****

计算机科学与技术学院

Java Web程序设计

大作业

|  |  |
| --- | --- |
| 班 级： | **软件1803** |
| 姓 名： | **李威** |
| 学 号： | **18110506084** |

2020年7 月

**目 录**

[第一章 系统分析与总体设计 1](#_Toc45538603)

[1.1功能需求分析 1](#_Toc45538604)

[1.2 系统体系结构设计 2](#_Toc45538605)

[1.3 数据库设计 5](#_Toc45538606)

[第二章 系统的实现 7](#_Toc45538607)

[2.1 登录窗体和系统主界面的设计 7](#_Toc45538608)

[2.2 基本数据管理模块（普通管理员）的设计与实现 10](#_Toc45538609)

[2.3 课程安排管理模块（管理员/高级管理员）的设计与实现 15](#_Toc45538610)

[2.4 学生成绩管理模块（管理员/高级管理员）的设计与实现 19](#_Toc45538611)

[2.5 基本数据管理模块（学生/教师）的设计与实现 23](#_Toc45538612)

# 第一章 系统分析与总体设计

## 1.1功能需求分析

易管·信息管理系统是针对学校人事处的大量业务处理工作而开发的管理软件，主要用于学校学生信息管理，总体任务是实现学生信息关系的系统化、科学化、规范化和自动化，其主要任务是用计算机对学生各种信息进行日常管理，如查询、修改、增加、删除，另外还考虑到学生选课，针对这些要求设计了学生信息管理系统。作为计算机应用的一部分，使用计算机对学生档案进行管理，有着手工管理所无法比拟的优点，如：检索迅速、查找方便、可靠性高、存储量大、保密性好、寿命长、成本低等。这些优点能够极大地提高学生档案管理的效率，也是学校向科学化、正规化管理发展的必要条件，更是各个高等院校与世界接轨的重要条件。本系统就是为了更好的管理学生信息而设计的。本系统分为四大功能系统，分别为：管理员使用系统、高级管理员使用系统、学生使用系统、教师使用系统。其中管理员使用系统和高级管理员使用系统中包含：基本数据管理模块，其中包括管理员管理模块、学生管理模块、教师管理模块、班级管理模块、课程管理模块，课程安排管理模块，其中包括学生选课安排模块、教师授课安排模块、公选课安排模块，学生成绩管理模块以及个人信息管理模块；学生使用系统包含：基本数据管理模块，其中包括我的成绩查询模块、我的选课查询模块、申请新课程模块，以及个人信息管理模块；教师使用系统包含：基本数据管理模块，其中包括学生成绩管理模块、我的授课查询模块、申请新授课模块，以及个人信息管理模块。

易管·信息管理系统的用户是普通管理员、高级管理员以及全校师生。本系统采用面向对象的设计思想，可视化的人机交互，系统采用MVC模式，采用了Action控制层+Service业务逻辑层 +Dao数据处理层 +Entity实体层四层架构，使用Servlet视图层做数据接口，分离前后端，是系统自由灵活，逻辑性强，前端使用ajax访问后端数据接口获取数据，使数据展现过程更加直接，提供更好的用户体验。前端采用了一系列插件，包括bootstrap框架，jQuery插件，sweetalert.js等插件，用户操作系统简洁高效，系统风格统一，致力于创造一个优美、舒适的用户界面。前端使用jsp技术创建页面。本系统使用了Filter过滤器，判断用户登录状态以及登录身份，保证系统的安全可靠。

## 1.2 系统体系结构设计

本节将根据需求分析中总结的用户需求设计系统的体系结构。系统的主要功能模块概述图如图1所示：

易管·信息管理系统

高级管理员

管理员

教师

学生

基本数据管理

个人信息管理

学生成绩管理

课程安排管理

基本数据管理

个人信息管理

学生成绩管理

课程安排管理

基本数据管理

个人信息管理

基本数据管理

个人信息管理

图 1系统功能模块概述图

不同的登录身份对应不同的系统和功能，高级管理员/管理员系统功能模块图如图2所示，学生/教师系统功能模块图如图3所示：

高级管理员/管理员

基本数据管理

课程安排管理

个人信息管理

学生成绩管理

班级信息维护

教师信息维护

学生信息维护

学生选课安排

公选课安排

教师授课安排

学生成绩管理

课程信息维护

高级

管理

员独

有

管理员维护

管理

员独

有

修改个人信息

图 2 高级管理员/管理员系统功能模块图

学生

基本数据管理

个人信息管理

我的成绩查询

申请新课程

我的选课查询

修改个人信息

教师

基本数据管理

个人信息管理

学生成绩管理

申请新授课

我的授课查询

修改个人信息

图 3 学生/教师系统功能模块图

在系统功能模块图的结构中，每一个结点都是一个功能模块。

每一个功能模块都需要根据不同的表完成相同的数据库操作：增加记录、删除记录、修改记录及查询显示记录信息等。

在本系统中，可以把功能模块分为以下主要的功能模块来描述。

1．管理员维护模块 包括管理员信息管理、管理员批量添加、管理员信息修改、管理员信息删除和批量删除、管理员信息查询。此集合中的功能模块的结构如图4所示：



图 4 管理员管理功能模块结构图

2．学生信息维护模块 包括学生信息管理、学生信息批量添加、学生信息修改、学生信息删除和批量删除、学生信息查询。此集合中的功能模块的结构与管理员维护模块类似。

3．教师信息维护模块 包括教师信息管理、教师信息批量添加、教师信息修改、教师信息删除和批量删除、教师信息查询。此集合中的功能模块的结构与管理员维护模块类似。

4．班级信息维护模块 包括班级信息管理、班级信息批量添加、班级信息修改、班级信息删除和批量删除、班级信息查询。此集合中的功能模块的结构与管理员维护模块类似。

5．课程信息维护模块 包括课程信息管理、课程信息批量添加、课程信息修改、课程信息删除和批量删除、课程信息查询。此集合中的功能模块的结构与管理员维护模块类似。

6．学生选课安排模块 包括学生选课信息管理、选课信息批量添加、通过和批量通过学生选课申请、删除和批量删除学生选课、选课信息查询、全部未通过选课查询。

7．教师授课安排模块 包括教师授课信息管理、授课信息批量添加、通过和批量通过教师授课申请、删除和批量删除教师授课、授课信息查询、全部未通过授课查询。

8．公选课安排模块 选择班级、课程和教师后，直接为所选班级的学生选课所选课程，并让所选教师授课所选课程。

9．学生成绩管理模块 包括学生成绩信息管理、修改学生成绩、删除和批量删除学生成绩、通过学号或课程编号查询学生成绩信息、根据班级、课程分类学生成绩信息。

10．个人信息管理模块 包括退出登录和修改维护个人信息。

## 1.3 数据库设计

本系统采用MariaDB数据库作为后台数据库。整个系统共使用了8张数据表，分别为：高级管理员信息表，管理员信息表，学生信息表，教师信息表，班级信息表，课程信息表，选课信息表，授课信息表。

下面分别介绍主要表的结构，如表1～表8所示：



表 1 高级管理员信息表



表 2 管理员用户表



表 3 学生信息表



表 4教师信息表



表 5 班级信息表



表 6课程信息表



表 7选课信息表



表 8授课信息表

# 第二章 系统的实现

本系统分为四大功能系统，分别为：管理员使用系统、高级管理员使用系统、学生使用系统、教师使用系统。下面主要介绍了登录窗体的设计与实现、系统主界面的设计等、学生信息维护模块的设计与实现、学生成绩管理模块的设计与实现等。

## 2.1 登录窗体和系统主界面的设计

当系统运行时，首先在浏览器地址栏输入下列地址:

<http://localhost:8888/Management-system/>

会自动跳转到登录界面，只有权限的管理员、高级管理员、学生和老师才能进入系统。在登录界面点击不同的身份后，会弹出不同的登录窗体，然后用户名及密码与当前身份数据库里的字段信息匹配时，用户才能进入对应的信息管理系统。登录成功后将显示系统主界面。如果登录用户不能通过系统的身份验证，此用户就不能进入系统。打开页面后跳转到登录界面如图5所示：



图 5初始登录界面

假如一位管理员身份的用户需要登录，点击“我的是管理员”之后，会出现管理员用户名、密码以及验证码的输入框，如图6所示：



图 6 点击“我是管理员”之后出现了登录窗体

此时若是要使用其他身份登录，点击“更换身份”按钮，登录窗体即会消失，然后点击其他身份登录即可。

当点击登录后，系统提示登录成功时，即会自动跳转到主页面，主界面右上角有会自动刷新的当前时间、实时天气、和用户名，其次是菜单导航栏，如图7所示（下图是普通管理员登录的界面，其他身份登录后菜单栏不同）：

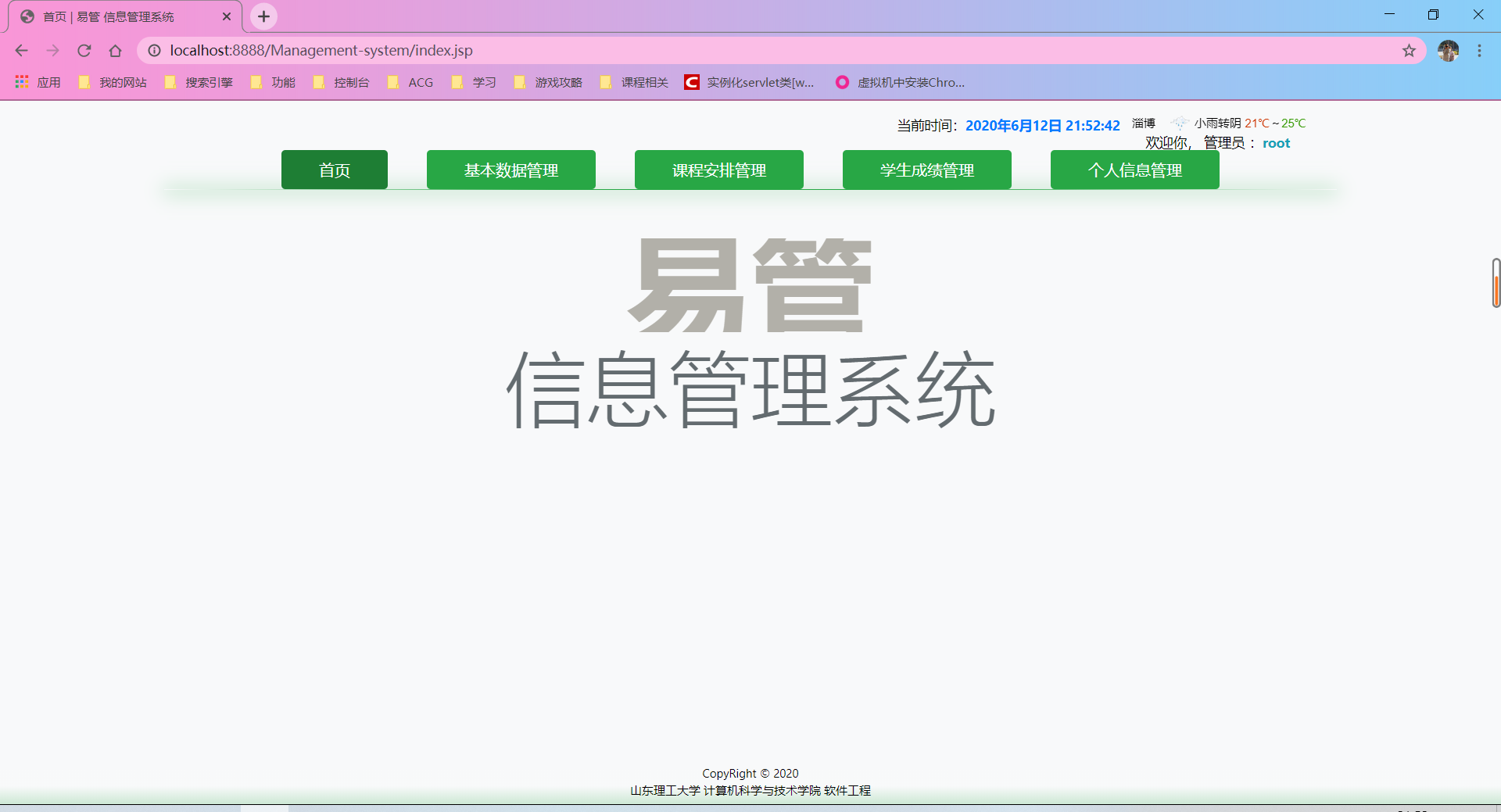


图 7 系统主界面（普通管理员身份）

登录方法中使用到的部分Ajax技术代码如下：

$.ajax({

type: "post",

url: "./LoginServlet",

data: {

tbuser: tbuser,

tbpwd: tbpwd,

identity: Identity

},

success: **function**(e) {

**if** (e == "success") {

// 登录成功

swal("成功", "登录成功!", "success");

