性，使得很多国际上的大型练习都使用了网络练习系统[1]。在我国，在线题库系统的实际应用要晚于国外，研究起步也比较晚，但是发展也比较迅速，研究和开发活动比较活跃，实际应用也比较多[2]。但是C/S结构存在开发繁琐，后期维护成本高，变更不够灵活的缺点，所以本系统采用B/S结构配合近年前端流行框架Vue进行开发。

在2009-2010年期间康皓等人研发的练习系统引进了一个多媒体试题库，能够包含图像识别、动画反映、音频和视频播放等各种内容，有效地解决了学生学习枯燥乏味的问题[3]。因此在本系统中添加了“算法游戏”作品模块，来提高学生的学习积极性。并且为了方便掌握学生的学习情况，对成绩的分布，使用图表来显示，成绩分布情况会十分清除。

## 1.3 研究内容

为了更好的实现《C语言程序设计》课程在线题库系统，本课题对在线题库系统进行了研究和分析，基于学生与教师的实用性和乐趣性进行功能模块上的需求分析与设计。需要实现对不同用户身份进行识别，相应展示不同的功能模块。对用户信息、题目信息、练习信息、交流信息等进行储存与管理。为了确保练习的时效性，为此设置了练习进行时的倒计时功能。针对平常的编程题目输入麻烦问题，本系统提供了在线的编辑器功能。经过调研以及深刻解析系统的工作流程，对系统的功能需求进行分析，对操作流程进行简化，最终设计了本系统所拥有的各个功能模块和系统所用到的数据表结构。具体包括以下方面内容：

（1）需求分析：本系统基于对市面现有题库系统功能的分析、总结，新增了乐趣学习方面需求，并设计了各个功能模块。

（2）系统功能模块设计：系统包括面向用户的客户端界面设计，系统后台服务器设计，用户和系统交互流程设计以及系统中各个模块的数据表设计。本系统存在三类用户，分别为学生用户、教师用户和管理员用户。学生用户有着单独的注册接口和登录接口，有着在线练习功能、成绩与历史练习存储以及展示功能，个人信息更改功能，学习心得交流功能，算法游戏视频存储与展示功能，相互评分功能。为教师用户在学生用户功能的基础上增加了题库试题的管理功能，学生用户整体成绩的图形化展示和表格展示功能。为管理员用户在教师用户的功能基础上增加了用户管理功能,可以对所有用户信息进行管理。

## 1.4 论文结构简介

本文主要分为四部分，介绍了系统从初期调查开始，经过需求分析，模块设计，最后系统得以实现。第一部分主要是调查国内外的研究现状。明确本系统实现的意义。第二部分对系统进行了可行性以及系统的主要需求分析。对各个模块的业务逻辑进行全面细致的分析，以求建造更好的系统结构。第三部分主要涉及到系统中所使用的核心技术。第四部分则对系统中数据库结构进行设计，对系统进行技术实现。展示一些用户界面。第五部分进行了系统的测试，分别从系统前后端连接，功能模块和不同设备兼容性进行测试。

# 2 可行性分析和需求分析

## 2.1 经济可行性

主要对实现系统的费用进行评估；系统客户端使用目前主流框架之一的Vue框架搭建，项目代码开源，有着良好的生态支持，官方提供了清晰的使用文档。服务端使用Node开发，有着和Vue相同的特点，生态良好，项目开源；在代码编写工具方面使用了VSCode，可供开发者免费使用；只有服务器租用中需要经济支持，对于第一次租用服务器的开发者来说，费用相对较低。所以在经济方面完全满足系统开发需求。

## 2.2 技术可行性

技术可行性：技术上的可行性是能否实现该系统的关键性因素。该系统采用B/S结构进行开发。数据库采用MongoDB，语言使用HTML、CSS、JavaScript、Node，在过去的学习中，个人对本次所使用的技术进行了较为全面的学习，尝试开发了博客系统和仿真网站的项目，所以在技术可行性方面满足系统开发需求。

## 2.3 需求分析

在进行系统功能需求分析时，按照各个模块的形式进行了各个模块中的功能需求分析，在进行合理分配后，使得前台用户端和后台管理端的耦合程度有效降低。保证系统平稳的运行。系统的需求结构如图2.1所示。

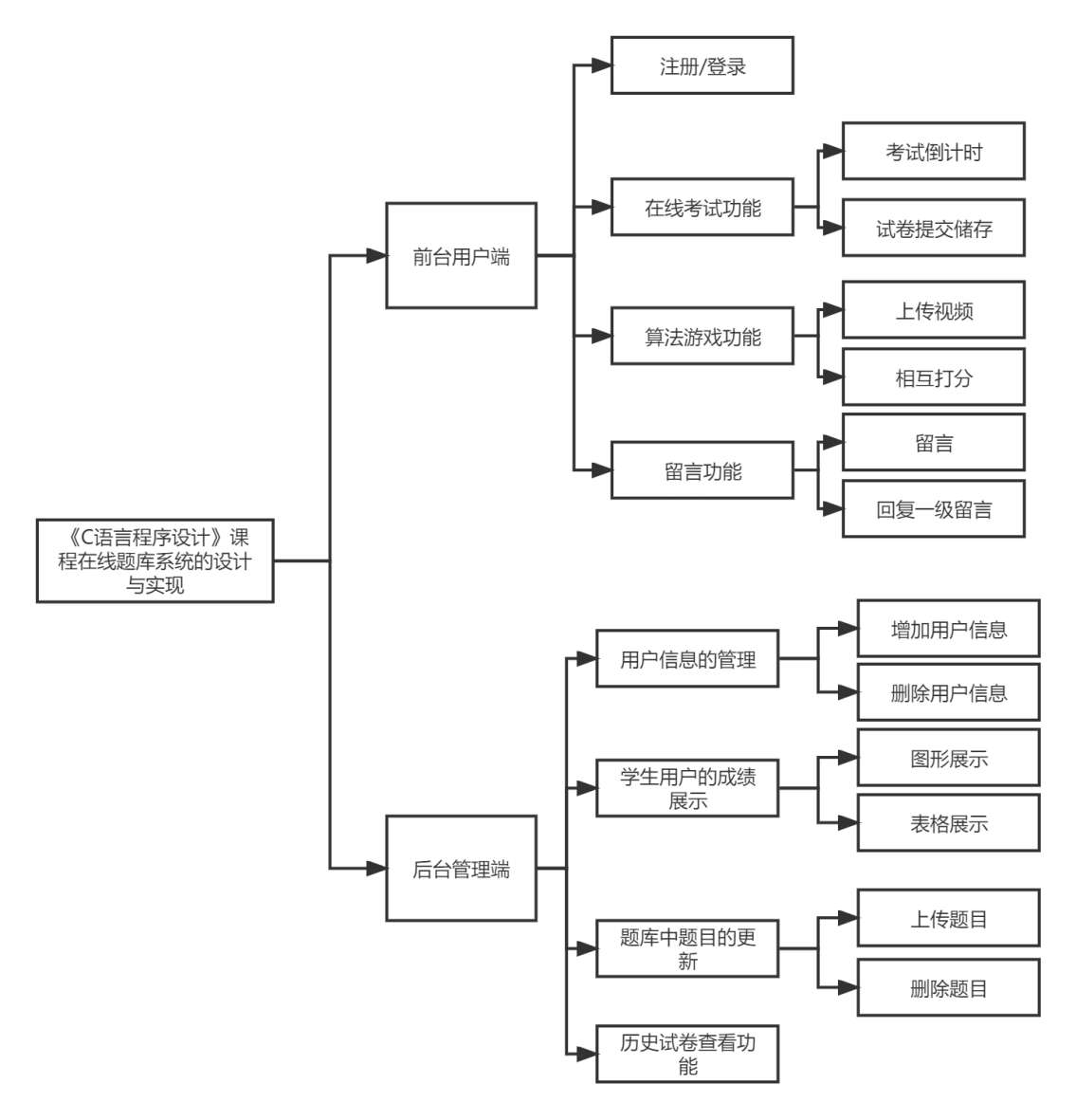


图2.1 系统功能需求图

### 2.3.1 注册登录需求分析

学生用户在没有账号的情况下，需要注册账号信息，在注册好用户信息后进行登录。在登录、注册时需要对用户所输入的账号和密码进行格式检测，在检测不通过时在客户端进行相应的信息提示。

教师用户账号的注册只能由管理员在系统中进行添加，防止不相关人员对题库进行错误管理。

### 2.3.2 在线练习需求分析

学生用户可以在本系统中进行自我练习，针对不同学生学习效果、进度不同的情况，需要设置不同难度比例的题目供自我练习所用。在自我练习时需要进行时间上的限制，设置时间到达系统自动提交的功能。在用户自己进行提交时，需要检测所有题目是否全部完成，若未完成则需要对用户进行提示，显示具体哪些题目没有完成，是否在这种情况下继续提交练习，提交练习后系统进行答案对比，得出本次练习得分。

### 2.3.3 留言模块需求分析

用户需要在这里进行留言提交和展示，在输入信息并且提交后，留言展示面板需要进行更新展示。

### 2.3.4 算法游戏模块需求分析

本模块需要实现用户上传视频作品，对同学作品相互打分的功能，并且本小组成员不得对本小组作品打分，不可对同一作品进行重复打分。在上传完成后，展示界面进行实时的数据更新。

### 2.3.5 题库管理模块需求分析

在本模块中需要实现用户对题库的更新管理，可以对陈旧题目进行删除，对新题目进行添加，在添加数据方面采用多种形式上传的方式，设置Excel文件和单个题目上传的模式。在删除或者添加题目成功后，数据展示页面需要进行实时的数据更新。

### 2.3.6 成绩展示模块需求分析

本模块需要实现所有学生用户的成绩展示，通过图表的形式可以让教师掌握到学生对知识的掌握情况。

### 2.3.7 用户管理模块需求分析

本模块需要实现管理员用户对学生和教师用户账号信息的管理，能对所有用户信息进行删除并且可以在这里添加教师用户的账号信息。在删除或者添加用户信息成功后需要在展示界面进行实时的更新。

### 2.3.8 个人中心模块需求分析

本模块需要实现用户对本账号的信息修改、查看历史练习的功能。

## 2.4 功能模块逻辑分析

### 2.4.1 登录注册模块逻辑分析

在用户输入网址后跳转至登录页面，如果学生用户没有账号信息则可以进行注册账号，输入学号和密码，在输入密码时需要输入进行两次比对，前后输入的密码是否一致，防止错按键盘导致注册密码与主观意识不符的情况，然后进行格式检测，如果格式不符设定，进行提示。最后将用户信息发送至后台进行储存。在后台首先在用户的数据库中进行以学号为关键字的查找，如果本账号已存在则返回提示信息，若没有此账号，则进行存储。在有账号的情况下直接进行登录，在输入账号密码后，点击登录按钮，执行登录函数中事件，系统将账号信息发送至后台，系统在后台通过用户账号信息在user信息表中查找比对，如果密码和账号和数据库中用户信息匹配则向客户端返回正确信息，页面路由进行跳转。若密码和账号匹配不上，则返回错误信息，给用户进行提示，重新输入账号，密码信息。本模块的流程图如图2.2。

