**课程设计报告**

**2025年 1月 7日**

目 录

目录

[课程设计选题 6](#_Toc187499964)

[1.系统需求分析 7](#_Toc187499965)

[1.1问题定义 7](#_Toc187499966)

[1.2系统功能需求与非功能需求 7](#_Toc187499967)

[1.2.1 学生的模块系统需求 7](#_Toc187499968)

[1.2.2管理员功能模块系统需求 8](#_Toc187499969)

[1.2.3授课老师功能模块系统需求 8](#_Toc187499970)

[1.2.4非功能需求 8](#_Toc187499971)

[1.3系统流程分析与逻辑模型 9](#_Toc187499972)

[1.3.1系统流程分析 9](#_Toc187499973)

[1.3.2数据流图（DFD） 9](#_Toc187499974)

[2.系统总体设计 11](#_Toc187499975)

[2.1系统设计方案 11](#_Toc187499976)

[2.2系统功能结构 12](#_Toc187499977)

[2.3数据库实体设计 13](#_Toc187499978)

[2.4数据库表结构设计 15](#_Toc187499979)

[3.系统主要功能实现 18](#_Toc187499980)

[3.1主要模块处理流程 18](#_Toc187499981)

[3.2学生用户登录 19](#_Toc187499982)

[3.3授课教师功能模块 19](#_Toc187499983)

[3.3.1教师登录 19](#_Toc187499984)

[3.3.2学生成绩管理 19](#_Toc187499985)

[3.4管理员功能模块 19](#_Toc187499986)

[3.4.1管理员登录 19](#_Toc187499987)

[3.4.2学生和教师信息管理 19](#_Toc187499988)

[结 论 20](#_Toc187499989)

课程设计选题

**编号：007**

#### 设计题目: 基于B/S模式的学生成绩管理信息系统的设计与实现

**1.课程设计任务**

1、设计内容

设计一个学生成绩信息管理信息系统，该系统的用户分别是：学生用户、授课教师和系统管理员。不同的用户拥有不同的权限，各自完成各自的管理功能，不同的用户看到不同的系统功能。创建后台数据库，然后编写程序实现对数据库的操作，按照要求完成所有的功能和模块。学生用户的主要功能模块：

（1）学生的登陆与注册

（2）学习成绩的查询

（3）与授课教师的在线交流

管理员的主要功能模块：

（1）学生帐户和教师帐户信息的管理

（2）学生成绩的管理（添加、删除、修改、查询）

（3）在线交流平台的管理

授课教师的主要功能模块

（1）授课教师的登陆和注册

（2）学生成绩的管理（添加、修改、查询）、与学生在线交流

2、主要数据库表单

（1）学生信息表单

主要字段有：学号、姓名、帐号、性别、学院、专业、联系方式、家庭住址

（2）教师情况表单

主要字段有：工号、教师姓名、性别所属系、职称、授课编号

（3）课程安排表单

主要字段有：课程号、课程名、学分、学期、学时

（4）学生选课情况表单

主要字段有：学号、课程号、成绩

**2. 课程设计报告书主要内容**

1、软件需求规格说明书；（不少于15页）

至少包括：系统功能需求、非功能需求。

2、软件设计说明书；（不少于15页）

至少包括：1）模块设计；2）用户界面设计；3）数据库设计

**1.**系统需求分析

## 1.1问题定义

学生成绩管理系统是一个集成了多项关键功能的综合性平台，旨在全面提升学校教学管理的效率与便捷性。其核心在于实现成绩的高效管理，并围绕这一核心，进一步细分为多个子系统以满足不同用户角色的需求。首先，系统设计了完善的身份系统和登录功能，这是确保信息安全与用户权限管理的基础。通过这一机制，系统能够明确区分学生、教师、管理员等不同身份的用户，为他们提供定制化的服务界面与操作权限。用户需通过注册流程建立个人账户，并在后续使用中通过登录验证身份，确保操作的合法性与数据的保密性。下面是具体描述：

1，对于学生而言，学生系统功能的设计充分考虑到了他们的实际需求。学生不仅能够方便地查询自己的考试成绩，实时了解自己的学业进展，还能通过内置的通讯模块与任课老师进行直接对话，就课程内容、作业疑问等事宜进行沟通，这极大地促进了师生之间的交流，有助于学习效率的提升。

2，教师系统则侧重于成绩管理的深化应用。教师不仅可以轻松查询、修改及增加学生的成绩记录，实现对学生学业表现的全面追踪与管理，还能通过系统平台与学生进行即时通讯，解答疑问，提供个性化学习建议，从而增强教学效果，促进师生互动。

3，管理员角色在系统中扮演着至关重要的角色。他们拥有最高级别的权限，能够对学生成绩进行全方位的增删改查操作，确保数据的准确无误。此外，管理员还能对学生和教师的个人信息进行管理与维护，资料更新，为整个校园信息管理体系的稳定运行提供坚实保障。

