|  |  |
| --- | --- |
| **项目编号** |  |
| **文档编号** |  |
| **密级** |  |

**作业管理系统详细设计**

**广西民族大学**

日期： 2019年6月23日

目录

[1．导言 3](#_Toc12952381)

[1.1 目的 3](#_Toc12952382)

[1.2 范围 4](#_Toc12952383)

[1.3 引用标准 4](#_Toc12952384)

[1.4 参考资料 4](#_Toc12952385)

[1.5 版本更新信息 4](#_Toc12952386)

[2．详细设计简述 5](#_Toc12952387)

[2.1 设计简介 5](#_Toc12952388)

[2.2 模块简介 6](#_Toc12952389)

[3、界面详细设计 6](#_Toc12952390)

[3.1、动态页面详细设计 6](#_Toc12952391)

[4、客户端模块详细设计 7](#_Toc12952392)

[4.1公用模块的详细设计 7](#_Toc12952393)

[4.1.1 表现层 8](#_Toc12952394)

[4.2、登录/注册模块 9](#_Toc12952395)

[4.2.1 登陆模块 9](#_Toc12952396)

[4.2.2 注册模块 32](#_Toc12952397)

[4.3、上传/下载模块 56](#_Toc12952398)

[4.3.1 表现层 57](#_Toc12952399)

[4.3.2 控制层 58](#_Toc12952400)

[4.3.3 业务逻辑层 64](#_Toc12952401)

[4.3.4 数据持久层 70](#_Toc12952402)

[4.3.5 域模型层 71](#_Toc12952403)

[4.4、修改/布置模块 72](#_Toc12952404)

[4.4.1 表现层 72](#_Toc12952405)

[4.4.2 控制层 73](#_Toc12952406)

[4.4.3 业务逻辑层 77](#_Toc12952407)

[4.4.4 数据持久层 79](#_Toc12952408)

[4.4.5 域模型层 81](#_Toc12952409)

[5、管理端模块详细设计 82](#_Toc12952410)

[5.1登陆模块 82](#_Toc12952411)

[5.2管理信息模块 94](#_Toc12952412)

[5.2.1 表现层 94](#_Toc12952413)

[5.2.2 控制层 96](#_Toc12952414)

[5.2.3 业务逻辑层 98](#_Toc12952415)

[5.2.4 数据持久层 100](#_Toc12952416)

[5.2.5 域模型层 101](#_Toc12952417)

[6、公共部分模块详细设计 103](#_Toc12952418)

[6.1 公共页面 103](#_Toc12952419)

# 1．导言

## 1.1 目的

该文档是对作业批改系统概要设计进行的详细说明，是为了指导和规范该系统开发而制定的详细开发设计文档，对该系统的结构体系以及模块内容进行详细说明。

本文档的预期读者包括：

* 设计开发人员
* 项目管理人员
* 测试人员
* 用户

## 1.2 范围

该文档内容涵盖作业批改系统的所有功能模块的详细设计描述，具体功能模块为登录模块、上传下载模块、数据管理模块。

## 1.3 引用标准

[1] 《软件工程案例教程 第2版》 韩万江等 机械工业出版社

## 1.4 参考资料

[1] 《软件项目管理案例教程 第3版》 韩万江等 机械工业出版社

## 1.5 版本更新信息

本文档的更新记录如下表1-0所示。

表1-1 文档更新记录

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 修改编号 | 修改日期 | 修改后版本 | 修改位置 | 修改内容概述 |
| 001 | 2019.6.24 | 0.1 | 全部 | 初始发布版本 |
| 002 | 2019.6.25 | 1.0 | 4章节 | 增加 |
| 003 | 2019.6.26 | 2.0 | 4章节 | 修改 |
| 004 | 2019.6.27 | 3.0 | 4、6章节 | 修改 |
| 005 | 2019.6.28 | 4.0 | 6、7章节 | 修改、增加 |
| 006 | 2019.7.1 | 5.0 | 5、7章节 | 修改、增加 |
| 007 | 2019.7.2 | 6.0 | 4、6章节 | 修改 |
| 008 | 2019.7.3 | 7.0 | 5、7章节 | 修改 |
| 009 | 2019.7.4 | 8．0 | 全部 | 修改，增加 |