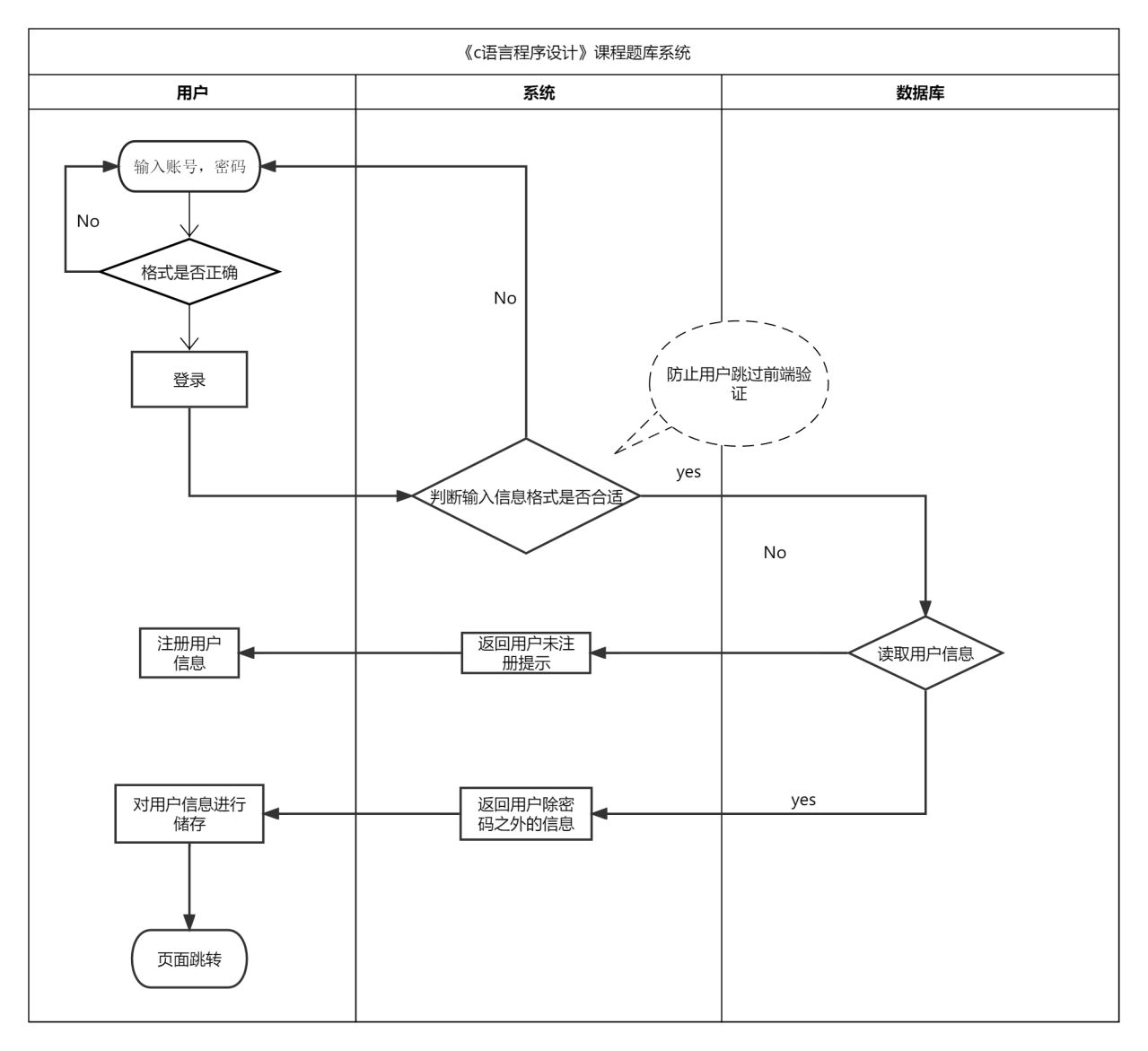


图2.2 登录模块逻辑分析图

### 2.4.2 在线练习模块逻辑分析

在考生进入系统后，首页设置三个等级的练习难度，考生可以根据自身的学习情况来选择练习题目的难易程度，然后跳转至考生须知界面，并将练习题目分为选择题，编程题两类，在点击开始练习后，进入练习页面，显示倒计时，题目详情，此时考生可以进行做答，在规定时间内答题完毕后，点击提交按钮，系统检测用户是否答题完成有没有遗漏题目，若有则进行提示，用户选择是否继续交卷，之后系统进行评分，并且将学生练习题储存至历史练习题目数据库中，以便未来进行复习，最后在页面中显示最终得分。 模块逻辑分析如图2.3所示。

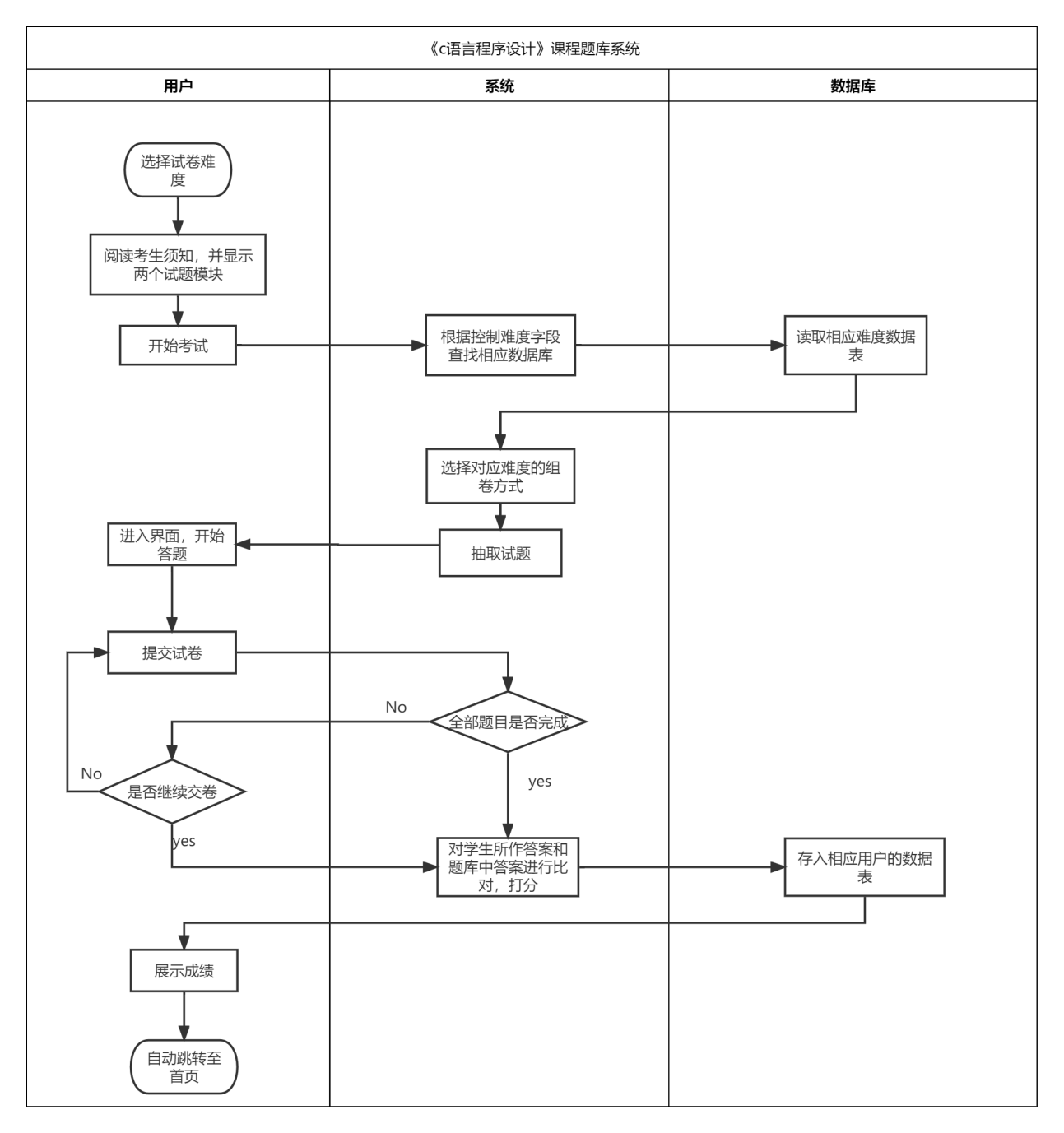


图2.3 在线练习模块逻辑分析图

### 2.4.3 留言模块逻辑分析

设置留言模块的目的是在为同学们提供一个可以进行学习交流的平台，用户进入本模块后，可以在题目的输入框中输入自己要交流的问题的题目，也就是对本问题的大概说明。在详细问题的输入框中进行对问题的详细说明。然后进行提交，在下方的留言展示进行展示，以便其他用户进行回复。

或者可以在浏览其他用户信息，对其他用户所留言的问题进行解答。模块逻辑分析如图2.4所示。

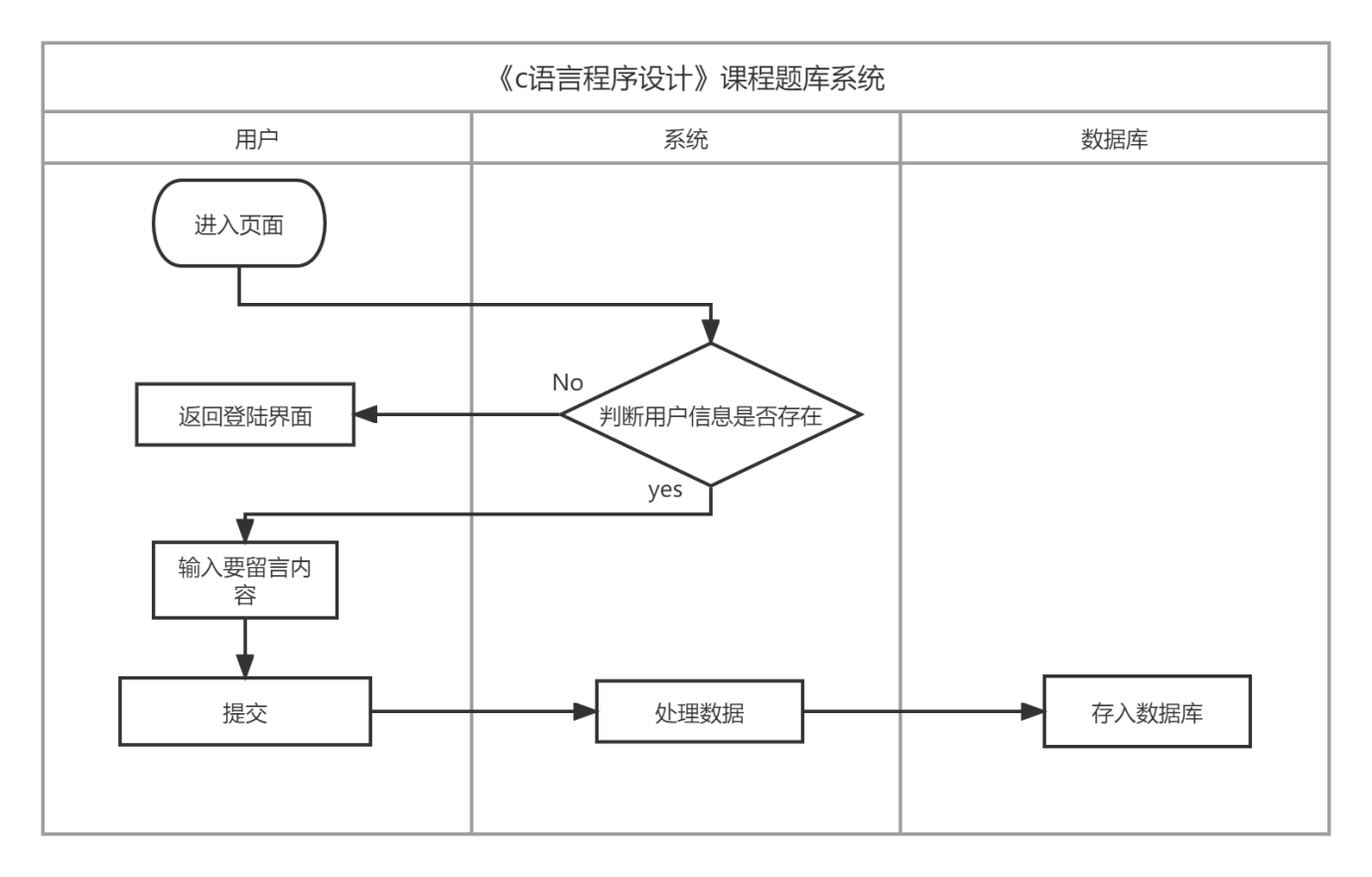


图2.4 留言模块逻辑分析图

### 2.4.4 算法游戏模块逻辑分析

在进入到算法游戏模块后，先判断该用户是否存在，若存在可以在上传视频的功能区输入自己的小组学生学号，然后对游戏视频进行上传。若这个小组的同学已经上传过视频，则不能再次进行上传，上传视频后可以对除自己所在小组的其他小组视频进行打分，在打分时首先要对该用户是否是该视频的作者小组成员进行判断，并且不能进行重复打分。最后打的分数要进行计算，得出平均值，向用户展示。模块逻辑分析如图2.5所示。