综上所述，学生成绩管理系统通过整合身份认证、成绩管理、即时通讯等核心功能，构建了一个高效、安全、互动的教学管理平台。它不仅解决了传统成绩管理方式中存在的繁琐与低效问题，还极大地促进了教学资源的优化配置，提升了授课效率与学习体验，为学校的现代化教学管理注入了新的活力。

## 1.2系统功能需求与非功能需求

### 1.2.1 学生的模块系统需求

一、成绩查询功能

1. **实时成绩查询**：学生能够即时查询自己的各科成绩，确保信息的时效性。

可根据课程搜索自己的成绩。

二、在线交流功能

1. **一对一私信**：学生可根据课程与授课老师进行一对一的在线私信交流，方便询问课程内容、作业疑惑、学习建议等个性化问题。
2. **交流记录保存**：所有在线交流记录自动保存，便于学生随时回顾，同时也作为师生沟通的历史证据，有助于解决可能的误解或争议。

### 1.2.2管理员功能模块系统需求

一、学生和教师信息管理

1. **全面信息查询**：管理员能够查询所有学生和教师的详细信息，包括但不限于姓名、学号/工号、学院/院系、联系方式等，确保信息的全面性和准确性。
2. **信息批量修改**：支持对学生和教师信息进行批量修改操作，如更新联系方式、调整学院/院系归属、修改姓名、职衔和性别等，提高管理效率。

二、成绩管理

1. **成绩查询**：管理员能够查询任意学生的所有课程成绩，支持按姓名条件进行筛选。
2. **成绩批量修改**：在特殊情况下（如成绩录入错误、课程调整等），管理员可对成绩进行批量修改，确保成绩的准确性和公正性。
3. **成绩删除**：对于因故需要删除的成绩记录，管理员可执行删除操作。
4. **成绩添加**：管理员可手动添加学生成绩并确保成绩不会超出合理数据的范围。

### 1.2.3授课老师功能模块系统需求

一、成绩管理

1. **实时成绩录入与更新**：授课老师能够实时录入和更新学生的自己所授课程的成绩，确保成绩的时效性和准确性。
2. **成绩查询**：授课老师能够查询自己所授课程的所有学生的课程成绩，支持按姓名条件进行筛选。
3. **成绩批量修改**：在特殊情况下（如成绩录入错误、课程调整等），授课老师可对成绩进行批量修改，确保成绩的准确性和公正性。
4. **成绩删除**：对于因故需要删除的成绩记录，授课老师可执行删除操作。
5. **成绩添加**：授课老师可手动添加学生成绩并确保成绩不会超出合理数据的范围。

二、学生互动与评估

1. **在线问答与讨论**：授课老师可根据课程对学生发起在线问答，解答学生疑问，促进师生互动，提高教学效果。
2. **交流记录保存**：所有在线交流记录自动保存，便于学生随时回顾，同时也作为师生沟通的历史证据，有助于解决可能的误解或争议。

### 1.2.4非功能需求

1. **系统确保输入框校验的严谨性，从而确保后端得到的数据一定正常：**  
   系统在设计时应考虑数据输入的正确性和安全性，通过前端严格的输入框校验机制，如格式验证、必填项检查、非法字符过滤等，有效避免无效或恶意数据的提交。
2. **系统采用路由跳转和异步页面加载技术，提高系统效率：**  
   为了实现更流畅的用户体验和更快的响应速度，系统应该引入先进的路由管理和异步页面加载机制。用户在进行页面切换时，系统能够迅速响应并加载所需内容，而无需重新加载整个页面，这大大减少了等待时间。同
3. **系统应该实施前后端分离的开发策略，降低维护成本并加快开发进度**：  
   前后端分离是现代Web开发中的一种重要架构模式，它将前端用户界面与后端服务逻辑解耦，使得两者可以独立开发、部署和迭代。这种策略不仅提高了系统的可扩展性和灵活性，还显著降低了维护成本。