// 延时1s跳转到index.jsp主页面

setTimeout(() => {

location.href = "./index.jsp"

}, 1000);

} **else** **if** (e == "failed") {

// 登录失败

Tbpwd.addClass("is-invalid");

tbpwdFeedback.addClass("invalid-feedback");

tbpwdFeedback.text("您的登录名或密码错误!");

swal("错误", "登录失败，您的登录名或密码错误!", "error");

} **else** {

// 未知错误

swal("错误", "出现未知错误!" + e, "error");

}

},

error: **function**(e) {

// 网络或服务器错误

swal("错误", "网络堵塞或服务器故障!", "error");

console.log(e);

}

});

登录方法中使用到的部分Servlet技术代码如下：

**if** ("user".equals(Identity)) {

Tbuser User = **new** Tbuser();

// 把用户名、密码封装在Tbuser对象中

User.setTbuser(Name);

User.setTbpwd(Pwd);

// 把user对象传给login方法

Tbuser u = userAction.login(User);

**if** (u != **null**) {

session.setAttribute("username", "user" + Name);

response.addCookie(**new** Cookie("identity", "user".toString()));

response.addCookie(**new** Cookie("uid", u.getTbid().toString()));

out.print("success");

} **else** {

out.print("failed");

}

}

登录方法中使用到的部分数据层（Dao）技术代码如下：

**public** Tbuser login(Tbuser user) {

String sql = "SELECT \* FROM tbuser WHERE tbuser=? AND tbpwd=?";

Object params[] = **new** Object[] { user.getTbuser(), user.getTbpwd() };

**return** **this**.selectOne(sql, Tbuser.**class**, params);

}

## 2.2 基本数据管理模块（普通管理员）的设计与实现

本部分的基本信息模块主要包括学生信息维护，教师信息维护，班级信息维护，课程信息维护。这四大模块具有相似性，操作、功能等方面具有相似性，这里具体讲述学生信息维护模块的设计与实现。

点击基本数据管理后，再点击次级导航栏的学生信息维护，出现的界面如图8所示：

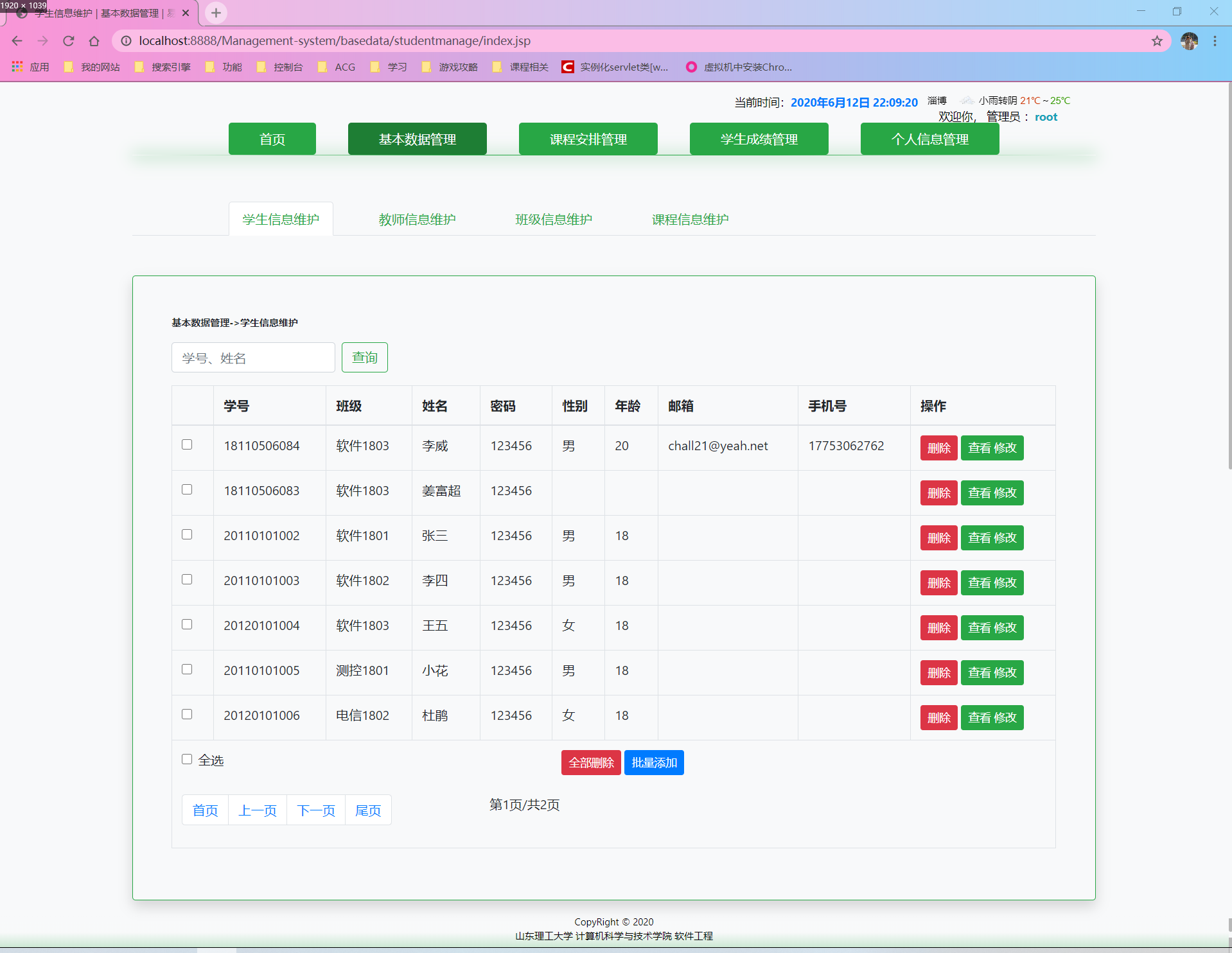


图 8 基本数据管理(普通管理员)的学生信息维护模块

在当前界面可以：对单条数据进行删除或者查看/修改，也可以点击全选按钮对数据进行批量操作，在主窗体上左上方有搜索输入框，可以模糊匹配学生学号姓名返回查询数据。页面默认显示7行数据，左下方有分页按钮，可以选择查看上一页、下一页、首页或者尾页，点击后退跳转到对应的页数，若是已经在首页，点击上一页或首页则没有效果，若是已经在尾页，同理。

点击底部批量添加按钮，跳转到学生信息批量添加页面，默认有一行信息输入栏，可以点击页面底部的“+”按钮添加多行数据栏，也可以点击每行数据栏后面都对应的“-”按钮对本行数据栏进行删除。其中学号、姓名、密码属性是必填项，班级属性是必选项，其他属性为选填项，如图9所示。点左下方批量提交按钮后，系统提示添加成功的信息时，即添加信息成功，自动跳转到主页面，如图10所示：