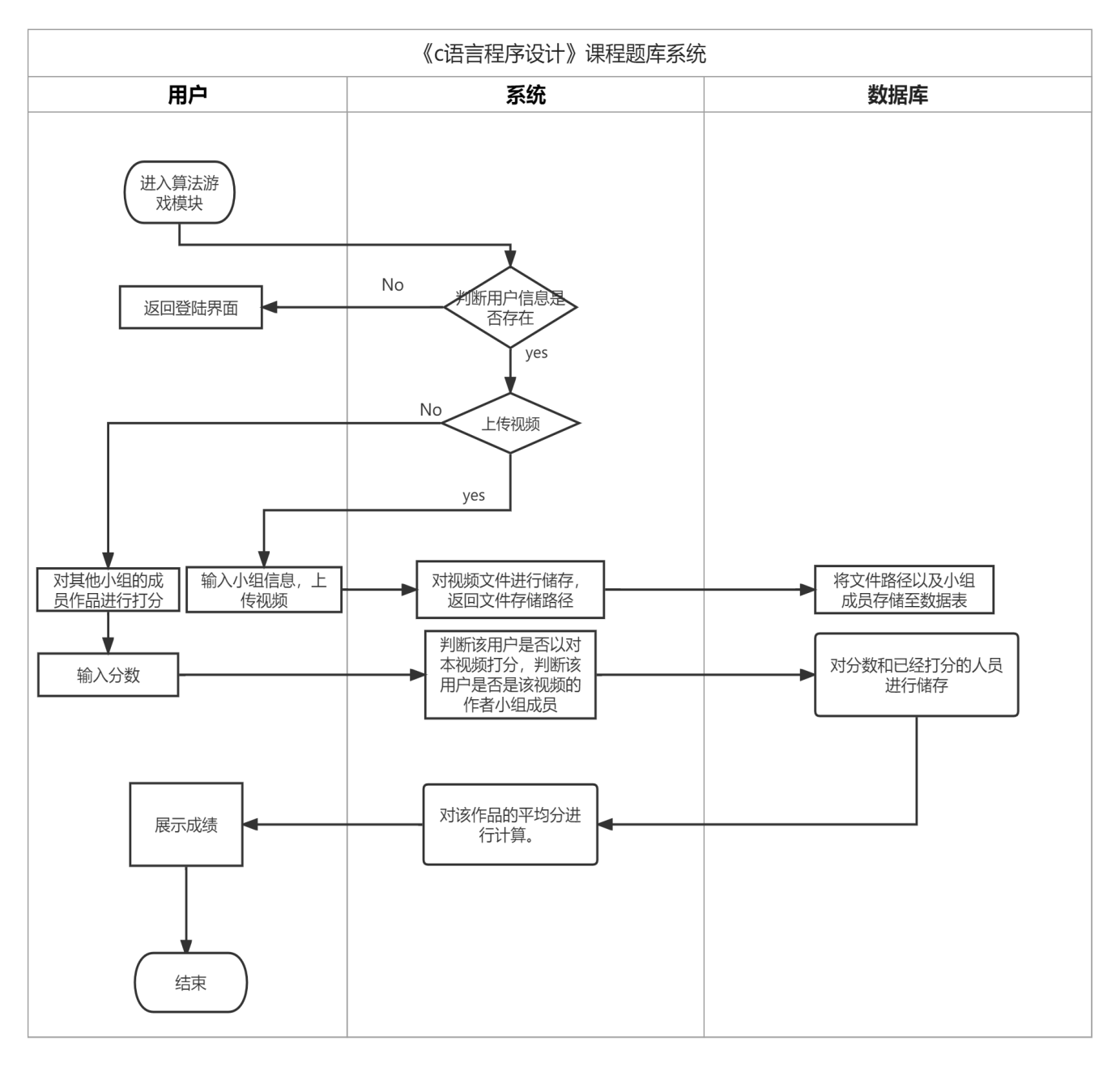


图2.5 算法游戏模块逻辑分析图

### 2.4.5 题库管理模块逻辑分析

本模块主要为教师用户和管理员用户展示。在用户进入页面后首先需要展示本题库中已经有的一些题目，并且在验证用户权限后，可以对已有的题目进行管理。比如删除，增加。增加题目时需要为用户提供难度不同的按钮。可以使用上传Excel文件，也可以进行单个题目的上传。模块逻辑分析图如图2.6所示。

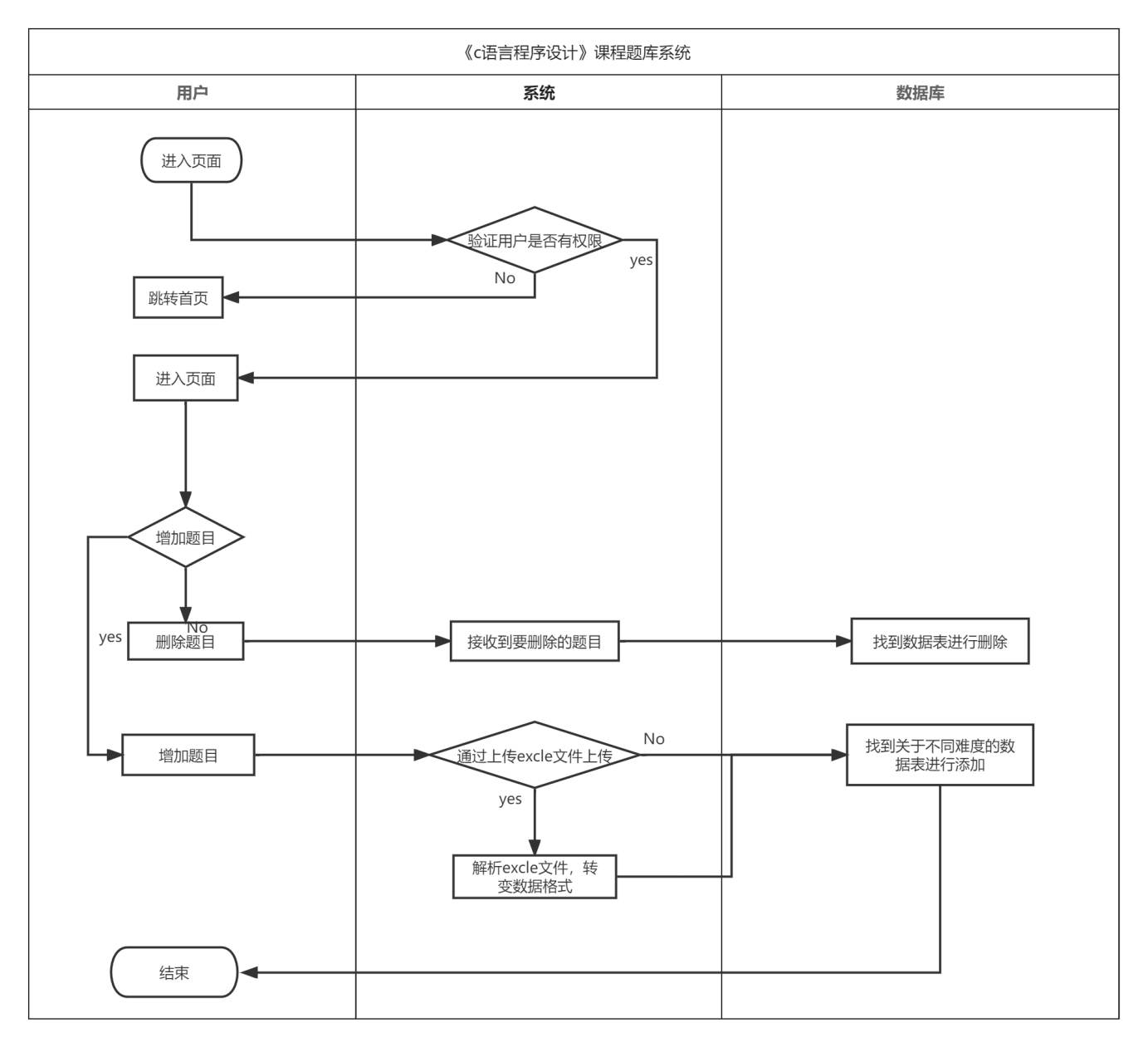


图2.6 题库管理模块逻辑分析图

### 2.4.6 用户管理模块逻辑分析

本模块只为管理员用户展示，在页面创建之前进行身份认证，对比身份字段，如果身份字段比对不通过，则跳转至系统首页。如果身份字段比对匹配，则渲染页面。管理员用户可以看到目前系统的所有用户，并且将用户的详细信息进行展示。可以对用户进行管理，比如删除，增加。并且教师用户只能由管理员用户在此模块进行增加。需要增加教师用户时。在增加的表单中输入要增加的教师账号，密码等信息，点击提交后，由系统对教师用户进行储存。模块逻辑分析图如图2.7所示。

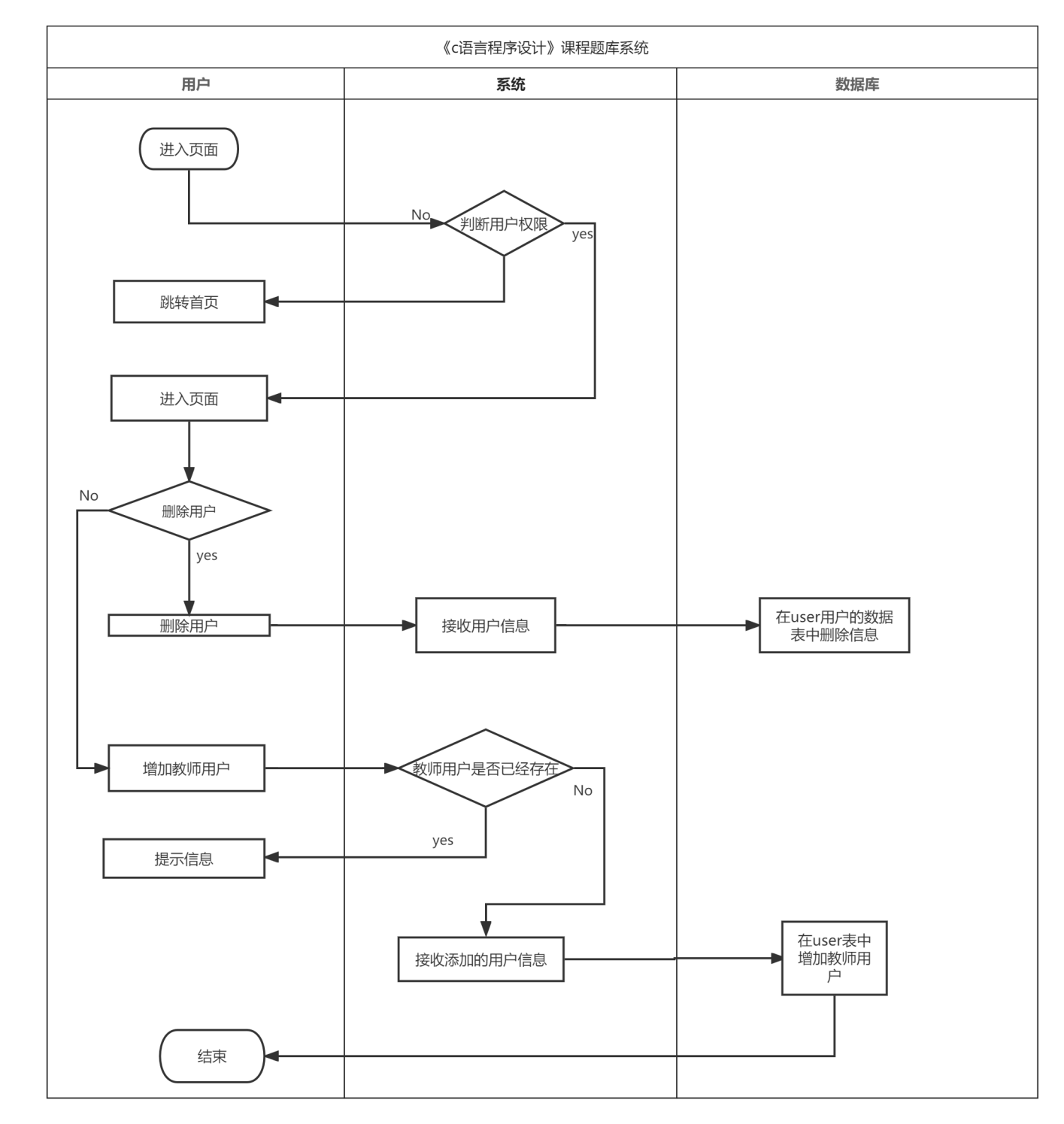


图2.7 用户管理模块逻辑分析图

### 2.4.7 成绩展示模块逻辑分析

本模块为教师用户和管理员用户设置。在此模块中教师、管理员可以看到所有学生用户的成绩情况。首先整体情况以饼状图展示，同时各个练习的详细情况以列表形式展示。可以看到练习所属学生和答卷日期，成绩等。还可以对练习详细答案进行查看。以便了解学生学习的具体情况。模块逻辑分析图如图2.8所示。