## 1.3系统流程分析与逻辑模型

### 1.3.1系统流程分析

1. 主要参与者

* 学生：注册、登录、查看成绩。
* 教师：登录、录入和修改成绩、查询成绩。
* 管理员：登录、管理用户。

1. 主要功能

* 学生管理：学生能注册、登录、查看自己成绩。
* 成绩管理：教师能够录入和修改学生成绩。
* 交流功能：教师和学生能够进行在线交流。

### 1.3.2数据流图（DFD）

数据流图通过图形化方式展示了不同实体之间的数据流动关系。以下基于B/S模式的学生成绩管理信息系统的数据流图描述：

**1. 数据流图的层级**

**层次0（上下文图）**：

* 展示系统与外部实体（学生、教师、管理员）的基本交互。

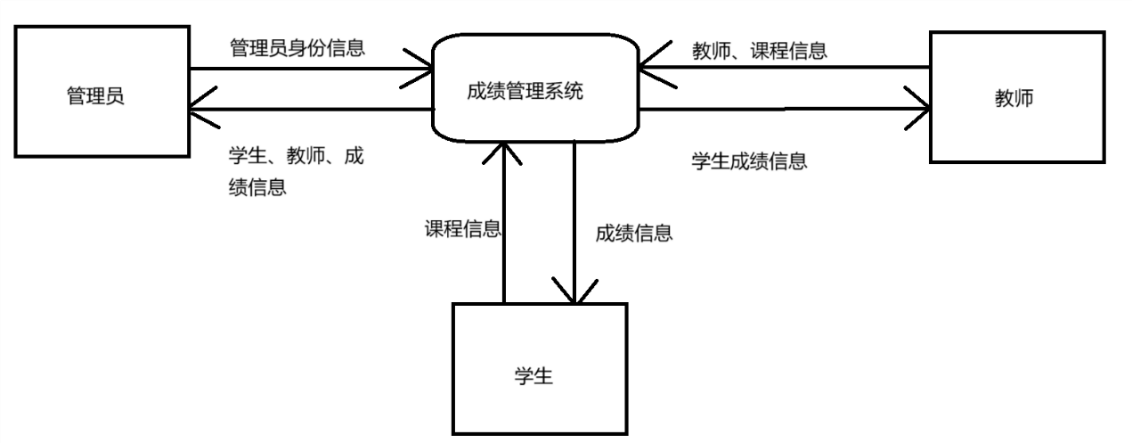


图1-1 系统数据流图

**层次1（详细数据流图）**：

* 对每个主要功能进行详细描述，包括数据流向、数据存储等。

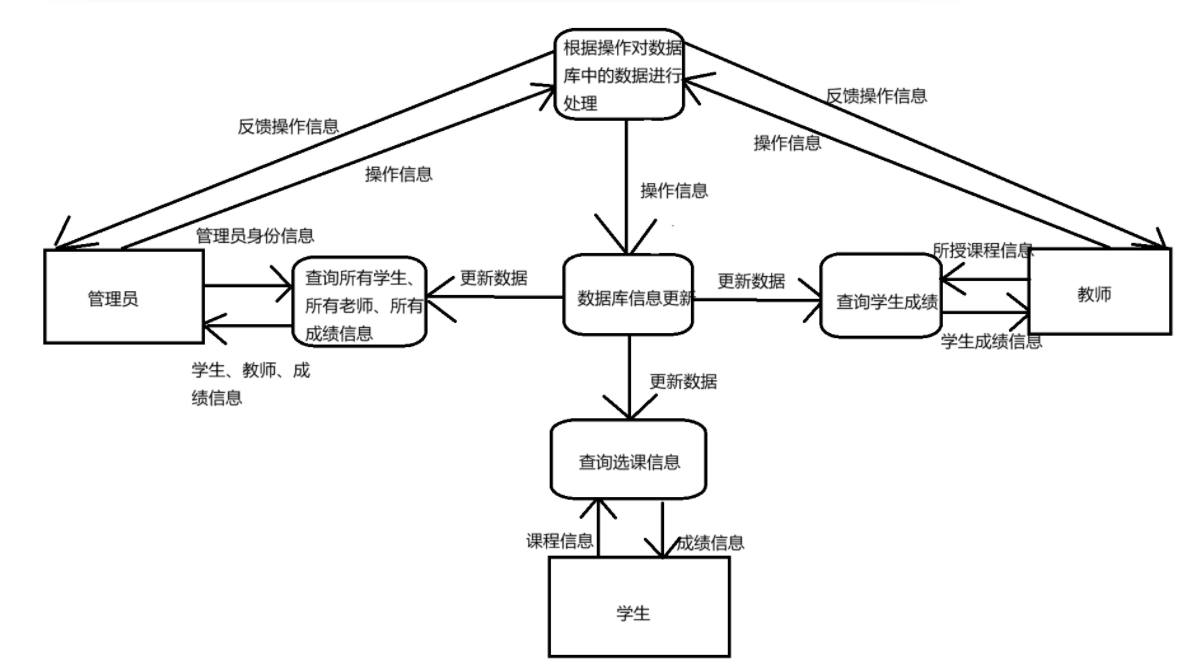


图1-1 系统数据流图

2.系统总体设计

基于B/S模式的学生成绩管理信息系统的设计方案包括系统架构设计、数据库设计、系统界面设计以及安全性设计等多个部分。以下是对这些设计组件的详细说明。

## 2.1系统设计方案

**一、系统架构设计**

学生成绩管理信息系统采用B/S（Browser/Server）架构，采用了前后端分离的开发策略，其主要组成部分包括：

1. **前端**：

**Vue框架基础**：

利用Vue组件化开发，提升代码复用性和可维护性。

使用Vue Router实现页面路由管理，支持多页面应用。

**数据绑定与交互**：

采用Vue的数据绑定机制，简化界面与数据模型间的同步。

实现双向数据绑定，提升用户交互体验。

**异步axios请求：**

封装axios请求方法，统一管理API接口。

1. **后端**：

后端采用了经典的三层架构设计，即Controller层、Service层和DAO层，这一设计在多个方面对项目产生了积极影响，特别是解耦、维护便利性和开放扩展性方面。以下是针对这三点的详细扩写：