图 9学生信息批量添加页面

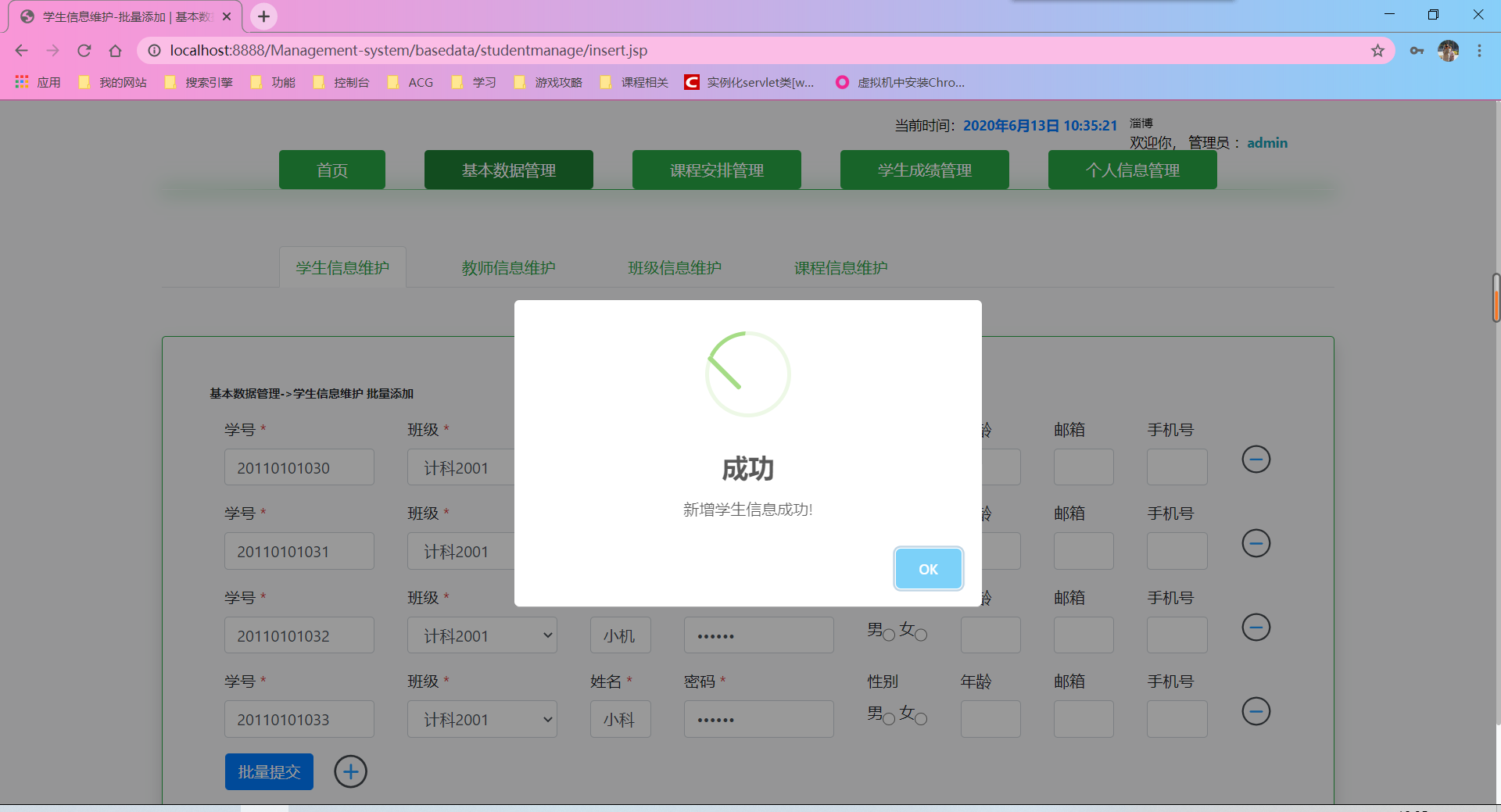


图 10点批量提交后，成功提交的系统提示信息

然后在学生信息维护主页面便可以查看到刚刚提交的学生信息。

在数据栏和数据底部都有删除按钮，点击后，若成功删除会出现删除成功的提示信息。成功删除后，删除的学生信息就不会再出现在信息界面，如图11所示：

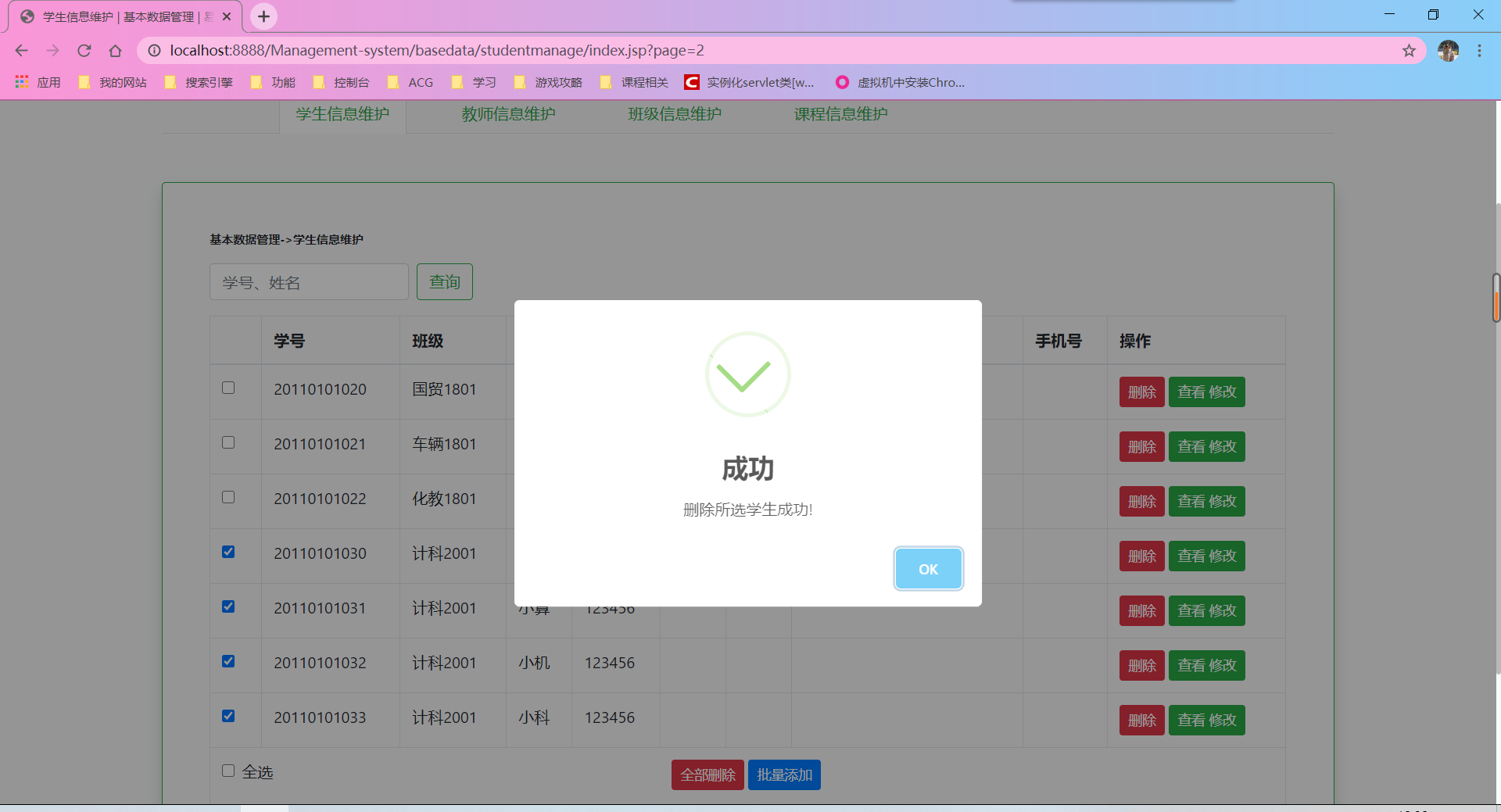


图 11批量删除按钮点击后删除所选学生信息

主页面左上方有搜索输入框和搜索按钮，输入学号或者学生姓名的关键字，点击查询即可得到查询结果，查询界面同样可以进行主页面相似的操作，包括数据栏单项数据的删除，查看/修改，全部删除和批量添加。搜索18级男生，即搜索关键词为1811，查询结果如图12所示：