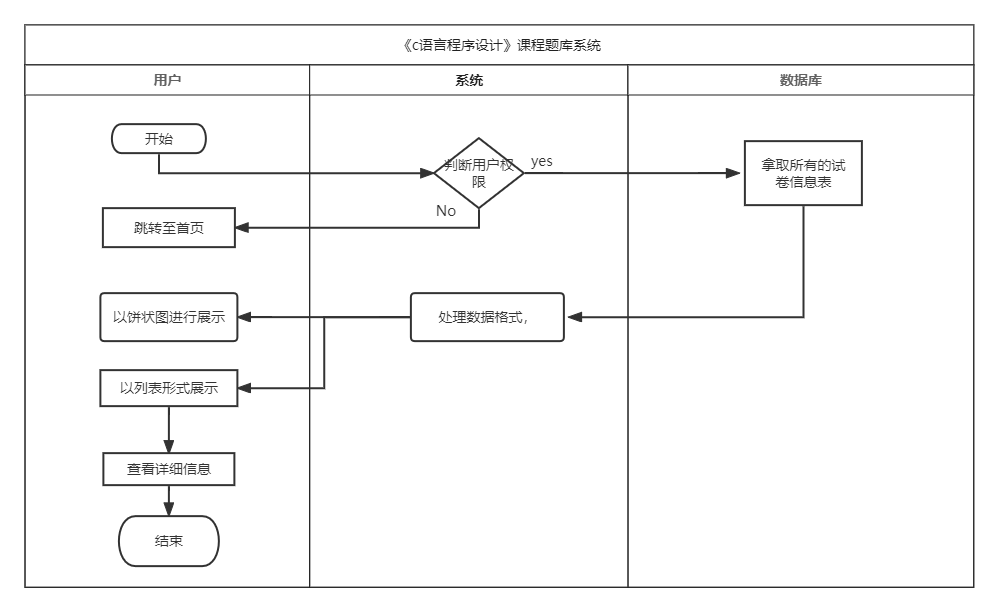


图2.8 成绩展示模块逻辑分析图

### 2.4.8 个人中心模块逻辑分析

在本模块中，可以对用户的历史成绩进行查看，可以对用户的头像，昵称，密码进行修改。修改密码时需要验证原来的密码，并且在输入新密码时需要进行二次验证防止误改密码。模块逻辑分析图如图2.9所示。

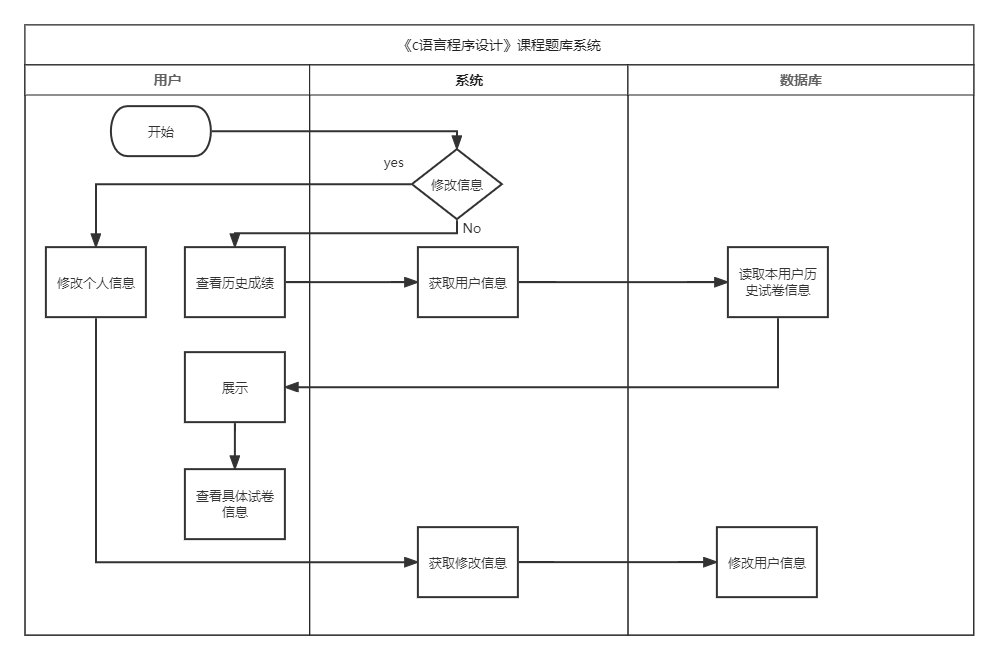


图2.9 个人中心模块逻辑分析图

# 3 系统中核心技术

## 3.1 Node.js程序

Node.js是一个运行在chromeJavascript运行环境下（俗称GoogleV8引擎）的开发平台，用来方便快捷的创建服务器端网络应用程序。

全站开发离不开服务器的搭建，而Node.js 作为一门新出现的后台语言，可以让前端程序员经过简单的学习快速构建可伸缩的应用程序。Node中有fs模块，它补足了Node在浏览器中不能操作文件的缺点，比如使用require方法加载了fs模块后，就可以对文件进行操作。提供读，写，删除，复制等方法。并且它还提供了用来创建服务器的HTTP模块。在使用第三方的模块都是需要npm加载，并且通过require（‘模块名’）来引入。

## 3.2 Express

Express 是一个 基于[Node.js](https://baike.baidu.com/item/node.js/7567977" \t "https://baike.baidu.com/item/eXpress/_blank)的服务端的开发框架, 有着简洁、灵活的特点，有着强大的路由，多模块支持的特性[4]。如path路径工具，fs操纵文件工具。有着丰富的HTTP工具和来自Connect框架的中间件，作为开发者的我们可以随心所欲的使用，并且Express中的API对开发者十分友好，可以方便快捷的创建功能强健的应用。Express和原生写法的区别就是可以进行快速开发，不需要手写很多的代码，使用一行命令就可以生成一个Express的基础代码模板，并且通过其灵活的扩展机制可以很方便的加入一些其他的功能，他还提供了详细的API文档，开发人员可以轻松使用。

Express框架的核心特征在于：可以设置中间件来响应 HTTP 请求。可以封装一个路由文件，用来规范，定义前台的 HTTP的请求动作，更加趋向于模块化开发，使得后续的代码维护更加的容易，简单，快捷。

## 3.3 Vue

“Vue.js”(读音/vju:/,类似于“view”)是一套构建用户界面的渐进式框架。与其他重量级框架不同的是，“Vue.js”采用自底向上增量开发的设计。“Vue.js”是个非常轻量级的工具。其本身具有响应式编程和组件化的特点[9]。

响应式编程简单来说，就是让数据状态和视图变化一致。在没有响应式编程的时代，在初始化界面时将数据在视图中渲染出，如果要对视图进行修改，则需要通过获取视图元素，利用相应的语法对DOM元素进行修改，保证数据和视图变化一致。但是Vue使用MVVM开发模式，使得开发者不用在关注视图和数据统一变化的问题，只需要关注数据的变化，不需要再手动获取DOM元素，让代码变的更加简洁。

组件化可以让开发者期望的代码复用得以实现。在Vue中开发者可以使用组件化这一特性，将封装的代码注册为可以复用的标签。

并且Vue也是有着丰富的生态环境。他提供了创建项目的脚手架vue-cli，不用再单独去配置webpack等配置。路由管理工具vue-router，用来实现各个页面中的跳转。全局状态变量管理工具Vuex。 HTTP请求工具axios。各种各样的UI库。使得Vue开发应变变得十分方便快捷。

## 3.4 Ajax技术

Ajax ( Asynchronous JavaScript and XML，异步 JavaScript和 XML)[37]最早山Jess JamesGarrett提出，区别于传统的Wcb应用，Ajax主要目的就是提高用户体验。

可以这样来形容，Ajax类似于一个产业链中的中间商角色，不过不同于中间商的是它并不赚取HTTP通信的差价。在我们在浏览器中上网时，通过输入网址，或者通过各种各样的表单，提交或者是拿去内容时，就是在和后台的服务器进行交互。传统的web应用和服务端交互是怎样的呢？它先是由服务器端的用户来引发一条HTTP申请到达后台的服务器,而后服务器对客户端的申请做出相应的处理,之后再给服务器端回复一条HTML的页面。在这个过程中，用户不能在客户端进行其他的操作，只能安静的等待，而且还需要宽带重新取请求整个页面。传统web应用交互逻辑图如图3.1所示。

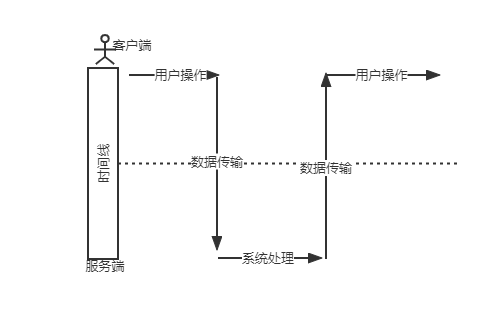


图3.1：传统web应用交互逻辑图

而Ajax用异步化的方式来建立起客户端和服务端的联系，在那些不需要交给服务端处理的流程直接交给Ajax来处理。经过发展，Ajax现在就像一个集合体，它集成了Css，JavaScript，XMLHTTPRequest，DOM。然后通过浏览器提供的XMLHTTPRequest（XHR）对象来发出HTTP请求和接收HTTP响应。实现了web应用中界面和数据处理的分离，这样的结构有利于开发人员的分工合作，提高效率。总结来说，Ajax的优点有：

（1）有了Ajax后，系统就可以在不刷新web应用的页面的前提下和服务器来通信，使得系统可更快的响应应用的交互操作，让客户端的等待时间减少，提高用户的使用体验。

（2）Ajax使用异步方式实现客户端和服务端的通信，在不打断用户的的操作的前提之下拥有了更加快速的响应速度，并且减少了不必要的数据传输，并节省了用户的数据流量的使用。

（3）有了Ajax之后，以前只能是服务端负担的工作可以由后台转移到前台，利用浏览器的闲置能力来处理，以达到减轻服务端工作负担的目的。提升系统的性能。

## 3.5 MongoDB

MongoDB 是一个基于分布式，文件存储的非关系型数据库，其特点是高性能、易部署、易使用，存储数据方便[11]。

在MongoDB中文档数据以BSON格式存储，可以存储结构十分复杂的数据。但是也是因为其灵活，存储文档数据结构可以十分复杂的特性，在设计文档结构时需要谨慎的设计文档模式，以便应对数据查询困难的问题。

# 4 系统的设计与实现

## 4.1 数据库设计

经过对系统的功能需求分析，各个模块的逻辑分析，为了更好的实现系统的功能，对数据库进行了详细的设计。

### 4.1.1 用户信息表（user）

用户信息表用来存储用户的信息，以实现具体功能呢为目的。做出了如下设计，具体字段有xingming（用户名称），name(账号)，pass(密码)，photo（头像路径），student（学生用户），teacher（教师用户），admin（管理员），date（创建日期），具体结构如表4.1所示。

表4.1 用户信息数据表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 默认值 | 是否必须 |
| 1 | xingming | 字符串（String） |  | true |
| 2 | name | 字符串（String） | /files/img/photo/default.jpg | true |
| 3 | pass | 字符串（String） |  | true |
| 4 | photo | 字符串（String） |  | true |
| 5 | student | 字符串（String） | true |  |
| 6 | teacher | 布尔值（Boolean） | false |  |
| 7 | admin | 布尔值（Boolean） | false |  |
| 8 | date | 数值（Array） | Date.now |  |

### 4.1.2 留言数据表（textarea）

留言数据表用来存储用户的留言信息，通过数据表关联，将user用户数据表关联值留言数据表，使得每个留言有各自的留言用户信息。具体字段设置如下：title（留言标题），textarea（留言内容），date（留言时间），user（关联至具体用户），children（每个留言信息的子留言信息），子留言数组中有textarea（子留言内容），date（子留言的留言时间），user（子留言用户），replyUser（子留言用户回复对象），具体结构如表4.2，4.3所示。

