数据库原理课程设计

**目 录**

[第一章 系统需求分析 1](#_Toc519797058)

[1.1 功能分析 1](#_Toc519797059)

[1.2 数据流图 2](#_Toc519797060)

[1.3 数据字典 5](#_Toc519797061)

[第二章 数据库逻辑结构设计 6](#_Toc519797062)

[2.1 E-R图设计 6](#_Toc519797063)

[2.2 关系模式 6](#_Toc519797064)

[第三章 软件结构设计 8](#_Toc519797065)

[3.1 系统软件结构图 8](#_Toc519797066)

[3.2 开发与运行环境 9](#_Toc519797067)

[3.3 详细设计 9](#_Toc519797068)

[第四章 程序模块调试及运行 18](#_Toc519797069)

[4.1 学生部分 18](#_Toc519797070)

[4.2 教师部分 20](#_Toc519797071)

[4.3 管理员部分 25](#_Toc519797072)

[第五章 软件使用说明书 32](#_Toc519797073)

[5.1 前端使用说明 32](#_Toc519797074)

[5.2 后端使用说明 32](#_Toc519797075)

[第六章 课程设计小结 32](#_Toc519797076)

[第七章 参考文献 33](#_Toc519797077)

[第八章 其他说明 33](#_Toc519797078)

## ****第一章 系统需求分析****

### 1.1 功能分析

1、总体功能要求

要求设计完成一个学生成绩管理系统，提供如下功能：学生查询成绩；教师查询、录入、修改成绩；管理员查询、录入、修改人员、课程基本信息。

2、身份功能划分

按照学生成绩管理系统的使用对象，可以将其划分为三类：学生、教师和管理员。他们分别对应不同的功能：

学生：登录系统、查看自己的成绩和个人基本信息、允许修改密码；

教师：登录系统、查询教师本人所教授课程的学生成绩、录入和修改成绩、查询个人基本信息、允许修改密码；

管理员：登录系统、查询人员和课程的基本信息、录入修改人员和课程的基本信息。

3、对于系统的功能模块划分如表1所示

表1 系统功能模块划分表

|  |  |  |
| --- | --- | --- |
| 模块 | 身份 | 功能 |
| 登录模块 | 学生 | 输入账号密码，验证成功即可登录系统 |
| 教师 | 输入账号密码，验证成功即可登录系统 |
| 管理员 | 输入账号密码，验证成功即可登录系统 |
| 查询模块 | 学生 | 查询学生本人成绩和个人基本信息 |
| 教师 | 查询所教课程的学生成绩及基本信息、查询个人基本信息 |
| 管理员 | 查询学生教师等人员的基本信息、查询课程的基本信息 |
| 录入模块 | 学生 | 无 |
| 教师 | 录入所教课程的学生成绩 |
| 管理员 | 录入学生教师等人员信息、录入课程信息 |
| 修改模块 | 学生 | 无 |
| 教师 | 修改所教课程的学生的成绩信息 |
| 管理员 | 修改学生教师等人员的基本信息、修改课程的基本信息 |
| 统计模块 | 教师 | 统计各分数段学生的成绩分布，画出直方图和饼图 |

### 1.2 数据流图

1、顶层数据流图

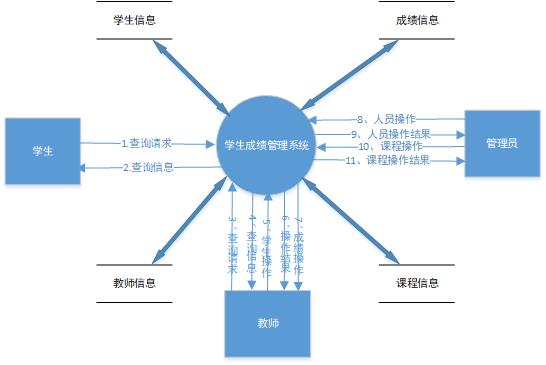


图1-1 学生成绩管理系统顶层数据流图

2、学生部分第二层数据流图

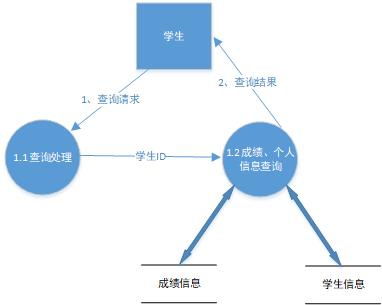


图1-2 学生部分第二层数据流图

3、教师部分第二层数据流图

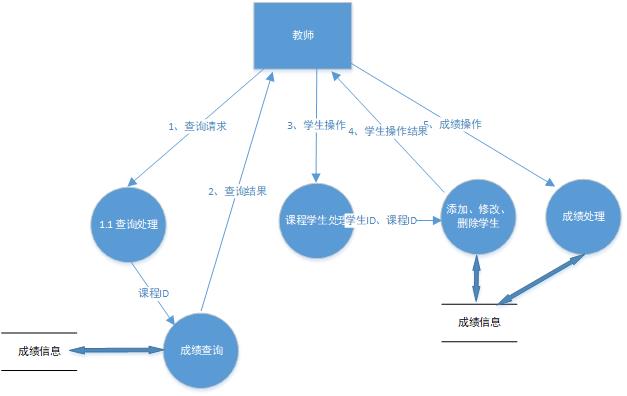


图1-3 教师部分第二层数据流图

4、管理员部分第二层数据流图

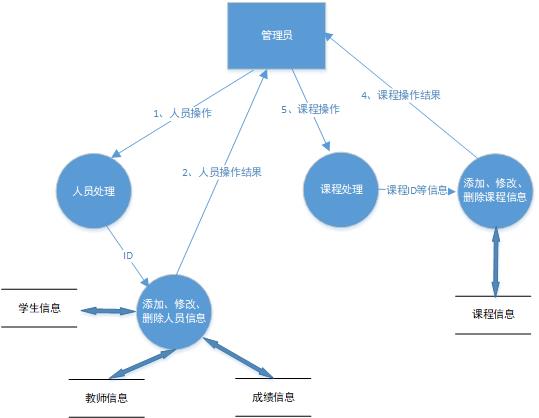


图1-4 管理员部分第二层数据流图

### 1.3 数据字典

1、student表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Table\_name | Field | Type | Null | Key | Extra |
| student | sID | varchar(20) | NO | PRI | 学生ID |
| student | sName | varchar(20) | NO |  | 学生姓名 |
| student | classno | varchar(20) | YES |  | 学生班级 |
| student | sex | varchar(10) | YES |  | 学生性别 |
| student | password | varchar(20) | NO |  | 学生登录密码 |

2、teacher表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Table\_name | Field | Type | Null | Key | Extra |
| teacher | tID | varchar(20) | NO | PRI | 教师ID |
| teacher | tName | varchar(20) | NO |  | 教师姓名 |
| teacher | email | varchar(30) | NO |  | 教师邮箱 |
| teacher | sex | varchar(10) | YES |  | 教师性别 |
| teacher | password | varchar(20) | YES |  | 教师登录密码 |

其中，默认只有一个管理员admin，信息存储在teacher表中。

3、course表

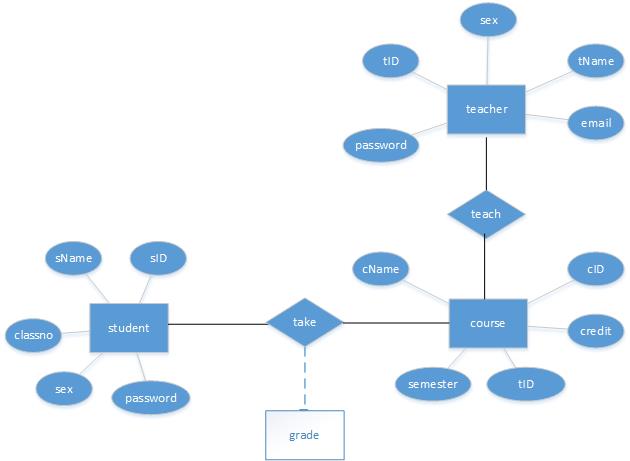
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Table\_name | Field | Type | Null | Key | Extra |
| course | cID | varchar(20) | NO | PRI | 课程ID |
| course | tID | varchar(20) | NO | MUL | 教师ID |
| course | tName | varchar(20) | YES |  | 课程名称 |
| course | credit | int(10) | YES |  | 课程学分 |
| course | semester | varchar(10) | YES |  | 课程学期 |