# 2．详细设计简述

## 2.1 设计简介

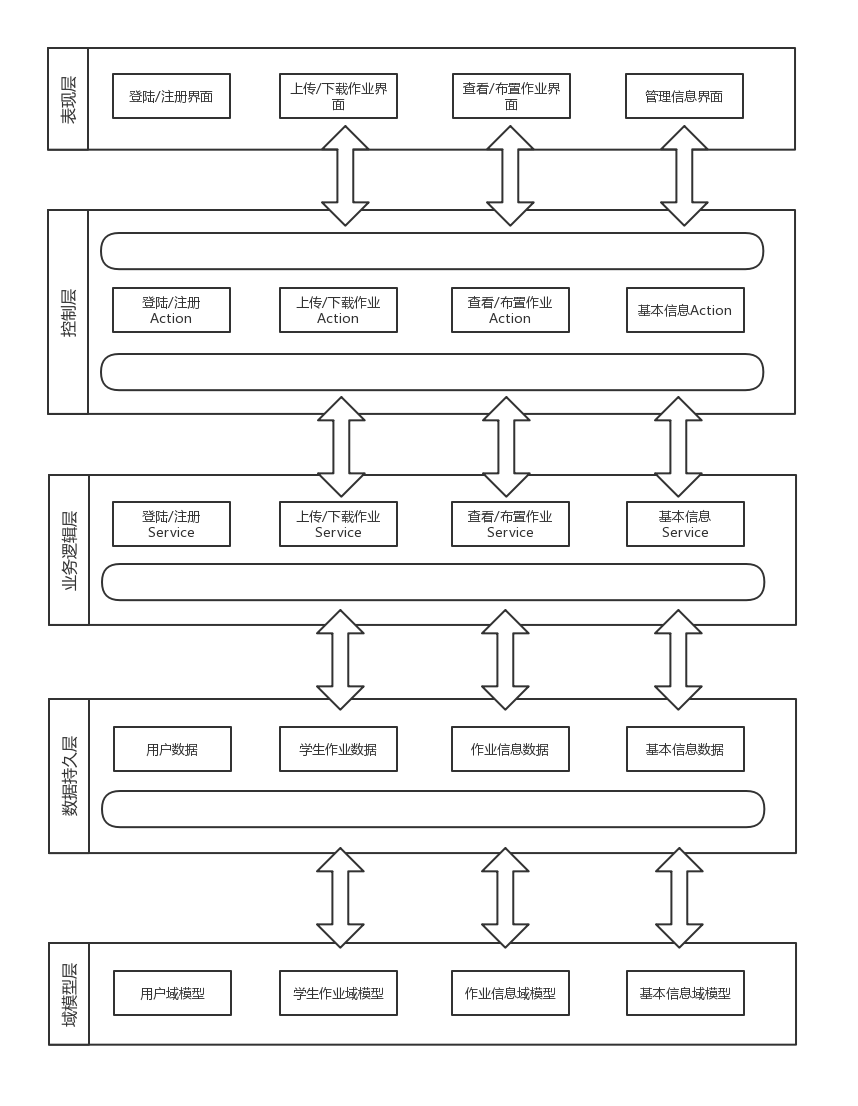
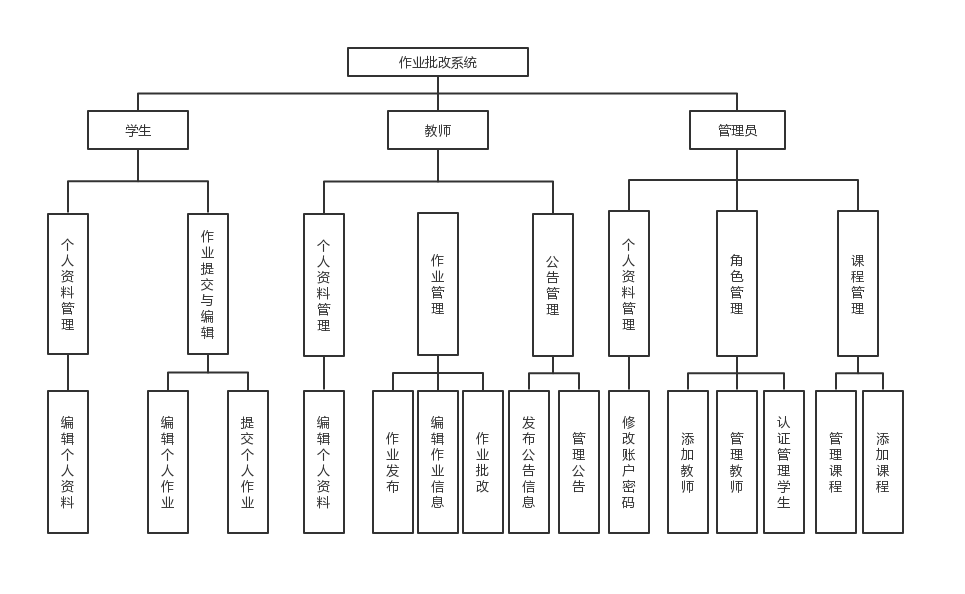
作业批改系统分层结构图如图2-1所示。

图 2-1作业批改系统示意图

**2.2 模块简介**

系统功能模块图如图2-2所示。



**3、界面详细设计**

作业批改系统内界面主要使用JSP技术进行设计编写。界面主要划分为页面上方的导航、左侧的管理功能选择以及主要操作区。

导航：通过选择可跳转至公告、首页、学院、班级、活动相关页面。

管理功能选择：选择用户权限内具体管理的数据。

主要操作区：显示该功能下的数据并进行相关编辑。

**3.1、动态页面详细设计**

由于本系统所有jsp\htm页面中均涉及Java语言以及数据库的相关使用，所以所有页面均属于动态页面。

作业批改系统动态页面见表1-1

表3-1 页面表现层设计

|  |  |  |
| --- | --- | --- |
| 界面 | JSP | 功能描述 |
| 用户登录界面 | main/index.jsp | 为用户展示系统登录界面，选择登录账户类型，提供注册和登录功能。 |
| 管理员登录页面 | Admin/index.jsp | 为管理员提供登录入口，并且为管理员登录入口提供保护功能。 |
| 用户注册界面 | Student/index.htm  Teacher/index.htm | 用户只有通过注册才能进入系统，通过选择身份并填写相关信息提交后由管理员进行审核通过才可进入系统。 |
| 系统用户功能界面 | Teacher/Left.htm  Student/left.htm | 系统界面左侧列出当前用户可进行编辑的操作 |
| 用户系统界面 | Admin/login.htm  Student/login.htm  Teacher/login.htm | 提示当前页面为该用户身份的提示页面 |
| 管理员管理信息界面 | Admin/top.htm | 可选择学生信息、教师信息、作业信息进行后台管理 |