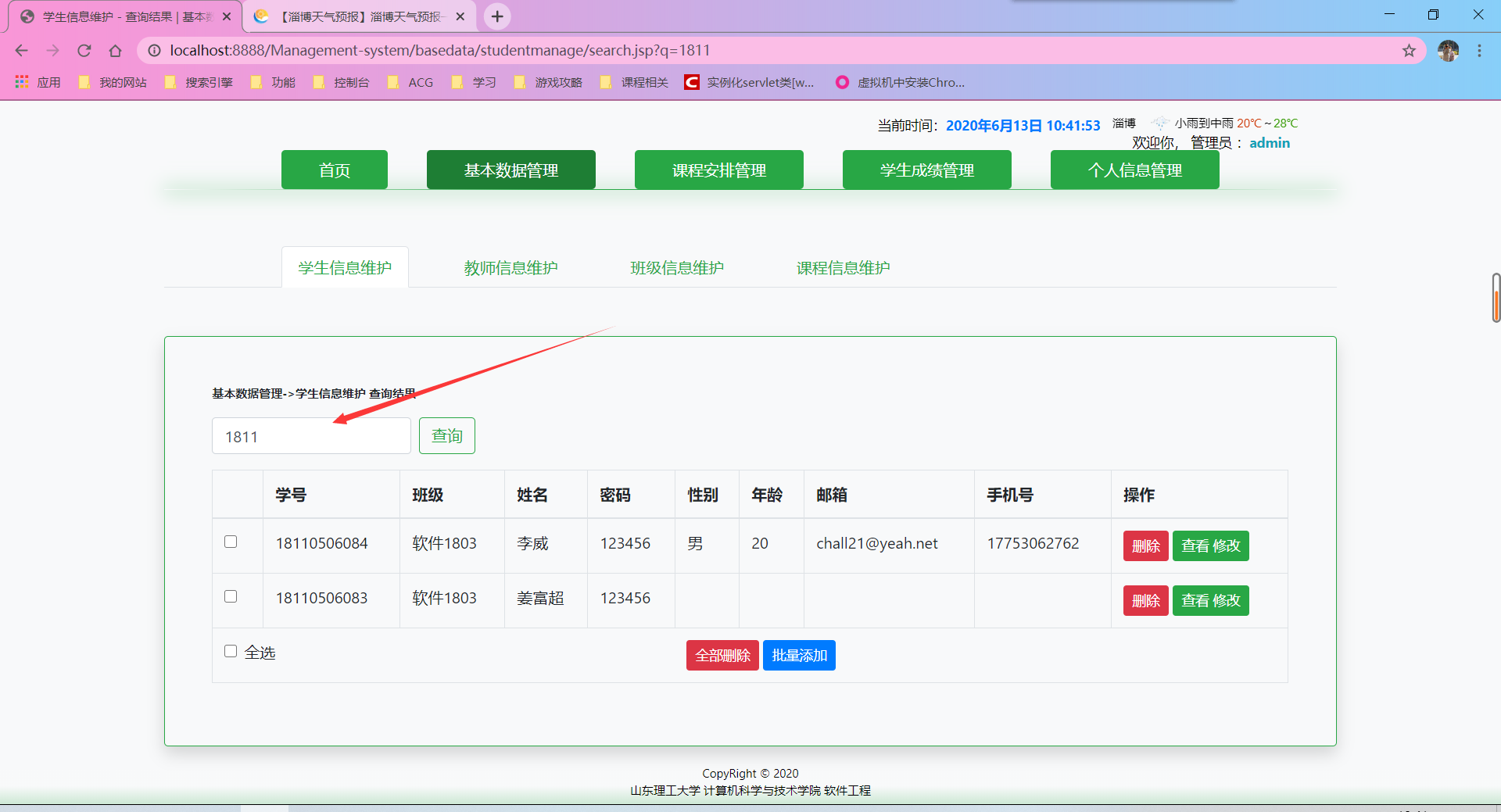


图 12搜索“1811”后出现的查询结果界面

每条数据栏后方都对应一个查看/修改按钮，点击后进入该条数据的详情界面。在这个界面中，可以对学生数据进行修改维护，点击保存后即可提交修改的数据。如图13所示：

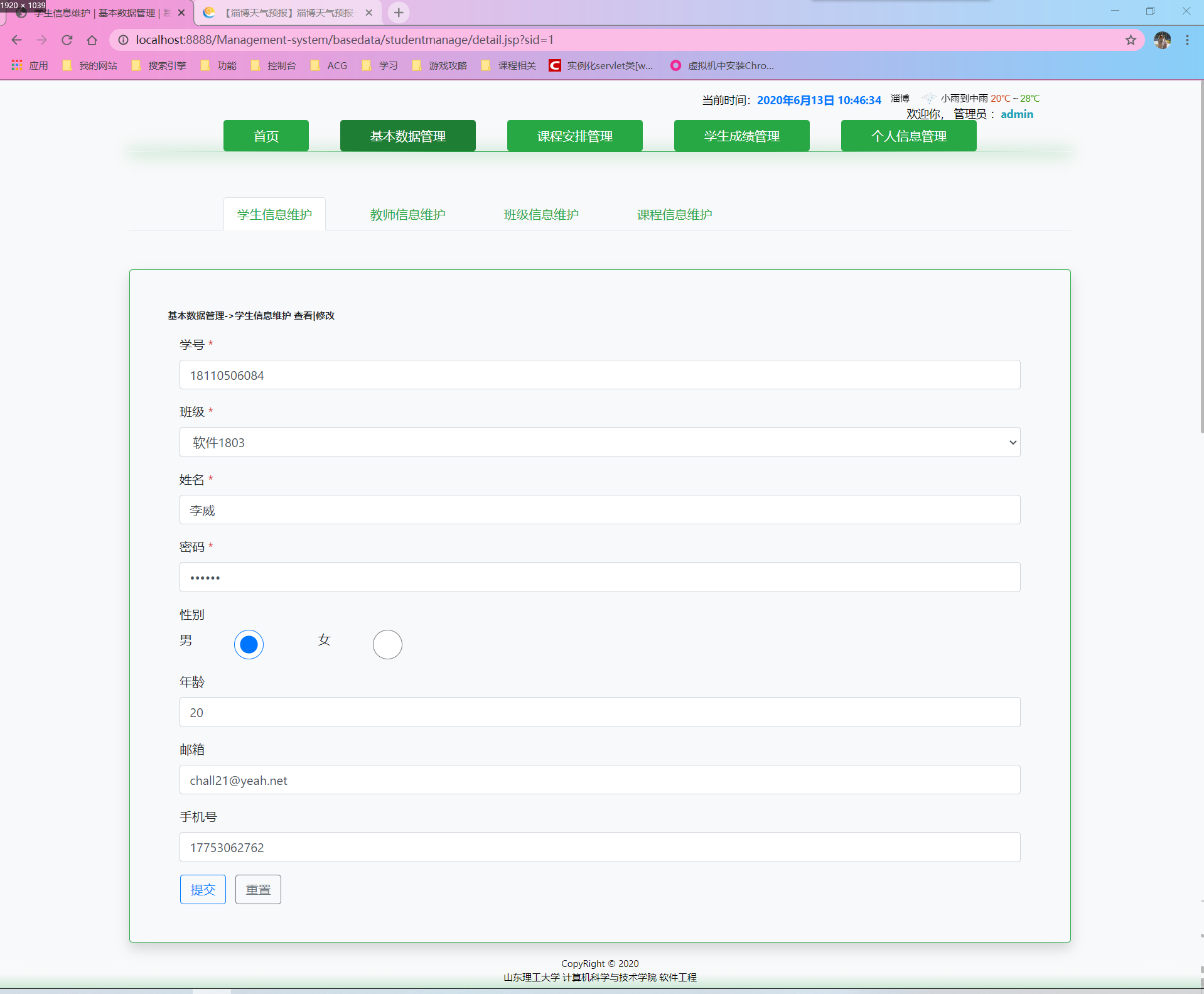


图 13学生姓名为"李威"的学生信息详情界面

其他功能模块，班级信息维护模块中，删除班级会同步删除班级内所有学生。班级信息维护模块中有一列为班级人数项，课程信息维护模块中有一列为课程人数项，此项不在数据库中存储，而是通过学生信息表和学生选课表分组查询出的。

学生信息数据展示中使用到的部分Ajax技术代码如下：

// 发送ajax 请求数据

$.ajax({

type: "post",

url: "../../StudentManageListServlet",

success: **function** (e) {

**if** (e == "empty") {

**return**;

}

// 解析管理员信息

**var** stus = JSON.parse(e);

// 获取管理员总数

**var** length = stus.length;

// 获取分页总数

**var** pagenum = Math.ceil(length \* 1.0 / MAXSHOW);

// 获取当前页数

**var** pagenow = getQueryVariable("page");

**if** (pagenow == **false**) {

pagenow = 1; // 默认为第一页

}

// 输出本页表格数据

**var** maxi = pagenow \* MAXSHOW < length ? pagenow \* MAXSHOW : length;

// 先查询空数据，对未定义属性进行填充空值

**for** (**var** i = (pagenow - 1) \* MAXSHOW; i < maxi; i++) {

**if** (!stus[i]['sex']) stus[i]['sex'] = "";

**else** **if** (stus[i]['sex'] == 1) stus[i]['sex'] = "男";

**else** stus[i]['sex'] = "女";

**if** (!stus[i]['sage']) stus[i]['sage'] = "";

**if** (!stus[i]['email']) stus[i]['email'] = "";

**if** (!stus[i]['phone']) stus[i]['phone'] = "";

}

**for** (**var** i = (pagenow - 1) \* MAXSHOW; i < maxi; i++) {

tbody.append('\

<tr sid=' + stus[i]['sid'] + ' class="user-item">\

<td>\

<div class="form-check">\

<input class="form-check-input" type="checkbox" name="checkuser">\

</div>\

</td>\

<td>' + stus[i]['sno'] + '</td>\

<td>' + stus[i]['cname'] + '</td>\

<td>' + stus[i]['sname'] + '</td>\

<td>' + stus[i]['spwd'] + '</td>\

<td>' + stus[i]['sex'] + '</td>\

<td>' + stus[i]['sage'] + '</td>\

<td>' + stus[i]['email'] + '</td>\

<td>' + stus[i]['phone'] + '</td>\

<td>\

<button type="button" class="dropbtn btn btn-danger btn-sm">删除</button>\

<button type="button" class="checkbtn btn btn-success btn-sm" onclick="location.href=\'./detail.jsp?sid=' + stus[i]['sid'] + '\'">查看 修改</button>\

</td>\

</tr>\

');

}

// 表格底部批量操作栏

tbody.append('\

<!-- 表格底部批量操作栏 -->\

<tr class="table-borderless">\

<td colspan="5">\

<div class="form-check">\

<input class="form-check-input" type="checkbox" id="checkAll">\

<label class="form-check-label" for="checkAll">全选</label>\

</div>\

</td>\

<td colspan="5">\

<button type="button" class="dropallbtn btn btn-danger btn-sm">全部删除</button>\

<button type="button" class="btn btn-primary btn-sm" onclick="location.href=\'./insert.jsp\'">批量添加</button>\

</td>\

</tr>\

');

// 定义上一页

**var** lastpage = pagenow - 1 > 1 ? pagenow - 1 : 1;

// 定义下一页

**var** nextpage = pagenow + 1 < pagenum ? pagenow + 1 : pagenum;

// 表格底部分页栏

tbody.append('\

<!-- 表格底部分页栏 -->\

<tr class="table-borderless">\

<td colspan="4">\

<nav aria-label="Page navigation example">\

<ul class="pagination">\

<li class="page-item"><a class="page-link" href="./index.jsp?page=1">首页</a></li>\

<li class="page-item"><a class="page-link" href="./index.jsp?page=' + lastpage + '">上一页</a></li>\

<li class="page-item"><a class="page-link" href="./index.jsp?page=' + nextpage + '">下一页</a></li>\

<li class="page-item"><a class="page-link" href="./index.jsp?page=' + pagenum + '">尾页</a></li>\

</ul>\

</nav>\

</td>\

<td colspan="6">\

<span>第<span class="nowpage">' + pagenow + '</span>页/共<span class="allpage">' + pagenum + '</span>页</span>\

</td>\

</tr>\

');

},

error: **function** (e) {

// 网络或服务器错误

swal("错误", "网络堵塞或服务器故障!", "error");

}

});

学生信息数据展示中使用到的部分Servlet技术代码如下：

**protected** **void** doPost(HttpServletRequest request, HttpServletResponse response)

**throws** ServletException, IOException {

response.setContentType("text/html;charset=UTF-8");

PrintWriter out = response.getWriter();

// 查询所有学生信息

List<Student> list = action.queryAll();

// 将信息集合转化为数组

Gson gson = **new** Gson();

String json = gson.toJson(list);

// 输出信息

out.print(json);

}

学生信息数据展示中使用到的部分数据层（Dao）技术代码如下：

**public** List<Student> queryAll() {

StringBuffer sb = **new** StringBuffer();

List<Object> list = **new** ArrayList<>();

sb.append("SELECT `student`.\*, `class`.`cname` ");

sb.append("FROM `student`, `class` ");

sb.append("WHERE `student`.`cno` = `class`.`cno` ");

sb.append("ORDER BY sid");

String sql = sb.toString();

**return** **this**.selectAll(sql, Student.**class**, list.toArray());

}

## 2.3 课程安排管理模块（管理员/高级管理员）的设计与实现

本部分的课程安排管理模块主要包括学生选课安排，教师授课安排，公选课安排。其中学生选课安排模块和教师授课安排模块在操作、功能等方面具有相似性，这里具体讲述学生选课模块和公选课安排模块的设计与实现。

首先点击导航菜单栏中的课程安排管理，进入课程安排管理主页面，再点击次级导航栏菜单中的学生选课安排，进入学生选课安排主界面，如图所示：

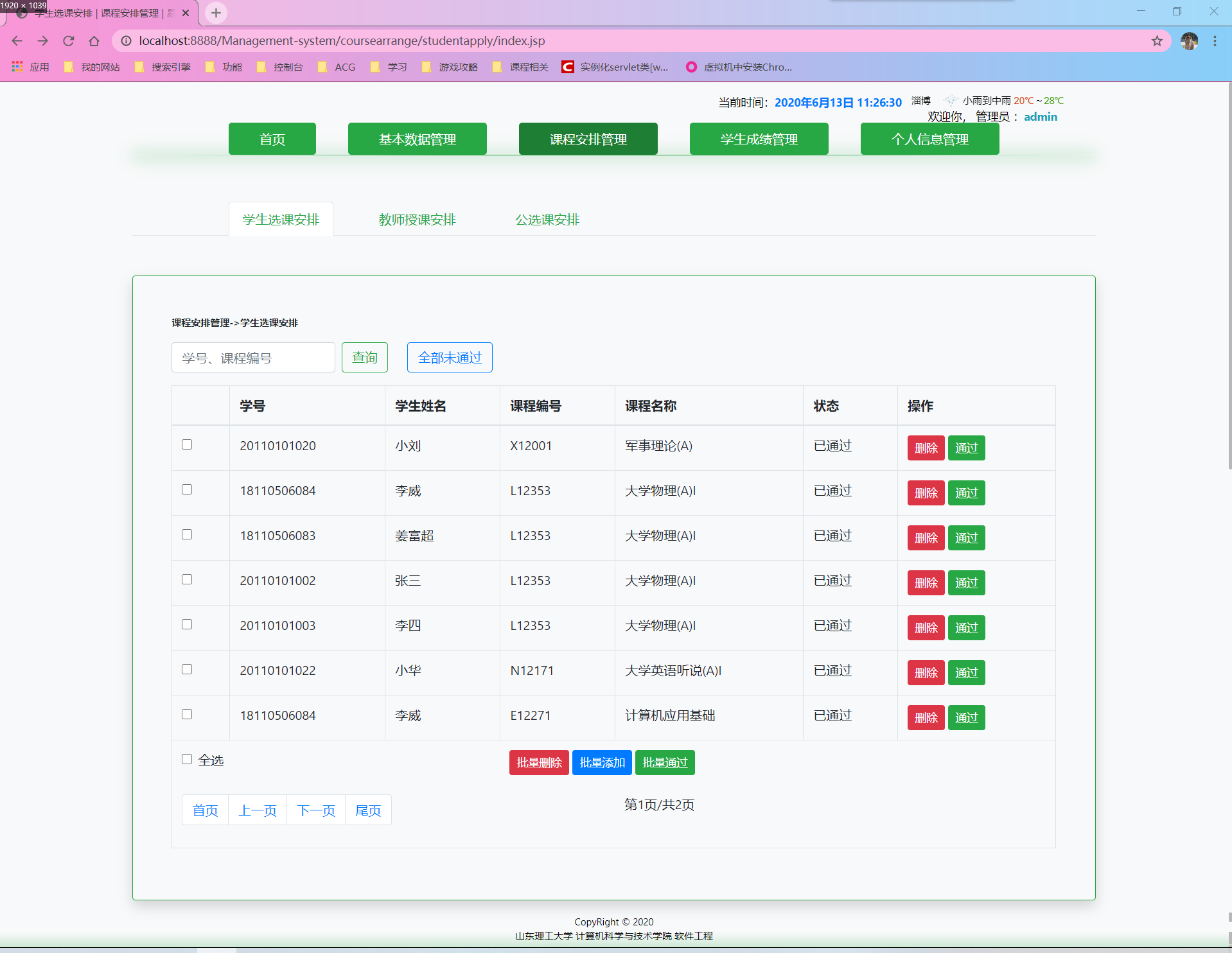


图 14 学生选课安排主界面

学生选课安排界面包括基本数据的查看，单项数据的删除和通过操作，批量数据的删除、通过、添加，其中删除功能和批量删除功能，查询功能和批量添加功能与2.2基本信息管理模块的设计和实现中，学生信息维护等模块功能类似，这里不再赘述。