4、take表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Table\_name | Field | Type | Null | Key | Extra |
| take | sID | varchar(20) | NO | PRI,MUL | 学生ID |
| take | tID | varchar(20) | NO | PRI,MUL | 课程ID |
| take | grade | double(10,0) | YES |  | 成绩 |

## 第二章 数据库逻辑结构设计

### 2.1 E-R图设计



### 2.2 关系模式

1、student表

关系模式用R（U,D,DOM,F），R指关系名，U指一组属性，D指域，DOM指属性到域的映射，F就是指数据依赖。

student表的关系模式为：

student（U,D,DOM,F）

其中，U = （sID,sName,classno,sex,password）

D:D1(由小于20位的字符串组成)

D2：（由2-4个任意汉字组成）

D3：（男，女）

DOM：

DOM（sID）= D1

DOM（sName）= D2

DOM（classno）= D2

DOM（sex）= D3

DOM（password）= D2

F：

sID –> sName

sID -> classno

sID -> sex

sID -> password

2、teacher表

teacher表的关系模式为：

teacher（U,D,DOM,F）

其中，U = （tID,tName,email,sex,password）

D:D1(由小于20位的字符串组成)

D2：（由2-4个任意汉字组成）

D3：（男，女）

D4:（正确的邮箱格式）

DOM：

DOM（tID）= D1

DOM（tName）= D2

DOM（email）= D4

DOM（sex）= D3

DOM（password）= D2

F：

tID –> tName

tID -> sex

tID -> password

3、course表

course表的关系模式为：

course（U,D,DOM,F）

其中，U = （cID,cName,tID,credit,semester）

D:D1(由小于20位的字符串组成)

D2：（由任意个汉字或英文字母组成）

D3：（春，秋）

D4:（大于0的数字）

DOM：

DOM（cID）= D1

DOM（cName）= D2

DOM（tID）= D1

DOM（credit）= D4

DOM（semester）= D3

F：

cID –> cName

cID -> tID

cID -> credit

4、take表

take表的关系模式为：

take（U,D,DOM,F）

其中，U = （sID,cID,grade）

D:D1(由小于20位的字符串组成)

D2:（0-100的数字）

DOM：

DOM（sID）= D1

DOM（cID）= D1

DOM（grade）= D2

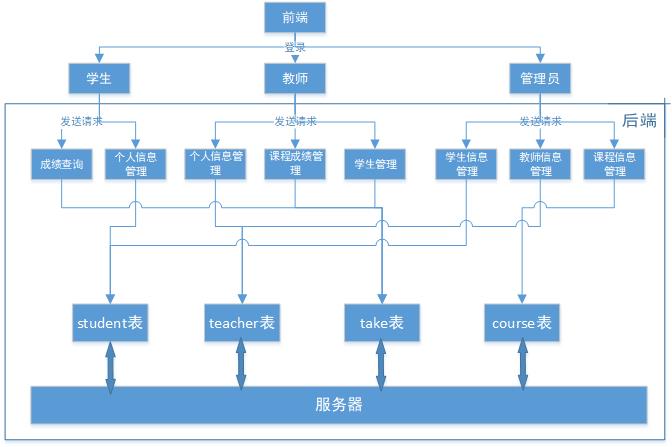
F：

（sID,cID）-> grade

## 第三章 软件结构设计

### 3.1 系统软件结构图

本部分从系统的设计与使用身份的角度对于学生成绩管理系统的功能进行了描述。



### 3.2 开发与运行环境

#### 3.2.1 开发环境

操作系统：Windows 10

数据库管理系统：MYSQL v5.7

开发工具：Visual Studio Code v1.25

#### 3.2.2 运行环境

### 3.3 详细设计

#### 3.3.1 模块划分及功能介绍

根据学生成绩管理系统的功能描述，可以将系统划分为5个模块：登录模块、查询模块、录入模块、修改模块、统计模块。模块功能描述如下所示：

|  |  |
| --- | --- |
| 模块名称 | 功能介绍 |
| 登录模块 | 用户输入账号及密码，系统识别用户身份并跳转至相应页面。 |
| 查询模块 | 以学生身份登录，可以查询自己的基本信息和成绩；以教师身份登录，可以查询成绩和个人基本信息；以管理员身份登录，可以查询基本信息情况。 |
| 录入模块 | 以教师的身份登录，可以录入成绩，录入所上课程的学生信息；以管理员的身份登录，可以录入人员和课程信息。 |
| 修改模块 | 以学生的身份登录，可以修改账户密码；以教师的身份登录，允许修改成绩，修改和删除所上课程所有学生的信息，修改账户密码；以管理员身份登录可以查询、修改、删除人员和课程的基本信息。 |
| 统计模块 | 以教师的身份登录，可以统计出某一门课程各分数段学生的成绩分布， 画出直方图和饼图。 |

#### 3.3.2 模块关系图

#### 3.3.2 详细设计

1、登录模块

2、查询模块

（1）学生查询个人信息：

前端发送请求：

getInfo() {

this.$http

.get("/stu?sID=" + this.$store.state.user.username)

.then(res => {

this.info = res.data;

});

}

后端处理：

app.get('/stu', function(req, res){

var sql = 'SELECT \* FROM student WHERE sID = \'' + req.query.sID + '\'';

connection.query(sql, function (err, result) {

if(err){

//错误处理

}

else{

temp = new Object();

temp.sID = result[0].sID;

temp.sex = result[0].sex;

temp.sName = result[0].sName;

temp.classno = result[0].classno;

res.send(temp);

}

// console.log(data); （调试打印信息）

});

});

（2）教师查询个人信息处理类似同上

（3）学生查询个人成绩

前端发送请求：

getScore() {

console.log(this.$store.state.user.username);

this.$http

.post("/stu/take", this.info)

.then(res => {

console.log(res);

this.tableData = res.data;

})

.catch(err => {

console.log(err);

});

}

后端处理：

app.post('/stu/take', function(req, res){

console.log(req.body);

var sql = 'SELECT \* FROM take NATURAL JOIN course WHERE sID = \'' + req.body.sID + '\'';

connection.query(sql, function (err, result) {

if(err){

//错误处理

}

else{

data = [];

for(var i = 0; i < result.length; i++)

{

temp = new Object();

temp.cID = result[i].cID;

temp.cName = result[i].cName;

temp.grade = result[i].grade;

data.push(temp);

}

res.send(data);

console.log(data);

}

});

});

（4）教师查询某一科目成绩

前端发送请求：

getScore: function () {

this.$http

.get(

"/tea/getGradeByCID?tID=" +

this.$store.state.user.username +

"&cID=" + this.curCourse.cID

)

.then(res => {

console.log(res);

this.score = res.data;

});

this.$http.get("/tea/pic?tID=" + this.$store.state.user.username +

"&cID=" + this.curCourse.cID)

.then(res => {

this.pieChartData.datasets[0].data = res.data;

this.barChartData.datasets[0].data = res.data;

this.$refs.piechart.update();

this.$refs.barchart.update();

}

);

}

后端处理：

var gInfo = [];

var sql = 'select \* from take natural join student natural join course ' +

'where tID = "' + req.query.tID +

'" and cID = "' + req.query.cID + '"';

console.log(sql);

connection.query(sql, function (err, result) {

if (err) {

console.log('[SELECT ERROR] - ', err.message);