# 4、客户端模块详细设计

客户端模块主要包括登陆/注册、上传/下载、修改/布置模块。

## 4.1公用模块的详细设计

[公用模块是指在系统在运行过程中将会被其他模块调用的基础资源]。

### 4.1.1 表现层

公用模块的表现层4-1-1

表 4-1-1

|  |  |  |
| --- | --- | --- |
| 界面 | JSP | 功能描述 |
| 登陆进入界面顶端 | top.html | 登陆的进入界面的顶端导航栏 |
| 登录出错页面 | Admin/Admin.jsp | 登入时出现超时时跳出弹窗进行提示 |
| 长时间未操作提示 | teacher/login.jsp  student/login.jsp | 在长时间未操作时弹出提示。 |

<div class="os-toolbar">

<div class="container row center-block">

<ul class="pull-left left-menu clearfix" id="nav-left-menu">

<li>

<a href="#" title="公告">公告</a>

</li>

<li>

<a href="#" title="首页">首页</a>

</li>

<li>

<a href="#" title="学院">学院</a>

</li>

<li>

<a href="#" title="班级">班级</a>

</li>

<li>

<a href="#" title="班级">活动</a>

</li>

</ul>

</div>

</div>

**4.2、登录/注册模块**

### 4.2.1 登陆模块

在登录模块时，系统内部的相应响应操作示意图如图4-2-1所示。

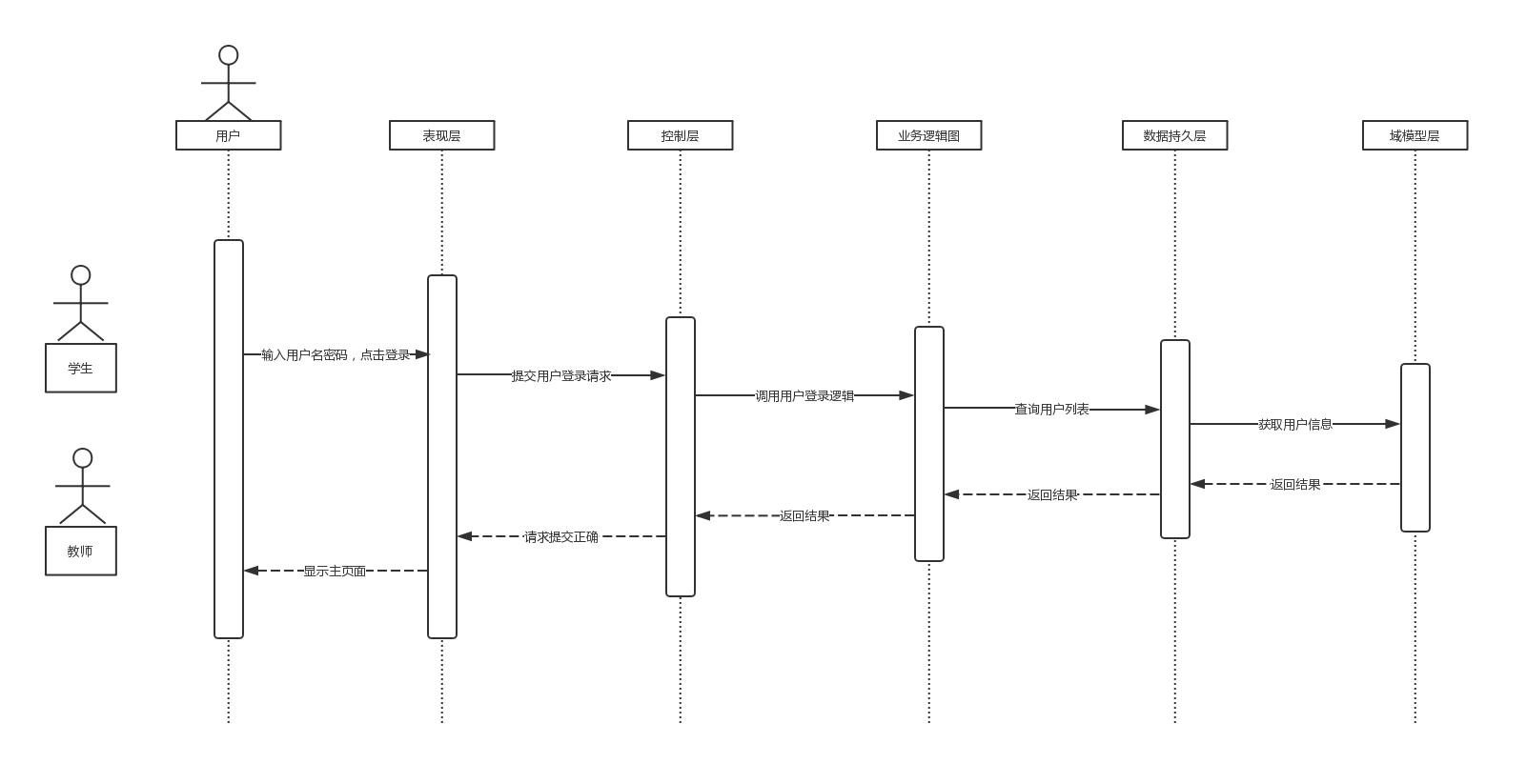


图 4-2-1-1 系统响应示意图

#### 4.2.1.1 表现层

登录模块的表现层主要完成不同用户的登录功能，登录页面时要求用户输入账号密码的基本信息，确认后页面给出响应消息，提示登录成功或失败的提示。表现层对应的JSP页面列表见表4-2-1-2所示。