学生选课安排模块中的通过操作，是相对于学生而言的。管理员新增的选课默认是通过状态，只有学生通过登录学生身份，申请新选课，默认是等待通过状态，需要管理员核实后决定是同通过学生的申请，若申请不符合要求，管理员可以删除此项选课申请。

点击数据栏上方的“全部未通过”按钮即可查看当前有哪些学生的选课申请未通过，如图15所示：

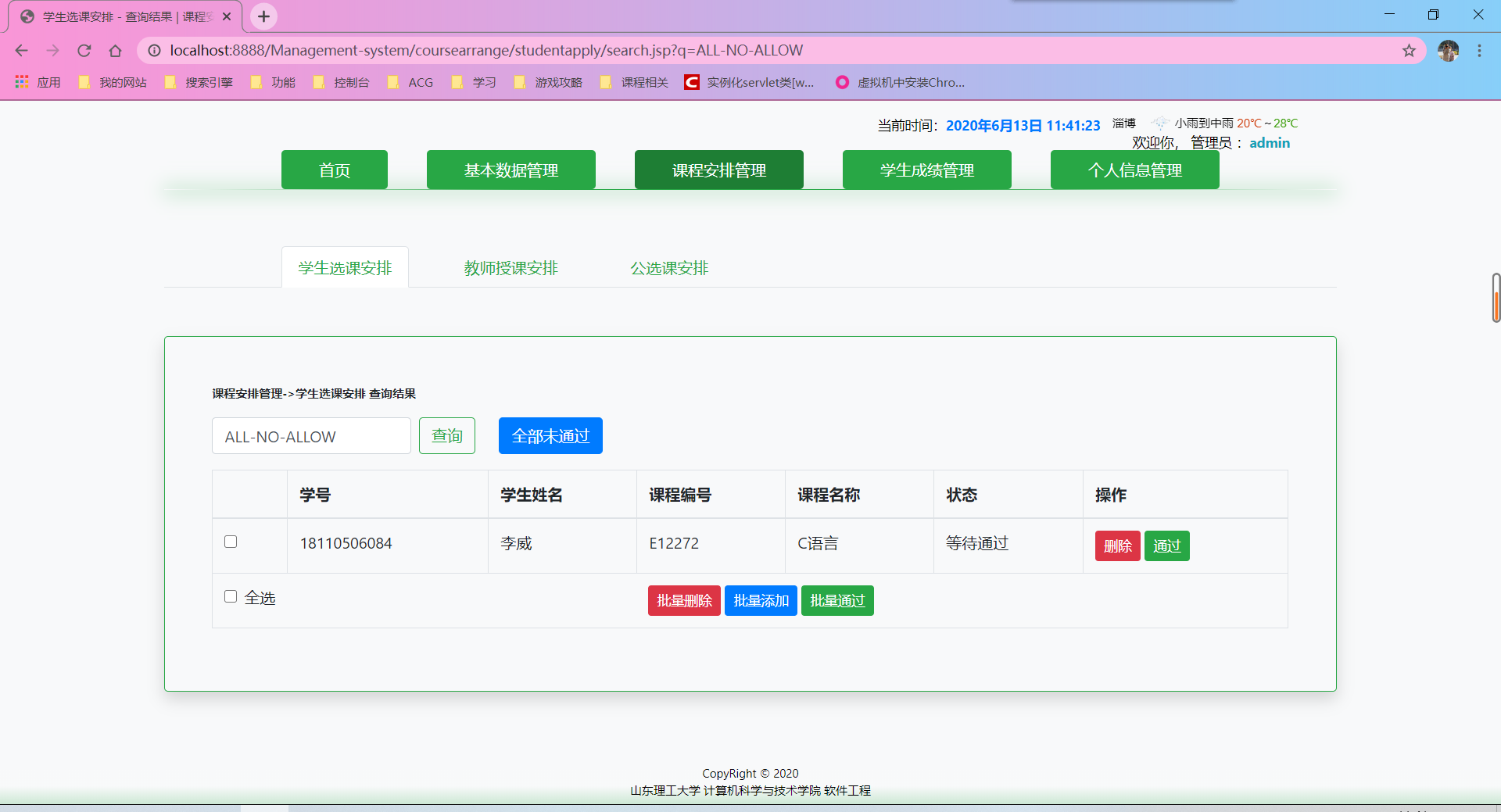


图 15“全部未通过”查询按钮点击后的查询结果

公选课安排模块只有一种功能，安排公选课。不同于学生选课安排或教师选课安排中的批量添加操作的是，公选课安排通过选择班级、课程、教师这三个选项，即可批量为所选班级学生安排新课程，默认是通过状态，可同时为教师安排新授课，默认是通过状态，十分方便，如图16所示：



图 16公选课安排模块，图为点击选择课程之后的状态

选择班级、教师、课程，点击提交后，系统出现添加公选课成功的提示即顺利添加。

公选课安排模块部分JavaScript代码（提交按钮点击事件）如下：

// 提交按钮点击事件

$("#Submitbtn").click(**function** () {

**var** insertItem = $(".insert-item");

**var** Tno = $(".Tno");

**var** Clno = $(".Clno");

**var** Cono = $(".Cono");

**var** TnoFeedback = $(".TnoFeedback");

**var** ClnoFeedback = $(".ClnoFeedback");

**var** ConoFeedback = $(".ConoFeedback");

// 检查数据长度

**if** (insertItem.length == 0) {

swal("糟糕", "没有添加任何公选课信息!", "warning");

**return**;

}

**var** i = 0;

// 检查教师、班级、课程

**var** tno = Tno.eq(i).val();

**if** (tno.length == 0) {

Tno.eq(i).addClass("is-invalid");

TnoFeedback.eq(i).addClass("invalid-feedback");

TnoFeedback.eq(i).text("请选择教师");

**return**;

}

**var** clno = Clno.eq(i).val();

**if** (clno.length == 0) {

Clno.eq(i).addClass("is-invalid");

ClnoFeedback.eq(i).addClass("invalid-feedback");

ClnoFeedback.eq(i).text("请选择班级!");

**return**;

}

**var** cono = Cono.eq(i).val();

**if** (cono.length == 0) {

Cono.eq(i).addClass("is-invalid");

ConoFeedback.eq(i).addClass("invalid-feedback");

ConoFeedback.eq(i).text("请选择课程!");

**return**;

}

// 发送ajax 提交批量插入请求

$.ajax({

type: "post",

url: "../../PublicApplyInsertServlet",

data: {

tno: tno,

clno: clno,

cono: cono

},

success: **function** (e) {

**if** (e == "success") {

// 插入数据成功

swal("成功", "新增公选课信息成功!", "success");

// 延时1s跳转刷新界面

setTimeout(() => {

location.replace("./index.jsp");

}, 1000);

} **else** **if** (e == "failed") {

// 插入数据失败

swal("错误", "批量新增公选课信息失败!", "error");

} **else** {

// 未知错误

swal("错误", "出现未知错误!", "error");

}

},

error: **function** (e) {

// 网络或服务器错误

swal("错误", "网络堵塞或服务器故障!", "error");

}

});

});

公选课安排模块中使用到的部分业务逻辑层（Service）技术代码如下：

**public** **int** publicChoose(String tno, String clno, String cono) {

// 先新增教师授课

Teaching teaching = **new** Teaching();

teaching.setCno(cono);

teaching.setTno(tno);

teaching.setStatus(1);

**this**.teachingDao.insertTeaching(teaching);

// 再根据clno查询班级内学生信息

List<Student> student = **this**.studentDao.queryStudentByCno(clno);

// 再把所有学生选课

**return** **this**.chooseDao.insertChooseBatchPublic(student, cono);

}

公选课安排模块中使用到的部分数据处理层（Dao）技术代码如下：

**public** **int** insertChooseBatchPublic(List<Student> student, String cno) {

StringBuffer sb = **new** StringBuffer();

List<Object> list = **new** ArrayList<>();

sb.append("INSERT INTO `choose` (`scid`, `sno`, `cno`, `status`, `grade`) VALUES ");

// 遍历Choose List

**for** (**int** i = 0; i < student.size(); i++) {

Student s = (Student) student.get(i);

list.add(s.getSno());

list.add(cno);

sb.append("(NULL, ?, ?, '1', NULL),");

}

sb.deleteCharAt(sb.length() - 1);

String sql = sb.toString();

**return** **this**.saveOrUpdate(sql, list.toArray());

}

## 2.4 学生成绩管理模块（管理员/高级管理员）的设计与实现

本部分的学生成绩管理模块主要学生成绩查看、按条件查询学生成绩，学生成绩单项数据的修改成绩和删除成绩，批量删除成绩，分页功能。

点击最上方导航栏的学生成绩管理，进入学生成绩管理主页面，再点击次级导航栏菜单的学生成绩管理进入学生成绩管理模块。

每一项成绩包含7个属性，分别是课程编号、课程名称、学号、姓名、班级、任课教师以及成绩，其中成绩存储在选课信息表中，对应每一条选课信息。只有选课状态为通过状态时，才能在此界面中查询出来。成绩属性值，值为“/”的为未公布成绩，需要管理员或者对应的授课教师才能修改发布。任课教师属性值，值为“/”的表示没有任课教师，值为多个教师名的是因为多个教师都在为此课程授课。清楚的显示每一条数据以便管理员准确查找对应学生，发布学生成绩，如图17所示：