} else if (result.length === 0) {

console.log('无');

} else {

for (var i = 0; i < result.length; i++) {

var temp = new Object();

temp.sID = result[i].sID;

temp.sName = result[i].sName;

temp.classno = result[i].classno;

temp.sex = result[i].sex;

temp.cID = result[i].cID;

temp.cName = result[i].cName;

temp.grade = result[i].grade;

gInfo.push(temp);

}

}

res.send(gInfo);

（5）管理员查询基本信息

前端发送请求：

getTeachers() {

this.$http

.get("/admin/teacher")

.then(res => {…} //构建请求报文

后端处理：

app.get('/admin/teacher', function (req, res) {

var sql = 'SELECT \* FROM teacher WHERE tID <> \'admin\'';

connection.query(sql, function (err, result) {

if (err) {

// 错误处理

}

else {

data = [];

for (var i = 0; i < result.length; i++) {

temp = new Object();

temp.tID = result[i].tID;

temp.tName = result[i].tName;

temp.email = result[i].email;

temp.password = result[i].password;

temp.sex = result[i].sex;

data.push(temp);

}

res.send(data);

}

});

3、录入模块

（1）教师录入成绩

前端发送请求：

addScore: function (index, row) {

this.teaFormTitle = "添加成绩";

this.teaFormModel = {

sID: "",

sName: "",

classno: "",

score: ""

};

}

commitAdd: function () {

this.$refs.teaForm.validate(valid => {

if (valid) {

console.log("add");

this.scoreModel.tID = this.$store.state.user.username;

this.scoreModel.sID = this.teaFormModel.sID;

this.scoreModel.cID = this.curCourse.cID;

this.scoreModel.newGrade = this.teaFormModel.score;

this.$http.put("/tea/newGrade", this.scoreModel).then(res => {

//请求响应处理

}

});

}

}

后端处理：

var sql = 'select \* from course where tID = "' + req.body.tID +

'" and cID = "' + req.body.cID + '"';

connection.query(sql, function (err1, result) {

if (err1) {

console.log('[SELECT ERROR] - ', err1.message);

res.send('查询错误');

} else if (result.length === 0) {

console.log('这个老师不教这门课');

res.send('您不教这门课');

} else {

sql = 'insert into take values("' + req.body.sID +

'", "' + req.body.cID +

'", ' + req.body.newGrade + ')';

console.log(sql);

connection.query(sql, function (err2) {

if (err2) {

console.log('[INSERT ERROR] - ', err2.message);

res.send('插入错误');

} else {

res.send('成功');

}

});

（2）管理员录入信息的处理过程与上类似

4、修改模块

（1）教师修改成绩

前端发送请求：

editScore: function (index, row) {

this.teaFormTitle = "编辑成绩";

this.teaFormModel = this.score[index];

}

commitEdit: function () {

this.$refs.teaForm.validate(valid => {

if (valid) {

this.scoreModel.tID = this.$store.state.user.username;

this.scoreModel.sID = this.teaFormModel.sID;

this.scoreModel.cID = this.curCourse.cID;

this.scoreModel.newGrade = this.teaFormModel.score;

this.$http.post("/tea/editGrade", this.scoreModel).then(res => {

//请求响应处理

}

}

后端处理：

var sql = 'select \* from course natural join take where tID = "' + req.body.tID +

'" and sID = "' + req.body.sID +

'" and cID = "' + req.body.cID + '"';

console.log(sql);

connection.query(sql, function (err1, result) {

if (err1) {

console.log('[SELECT ERROR] - ', err1.message);

res.send('查询错误');

} else if (result.length === 0) {

console.log('没有选了这门课的这个学生或这个老师不教这门课');

res.send('没有选了这门课的这个学生或您不教这门课');

} else {

sql = 'update take set grade = ' + req.body.newGrade +

' where sID = "' + req.body.sID +

'" and cID = "' + req.body.cID + '"';

console.log(sql);

connection.query(sql, function (err2) {

if (err2) {

console.log('[UPDATE ERROR] - ', err2.message);

res.send('修改错误');

} else {

res.send('成功');

}

（2）教师添加、删除学生进行的处理过程类似同上，只是相应的SQL语句发生改变。

（3）管理员修改基本信息

前端发送请求：

handleEditFromMenu(index, row) {

this.crsFormModel.cID = row.cID;

this.crsFormModel.tID = row.tID;

this.crsFormModel.cName = row.cName;

this.crsFormModel.credit = row.credit;

this.crsFormModel.semester = row.semester;

this.cIDInputDisabled = true;

this.showEdtCrsForm();

},

handleEditFromForm() {

this.$refs.crsForm.validate(valid => {

if (valid) {

this.edtCourse(this.crsFormModel);

} else {

return false;

}

});

},

edtCourse(course) {

this.hideCrsForm();

this.$http

.put("/admin/course/" + course.cID, course)

.then(res => {

//请求响应处理 }

}

后端处理：

app.put('/admin/course/:id', function (req, res) {

// update course info

console.log(req.body)

var sql = 'UPDATE course SET tID = \'' + req.body.tID + '\', cName = \'' + req.body.cName + '\', credit = \'' +

req.body.credit + '\', semester = \'' + req.body.semester + '\' WHERE cID = \'' + req.params.id + '\'';

// console.log(sql);

connection.query(sql, function (err, result) {

if (err) {

console.log('[SELECT ERROR] - ', err.message);

res.json({ restype: 'failed' });

}

else {

res.json({ restype: 'success' });

}

});

});

（4）管理员删除等操作处理过程同上类似，只是相应的SQL语句发生变化即可。

5、统计模块

前端发送请求：

教师在请求某一门全部成绩时，发送请求同上2（4）

后端处理：

将各分数段的人数进行统计

var info = [0, 0, 0, 0, 0];

var sql = 'select grade from take natural join student natural join course ' +

'where tID = "' + req.query.tID +

'" and cID = "' + req.query.cID + '"';

console.log(sql);

connection.query(sql, function (err, result) {

if (err) {

console.log('[SELECT ERROR] - ', err.message);

} else if (result.length === 0) {

console.log('无');

} else {

for (var i = 0; i < result.length; i++) {

if (result[i].grade >= 90) {

info[4]++;

} else if (result[i].grade >= 80) {

info[3]++;

} else if (result[i].grade >= 70) {

info[2]++;

} else if (result[i].grade >= 60) {

info[1]++;

} else {

info[0]++;