表4.2 留言信息数据表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 默认值 | 是否必须 |
| 1 | title | 字符串（String） |  | true |
| 2 | textarea | 字符串（String） |  | true |
| 3 | date | 字符串（String） | Date.now | true |
| 4 | user | 对象（Object） |  | true |
| 5 | children | 数组（Array） |  |  |

表4.3 留言信息children数据表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 默认值 | 是否必须 |
| 1 | textarea | 字符串（String） |  | true |
| 2 | date | 字符串（String） | Date.now | true |
| 3 | user | 对象（Object） |  | true |
| 4 | replyUser | 对象（Object） |  | true |

### 4.1.3 算法游戏数据表（vedio）

算法游戏数据表 用来存储用户上传的算法游戏的视频的信息，具体的字段名如下：name（小组成员），date（上传时间），newpath（视频的存储地址），arr（已打分人员信息），score（得分信息），具体结构如表4.4所示。

表4.4 算法游戏数据表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 默认值 | 是否必须 |
| 1 | name | 字符串（String） |  | true |
| 2 | date | 字符串（String） | Date.now | true |
| 3 | newpath | 字符串（String） |  | true |
| 4 | arr | 数组（Array） |  | true |
| 5 | score | 数组（Array） |  | true |

### 4.1.4 题库数据表

题库的数据库表用来存储所有的题目信息，由于系统设置三类练习难度，所以使用不同难度的数据表将题目分类，分别为简单（xeasy），适中（xmiddle），困难（xdifficult），具体字段设置如下：difficult（题目的难度），title（题目的题干），选项为A，B，C，D。answer（正确答案选项），date（系统录入本题目的时间）。编程题目的数据表设置分为besay和bdifficult两个难度。具体字段和选择题目的数据库设置类似，不设置A，B，C，D四个选项，仅设置answer字段信息。具体结构如表4.5所示。

表4.5 题库信息数据表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 默认值 | 是否必须 |
| 1 | difficult | 字符串（String） |  | true |
| 2 | title | 字符串（String） |  | true |
| 3 | A | 字符串（String） |  | true |
| 4 | B | 字符串（String） |  | true |
| 5 | C | 字符串（String） |  | true |
| 6 | D | 字符串（String） |  | true |
| 7 | answer | 字符串（String） |  | true |
| 8 | data | 数值 | Date.now | true |

### 4.1.5 历史练习信息数据表

历史练习信息数据表用来存储学生用户的历史练习信息，用来向用户展示已经做过的练习信息，以便以后进行复习，具体字段名设置如下：list（练习的具体信息，为一个数组），user（答题用户信息），使用数据表关联，将用户数据表和练习信息进行关联。score（得分信息），difficult（练习的难度分类），date（练习存储的时间记录），具体结构表如表4.6所示。

表4.6 历史练习信息数据表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 默认值 | 是否必须 |
| 1 | list | 数组（Array） |  | true |
| 2 | score | 字符串（String） |  | true |
| 3 | user | 对象（Object） |  | true |
| 4 | difficult | 字符串（String） |  | true |
| 5 | date | 数值（Number） | Date.now | true |

### 4.1.6 系统中数据表关系

在本系统中共有五类数据表，其中的user数据表关联历史练习数据表和留言数据表。具体关系图如图4.1所示。

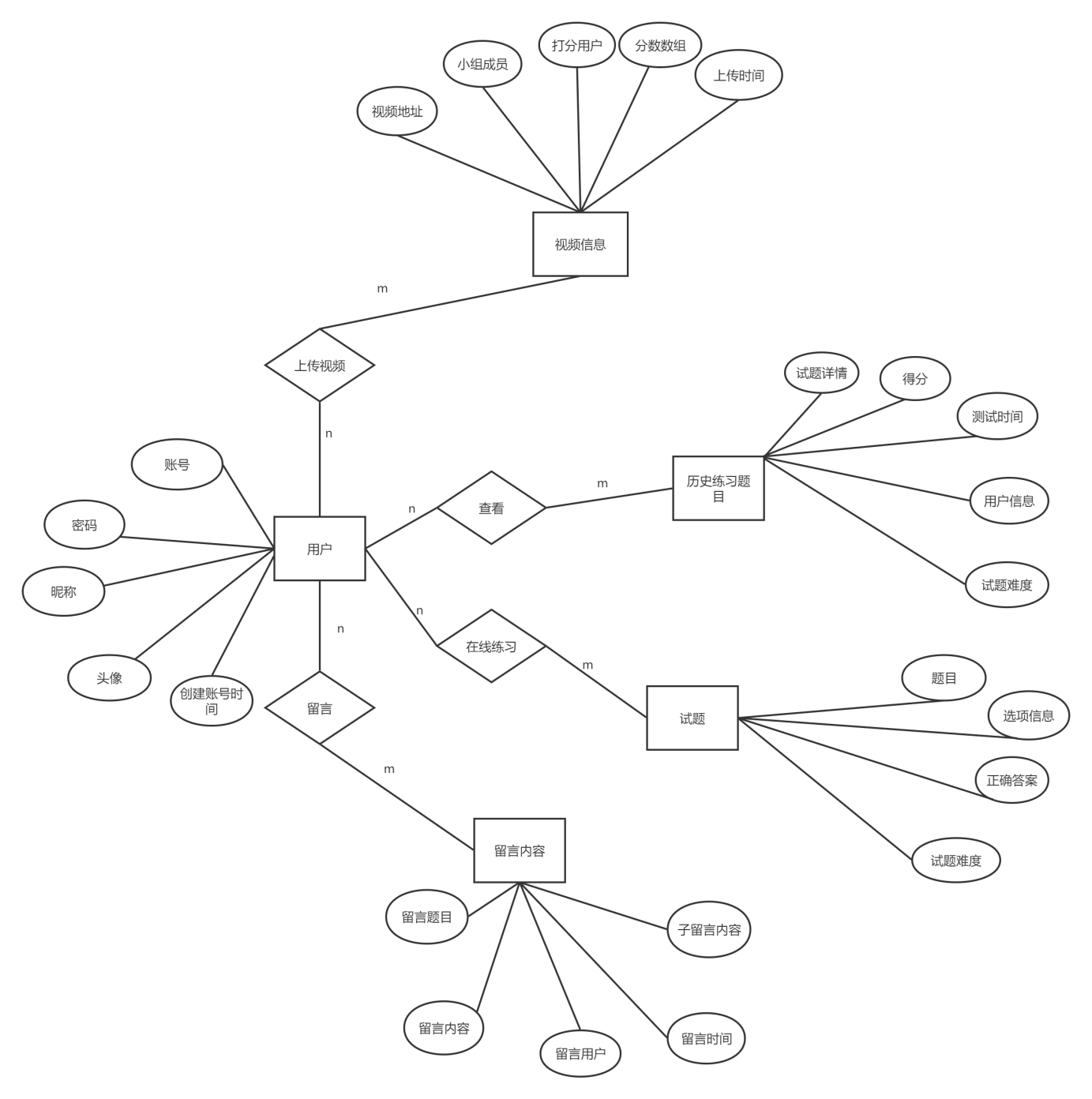


图4.1 系统数据表联系图

## 4.2 登录模块的设计和实现

在本系统中用户输入网页地址后，首先映入眼帘的是本系统的登陆界面。在设计之初，选择在本页面做出一点动态效果，以达到吸引用户的目的。本页面中主要是负责系统验证用户权限和注册学生用户。由于是客户端直面用户，所以设置了一些必要的交互效果提示。比如注册账号时，账号的格式输入不正确，在输入框失去焦点时，会进行验证，验证不通过，输入框的边框颜色变红。注册时需要输入两次密码，前后密码输入不正确，输入框会变红，并在输入框下方出现提示信息。在注册成功后，页面中会弹出用户账号注册成功的提示信息。并且页面从注册界面变为登录账号界面。在登录账号时，在前端需要进行数据格式的验证，在后端服务器中也需要进行验证，目的是防止一些通过浏览器调试工具的用户直接在控制台中进行数据发送，从而跳过前端的数据验证。登陆成功后会在vuex中进行用户信息的存储，在本系统中后续需要向后台传输身份信息时都在vuex中的user信息表中拿取身份信息。

页面的星空特效使用particles插件实现，登录或者注册的表单的验证和提交使用ElementUI实现。具体的核心思想如下。

表单的数据格式设置采用正则表达式实现，正则表达式简单来说就是一串文本字符，可以使用test方法来判断字符串是否合乎开发者指定的规则。在这里指定的账号规则是12位的数子，密码的规则是6-18位字符串，字符串中可以包括数子，字母和“,.?;<>/|\\:'"!@#$%^&\*()+-”符号。

密码的正则规则：let reg = /^[\w,.?;<>/|\\:'"!@#$%^&\*()+-]{6,18}$/;

账号的正则规则：let reg = /^[\d]{12,12}$/;

具体的验证效果实现使用el-form中的validator函数。如果验证不通过进行具体信息的提示。在字符串的规则验证通过后，点击登录按钮，则进行数据的发送，建立前后端的数据通信，后端接收数据，进行数据库查询，user信息表中是否有该账号，以及输入密码是否和数据表中存储的字段信息相同。

具体的数据发送使用axios技术实现，在本系统中涉及到发送或者请求数据的代码皆是通过axios实现，请求数据方式使用Get方式，发送数据使用Post方式。后续关于发送请求数据的过程不再赘述。核心代码如下。

|  |
| --- |
| this.$refs[loginFrom].validate(async (valid) => {          if (valid) {            let { data } = await this.$axios({              method: "POST",              url: "/login",              data: this.loginFrom,            });             if (data.code) {              return this.$message.error(data.msg);            }            this.$message({              message: "用户登录成功",              type: "success",              duration: 1000,            });            this.updateuser(data.data)            //关闭登录页面            this.close();            this.$router.replace('/')          } else {            this.$message.error("请输入正确的用户名或密码");            return false;          }        }); |

## 4.3 题库管理模块的设计和实现

题库管理具体分为删除题目信息和增加题目信息。在本系统中增加题目信息的方式分为单个题目上传和Excel文件上传。在练习的难度处理时采用多个不同难度的数据库组合抽题的方式，所以在上传题目的时候是需要特别标明本次上传题目的难度。选择上传单个题目时，在第一行的有三个难度的按钮，在上传时进行选择即可。如果选择Excel文件上传文件，则需要对Excel文件按照特定格式进行编辑。方便系统解析Excel文件进行解析。在解析文件过程中使用js提供的FileReader对象，利用readAsBinaryString()方法将文件转为二进制字符串，使用xlsx插件进行解析。通过xlsx提供的方法将二进制字符串转换为json格式数据。具体的代码如下。

|  |
| --- |
| //使用FileReader对象读取文件，并转为二进制   let reader = new FileReader          reader.readAsBinaryString(file)          reader.onload = ev => {              resolve(ev.target.result)          }  //将二级制字符串变为json数据输出。  let workbook = XLSX.read(data, { type: "binary" }),          worksheet = workbook.Sheets[workbook.SheetNames[0]];        data = XLSX.utils.sheet\_to\_json(worksheet); |

在使用文件上传时，可能会因为一些错误编辑，使得文件中的题目重复，所以在客户端的代码运行中需要对上传的数据进行去重，去重算法在这里不再赘述。在后台接收到数据后，首先还是要检测本次上传的数据在数据库中是否存在，如果存在则对重复题目不进行储存，不存在则进行储存。不只是使用Excel文件上传题目时需要进行去重，在进行单个题目上传时，也是需要查看数据库中是否已经存在本次上传的题目，若存在需要给客户端用户一个提示信息。

在上传题目后，可以在题目的展示页面进行查看，如果上传错误或者时题库中的题目已经不适合如今的教学发展，则可以对本题目进行删除的操作。前边已经说过MongoDB数据库的唯一值为id，在需要进行删除时，将id值传给后台，系统进行数据库的查找，删除工作。具体代码不在这里展示。题库管理界面如图4.2所示。