表 4-2-1-2 登录模块表现层JSP列表

|  |  |  |
| --- | --- | --- |
| 界面 | JSP | 功能描述 |
| 登录页面 | main/index.jsp | 用户（教师、学生、管理员）登入功能，当登入出错时给出提示。 |

main/index.jsp的流程图如图4-2-1-3所示。

进入游首页

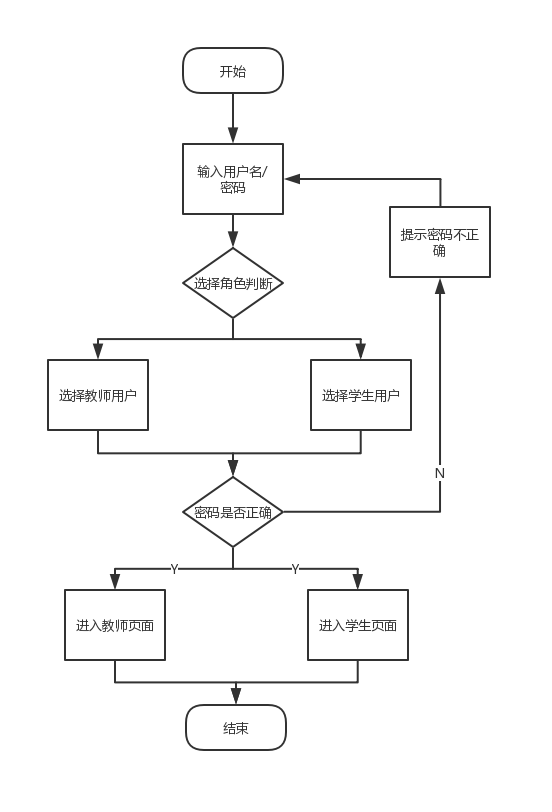


图 4-2-1-3 登录模块详细设计流程图

#### 4.2.1.2 控制层

登录模块的控制层负责接受来自main/index.jsp的用户输入，同时调用登入模块的业务逻辑接口，将用户名与密码等用户关键信息传递到业务逻辑层进行判定。等到业务逻辑处理完成之后，将来自业务逻辑层的相应信息传到表现层，并决定显示页面。登录模块控制层列表见表4-2-1-4所示。

表 4-2-1-4 登录模块控制层列表

|  |  |  |  |
| --- | --- | --- | --- |
| 事件 | Action | 转移说明 | 出口 |
| 登录 | servletMain.java  login.java | SUCCESS | teacher/adver.jsp  student/adver.jsp  admin/admin.jsp  —登入成功，显示提示窗口。 |
| ERROR | servletMain.jsp  login.jsp  ---登入失败，显示提示窗口。 |

在控制层中servletMain.java的描述如下所示：

public class servletMain extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doPost(request,response);

}

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

request.setCharacterEncoding("utf-8");

//Servlet接收

PrintWriter out = response.getWriter();

HttpSession session=request.getSession();

String strUserName=(String)request.getParameter("username");

String strPassword=(String)request.getParameter("password");

String strSelect=(String)request.getParameter("select");

//JavaBean处理

Data data=new Data();

Show show=new Show();

int intT=0;

if(strSelect!=null&&strSelect.equals("student")){

intT=data.getRowCount("tb\_student WHERE name='"+strUserName+"' AND password='"+strPassword+"'");

if(intT>0){

session.setAttribute("name",strUserName);

session.setAttribute("use","student");

response.sendRedirect("student/login.jsp");

}else{

out.print(show.errorBox("检查你的用户名或密码！","验证错误"));

return;

}

}else if(strSelect!=null&&strSelect.equals("teacher")){

intT=data.getRowCount("tb\_teacher WHERE name='"+strUserName+"' AND password='"+strPassword+"'");

if(intT>0){

session.setAttribute("name",strUserName);

session.setAttribute("use","teacher");

response.sendRedirect("teacher/login.jsp");

}else{

out.print(show.errorBox("检查你的用户名或密码！","验证错误"));

return;

}

}else{

out.print(show.errorBox("检查你的用户名或密码！","验证错误"));

return;

}

}

}

Login.java

public class login extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doPost(request,response);

}

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

//Servlet接收

PrintWriter out=response.getWriter();

HttpSession session=request.getSession();

String strName=(String)request.getParameter("name");

String strPassword=(String)request.getParameter("password");

//JavaBean处理

Data data=new Data();

Show show=new Show();

int intT=0;

if(strName!=null||strPassword!=null){

intT=data.getRowCount("tb\_admin WHERE name='"+strName+"' AND password='"+strPassword+"'");

if(intT>0){

session.setAttribute("admin",strName);

response.sendRedirect("admin/admin.jsp");

}else{

out.print(show.errorBox("请检查你的用户名密码！","验证信息"));

return;

}

}else{

out.print(show.errorBox("请检查你的用户名密码！","验证信息"));

return;

}

}

}

#### 4.2.1.3 业务逻辑层

登录模块的业务逻辑层主要完成对用户登录逻辑的判定，Servlet响应单选按钮，判断进入不同用户验证。数据库查询，与数据库存入的信息比对，验证身份。页面跳转，响应重定向。比如：登录模块业务逻辑层列表如图4-2-1-5所示。

表 4-2-1-5登陆模块业务逻辑层列表

|  |  |  |  |
| --- | --- | --- | --- |
| 事件 | Service | 调用说明 | 出口 |
| 登录 | servletMain.java  login.java | 调用JavaBean将数据插入数据库 | 返回给Action |

servletMain.java方法：

public class servletMain extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doPost(request,response);

}

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

request.setCharacterEncoding("utf-8");

//Servlet接收

PrintWriter out = response.getWriter();

HttpSession session=request.getSession();

String strUserName=(String)request.getParameter("username");

String strPassword=(String)request.getParameter("password");

String strSelect=(String)request.getParameter("select");