三层架构实现了项目的高度解耦：  
通过将后端系统划分为Controller层、Service层和DAO层，每一层都承担了不同的职责。Controller层负责接收前端请求并返回响应，是用户与后端系统交互的桥梁；Service层则封装了业务逻辑，处理具体的业务操作；DAO层则专注于数据的持久化操作，与数据库进行交互。

三层架构使得项目维护更加轻松：  
由于三层架构实现了项目的高度解耦，因此当系统需要维护或升级时，开发人员可以更加专注于某一特定层的问题。例如，如果前端界面需要更新，开发人员只需关注Controller层的相关代码；如果需要修改业务逻辑，则只需调整Service层；而数据库结构或查询方式的变更则主要影响DAO层。

三层架构促进了项目的开放性和扩展性：  
三层架构的设计使得后端系统具有更好的开放性和扩展性。由于各层之间通过接口进行通信，因此可以很容易地引入新的技术或框架来替换某一层的实现。例如，如果需要提升系统的性能，可以考虑使用更高效的数据库访问技术或缓存机制来优化DAO层；

**数据库**：

* + 使用关系型数据库mysql存储用户信息和成绩数据。
  + 设计数据库表格，包括用户表、成绩表、课程表等。

## 2.2系统功能结构

系统主要有三大块功能，分别是：教师学生登录注册、成绩管理、在线交流。其中，教师和学生可以通过登录注册模块填写相应的信息进行注册，再登录该网站。进入网站后学生、教师和管理员都可以对应信息；其中教师与学生可在登录后进行在线交流。系统功能结构图，如图2-1所示。

学生成绩管理信息系统模块

学生用户

管理员

信息查询

授课老师

成绩查询

在线交流

信息管理

成绩管理

成绩管理

在线交流

图2-1 系统后台功能结构图

## 2.3数据库实体设计

数据库主要功能是存储和管理系统中的全部数据，并确保它们的安全性和共享性。为实现这样的功能，数据库中的数据需要按时间组织存储，以方便用户管理和使用。另外，数据在数据库中应该具有一定的共享性和独立性，以满足系统内各模块对数据的使用需求，并保护数据的安全性。设计数据库时，需要清晰地描述每个实体之间的联系和关系，以便数据的检索、更新和删除。在此基础上，可通过E-R图来展示数据库结构，主要包括各实体、属性和关系，以指导开发过程中的数据库建模和实现，系统的E-R图，如图2-2所示。

管理员

管理

1

n

授课教师

管理

学生用户

n

1

选课

课程表

管理

图2-2 系统E-R图

学生

教师

聊天

图2-3 聊天E-R图

## 2.4数据库表结构设计

本系统运用了MySQL数据库，根据不同功能模块对数据的相关要求，建立了如下表结构：用户表、学生表、教师表、课程表、成绩表、聊天记录表、管理员表。

（1）用户表：该表用于储存用户信息。包括：用户帐号、用户密码、用户角色。用户信息表如表2-3所示。

表2-3 user表（用户表）

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **注释** | 约束 |
| account | varchar(255) | 用户账号 | NOT NULL（主键） |
| password | varchar(255) | 用户密码 | NOT NULL |
| role | int | 用户角色（如 0 - 学生, 1 - 教师, 2 - 管理员） | default NULL |

（2）学生表：该表用于储存学生信息。包括：学生编号、学生姓名、用户账号、性别、学院、专业、电话、地址。用户信息表如表2-4所示。

表2-4student表（学生表）

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **注释** | 约束 |
| sno | varchar(255) | 学生编号 | NOT NULL（主键） |
| name | varchar(255) | 学生姓名 | NOT NULL |
| account | varchar(255) | 用户账号（外键） | NOT NULL（外键） |
| sex | enum('男', '女') | 性别 | NOT NULL |
| college | varchar(255) | 院校 | NOT NULL |
| major | varchar(255) | 专业 | NOT NULL |
| phone | varchar(255) | 电话 | NOT NULL |
| address | varchar(255) | 地址 | NOT NULL |

（3）教师表：该表用于储存教师信息。包括：教师编号、教师姓名、用户账号、性别、学院、职称。用户信息表如表2-5所示。

表2-5 teacher表（教师表）

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **注释** | 约束 |
| tno | varchar(255) | 教师编号 | NOT NULL（主键） |
| name | varchar(255) | 教师姓名 | NOT NULL |
| account | varchar(255) | 用户账号（外键） | NOT NULL（外键） |
| sex | enum('男', '女') | 性别 | NOT NULL |
| college | varchar(255) | 学院 | NOT NULL |
| Title | varchar(255) | 职称 | NOT NULL |

