**软件设计说明书**

**2025年 1月 7日**

目录

[**一、** **总体设计** 3](#_Toc187499713)

[1.1系统设计方案 3](#_Toc187499714)

[1.2系统功能结构 3](#_Toc187499715)

[**二、** **数据库设计** 3](#_Toc187499716)

[2.1数据库实体设计 3](#_Toc187499717)

[2.2数据库表结构设计 3](#_Toc187499718)

[**三、** **具体模块详细设计** 3](#_Toc187499719)

[3.1主要模块处理流程 3](#_Toc187499720)

[3.2用户登录 3](#_Toc187499721)

[3.3授课教师功能模块 3](#_Toc187499722)

[3.3.1教师登录 3](#_Toc187499723)

[3.3.2学生成绩管理 3](#_Toc187499724)

[3.4管理员功能模块 3](#_Toc187499725)

[3.4.1管理员登录 3](#_Toc187499726)

[3.4.2学生和教师信息管理 3](#_Toc187499727)

1. **总体设计**

## 1.1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。

## 1.2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

1. **数据库设计**

## 2.1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

## 2.2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） |

1. **具体模块详细设计**

## 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用户登录

这里只演示学生的，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。

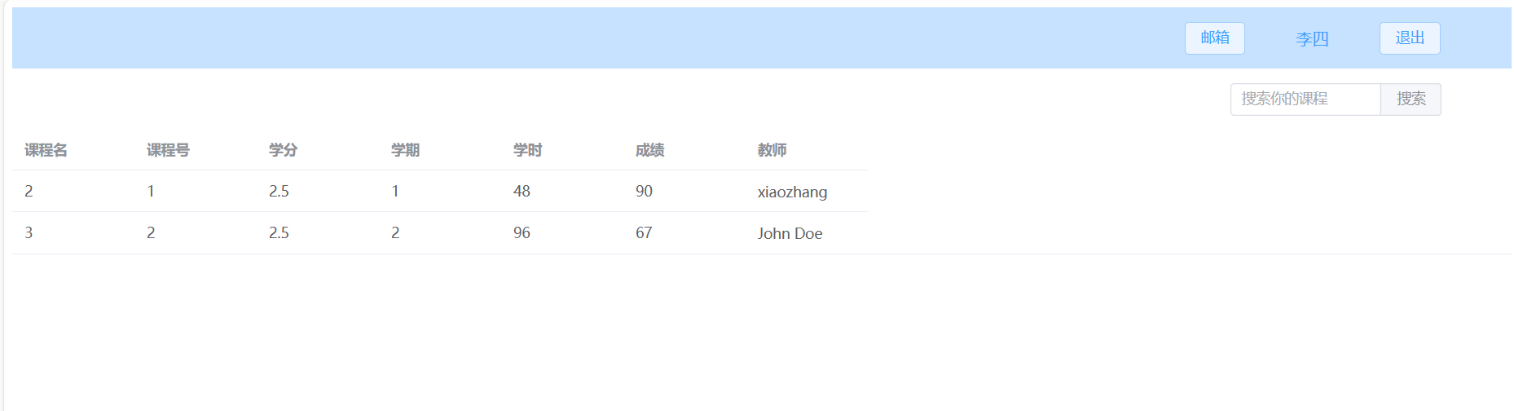


图4-1 学生用户登录界面

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) {

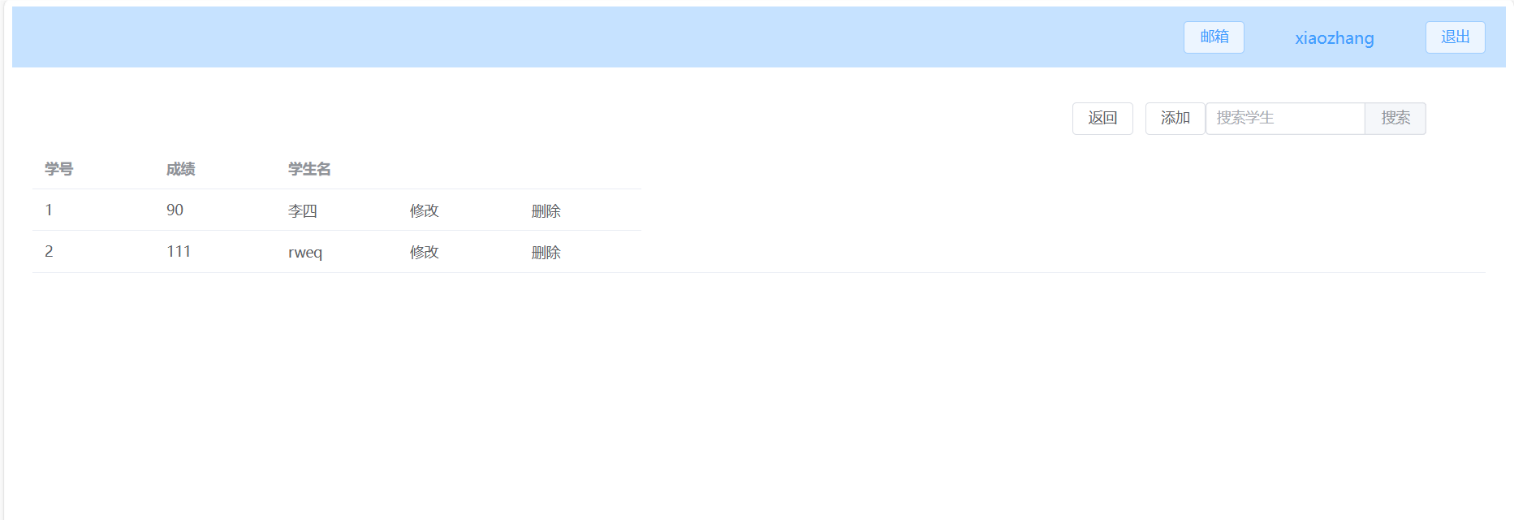
console.error('An error occurred during login:', error);

}

};

### 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。

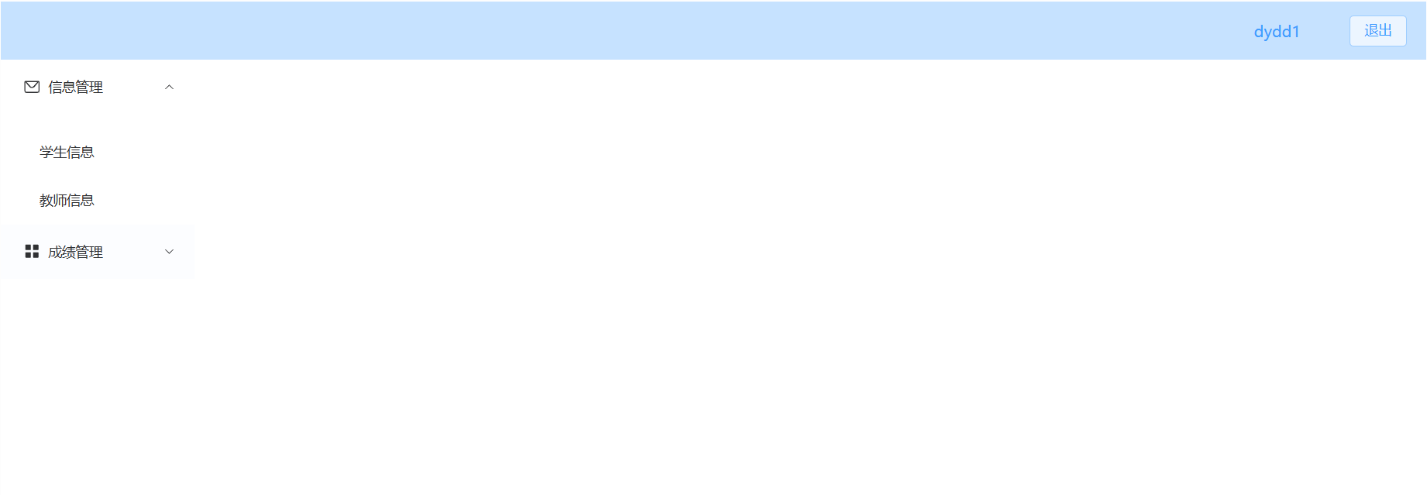


图4-5 管理员管理界面

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();

}