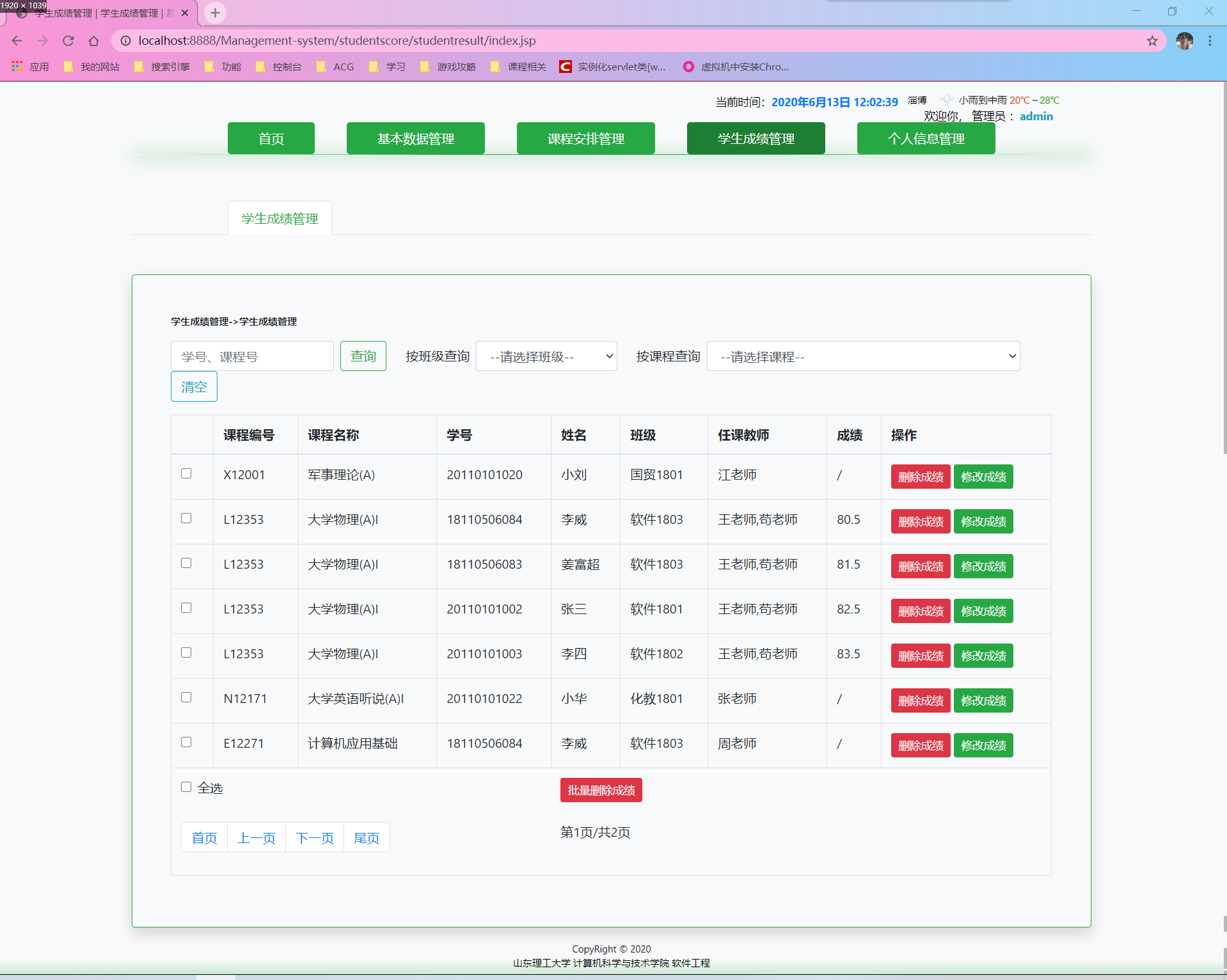


图 17学生成绩管理主页面

点击“删除成绩”不会删除本条选课记录，选课记录只能从学生选课安排模块中删除，删除成绩会把学生成绩值置空，在界面上显示就是“/”这种形式。批量删除成绩就是批量置空学生成绩。

点击单项数据栏后方的修改成绩按钮，会弹出一个成绩输入框，输入学生成绩后点击保存即可发布该学生成绩信息，发布后学生登录系统即可在我的成绩查询模块中查询其成绩。这里点击课程编号为L2353，姓名为李威的数据栏的修改成绩按钮，如图18所示：

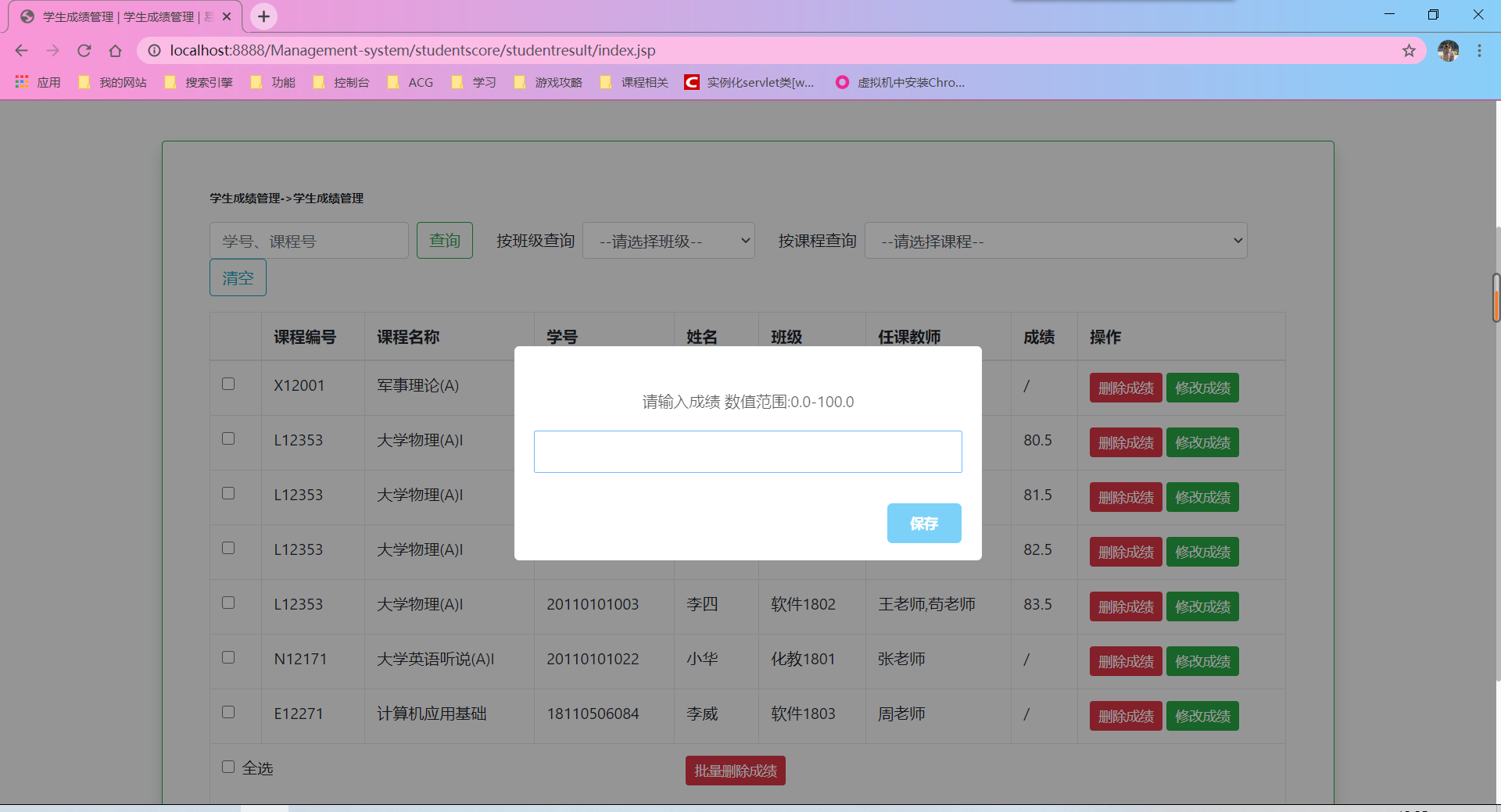


图 18修改成绩按钮点击后弹出成绩输入框

假设该学生的该课程成绩正确数值为90，在成绩输入框里输入90并保存，可以看到学生成绩修改成功。

数据栏上方包括通过学号、课程号模糊查询学生成绩，按班级查询成绩，按课程查询成绩。输入框输入关键词查询与前文功能相似，按班级查询成绩，点击选择一个班级，就会返回本班级里所有的学生信息，按课程查询同理。这里选择班级为软件1803，查询结果如图19所示：

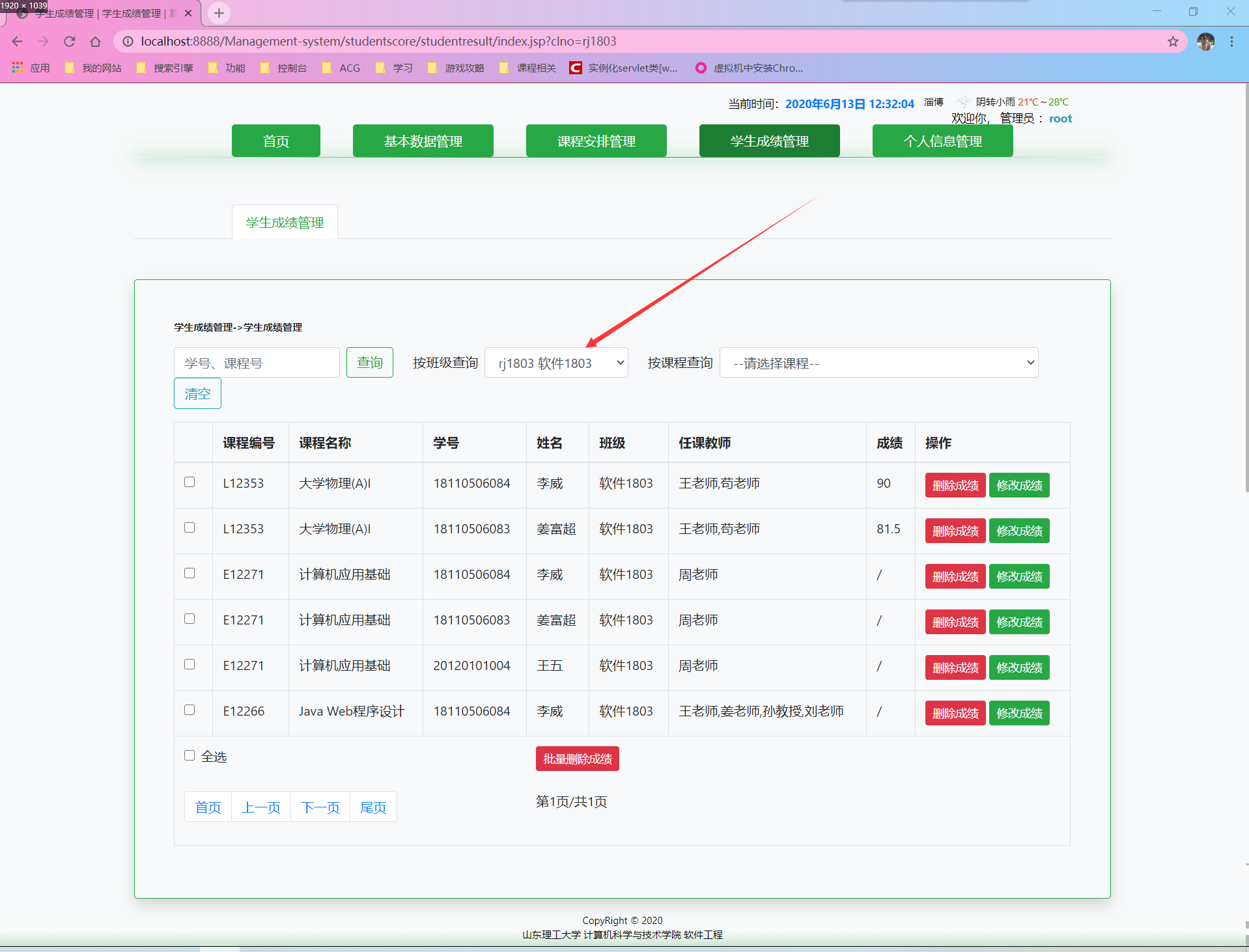


图 19 按班级查询 软件1803 的查询结果

学生成绩管理模块部分JavaScript代码（修改成绩按钮点击事件）如下：

// 修改成绩点击事件

$("body").on("click", ".checkbtn", **function**() {

**var** index = $(".checkbtn").index(**this**);

// 获取选课编号 scid

**var** userItem = $(".user-item");

**var** scid = userItem.eq(index).attr("scid");

swal({

text: '请输入成绩 数值范围:0.0-100.0',

content: "input",

button: {

text: "保存",

closeModal: **false**,

},

})