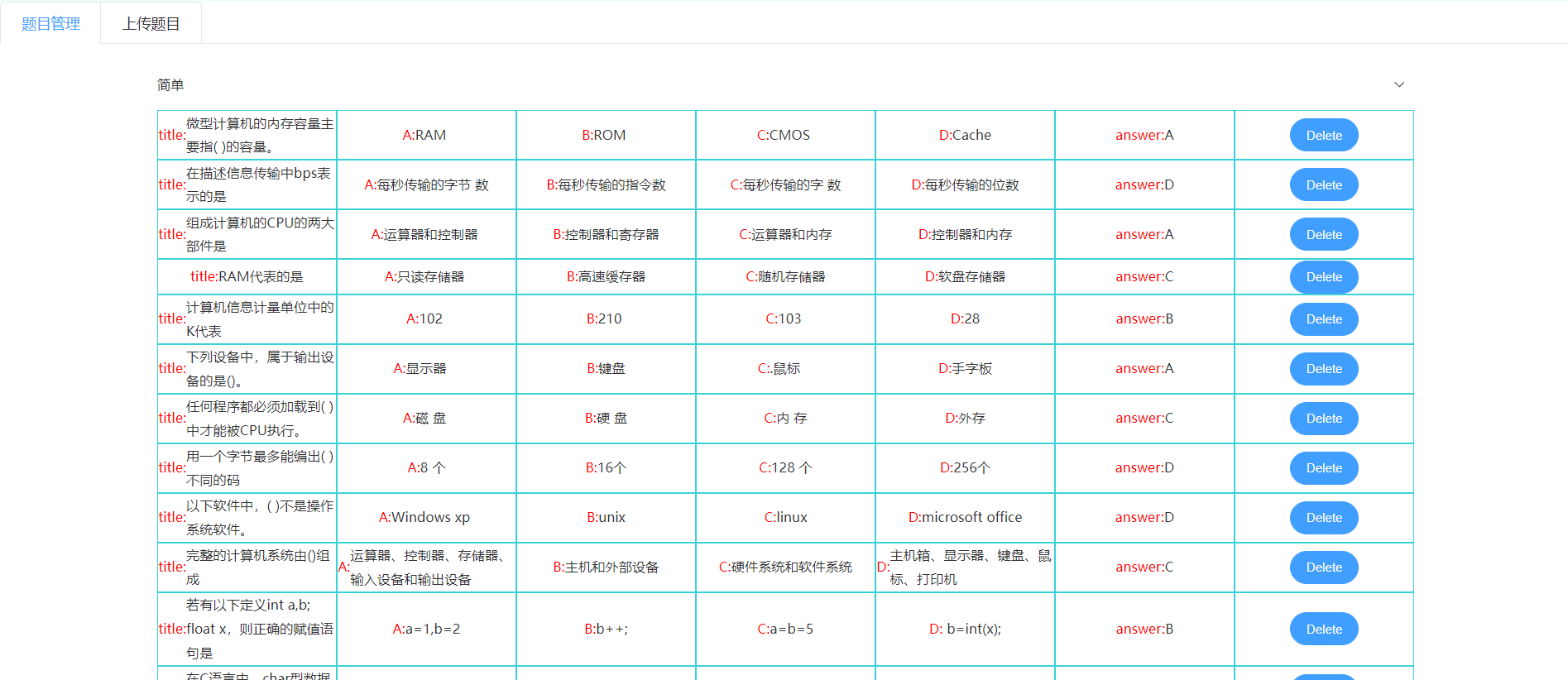


图4.2 题库管理界面

## 4.4 在线练习模块的设计和实现

本页面提供的主要功能是在线练习，在线练习分为简单、适中、困难三个等级，学生用户可以根据自身的学习程度来选择练习的难度。在选择难度后，首先进入考生须知界面，在这里会看到，练习分为两部分，一部分是选择题，另一部分是编程题目。进入练习的界面后，会有一个倒计时功能，在规定时间内，需要完成本模块中的题目，如果倒计时为零，则系统自动提交本模块的练习题。如果在倒计时没有完毕前提交练习题，系统会检测用户有没有将本模块的题目全部完成，如果没有完成全部题目，系统会弹出提示选择框，提示用户是否确认提交，并且会在提示信息中具体显示到底是那个题目没有完成。用户可以选择确认提交或者取消提交，继续答题。倒计时的实现如下：设置一个总时间的秒数值，通过封装时间处理函数，将时间的秒数值显示在客户端的页面中。在页面创建时启动无限循环的定时器，定时器的规则是每隔一秒，总时间的秒数值进行减一。然后在页面中监听时间的秒数值，单时间的秒数值变为零时，页面直接进行跳转。由于在进入页面时使用的replace的路由跳转方式，所以本页面不会记录在浏览器的地址记录当中，所以在练习时没有用户可以回退至上次的答题界面。在路由跳转的时候，系统将本模块的答题详情记录在本次的练习答题详情数据中，在两个模块中的试题都完成的时候，统一发送至后台。倒计时的具体代码实现如下。

|  |
| --- |
| Time() {        setInterval(() => {          if (this.seconds > 0) {            this.seconds -= 1;            this.countDown();          } else {            this.count = `00分00秒`;          }        }, 1000);      }, |

在答题完毕后，得分的计算实现在客户端，在创建界面时从后台拿到练习信息，答题完毕后将练习信息中的正确答案和考生答案进行比较，从而计算出练习得分。在线练习界面如图4.3所示。



图4.3 在线练习界面

## 4.5 成绩展示模块的设计和实现

在本模块使用ECharts开源库，通过对数据库中的成绩数据进行分类，形成特定数据格式，通过ECharts提供的图形组件和API进行页面中图形的渲染。作用主要是通过所有的学生用户的练习成绩来大体了解学生的学习情况。主要设置两个小模块。整体成绩的饼状图展示，所有学生用户的具体成绩和练习展示，以列表形式展示，练习详情展示采用个人中心历史练习同样的方式。成绩展示模块界面如图4.4所示。

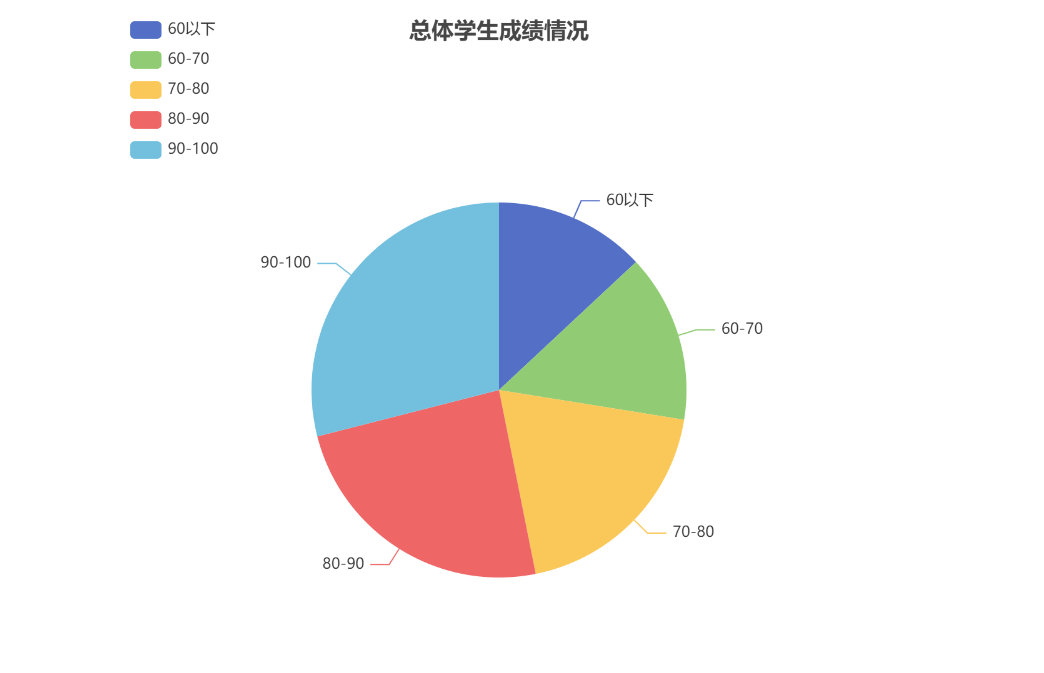


图4.4 成绩展示模块饼状图界面

## 4.6 个人中心模块的设计和实现

一个系统不可避免的就是个人信息的修改，本模块不仅提供个人信息的修改，还提供用户个人历史练习信息的查看。

个人信息的修改包括更改用户的头像，昵称，用户账号密码。上传头像类似与上传视频。修改用户名和修改密码类似，将用户的id信息，发送到后台。根据id信息查找到用户信息，对数据对象进行更新。

历史练习信息，每个用户在使用系统的过程中，练习信息必定大于一。所以在设计数据表时，将一个练习中的单个题目，答案作为一个对象，将所有的对象存储在arr数组中。展示时需要对arr数组进行处理。个人中心界面如图4.5所示。

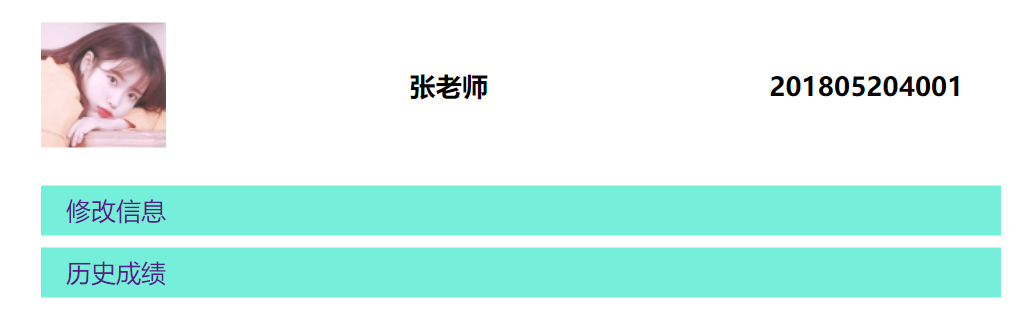


图4.5 个人中心界面

## 4.7 算法游戏模块的设计和实现

设置本模块是为了增加学生们的学习乐趣，为了让学生可以有一个更容易理解算法的核心思想。具体实现就是学生以小组为单位，录制关于一个算法的实现的视频，进行上传，其他同学可以对该视频，进行打分。在对该模块的数据库设置时考虑到自己小组用户不能对自己小组的作品进行打分，已经打分的人员不能进行二次打分。所以增加了一个存储已经打分的用户的数组。再进行打分时，需要两次判断验证，第一次是判断打分用户是否是本小组成员，第二次是判断该用户是否已经打分。这两次判断都交予前端，可以减轻后台的负担。不论打分失败还是成功，客户端用户都可以看到提示信息。每个作品的最后得分为平均分。具体实现方式为分数数组的总和来除以分数数组的长度。

在上传作品时，需要对小组成员同时上传。在这里同样使用正则表达式进行数据规则设置。由于作品为视频，但是数据库中是不可以存储视频文件的，所以处理方式是在系统文件夹中，新建一个用来储存视频文件的vedio文件夹。数据库中存储的是文件的相对路径。上传文件使用el-upload组件进行视频文件的上传。在上传进度（UploadProgressEvent）函数中得到file.percent来处理得到文件的上传进度。在后台通过multer工具设置在磁盘中存储文件。设置文件存储地址，文件存储时的名称中添加当时时间戳，防止文件夹中的文件名称重复。具体实现代码如下。

|  |
| --- |
| const upload = multer({    storage: multer.diskStorage({      destination(req, file, cb) {        cb(null, path.join(\_\_dirname, "../../public/files/vedio"))      },      filename(req, file, cb) {        let ext = '.mp4'        let fileName = Date.now().toString(36) + ext        req.fileName = fileName        cb(null, fileName)      }})  }).single("file") |

算法游戏界面如图4.6所示。



图4.6 算法游戏界面

## 4.8 留言模块的设计和实现

本界面为用户提供一个学习交流的平台，在这里主要是两个界面模块，一个是提交留言的模块，一个是展现所有留言的模块。可以在右边的按钮处关闭提交留言的模块。提交信息的提交信息方式和登录模块相似，采用表单提交。不同的地点是，留言需要进行留言时间的展示，所以在后台创建数据库时，记录了创建本留言的时间戳，经过封装时间展示格式函数处理，变为用户所见的时间格式。时间戳处理函数如下。

|  |
| --- |
| export default function formdate (val){      let date = new Date(val),          Y = date.getFullYear(),          M = date.getMonth()+1,          D = date.getDate(),          hh = date.getHours(),          mm = date.getMinutes(),          ss = date.getSeconds();      M  < 10 && (M  = '0' + M)      D  < 10 && (D  = '0' + D)      hh < 10 && (hh = '0' + hh)      mm < 10 && (mm = '0' + mm)      ss < 10 && (ss = '0' + ss)       return `${Y}-${M}-${D} ${hh}:${mm}:${ss}`  } |