（4）课程表：该表用于储存课程信息。包括：课程名称、教师编号、课程编号、学分、学期、创建时间。课程表如表2-6所示。

表2-6 course表（课程表）

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **注释** | 约束 |
| tno | varchar(255) | 教师编号 | NOT NULL（主键） |
| cname | varchar(255) | 课程名称 | NOT NULL（主键） |
| cno | varchar(255) | 课程编号 | NOT NULL |
| credit | double | 学分 | NOT NULL |
| semester | int(0) | 学期 | NOT NULL |
| ctime | int(0) | 创建时间 | NOT NULL |

（5）成绩表：该表用于储存学生成绩信息。包括：学生编号、课程编号、学生成绩。成绩表如表2-7所示。

表2-7 sc表（成绩表）

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **注释** | 约束 |
| sno | varchar(255) | 学生编号（外键） | NOT NULL（主键） |
| cno | varchar(255) | 课程编号（外键） | NOT NULL（主键） |
| sscore | int | 学生成绩 | default NULL |

（6）聊天记录表：该表用于储存聊天信息。包括：发送者账号、接收者账号、消息内容、消息发送时间、消息是否已读。聊天记录表如表2-8所示。

表2-8 chat表（聊天记录表）

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **注释** | 约束 |
| sender | varchar(255) | 发送者账号 | NOT NULL（主键） |
| receiver | varchar(255) | 接收者账号 | NOT NULL |
| msg | varchar(255) | 消息内容 | NOT NULL |
| date | datetime | 消息发送时间 | NOT NULL |
| isread | varchar(255) | 消息是否已读 | NOT NULL（主键） |

（7）管理员表：该表用于储存管理员信息。包括：管理员编号、管理员姓名、用户账号。聊天记录表如表2-9所示。

表2-9 管理员表（admin表）

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **注释** | 约束 |
| admin\_id | varchar(255) | 管理员编号 | NOT NULL（主键） |
| name | varchar(255) | 管理员姓名 | NOT NULL |
| account | varchar(255) | 用户账号（外键） | NOT NULL（外键） |

3.系统主要功能实现

## 3.1主要模块处理流程

学生和教师需要有属于自己的账号和密码，而且必须是正确的，这样才能顺利登录到自己专属的系统中。进入系统之后，学生可以查看自己的成绩；教师可对学生的成绩信息进行管理。具体流程如下图3-1所示。

输入账号、密码及身份

信息是否正确

否

信息错误

数据库

是

功能界面

功能处理

系统登录界面

开始

结束

图3-1 学生和授课教师操作流程图

想要管理系统信息，管理员须先登录系统。若管理员输入的账号和密码正确，则可进入后台系统管理学生、授课教师信息；若输入的账号密码错误，跳回第一步进行重新输入。具体登录流程图如图3-2所示。

输入用户名、

密码等信息

选择管理员身份

用户名、密码是否

为空

是

提示信息

不能为空

用户名、密码是否

正确

否

否

提示信息不正确

是

功能界面

结束

开始

功能处理

数据库

图3-2 管理员操作流程图

## 3.2学生用户登录

当学生用户输入个人的账号和密码及身份尝试登录系统时，如果输入正确，系统将直接进入系统进行操作。学生用户登录界面如图4-1所示。

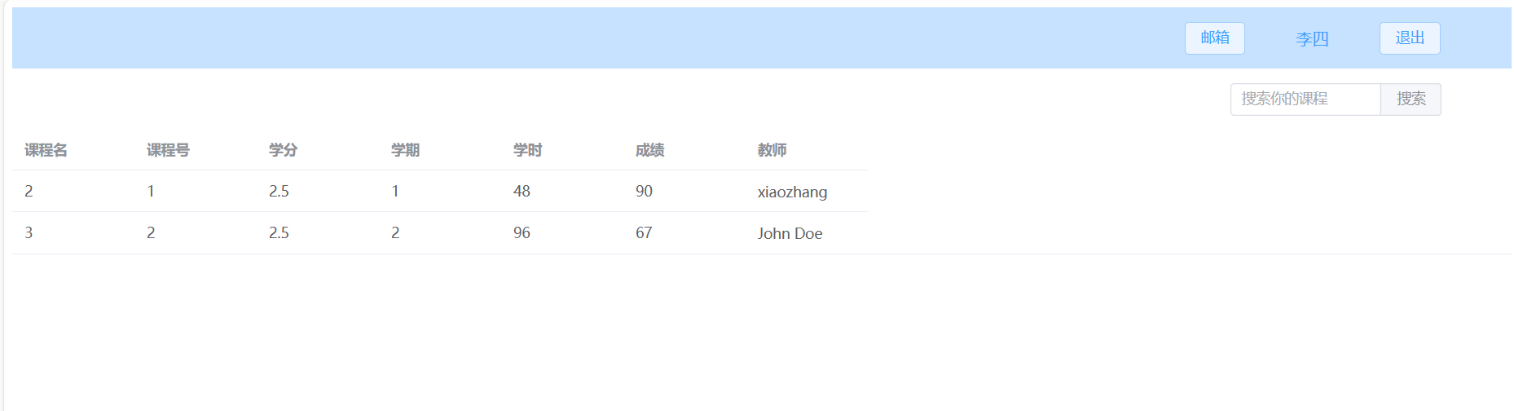


图4-1 学生用户登录界面

主要实现代码如下：

// 创建响应式数据

const user = ref({ account: '', password: '',role:'' });