}

}

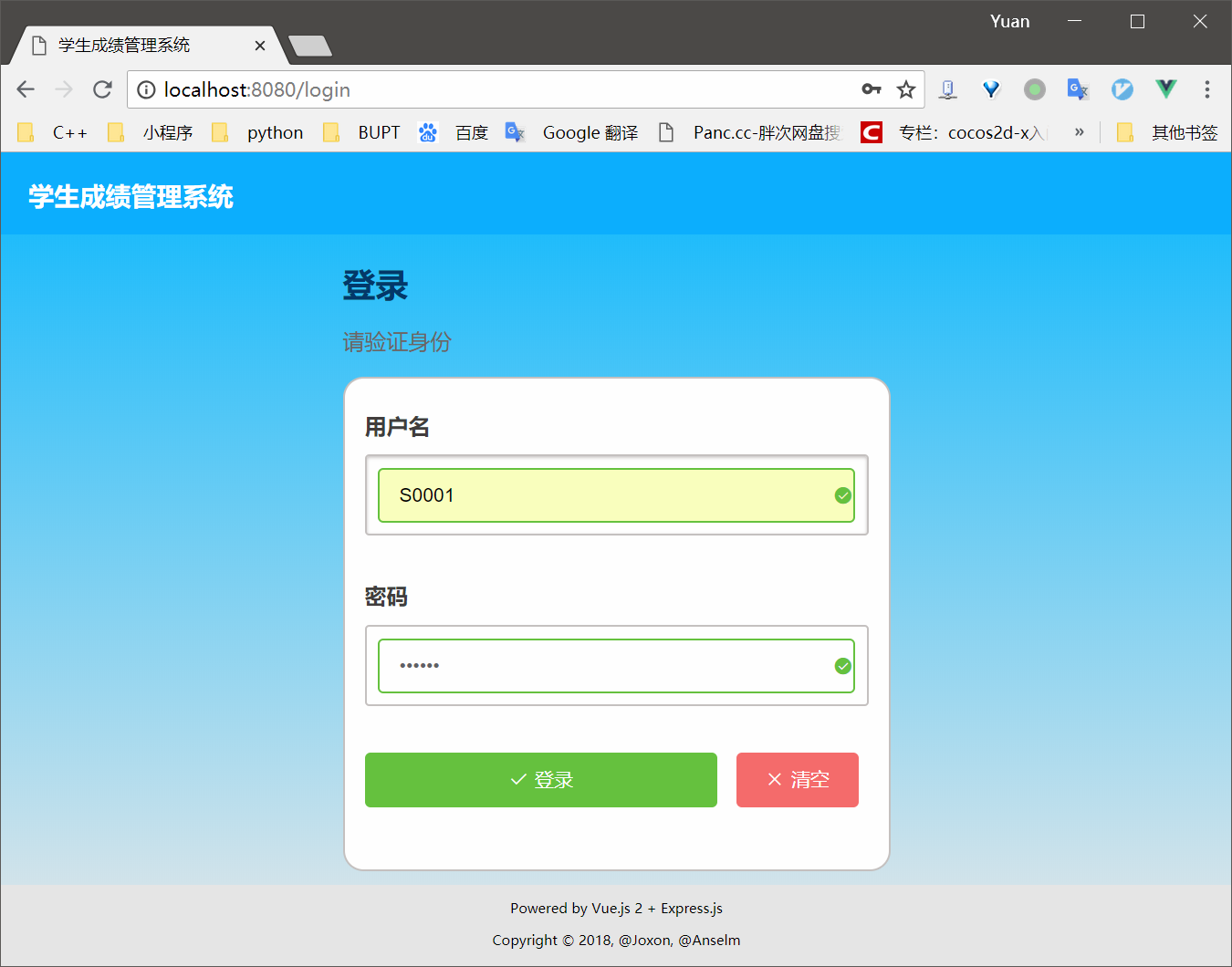
}

res.send(info);