.then(name => {

**if** (!name) **throw** **null**;

// 发送ajax请求

$.ajax({

type: "post",

url: "../../StudentResultUpdateServlet",

data: {

scid: scid,

grade: name

},

success: **function** (e) {

**if** (e == "success") {

// 修改成功

swal("成功", "修改成绩成功!", "success");

// 延时1s刷新界面

setTimeout(() => {

location.reload();

}, 1000);

} **else** **if** (e == "failed") {

// 修改失败

swal("糟糕", "修改成绩失败!", "error");

} **else** {

// 未知错误

swal("错误", "出现未知错误!" + e, "error");

}

},

error: **function** (e) {

// 网络或服务器错误

swal("错误", "网络堵塞或服务器故障!", "error");

}

});

});

});

学生成绩管理模块中使用到的部分业务逻辑层（Service）技术代码如下：

**public** List<Choose> queryAllGrade() {

List<Choose> chooses = **this**.chooseDao.queryAllGrade();

List<Teaching> teaching = **this**.teachingDao.queryAllGroupTeacher();

// 将授课信息合并

**for** (**int** i = 0; i < chooses.size(); i++) {

Choose sc = chooses.get(i);

**for** (**int** j = 0; j < teaching.size(); j++) {

Teaching tc = teaching.get(j);

**if** (sc.getCno().equals(tc.getCno())) {

sc.setTeacher(tc.getTeacher());

**break**;

}

}

}

**return** chooses;

}

学生成绩管理模块中使用到的部分数据处理层（ChooseDaoImpl）技术代码如下：

**public** List<Choose> queryAllGrade() {

String sql = "SELECT DISTINCT `choose`.`scid`, `choose`.`grade`, `student`.`sid`, `student`.`sno`, `student`.`sname`, `course`.\*, `class`.`cid` AS clid, `class`.`cno` AS clno, `class`.`cname` AS clname " +

"FROM `choose`, `teaching`, `student`, `teacher`, `class`, `course` " +

"WHERE `choose`.`status` = 1 " +

"AND `choose`.`sno` = `student`.`sno` AND `student`.`cno` = `class`.`cno` " +

"AND `choose`.`cno` = `course`.`cno` ";

**return** **this**.selectAll(sql, Choose.**class**);

}

学生成绩管理模块中使用到的部分数据处理层（TeachingDaoImpl）技术代码如下：

**public** List<Teaching> queryAllGroupTeacher() {

StringBuffer sb = **new** StringBuffer();

List<Object> list = **new** ArrayList<>();

sb.append("SELECT `teaching`.`tcid`, `teaching`.`cno`, GROUP\_CONCAT(`teacher`.`tname` SEPARATOR ',') AS teacher ");

sb.append("FROM `teaching`, `teacher` ");

sb.append("WHERE `teacher`.`tno` = `teaching`.`tno` GROUP BY `teaching`.`cno`");

String sql = sb.toString();

**return** **this**.selectAll(sql, Teaching.**class**, list.toArray());

}

## 2.5 基本数据管理模块（学生/教师）的设计与实现

学生使用学号、密码登录系统后，主菜单导航栏有三项，分别是首页、基本数据管理、个人信息管理，基本数据管理模块下有三个功能模块，分别是我的成绩查询、我的选课查询、申请新课程。这里登录学生姓名为李威的学生账号，点击主菜单导航栏的基本数据管理，再点击次级导航栏的我的成绩查询，进入我的成绩查询功能模块。

如图20所示，图中显示学生李威的三条成绩，其中一条已公布，两条未公布：



图 20学生系统 基本数据管理 我的成绩查询页面

点击次级导航栏的我的选课查询，进入我的选课查询功能模块。这里显示了四条数据，三条已通过，一条等待通过。等待通过是等待管理员审核，已通过的可以在成绩查询界面查询到成绩，如图21所示：



图 21学生系统 基本数据管理 我的选课查询页面

点击次级导航栏的申请新课程，进入申请新课程功能模块。这里选择一个课程之后，点击申请显示申请成功即可。申请成功后是等待通过状态，需要管理员审核后通过选课，如图22所示：

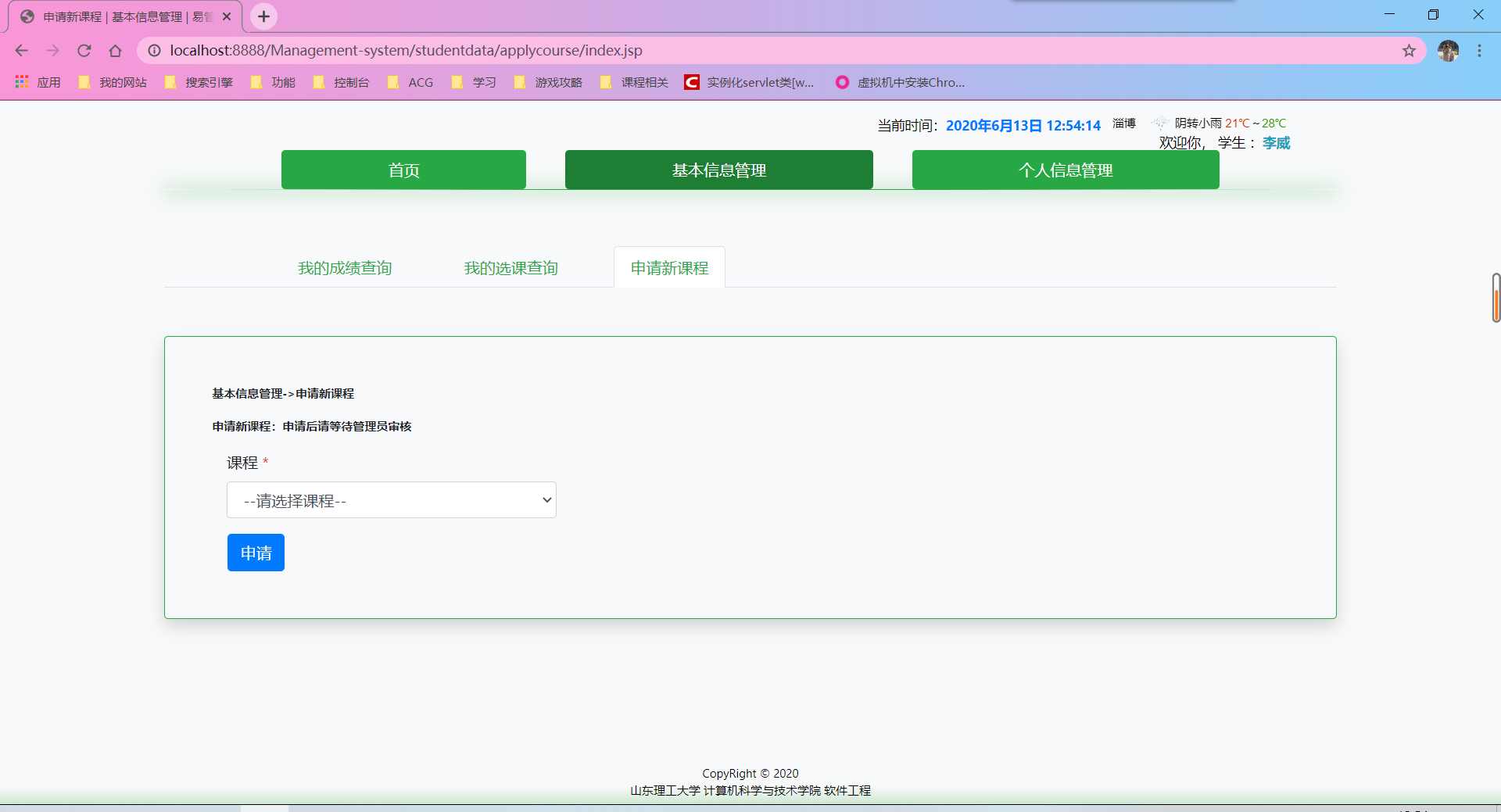


图 22学生系统 基本数据管理 申请新课程页面

教师使用教职工号、密码登录系统后，主菜单导航栏有三项，分别是首页、基本数据管理、个人信息管理，基本数据管理模块下有三个功能模块，分别是学生成绩管理、我的授课查询、申请新授课。其中我的授课查询、申请新授课与学生系统我的选课查询，申请新课程功能、操作、界面类似，这里不再赘述。