// 获取路由实例

const router = useRouter();

// 定义登录方法

const input = async () => {

try {

var res

var res2 = await managerLogIn(user.value);

if (user.value.role=='1'){

res = await studentLogIn(user.value);

}

else if(user.value.role=='2'){

res = await teacherLogIn(user.value);

}

else if(user.value.role=='3'){

res = await managerLogIn(user.value);

}

else

{

tip.value.tips='请选择您的身份';

}

console.log(res);

if (Object.keys(res.data).length !== 0 ) {

if (user.value.role==res2.data.role){

if (user.value.role=='1'){

sessionStorage.setItem('studentData', JSON.stringify(res.data));

router.push('/StudentMain');

}

else if(user.value.role=='2'){

sessionStorage.setItem('teacherData', JSON.stringify(res.data));

router.push('/TeacherMain');

}

else if(user.value.role=='3'){

sessionStorage.setItem('managerData', JSON.stringify(res.data));

router.push('/ManagerMain');

}

}

else tip.value.tips='请选择正确的身份';

} else {

// 可能需要在这里处理登录失败的情况

tip.value.tips='账号不存在或密码错误';

console.error('Login failed');

}

} catch (error) {

console.error('An error occurred during login:', error);

}

};

## 3.3授课教师功能模块

### 3.3.1教师登录

当教师输入个人的账号和密码及身份尝试登录系统时，如果输入正确，系统将直接进入系统进行操作。教师登录界面如图4-2所示。



图4-2 教师登录界面

主要实现代码如下：

// 创建响应式数据

const user = ref({ account: '', password: '',role:'' });

// 获取路由实例

const router = useRouter();

// 定义登录方法

const input = async () => {

try {

var res

var res2 = await managerLogIn(user.value);

if (user.value.role=='1'){

res = await studentLogIn(user.value);

}

else if(user.value.role=='2'){

res = await teacherLogIn(user.value);

}

else if(user.value.role=='3'){

res = await managerLogIn(user.value);

}

else

{

tip.value.tips='请选择您的身份';

}

console.log(res);

if (Object.keys(res.data).length !== 0 ) {

if (user.value.role==res2.data.role){

if (user.value.role=='1'){

sessionStorage.setItem('studentData', JSON.stringify(res.data));

router.push('/StudentMain');

}

else if(user.value.role=='2'){

sessionStorage.setItem('teacherData', JSON.stringify(res.data));

router.push('/TeacherMain');

}

else if(user.value.role=='3'){

sessionStorage.setItem('managerData', JSON.stringify(res.data));

router.push('/ManagerMain');

}

}

else tip.value.tips='请选择正确的身份';

} else {

// 可能需要在这里处理登录失败的情况

tip.value.tips='账号不存在或密码错误';

console.error('Login failed');

}

} catch (error) {

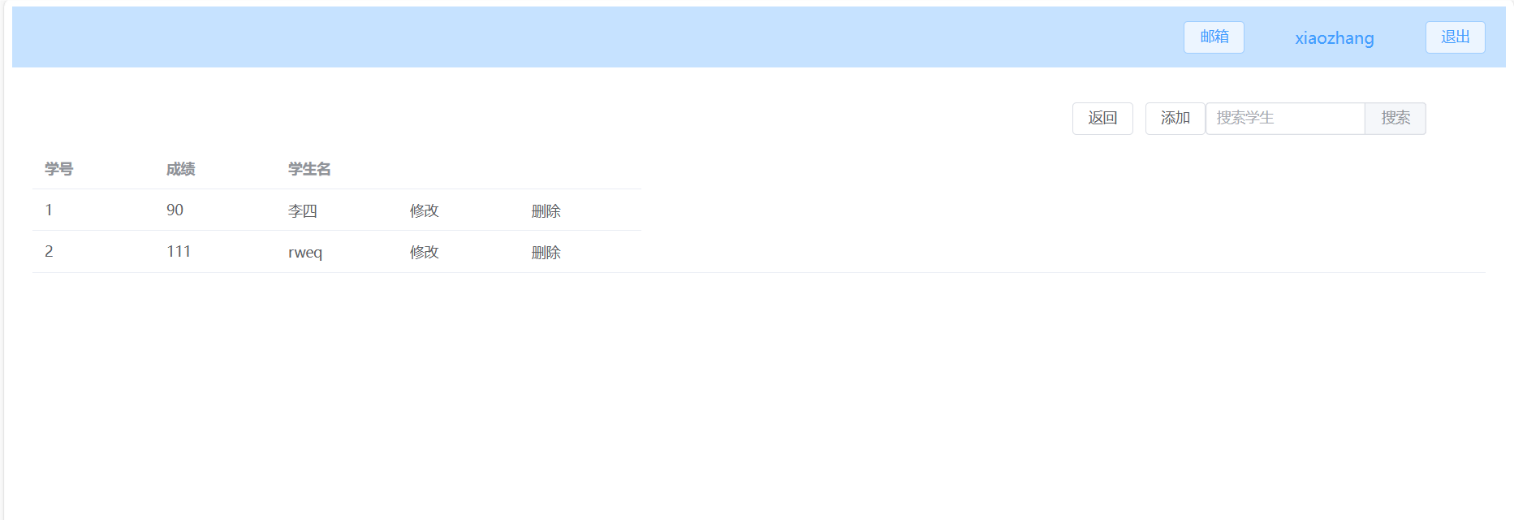
console.error('An error occurred during login:', error);