//JavaBean处理

Data data=new Data();

Show show=new Show();

int intT=0;

if(strSelect!=null&&strSelect.equals("student")){

intT=data.getRowCount("tb\_student WHERE name='"+strUserName+"' AND password='"+strPassword+"'");

if(intT>0){

session.setAttribute("name",strUserName);

session.setAttribute("use","student");

response.sendRedirect("student/login.jsp");

}else{

out.print(show.errorBox("检查你的用户名或密码！","验证错误"));

return;

}

}else if(strSelect!=null&&strSelect.equals("teacher")){

intT=data.getRowCount("tb\_teacher WHERE name='"+strUserName+"' AND password='"+strPassword+"'");

if(intT>0){

session.setAttribute("name",strUserName);

session.setAttribute("use","teacher");

response.sendRedirect("teacher/login.jsp");

}else{

out.print(show.errorBox("检查你的用户名或密码！","验证错误"));

return;

}

}else{

out.print(show.errorBox("检查你的用户名或密码！","验证错误"));

return;

}

}

}

Servlet响应单选按钮，判断进入不同用户验证。数据库查询，与数据库存入的信息比对，验证身份。页面跳转，响应重定向。

login.java主要实现方法:

public class login extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doPost(request,response);

}

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

//Servlet接收

PrintWriter out=response.getWriter();

HttpSession session=request.getSession();

String strName=(String)request.getParameter("name");

String strPassword=(String)request.getParameter("password");

//JavaBean处理

Data data=new Data();

Show show=new Show();

int intT=0;

if(strName!=null||strPassword!=null){

intT=data.getRowCount("tb\_admin WHERE name='"+strName+"' AND password='"+strPassword+"'");

if(intT>0){

session.setAttribute("admin",strName);

response.sendRedirect("admin/admin.jsp");

}else{

out.print(show.errorBox("请检查你的用户名密码！","验证信息"));

return;

}

}else{

out.print(show.errorBox("请检查你的用户名密码！","验证信息"));

return;

}

}

}

#### 4.2.1.4 数据持久层

登录模块的是数据持久层能对用户数据进行增删改查。登入模块数据持久层列表见表4-2-1-6所示。

表 4-2-1-6 登入模块数据持久层列表

|  |  |  |  |
| --- | --- | --- | --- |
| 事件 | DAO | 调用数据模型 | 说明 |
| 登录 | studentMreg.java  teacherMreg.java  userReg.java | 调用 Scourse.java  Student.java  Teacher.java | 对用户信息进行增删改查操作 |

超链接，点击进入相应处理页面(a)。修改注册信息，未填完整，提示信息未填完整，密码与确认密码不一致，提示确认密码输入错误，请重新输入，正确填写，则会提示修改成功（JavaScript）。提交页面交给Servlet处理，在其中调用JavaBean将数据插入数据库，并执行判断，成功返回主页面。

studentMreg.java

public class studentMreg extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doPost(request,response);

}

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

request.setCharacterEncoding("utf-8");

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

PrintWriter out = response.getWriter();

//接收数据

HttpSession session=request.getSession();

String strUserName=(String)session.getAttribute("name");

String specialty=(String)request.getParameter("specialty");

String classNum=(String)request.getParameter("classNum");

String strPassword=(String)request.getParameter("password");

String strrPassword=(String)request.getParameter("rpassword");

String strEmail=(String)request.getParameter("email");

//处理数据

Show show=new Show();

Str str=new Str();

Student student=new Student();

student.setSpecialty(specialty);

student.setClass\_num(classNum);

student.setPassword(str.dbEncode(strPassword));

student.setEmail(str.dbEncode(strEmail));

if(!strrPassword.equals(strPassword)) {

out.print(show.errorBox("新的密码与确认密码不符!","错误信息"));

}else {

if(student.getSpecialty().equals("")||student.getClass\_num().equals("")||student.getPassword().equals("")||student.getEmail().equals("")){

out.print(show.errorBox("请添写完整信息!","错误信息"));

return;

}else{

int intT=0;

Data data=new Data();

intT=data.insert("UPDATE tb\_student SET password='"+student.getPassword()+"',email='"+student.getEmail()+"',specialty='"+student.getSpecialty()+"',classnum='"+student.getClass\_num()+"' WHERE name='"+strUserName+"'");

if(intT<=0){

out.print(show.errorBox("修改失败，数据库错误!","错误信息"));

return;

}else{

out.print("<script>alert('修改成功！');document.location='student/reginfo.jsp';</script>");

}

}

}

}

}

teacherMreg.java

public class studentMreg extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doPost(request,response);

}

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

request.setCharacterEncoding("utf-8");

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

PrintWriter out = response.getWriter();

//接收数据

HttpSession session=request.getSession();

String strUserName=(String)session.getAttribute("name");

String specialty=(String)request.getParameter("specialty");

String classNum=(String)request.getParameter("classNum");

String strPassword=(String)request.getParameter("password");

String strrPassword=(String)request.getParameter("rpassword");

String strEmail=(String)request.getParameter("email");

//处理数据

Show show=new Show();

Str str=new Str();

Student student=new Student();

student.setSpecialty(specialty);

student.setClass\_num(classNum);

student.setPassword(str.dbEncode(strPassword));

student.setEmail(str.dbEncode(strEmail));

if(!strrPassword.equals(strPassword)) {

out.print(show.errorBox("新的密码与确认密码不符!","错误信息"));

}else {

if(student.getSpecialty().equals("")||student.getClass\_num().equals("")||student.getPassword().equals("")||student.getEmail().equals("")){

out.print(show.errorBox("请添写完整信息!","错误信息"));

return;