在留言展示模块中针对每个一级留言，还可以对每一个一级留言进行回复，为了优化用户体验，初始状态下，每个一级留言下的输入框是不可见的。为了添加可以控制单个输入框显示的按钮，在页面创建时请求到的留言对象下使用filter方法为每个留言对象添加三个属性，分别为ifShowReply（布尔类型，初始值为false），reply（字符类型），replayUser（对象类型）。ifShowReply用来控制每个留言框是否显示，reply用来储存每个留言的回复内容，replayUser用来存储回复的是谁。增加了属性之后，就可以对每个一级留言进行响应的交互操作。在提交子留言时，将数据发送至后台，系统对数据进行解析，查找数据表并进行更改。因为MongoDB数据库在生成每条数据信息时都会赋予一个唯一的id值，所以通过唯一值id进行数据库的查找和更新，核心代码如下。

|  |
| --- |
| textareaDB.findByIdAndUpdate(\_id,{              $push:{                  children:{                      textarea,                      user:userId,                      replyUser:replayUserId                  }}}) |

数据更新成功，通过客户端的消息提示给予用户提示信息，然后再对留言模块的信息重新请求，拿到数据后进行页面刷新。留言界面如图4.7所示。



图4.7 留言界面

## 4.9 用户管理模块的设计和实现

用户信息是一个系统中相对重要的数据，所以在设计之初，便设定为只有管理员可以进行用户的信息管理，通过管理界面可以对数据库进行增删。

代码中主要通过js中使用MongoDB的一些数据库操作语句进行实现。用户管理实现界面如图4.8 所示。

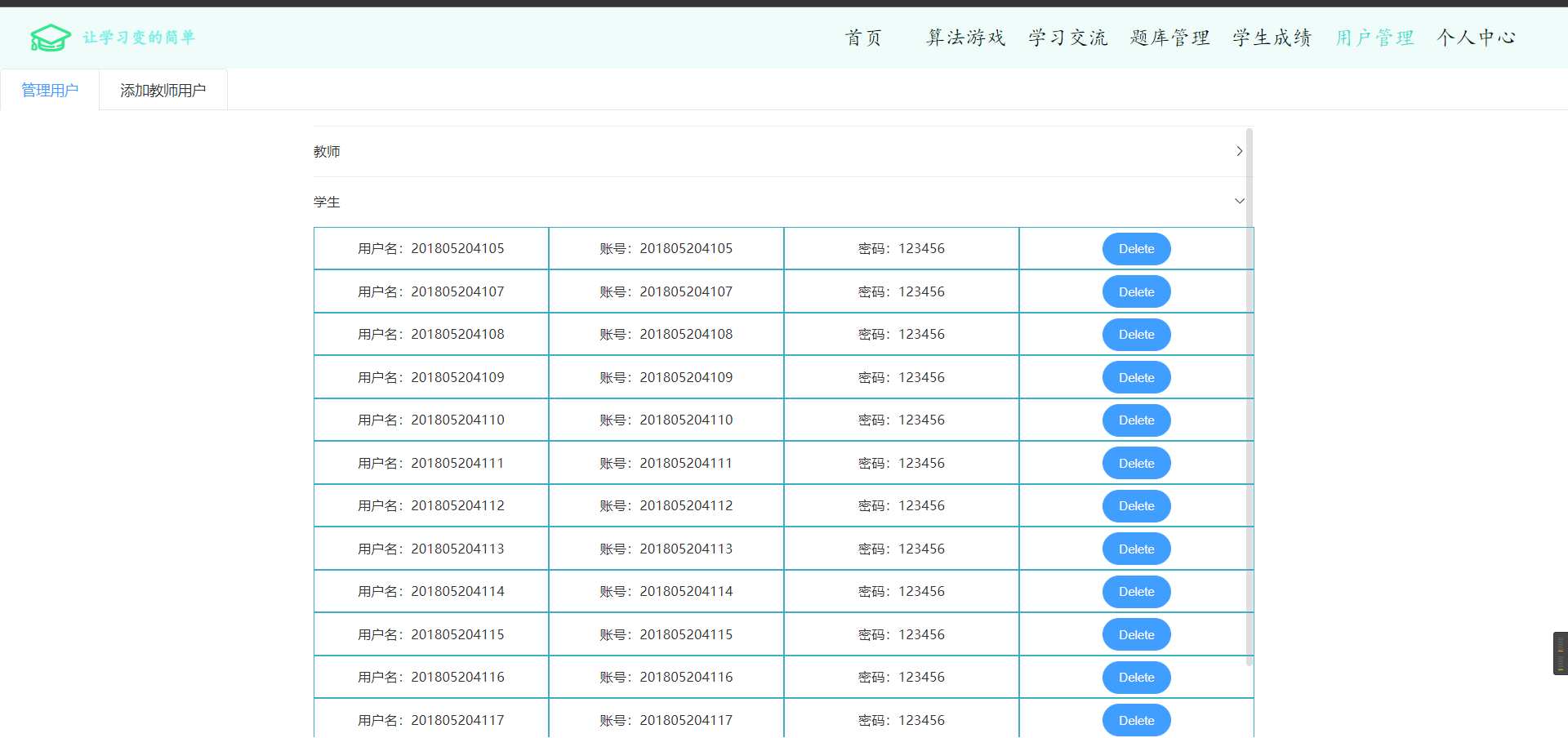


图4.8 用户管理界面

# 5 系统测试

## 5.1 系统连接测试

### 5.1.1 前后端连接测试

由于客户端和服务端使用同一个ip地址，不同端口号，所以在前后端连接需要做跨域设置，在进行连接后可以进行数据发送或者接收。在这里使用发送数据的方式练习前后端是否连接。

在登录界面输入账号，密码。在服务器端使用代码输出客户端传输的变量，查看变量是否存在。登录图如5.1，变量检验图如5.2。



图5.1 登录图

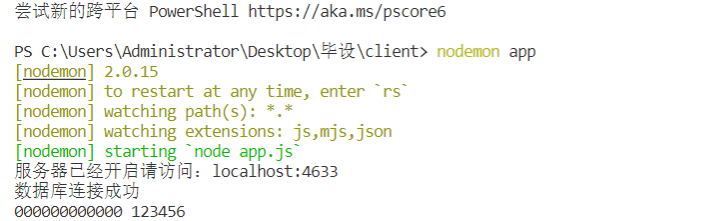


图5.2 变量验证图

综上所示，系统前后端连接正常。

### 5.1.2 数据库连接测试

在Node中连接MongoDB数据库，可以使用输出提示信息来检验系统是否和数据库连接正常。在代码中写入连接数据库代码，在终端中启动系统，然后在终端中查看提示信息，若显示‘数据库连接成功’，则系统数据库连接成功。系统启动终端图如5.3。

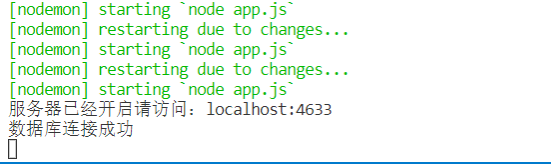


图5.3 系统启动终端图

## 5.2 系统功能测试

### 5.2.1 输入数据检测功能测试

在本系统中用户在客户端中提交信息后，系统需要对提交的信息进行验证。比如提交数据的格式，数据长度，数据类型等。

在这里练习用例为，用户密码验证。在登录模块的设计和实现中提到了密码的验证规则，这里练习的是错误输入后，客户端的输入检测功能和提示显示。如图5.4所示。



图5.4 密码输入验证和用户提示信息图

### 5.2.2 数据提交功能测试

本系统中，数据提交采用JSON格式发送，数据提交后进行数据库的相关操作，例如存储。在这里使用留言的提交为例，在用户输入想要留言的内容后，点击提交后可以在数据库中找到相应的数据，提交成功后，留言面板中便有本次留言内容。

例如留言题目为：数据提交，留言内容为：本次数据提交仅供练习留言提交是否正确，如图5.5。

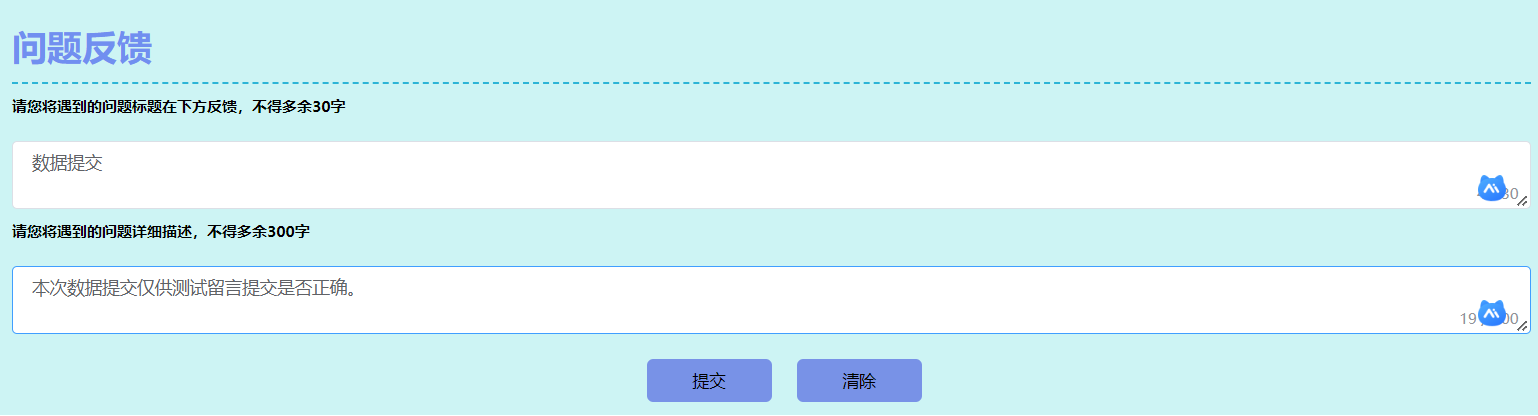


图5.5 客户端留言提交

点击提交后在数据库可视化软件中显示如图5.6所示。

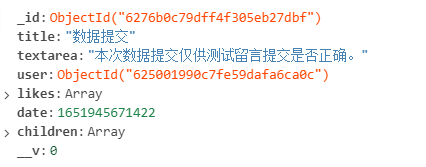


图5.6 留言提交数据库显示

存入数据库成功后，重新拿取信息，在留言面板中显示，如图5.7所示。

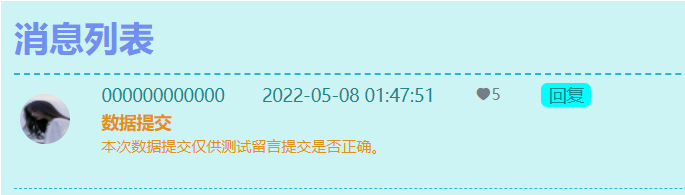


图5.7 留言面板显示图

### 5.2.3 文件提交功能测试

本系统中在算法游戏和选择题目Excel文件上传使用文件形式上传，这里使用算法游戏上传练习为例。在用户选定文件后，点击按钮，系统会对文件进行上传，如果上传成功，可以在后台指定的文件夹下出现本次上传文件，上传成功，客户端进行提示。上传文件界面如图5.8，文件上传成功后的界面如图5.9，文件夹下的文件如图5.10。