}

};

### 3.3.2学生成绩管理

教师进入系统后，点击“管理”可对学生成绩进行管理操作。操作界面如图4-3所示。

图4-3 教师管理成绩界面

主要实现代码如下：// 修改 删除点击事件处理函数

const handleAlter = (row) => {

dialogVisibleAlter.value=true;

storeSno.value=row.sno;

console.log('AstoreSno', storeSno);

//router.push({name: 'routeName',params:{ cno:row.cno}});

};

const handleDel = (row) => {

dialogVisibleDel.value=true;

storeSno.value=row.sno;

console.log('DstoreSno', storeSno);

//router.push({name: 'routeName',params:{ cno:row.cno}});

};

const handleAdd = () => {

dialogVisibleAdd.value=true;

//router.push({name: 'routeName',params:{ cno:row.cno}});

};

const input0 = ref();

const input1 = ref();

const input2 = ref();

const input3 = ref();

//搜索功能

const SearchStudent =async()=>{

console.log("input",input0.value)

const cnoSname={name:input0.value,cno:storeCno.value}

const res = await searchStudentScoreByCnoSname(cnoSname);

items.value=res.data;

console.log("name",res);

}

//修改内容处理

const modify = async()=>{

const score ={sno:storeSno.value,cno:storeCno.value,sscore:input1.value};

const res = await TeacherAltScore(score);

console.log(res);

window.location.reload();

}

//删除内容处理

const del = async()=>{

const score ={sno:storeSno.value,cno:storeCno.value,sscore:input1.value};

const res = await TeacherDelScore(score);

console.log(res);

window.location.reload();

}

//添加内容处理

const add = async()=>{

const score ={sno:input2.value,cno:storeCno.value,sscore:input3.value};

const res = await TeacherAddScore(score);

console.log(res);

window.location.reload();

}

## 3.4管理员功能模块

### 3.4.1管理员登录

当管理员输入个人的账号和密码及身份尝试登录系统时，如果输入正确，系统将直接进入系统进行操作。管理员登录界面如图4-4所示。



图4-4 管理员登录界面

主要实现代码如下：

// 创建响应式数据

const user = ref({ account: '', password: '',role:'' });

// 获取路由实例

const router = useRouter();

// 定义登录方法

const input = async () => {

try {

var res

var res2 = await managerLogIn(user.value);

if (user.value.role=='1'){

res = await studentLogIn(user.value);

}

else if(user.value.role=='2'){

res = await teacherLogIn(user.value);

}

else if(user.value.role=='3'){

res = await managerLogIn(user.value);

}

else

{

tip.value.tips='请选择您的身份';

}

console.log(res);

if (Object.keys(res.data).length !== 0 ) {

if (user.value.role==res2.data.role){

if (user.value.role=='1'){

sessionStorage.setItem('studentData', JSON.stringify(res.data));

router.push('/StudentMain');

}

else if(user.value.role=='2'){

sessionStorage.setItem('teacherData', JSON.stringify(res.data));

router.push('/TeacherMain');

}

else if(user.value.role=='3'){

sessionStorage.setItem('managerData', JSON.stringify(res.data));

router.push('/ManagerMain');

}

}

else tip.value.tips='请选择正确的身份';

} else {

// 可能需要在这里处理登录失败的情况

tip.value.tips='账号不存在或密码错误';

console.error('Login failed');

}

} catch (error) {

console.error('An error occurred during login:', error);

}

};