}else{

int intT=0;

Data data=new Data();

intT=data.insert("UPDATE tb\_student SET password='"+student.getPassword()+"',email='"+student.getEmail()+"',specialty='"+student.getSpecialty()+"',classnum='"+student.getClass\_num()+"' WHERE name='"+strUserName+"'");

if(intT<=0){

out.print(show.errorBox("修改失败，数据库错误!","错误信息"));

return;

}else{

out.print("<script>alert('修改成功！');document.location='student/reginfo.jsp';</script>");

}

}

}

}

}

userReg.java

public class userReg extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doPost(request,response);

}

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

request.setCharacterEncoding("utf-8");

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

String strSelect=(String)request.getParameter("select");

if(strSelect!=null&&strSelect.equals("teacher")){

PrintWriter out = response.getWriter();

//收集数据

String strName=(String)request.getParameter("name");

String strEmail=(String)request.getParameter("email");

String strPassword=(String)request.getParameter("password");

//处理数据

Str str=new Str();

Show show=new Show();

Teacher teacher=new Teacher();

teacher.setEmail(str.dbEncode(str.toChinese(strEmail)));

teacher.setName(str.dbEncode(str.toChinese(strName)));

teacher.setPassword(str.dbEncode(str.toChinese(strPassword)));

if(teacher.getName().equals("")||teacher.getPassword().equals("")||teacher.getEmail().equals("")){

out.print(show.errorBox("请添写完整信息！","错误信息"));

return;

}else{

int intT=0;

Data data=new Data();

intT=data.getRowCount("tb\_teacher WHERE name='"+teacher.getName()+"'");

if(intT>0){

out.print(show.errorBox("该用户名已被注册！","注册信息"));

return;

}else{

intT=data.insert("INSERT INTO tb\_teacher(password,name,email) VALUES('"+teacher.getPassword()+"','"+teacher.getName()+"','"+teacher.getEmail()+"')");

if(intT<=0){

out.print(show.errorBox("注册失败，数据库错误!","错误信息"));

return;

}else{

out.print("<script>alert('注册成功！')</script>");

response.sendRedirect("main/index.jsp");

}

}

}

}else {

request.setCharacterEncoding("utf-8");

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

PrintWriter out = response.getWriter();

//接收数据

String strEmail=(String)request.getParameter("email");

String strName=(String)request.getParameter("name");

String strPassword=(String)request.getParameter("password");

//处理数据

Str str=new Str();

Show show=new Show();

Student student=new Student();

student.setEmail(strEmail);

student.setName(strName);

student.setPassword(strPassword);

if(student.getName().equals("")||student.getPassword().equals("")||student.getEmail().equals("")){

out.print(show.errorBox("请添写完整信息！","错误信息"));

return;

}else{

int intT=0;

Data data=new Data();

intT=data.getRowCount("tb\_student WHERE name='"+student.getName()+"'");

if(intT>0){

out.print(show.errorBox("该用户名已被注册！","注册信息"));

return;

}else{

intT=data.insert("INSERT INTO tb\_student(password,name,email) VALUES('"+student.getPassword()+"','"+student.getName()+"','"+student.getEmail()+"')");

if(intT<=0){

out.print(show.errorBox("注册失败，数据库错误!","错误信息"));

return;

}else{

out.print("<script>alert('注册成功！');document.location='main/index.jsp';</script>");

}

}

}

}

}

}

#### 4.2.1.5 域模型层

登录模块用到域模型层中User.java，User.java是一个公用域模型，在涉及到用户信息查询等操作时，就会调用到该模型，登录模块域模型层列表如表4-2-1-7所示。

表 4-2-1-7登入模块域模型层列表

|  |  |
| --- | --- |
| 域模型 | 描述 |
| Teacher.java  Student.java | 对用户信息的增、改、查操作 |

Teacher.java主要属性与方法：

public class Teacher {

private String tname;

private String password;

private String name;

private String email;

private String specialty;

private String cname;

Student.java主要属性与方法：

public class Student {

private String name;

private String password;

private String email;

private String specialty;

private String class\_num;