## 第四章 程序模块调试及运行

### 4.1 学生部分

（1）登录界面



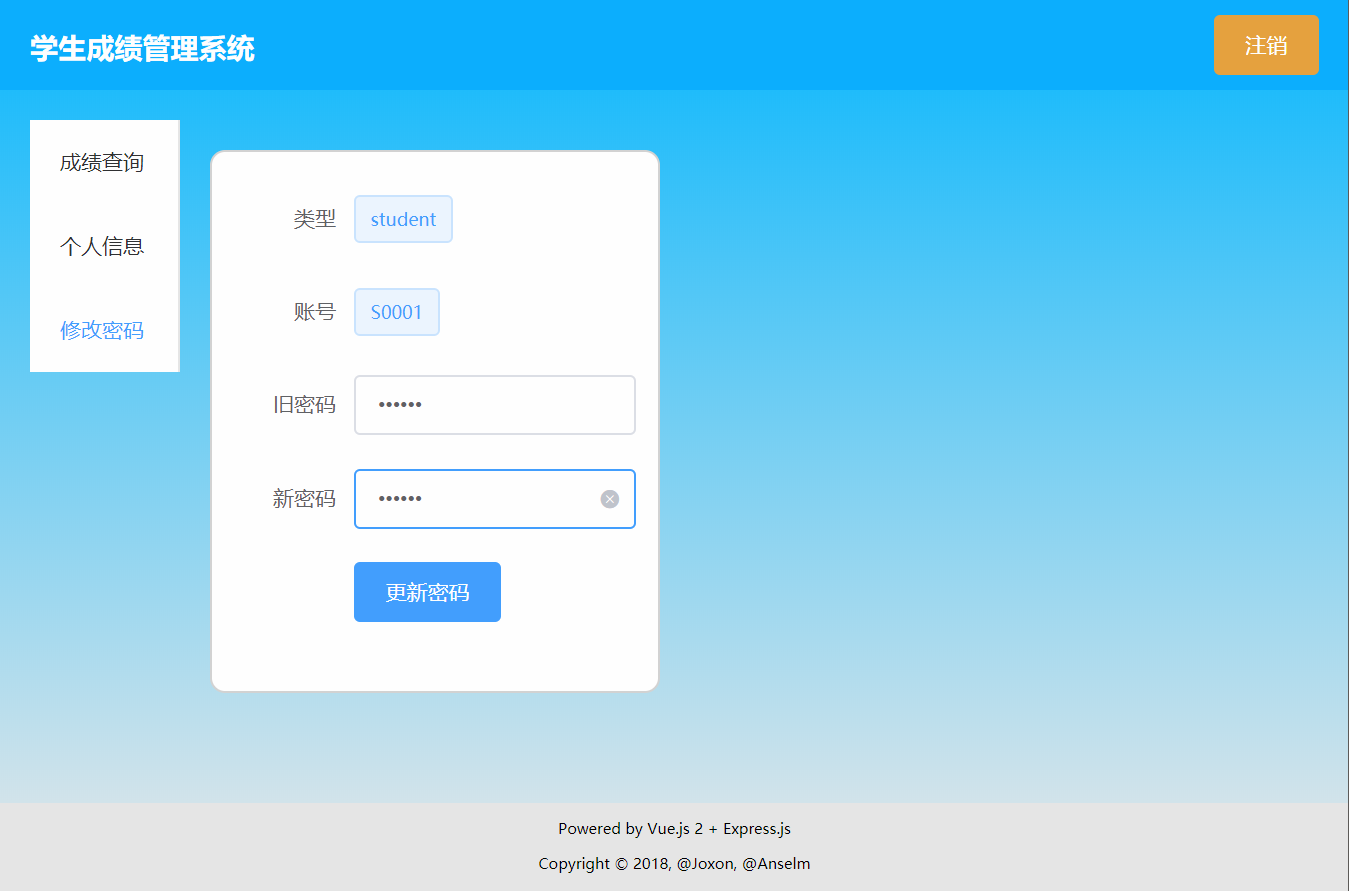
（2）登陆成功成绩查询



（3）查询个人信息



（4）修改账户登录密码



### 4.2 教师部分

（1）登录界面



（2）登陆成功成绩查询



（3）修改学生成绩



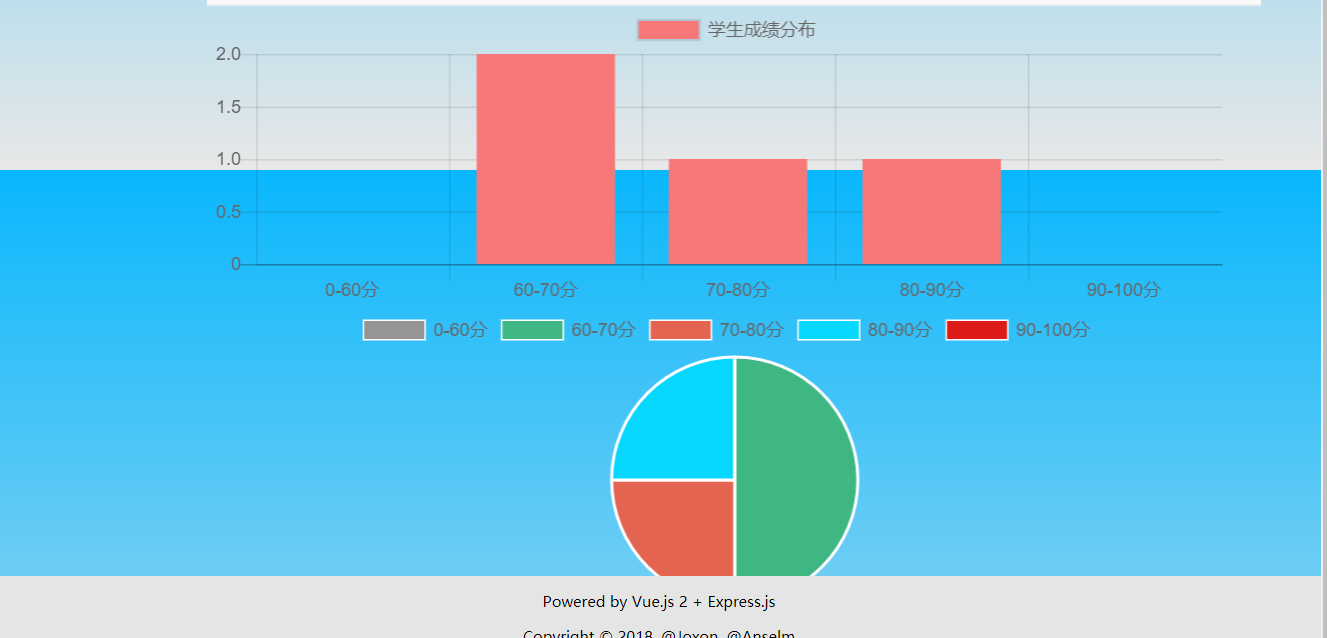


（4）录入学生成绩





（5）成绩直方图和饼状图统计



（6）学生管理界面

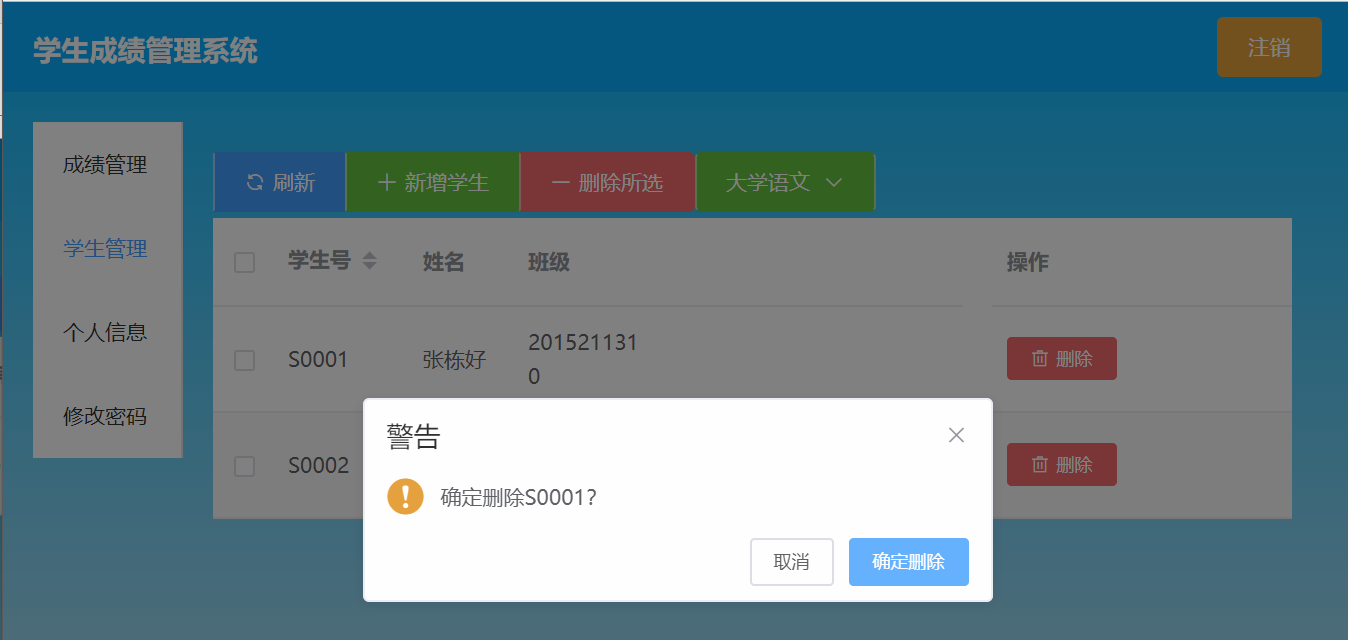


（7）为课程新增学生





（8）删除学生





（9）查看个人信息

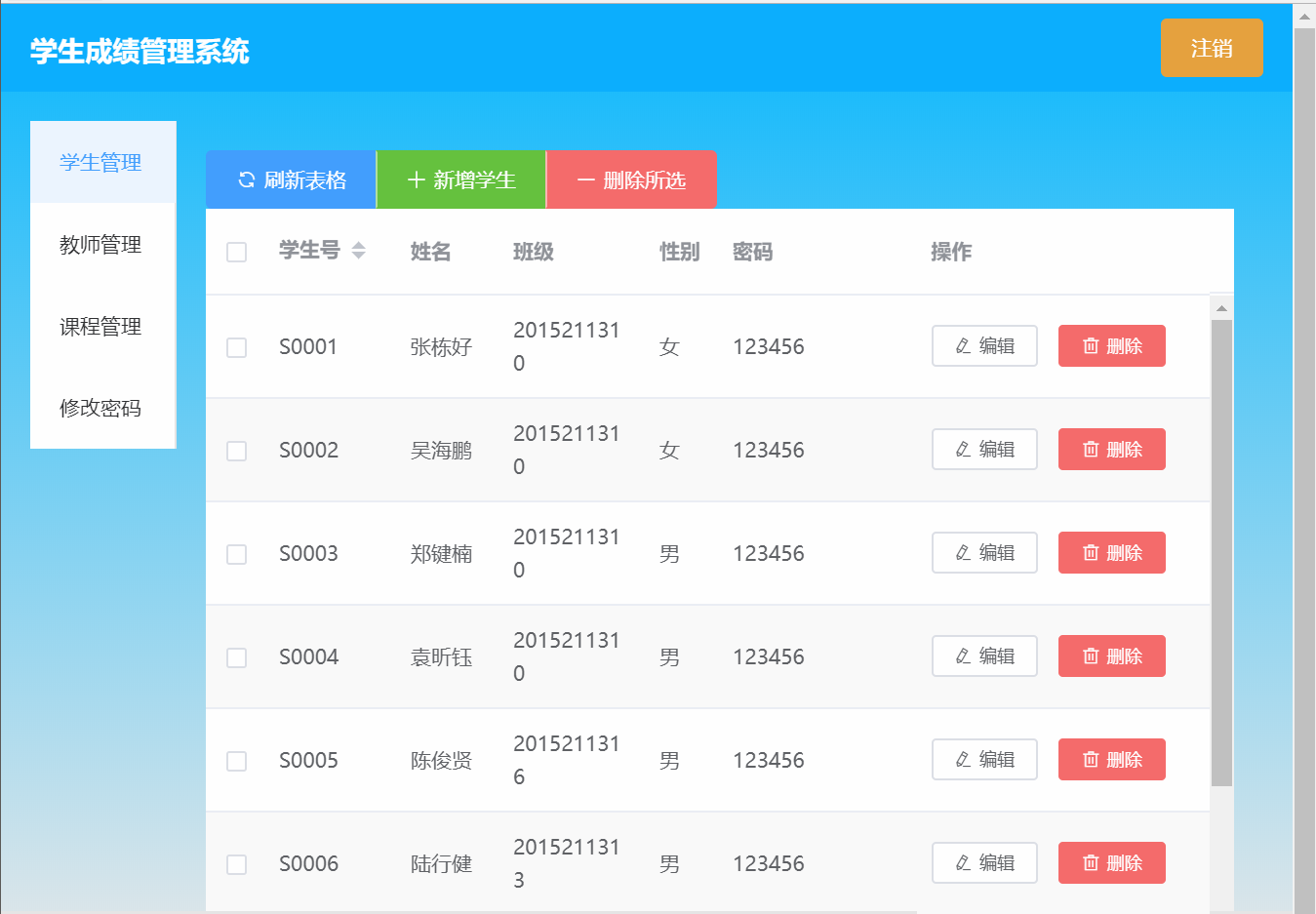


### 4.3 管理员部分

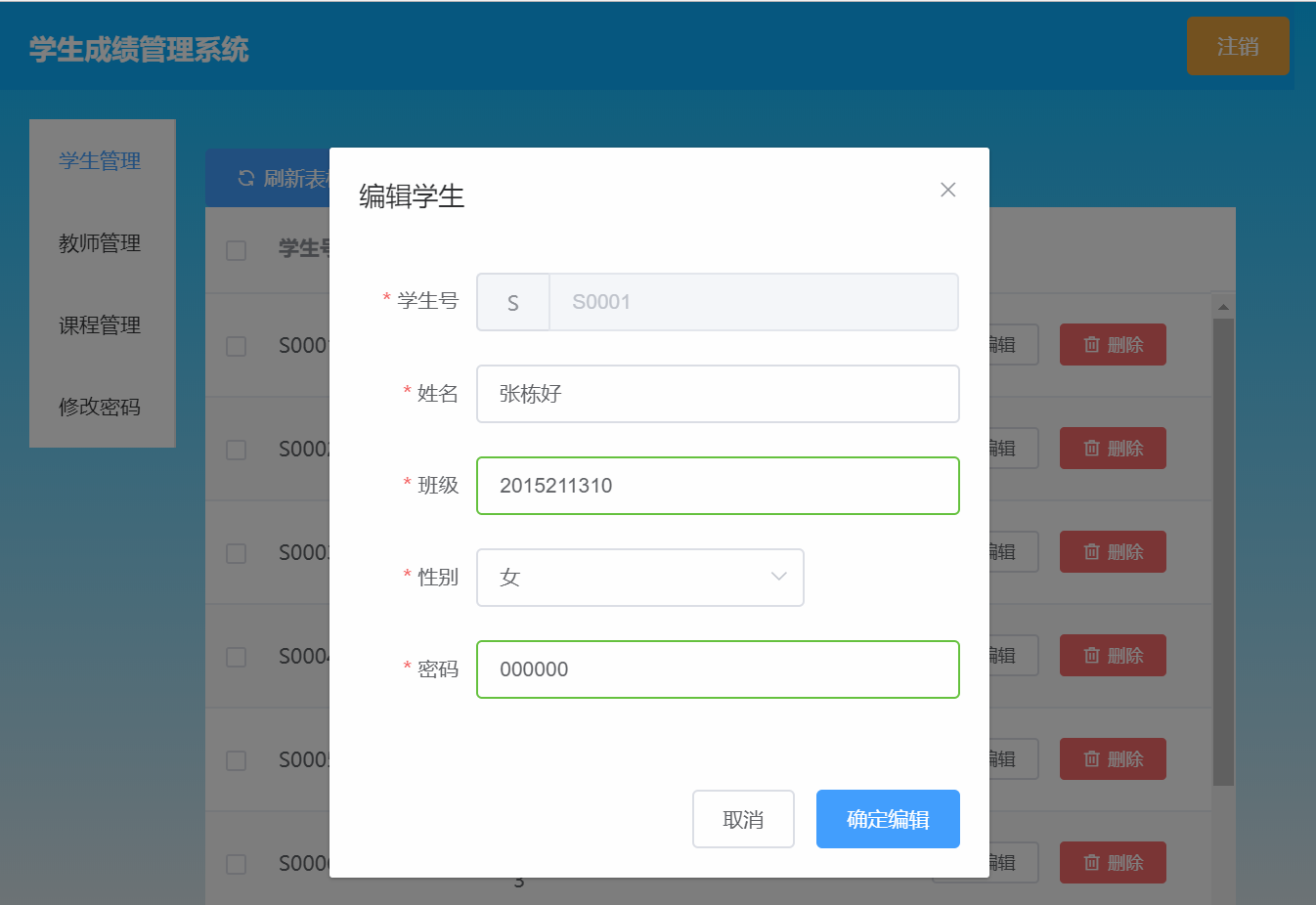
（1）登录界面



（2）登陆之后学生管理界面

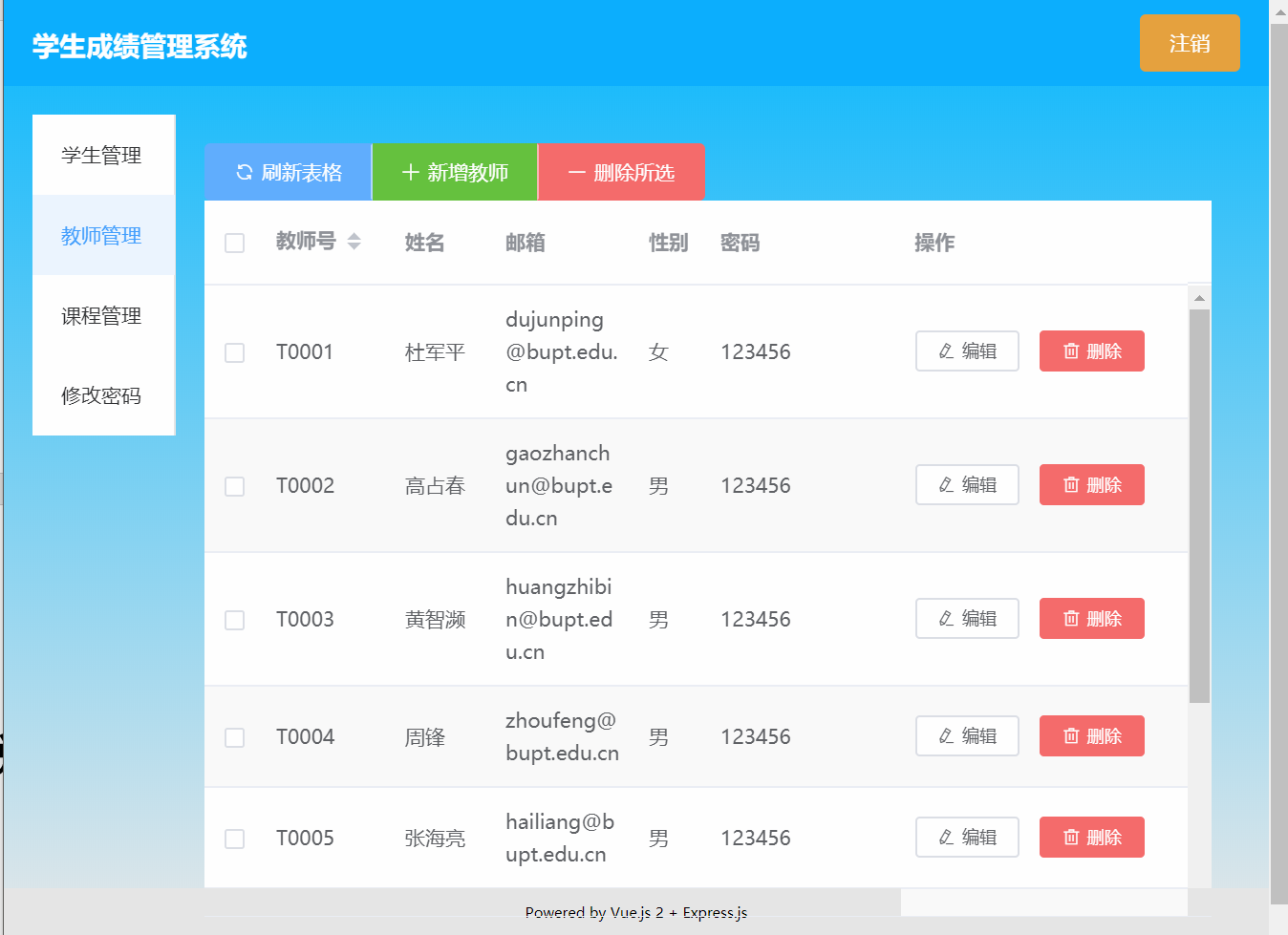


（3）编辑学生信息

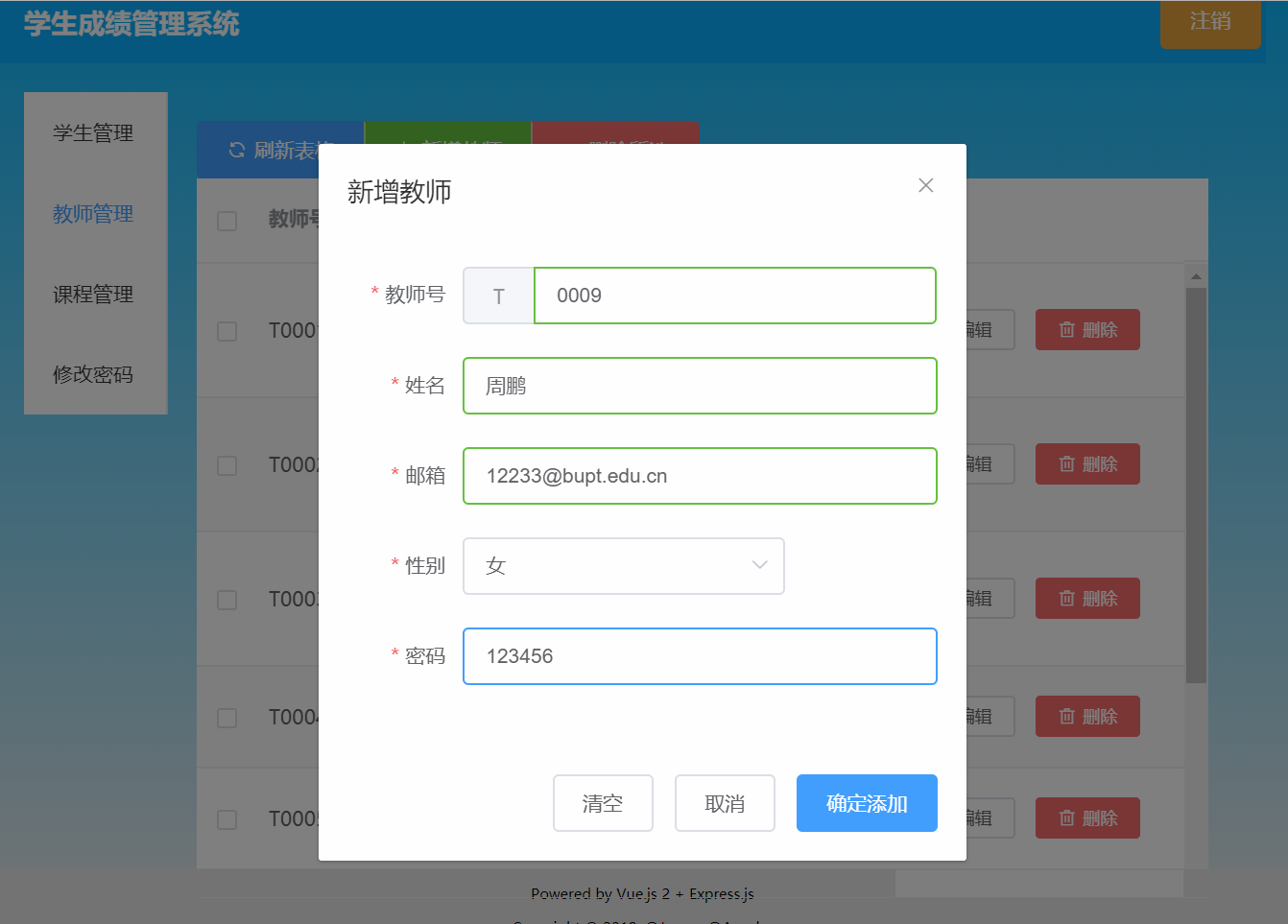




（4）教师管理界面

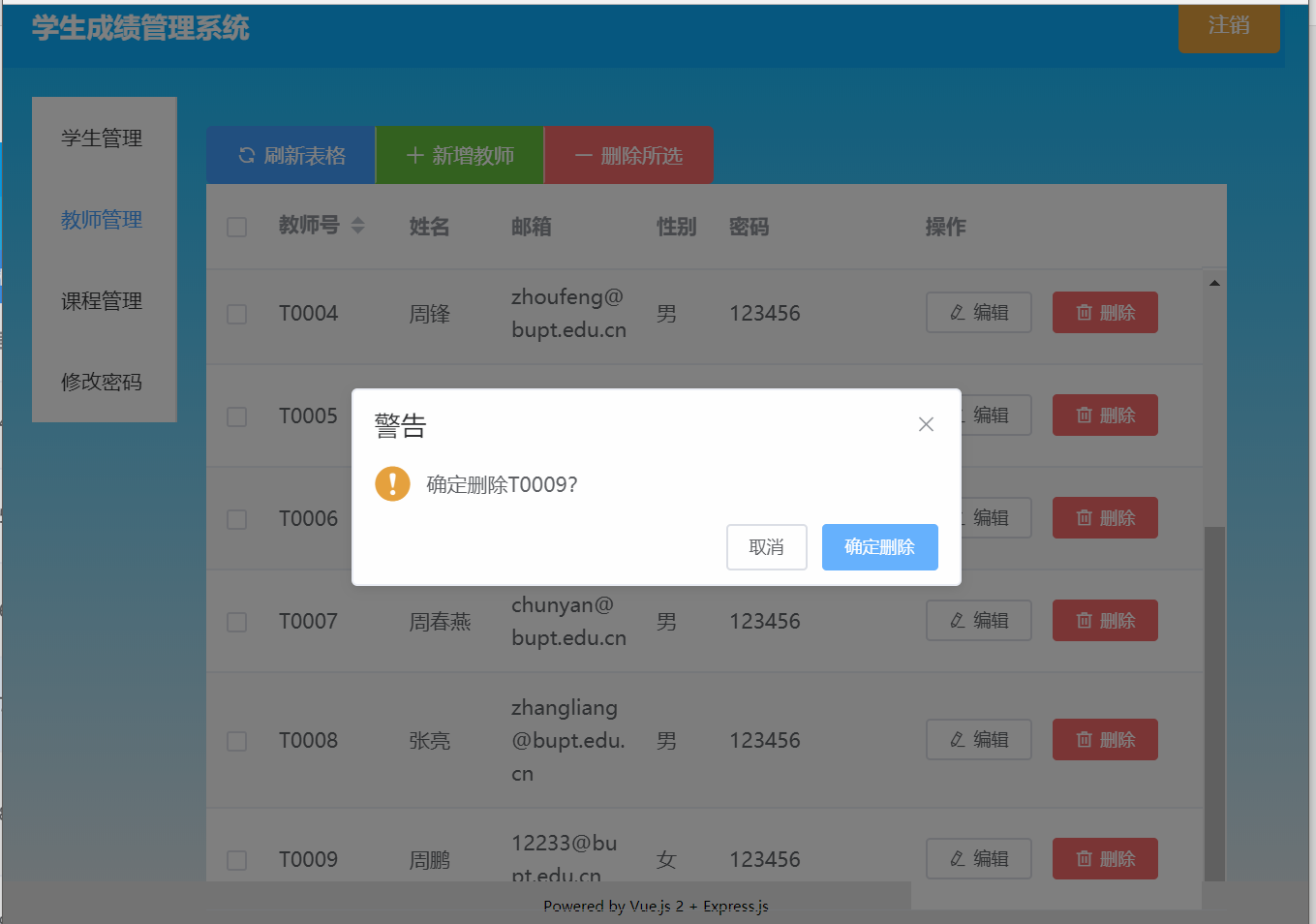