### 3.4.2学生和教师信息管理

管理员进入系统后，点击“信息管理”和“成绩管理”可对学生和教师信息及学生成绩进行管理操作。操作界面如图4-5所示。

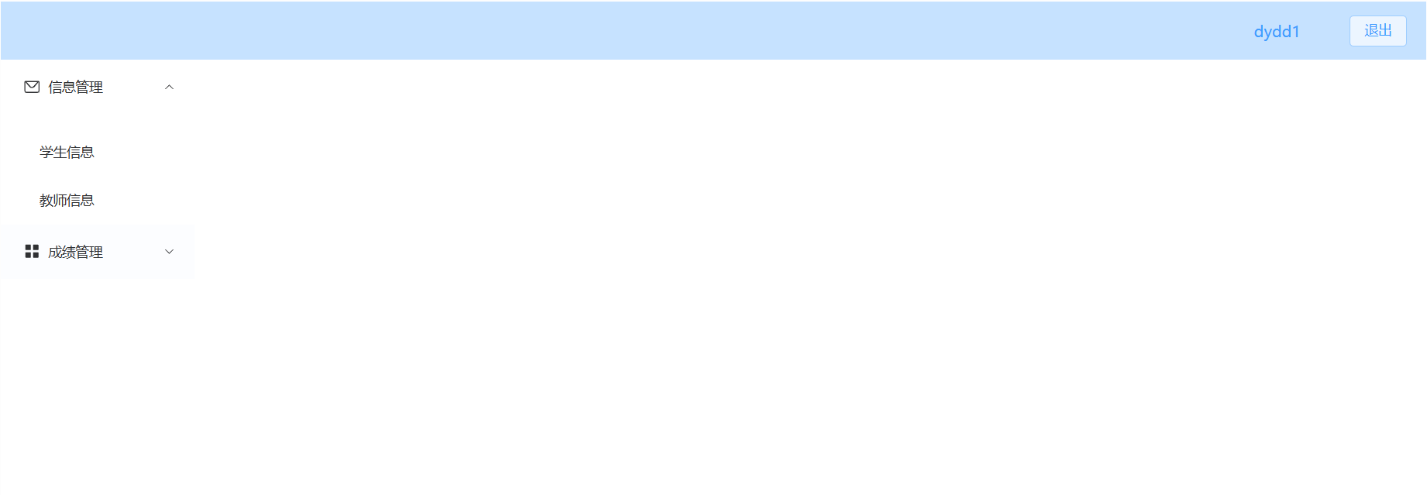


图4-5 管理员管理界面

主要实现代码如下：

const modify = async()=>{

student.value.name=input1.value;

student.value.sex=input2.value;

student.value.college=input3.value;

student.value.major=input4.value;

student.value.phone=input5.value;

student.value.address=input6.value;

console.log('AfSno', student.value);

const res = await UpdateStudent(student.value);

console.log(res);

window.location.reload();

}

结 论

本课程设计基于B/S（Browser/Server）模式，开发了一款学生成绩管理信息系统，旨在为教育机构提供高效、便捷的成绩管理和查询平台。该系统实现了学生、教师及管理员的多层次用户管理，充分利用了B/S架构的优势，使得系统可以通过浏览器进行访问，大大降低了终端对客户端软件的依赖。

系统的核心功能包括学生成绩录入、查询、统计与管理。教师可以快速录入学生的成绩信息，系统自动进行数据校验，确保数据的准确性。同时，学生也可以通过个人账户实时查询自己的成绩和学业信息，提升了信息透明度，增强了学生的自主学习意识。

由于采用了B/S架构，该系统具备良好的可扩展性和维护性。通过后端数据库与前端用户界面的分离，开发者可以在不影响用户体验的前提下，进行后端的功能更新和优化。同时，管理员可以轻松管理用户权限，确保系统的安全性。

系统在性能上也进行了优化，通过合理的数据库设计和高效的查询语句，明显提升了数据处理的速度。此外，为了提升用户体验，我们还加入了友好的界面设计和便捷的操作流程，让用户在操作时感到舒适与高效。

然而，在实际应用中仍存在一些不足之处，比如在输入框校验方面，虽然系统已经设置了一定的数据校验机制以确保数据的准确性，但在实际应用中仍发现了一些问题。例如，某些特殊字符或格式不正确的数据有时能够绕过校验机制被录入系统，这可能导致后续数据处理出现错误或异常。此外，对于某些必填项，系统未能实现强制校验，导致部分关键信息缺失，影响了数据的完整性和可用性。因此，需要进一步完善输入框校验机制，提高数据录入的准确性和完整性。

在系统效率方面，虽然通过合理的数据库设计和高效的查询语句已经显著提升了数据处理速度，但在面对大规模数据或复杂查询时，系统响应速度仍有待提升。特别是在高并发访问场景下，系统可能会出现延迟或卡顿现象，影响了用户体验。为了解决这个问题，可以考虑采用更先进的数据库技术或优化查询算法，以提高系统处理大规模数据的能力。同时，也可以对系统进行负载均衡和分布式部署，以分散访问压力，提升系统整体性能。

此外，在系统安全性方面也存在一定的隐患。虽然管理员可以轻松管理用户权限，确保系统的基本安全性，但随着网络攻击手段的不断升级和多样化，系统仍可能面临各种安全威胁。因此，需要进一步加强系统的安全防护措施，如采用更安全的身份验证机制、加密传输数据、定期备份和恢复数据等，以确保系统数据的安全性和完整性。