**4.2.2 注册模块**

在注册模块时，系统内部的相应响应操作示意图如图4-2-2-1所示。

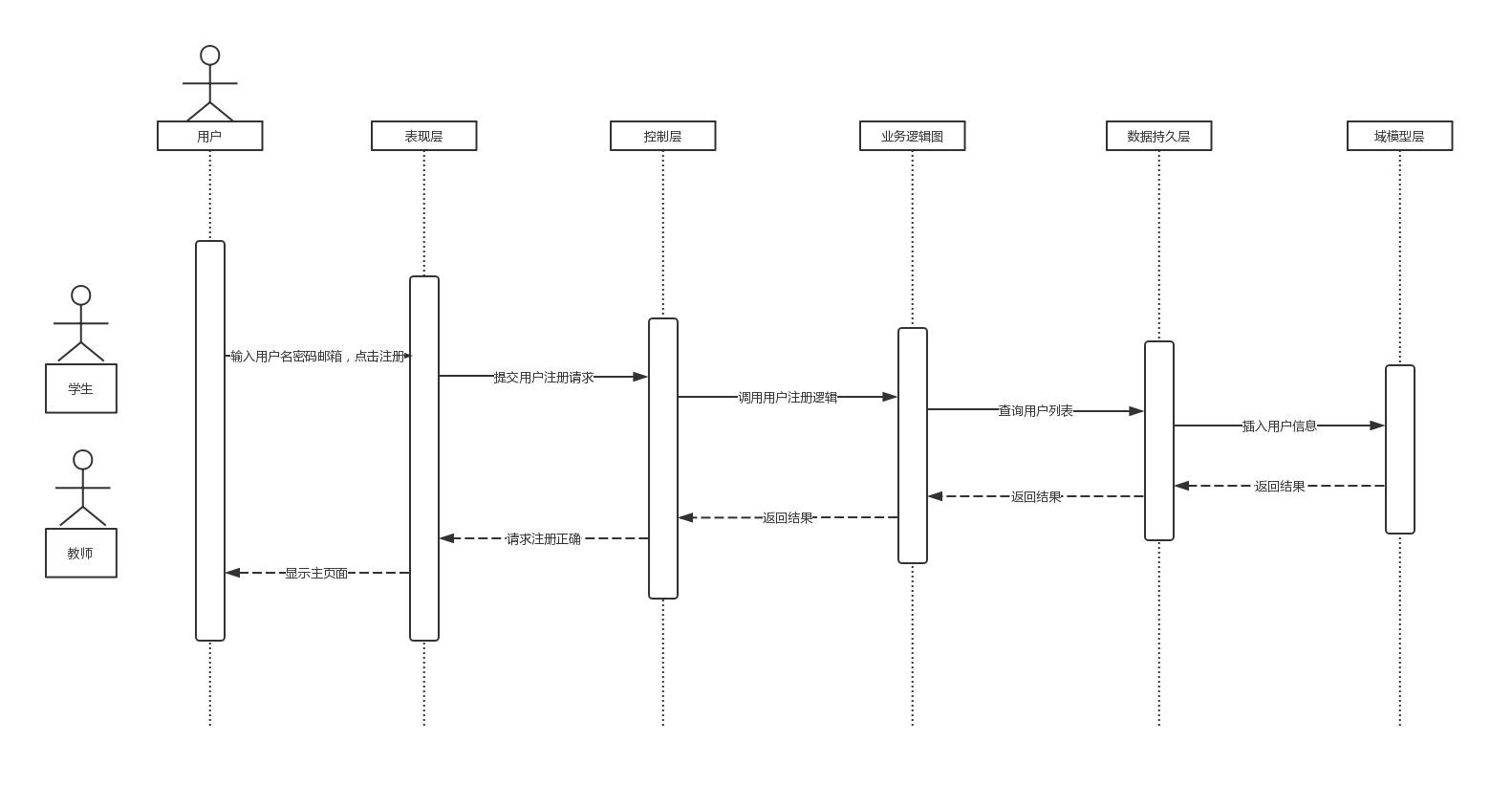


图 4-2-2-1 系统响应示意图

**4.2.2.1 表现层**

注册模块的表现层主要完成不同用户的注册功能，注册页面时要求用户输入账号密码等基本信息，确认后页面给出响应消息，提示注册成功或失败的提示。表现层对应的JSP页面列表见表4-2-2-2所示。

表 4-2-2-2 注册模块表现层JSP列表

|  |  |  |
| --- | --- | --- |
| 界面 | JSP | 功能描述 |
| 注册页面 | main/index.jsp | 用户（教师、学生、管理员）注册功能，当注册出错时给出提示。 |

main/index.jsp的流程图如图4-2-2-3所示。

进入游首页

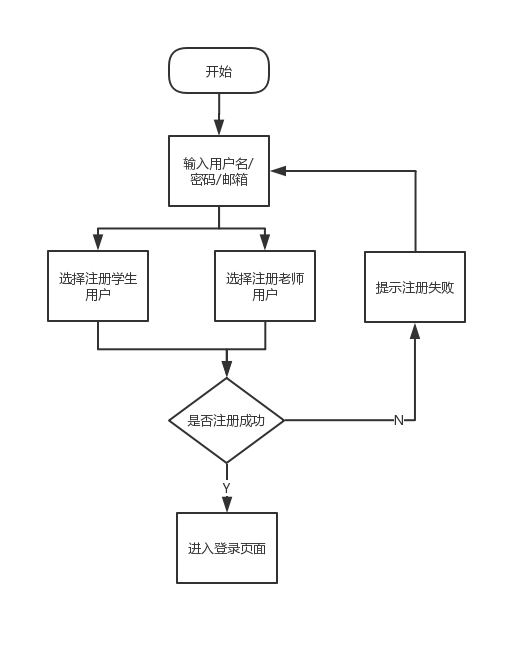


图 4-2-2-3 注册模块详细设计流程图

**4.2.2.2 控制层**

注册模块的控制层负责接受来自main/index.jsp的用户输入，同时调用注册模块的业务逻辑接口，将用户名密码邮箱关键信息传递到业务逻辑层进行判定。等到业务逻辑处理完成之后，将来自业务逻辑层的相应信息传到表现层，并决定显示页面。注册模块控制层列表见表4-2-2-4所示。

表 4-2-2-4 注册模块控制层列表

|  |  |  |  |
| --- | --- | --- | --- |
| 事件 | Action | 转移说明 | 出口 |
| 注册 | userReg.java | SUCCESS | main/index.jsp  —注册成功，显示提示窗口，跳转到登陆页面。 |
| ERROR | main/index.jsp  ---注册失败，显示提示窗口，重新输入信息。 |

在控制层中userReg.java的描述如下所示：

public class userReg extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doPost(request,response);

}

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

request.setCharacterEncoding("utf-8");

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

String strSelect=(String)request.getParameter("select");

if(strSelect!=null&&strSelect.equals("teacher")){

PrintWriter out = response.getWriter();

//收集数据

String strName=(String)request.getParameter("name");

String strEmail=(String)request.getParameter("email");

String strPassword=(String)request.getParameter("password");

//处理数据

Str str=new Str();