（5）添加教师





（6）删除教师

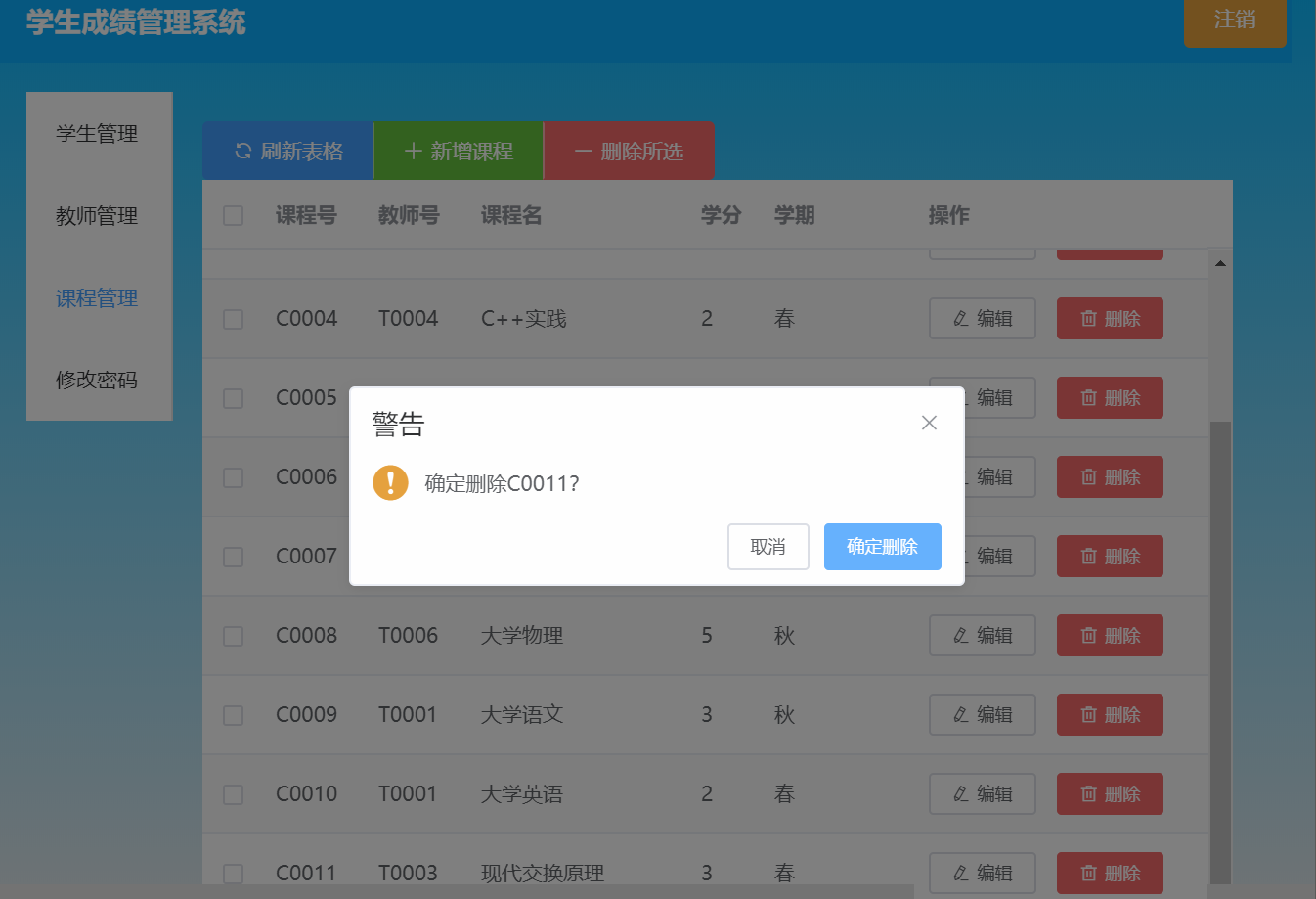




（7）课程管理界面



（8）删除课程





## 第五章 软件使用说明书

### 5.1 前端使用说明

1、安装Nodejs环境，可从 https://nodejs.org/en/ 选择合适版本：

2、安装cnpm加速安装。

打开命令行窗口，输入以下命令：

npm install -g cnpm --registry=https://registry.npm.taobao.org

3、升级 npm

cnpm install npm -g

4、在package.json所在的目录下打开命令行，安装依赖：

cnpm install

5、开发环境运行：

npm start 或 npm run dev，然后在浏览器打开http://localhost:8080/

6、开发完成后，生成html等静态资源

npm run build

7、部署到后端服务器

### 5.2 后端使用说明

1、安装Nodejs环境，可从以下网址选择合适版本：

http://nodejs.cn/download/

2、安装全局npm，打开命令行窗口，输入以下命令：

npm install npm -g