教师系统的学生成绩管理界面如图23所示：

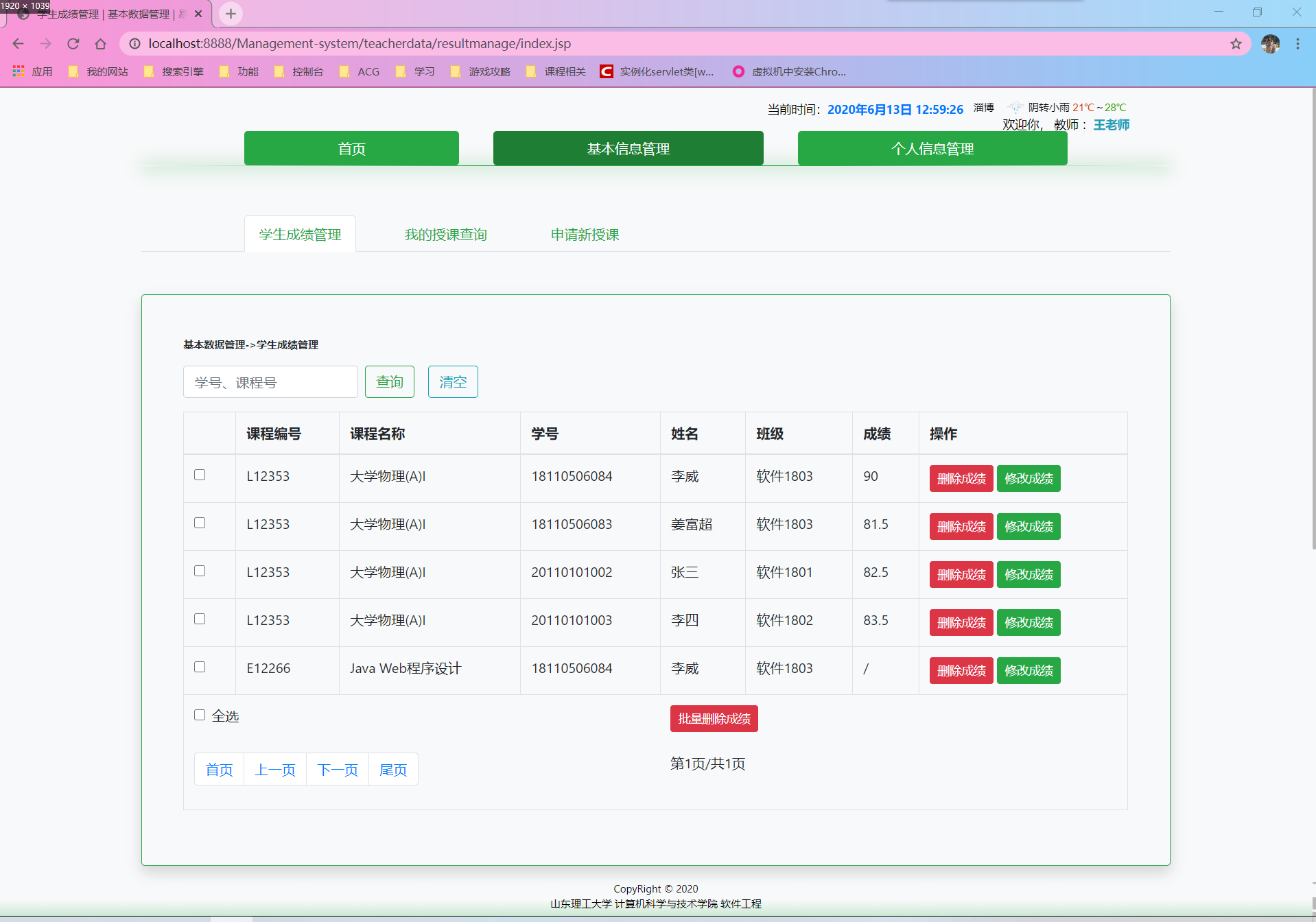


图 23教师系统 基本数据管理 学生成绩管理页面

教师系统的学生成绩管理功能模块与管理员/高级管理员系统的学生成绩管理功能类似，不同的是教师系统中只能查询到自己所授课科目的学生成绩信息。

学生身份和教师身份都可以对个人信息进行维护，不同的是学生可修改范围仅限个人信息，班级和学号不可修改，教师可修改范围仅限个人信息，教职工号不可修改。

所有身份登录系统，在个人信息管理模块都有退出登录功能，点击退出登录后即可退出系统，退出登录提示成功后跳转到登录界面，如图24：

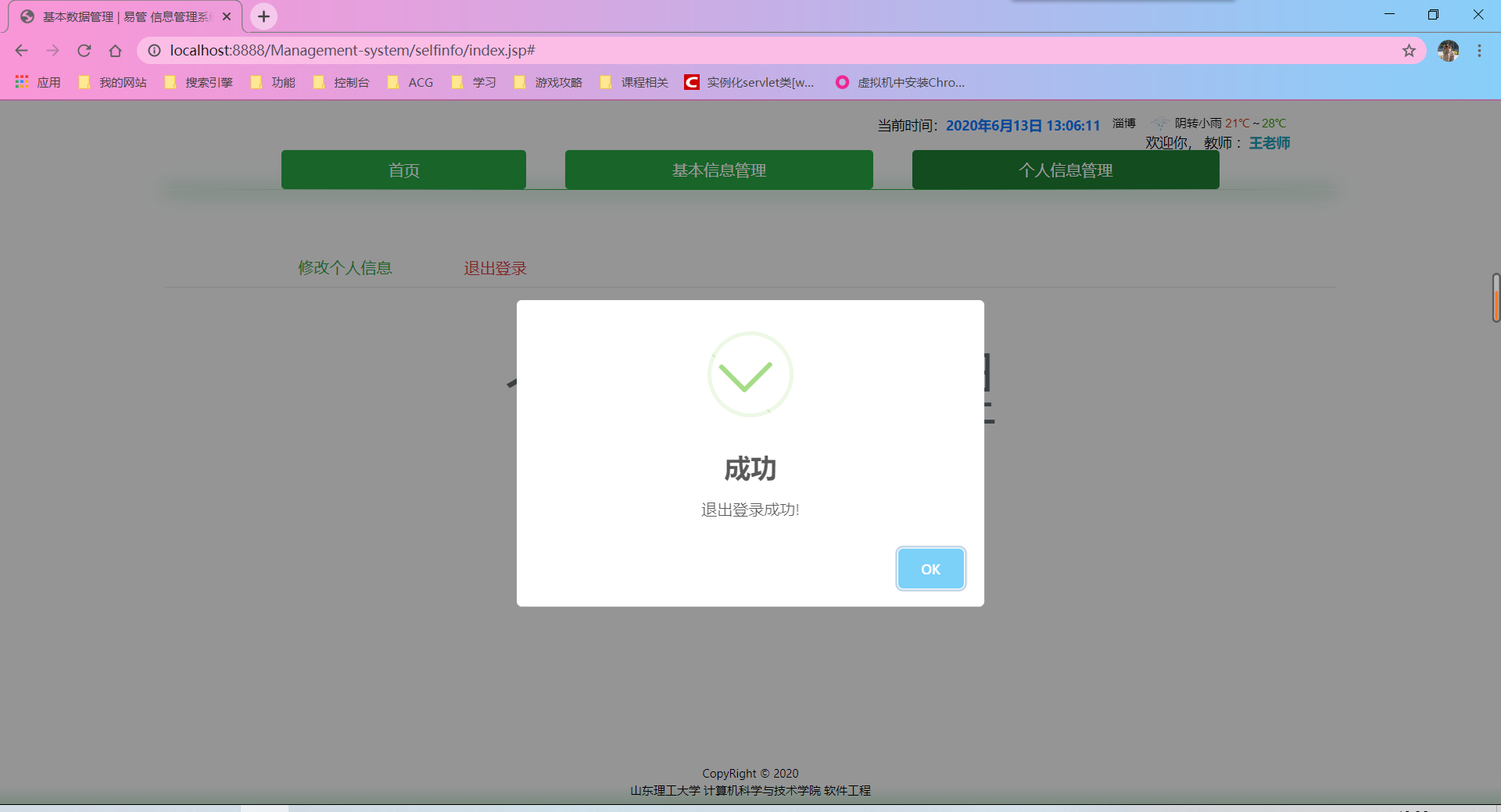


图 24 退出登录按钮点击后 系统弹出框 随后跳转到登录界面

退出登录Servlet实现部分代码：

**protected** **void** doGet(HttpServletRequest request, HttpServletResponse response)

**throws** ServletException, IOException {

response.setContentType("text/html;charset=UTF-8");

PrintWriter out = response.getWriter();

// 将Session 对象中的username 对象移除

request.getSession().removeAttribute("username");

// 清除本次使用的Cookie

Cookie cookie = **new** Cookie("identity", **null**);

cookie.setMaxAge(0);

Cookie aid = **new** Cookie("aid", **null**);

aid.setMaxAge(0);

Cookie uid = **new** Cookie("uid", **null**);

uid.setMaxAge(0);

Cookie sid = **new** Cookie("sid", **null**);

sid.setMaxAge(0);

Cookie tid = **new** Cookie("tid", **null**);

tid.setMaxAge(0);

response.addCookie(cookie);

response.addCookie(aid);

response.addCookie(uid);

response.addCookie(sid);

response.addCookie(tid);

out.print("success");

}

过滤器Filter实现部分代码（用于判断登录状态）：

**public** **void** doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain filterChain)

**throws** IOException, ServletException {

// 获得在下面代码中要用的request,response,session对象

HttpServletRequest httpRequest = (HttpServletRequest) servletRequest;

HttpServletResponse httpResponse = (HttpServletResponse) servletResponse;

// 获取请求URL

String servletPath = httpRequest.getServletPath();

// 检测1中获取的servletPath是否为不需要检测的URl中的一个.若是,放行

List<String> urls = Arrays.*asList*(uncheckedUrls.split(","));

**if** (urls.contains(servletPath)) {

filterChain.doFilter(httpRequest, httpResponse);

**return**;

}

// 从session中获取SessionKey对应值,若值不存在,则重定向到redirectUrl

Object user = httpRequest.getSession().getAttribute("username");

**if** ((user == **null**)) {

httpResponse.sendRedirect(httpRequest.getContextPath() + redirectUrl);

**return**;

}

// 若存在,则放行

filterChain.doFilter(httpRequest, httpResponse);

}

头部jsp框架部分代码（head.jsp用于被其他界面引用）：

<!-- 头部信息栏 -->

<div class=*"headinfo text-body"*>

<p>

当前时间：<span class=*"nowtime text-primary mr-2"*>XX-XX-XX xx:xx:xx</span>

<iframe frameborder=*"0"* scrolling=*"no"* hspace=*"0"* width=*"230"* height=*"20"* src=*"https://i.tianqi.com/?c=code&a=getcode&id=26&h=60&w=610"*></iframe>

</p>

<c:if test=*"*${cookie['identity'].value **ne** 'admin'}*"*>

<p class=*"mr-5"*>欢迎你，

<c:if test=*"*${cookie['identity'].value **eq** 'user'}*"*>管理员</c:if>

<c:if test=*"*${cookie['identity'].value **eq** 'student'}*"*>学生</c:if>

<c:if test=*"*${cookie['identity'].value **eq** 'teacher'}*"*>教师</c:if>

：<span class=*"username text-info"*>XXXX</span></p>

</c:if>

</div>

<!-- 头部导航栏 -->

<nav class=*"head-nav nav nav-pills nav-fill bg-light"*>

<a class=*"nav-item nav-link btn-success* <%=array[0] %>*"* href=*"/*<%=project %>*/index.jsp"*>首页</a>

<c:if test=*"*${(cookie['identity'].value **eq** 'user') || (cookie['identity'].value **eq** 'admin')}*"*>

<a class=*"nav-item nav-link btn-success* <%=array[1] %>*"* href=*"/*<%=project %>*/basedata/index.jsp"*>基本数据管理</a>

<a class=*"nav-item nav-link btn-success* <%=array[2] %>*"* href=*"/*<%=project %>*/coursearrange/index.jsp"*>课程安排管理</a>

<a class=*"nav-item nav-link btn-success* <%=array[3] %>*"* href=*"/*<%=project %>*/studentscore/index.jsp"*>学生成绩管理</a>

<a class=*"nav-item nav-link btn-success* <%=array[4] %>*"* href=*"/*<%=project %>*/selfinfo/index.jsp"*>个人信息管理</a>

</c:if>

<c:if test=*"*${cookie['identity'].value **eq** 'student'}*"*>

<a class=*"nav-item nav-link btn-success* <%=array[1] %>*"* href=*"/*<%=project %>*/studentdata/index.jsp"*>基本信息管理</a>

<a class=*"nav-item nav-link btn-success* <%=array[4] %>*"* href=*"/*<%=project %>*/selfinfo/index.jsp"*>个人信息管理</a>

</c:if>

<c:if test=*"*${cookie['identity'].value **eq** 'teacher'}*"*>

<a class=*"nav-item nav-link btn-success* <%=array[1] %>*"* href=*"/*<%=project %>*/teacherdata/index.jsp"*>基本信息管理</a>

<a class=*"nav-item nav-link btn-success* <%=array[4] %>*"* href=*"/*<%=project %>*/selfinfo/index.jsp"*>个人信息管理</a>

</c:if>

</nav>