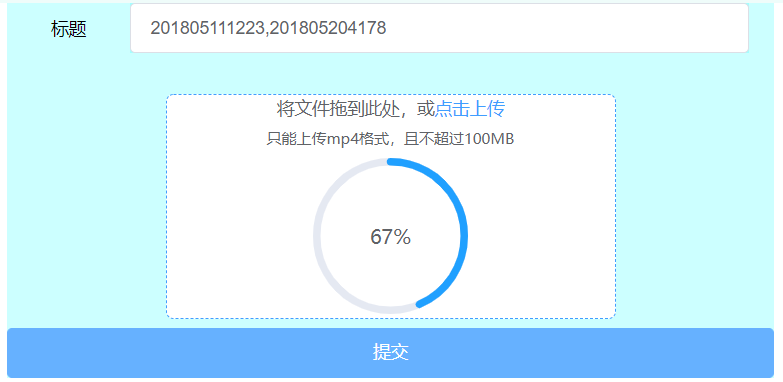


图5.8 上传文件界面

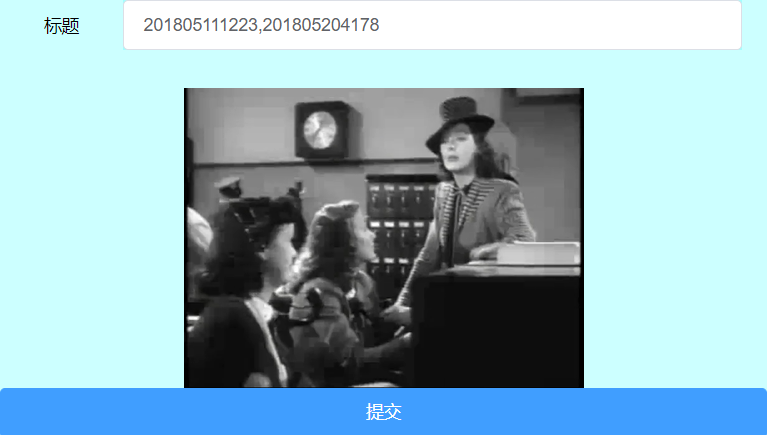


图5.9 文件上传成功后的界面



图5.10 文件夹下的文件

### 5.2.4 数据管理功能测试

本系统中数据管理包括题库中题目管理，用户管理。管理分为上传和更改，在这里使用上传编程题目为例。在客户端界面中输入题目，答案，选择题目难度后点击提交，如果上传成功，则在数据库中可见。上传界面图如图5.11，数据库图如图5.12。

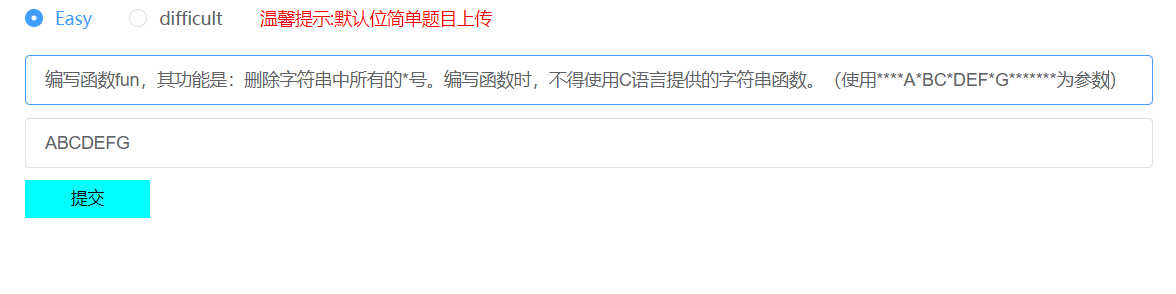


图5.11 上传界面图

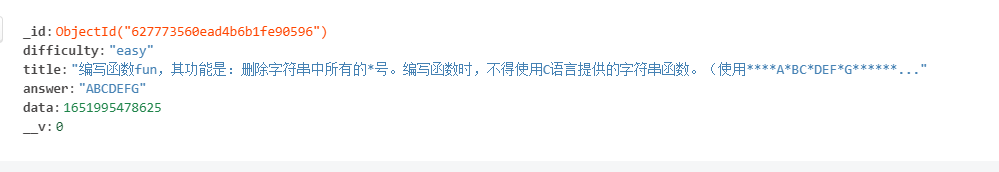


图5.12 上传后数据库图

### 5.2.5 在线练习功能测试

在进入在线练习界面后，会有题目数据的显示，倒计时功能，提交时信息提示，得分计算，在此模拟真实用户进行练习。进入首页选择试题难度，进行练习，如果在没有完成本次练习所有题目的情况下就进行提交练习操作。答题界面如图5.13，提交题目界面如图5.14。

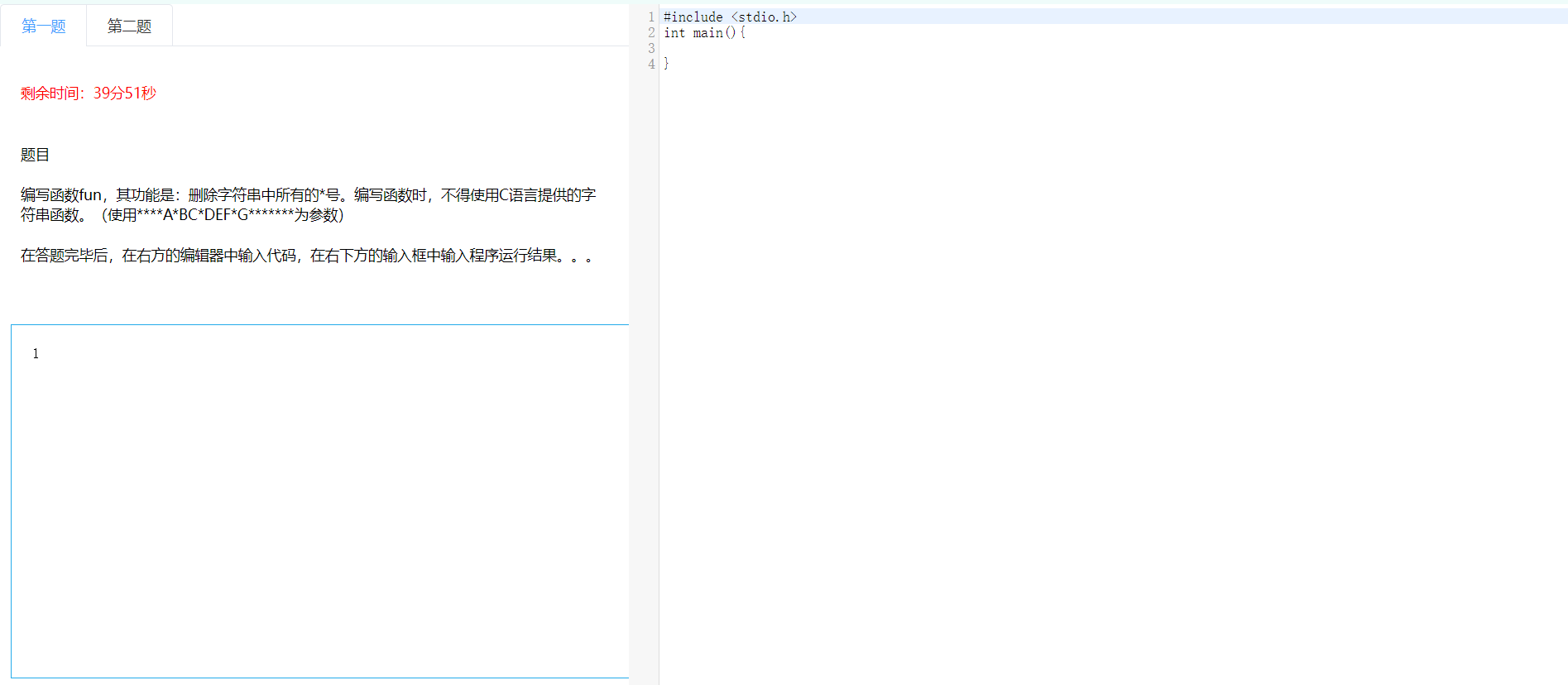


图5.13 答题界面

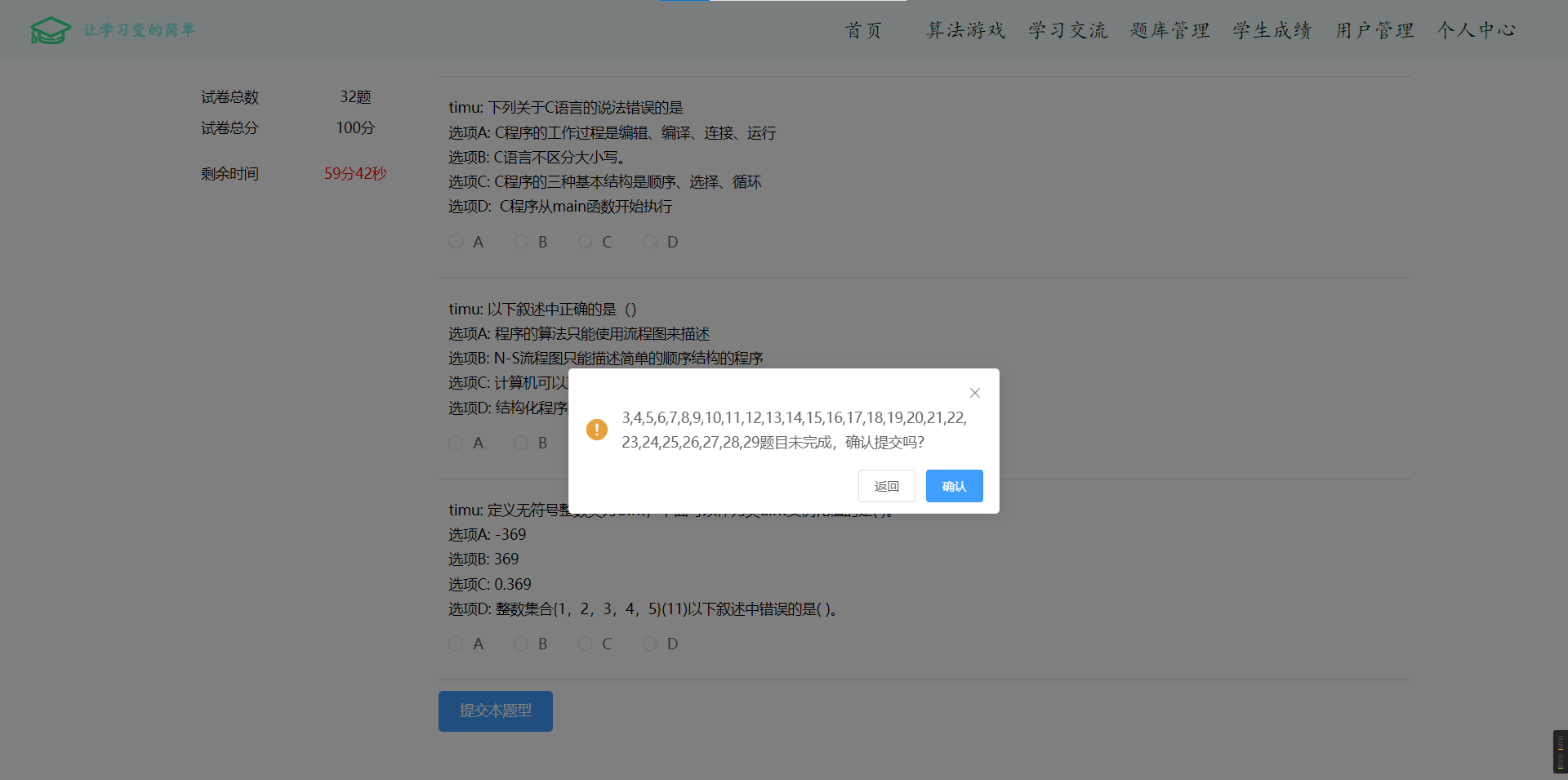


图5.14 提交练习界面

## 5.3 兼容性测试

本系统在手机端和PC端都可以使用，在这里分别对PC端和手机端的界面显示进行练习，在这里使用首页和单个题目上传在PC端和手机端用户界面作为示例。

### 5.3.1 PC端显示测试

首页界面图如图5.15所示，单个题目上传界面如图5.16所示。



图5.15 首页界面图



图5.16 留言交流图

### 5.3.2 手机端显示测试

首页界面图如图5.17所示，单个题目上传界面如图5.18所示。

图5.17 首页界面图 图5.18 留言交流图

### 

# 结 论

本文基于web开发技术，前端使用vue，后台使用nodejs，完成了本系统的设计开发。在系统实现的过程中本人学到了很多知识，让我对开发一个完整系统有了全面的了解，同时还学习了非关系型数据库MongoDB的使用，对数据库结构设计，操作有了更深的体会。系统的实现建立在精准的需求分析基础之上，以模块逻辑分析为踏板，实现了在线练习，系统组卷，用户留言，算法游戏的上传，打分，个人信息修改，历史练习查看，题库题目管理，学生成绩展示，用户管理的功能。系统改变了传统的学习模式，学习由线下走向线上，很大程度上解决了老师的重复工作量，解决了学生和教师时间不统一的问题。但是系统还有改进空间，例如：首页第一次渲染效率、页面不够精美、手机端操做不方便、编程题打分不够完善。随着技术能力的增长，我相信系统会越来越完善。总体来说系统各模块运行稳定、功能实现无误。