3、使用npm 下载express 以及相关依赖库

在命令行窗口依次输入下述命令：

npm install express

npm install mysql

npm isntall body-parser

npm install cookie-parser

npm install express-session

4、在命令行窗口输入下述命令，安装 nodemon 用来启动项目

npm install -g nodemon

5、在命令行窗口输入下述命令，启动程序

nodemon app.js

## 第六章 课程设计小结

在本次数据库原理课程设计中，我们实现了基于B/S框架的学生成绩管理系统。在整个完成的过程中，我们经历了数据库模式的分析、数据表的建立、前后端框架的选择、前端的构建、后端服务器的构建、前后端功能的调试以及程序的正常运行等过程，对于我们而言收获颇丰。首先，是对于数据库中表的建立时表项的选择以及不同表之间的模式关系的建立有了更为深入的理解。不同表之间关系的建立不仅要能够清晰的表示出学生、教师、课程、成绩等之间的关系模式，同时表之间的关系也要避免过于复杂，才会有利于后期的管理。同时，在建立表时，需要设定适当的主键和外键约束，从而避免数据表中一些非法的数据插入和更改，使得数据的管理更加方便、有效。同时，在实际的使用过程中，可以适当建立视图，存储在数据库结构中，方便不同表之间的查询处理。

在具体的实现过程中，我们分为了前端和后端部分，针对不同模块的功能进行了实现。在前端的页面设计及相关处理中我们使用了Vue框架，在后端服务器的搭建过程中，我们使用了express框架。在具体的实现过程中，遇到了前端页面跳转、数据显示、请求包的设计等问题，后端遇到了请求包解析、页面拦截、身份识别、登录超时等问题，最终我们通过学习、讨论、查询相关资料等方式解决了这些问题，同时，对于前端输入的数据进行了合法性的检测，以避免非法数据造成的错误查询和处理，最终实现了课程设计要求的全部功能。

通过这次课程设计，一方面使得我们对于数据库的相关知识以及网页设计中的一些前后端问题有了更为深入的理解，这也为我们以后的学习积累了一些经验，另一方面，我们再一次体会到了团队的力量。虽然，在这个过程中，我们遇到了很多的问题，有时会让我们短暂的迷茫、不知所措，但是，最终通过我们彼此相互交流讨论、共同学习，一点点地解决了全部的问题。希望这种团队精神能够在以后的学习、工作中继续鼓舞着我们战胜困难、挑战自我。

## 第七章 参考文献

[1] kevin\_zjy，《关于Vue2.0，Express实现的简单跨域》，

<https://www.cnblogs.com/kevin-zjy-blog/p/7357220.html>

[2] m0\_37828249，《Vue项目路径别名设置》

## 第八章 其他说明

源代码以附件的形式进行提交。