Show show=new Show();

Teacher teacher=new Teacher();

teacher.setEmail(str.dbEncode(str.toChinese(strEmail)));

teacher.setName(str.dbEncode(str.toChinese(strName)));

teacher.setPassword(str.dbEncode(str.toChinese(strPassword)));

if(teacher.getName().equals("")||teacher.getPassword().equals("")||teacher.getEmail().equals("")){

out.print(show.errorBox("请添写完整信息！","错误信息"));

return;

}else{

int intT=0;

Data data=new Data();

intT=data.getRowCount("tb\_teacher WHERE name='"+teacher.getName()+"'");

if(intT>0){

out.print(show.errorBox("该用户名已被注册！","注册信息"));

return;

}else{

intT=data.insert("INSERT INTO tb\_teacher(password,name,email) VALUES('"+teacher.getPassword()+"','"+teacher.getName()+"','"+teacher.getEmail()+"')");

if(intT<=0){

out.print(show.errorBox("注册失败，数据库错误!","错误信息"));

return;

}else{

out.print("<script>alert('注册成功！')</script>");

response.sendRedirect("main/index.jsp");

}

}

}

}else {

request.setCharacterEncoding("utf-8");

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

PrintWriter out = response.getWriter();

//接收数据

String strEmail=(String)request.getParameter("email");

String strName=(String)request.getParameter("name");

String strPassword=(String)request.getParameter("password");

//处理数据

Str str=new Str();

Show show=new Show();

Student student=new Student();

student.setEmail(strEmail);

student.setName(strName);

student.setPassword(strPassword);

if(student.getName().equals("")||student.getPassword().equals("")||student.getEmail().equals("")){

out.print(show.errorBox("请添写完整信息！","错误信息"));

return;

}else{

int intT=0;

Data data=new Data();

intT=data.getRowCount("tb\_student WHERE name='"+student.getName()+"'");

if(intT>0){

out.print(show.errorBox("该用户名已被注册！","注册信息"));

return;

}else{

intT=data.insert("INSERT INTO tb\_student(password,name,email) VALUES('"+student.getPassword()+"','"+student.getName()+"','"+student.getEmail()+"')");

if(intT<=0){

out.print(show.errorBox("注册失败，数据库错误!","错误信息"));

return;

}else{

out.print("<script>alert('注册成功！');document.location='main/index.jsp';</script>");

}

}

}

}

}

}

**4.2.2.3 业务逻辑层**

注册模块的业务逻辑层主要完成对用户注册逻辑的判定，数据库查询，与数据库存入的信息比对，查看是否重复信息。页面跳转，响应重定向。比如：注册模块业务逻辑层列表如图4-2-2-5所示。

表 4-2-2-5注册模块业务逻辑层列表

|  |  |  |  |
| --- | --- | --- | --- |
| 事件 | Service | 调用说明 | 出口 |
| 注册 | userReg.java | 调用JavaBean将数据插入数据库 | 返回给Action |

userReg.java方法：

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

request.setCharacterEncoding("utf-8");

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

String strSelect=(String)request.getParameter("select");

if(strSelect!=null&&strSelect.equals("teacher")){

PrintWriter out = response.getWriter();

//收集数据

String strName=(String)request.getParameter("name");

String strEmail=(String)request.getParameter("email");

String strPassword=(String)request.getParameter("password");

//处理数据

Str str=new Str();

Show show=new Show();

Teacher teacher=new Teacher();

teacher.setEmail(str.dbEncode(str.toChinese(strEmail)));

teacher.setName(str.dbEncode(str.toChinese(strName)));

teacher.setPassword(str.dbEncode(str.toChinese(strPassword)));

if(teacher.getName().equals("")||teacher.getPassword().equals("")||teacher.getEmail().equals("")){

out.print(show.errorBox("请添写完整信息！","错误信息"));

return;

}else{

int intT=0;

Data data=new Data();

intT=data.getRowCount("tb\_teacher WHERE name='"+teacher.getName()+"'");

if(intT>0){

out.print(show.errorBox("该用户名已被注册！","注册信息"));

return;

}else{

intT=data.insert("INSERT INTO tb\_teacher(password,name,email) VALUES('"+teacher.getPassword()+"','"+teacher.getName()+"','"+teacher.getEmail()+"')");

if(intT<=0){

out.print(show.errorBox("注册失败，数据库错误!","错误信息"));

return;

}else{

out.print("<script>alert('注册成功！')</script>");

response.sendRedirect("main/index.jsp");

}

}

}

}else {

request.setCharacterEncoding("utf-8");

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

PrintWriter out = response.getWriter();

//接收数据

String strEmail=(String)request.getParameter("email");

String strName=(String)request.getParameter("name");

String strPassword=(String)request.getParameter("password");

//处理数据

Str str=new Str();

Show show=new Show();

Student student=new Student();

student.setEmail(strEmail);

student.setName(strName);

student.setPassword(strPassword);

if(student.getName().equals("")||student.getPassword().equals("")||student.getEmail().equals("")){

out.print(show.errorBox("请添写完整信息！","错误信息"));

return;

}else{

int intT=0;

Data data=new Data();

intT=data.getRowCount("tb\_student WHERE name='"+student.getName()+"'");

if(intT>0){

out.print(show.errorBox("该用户名已被注册！","注册信息"));

return;

}else{

intT=data.insert("INSERT INTO tb\_student(password,name,email) VALUES('"+student.getPassword()+"','"+student.getName()+"','"+student.getEmail()+"')");

if(intT<=0){

out.print(show.errorBox("注册失败，数据库错误!","错误信息"));

return;

}else{

out.print("<script>alert('注册成功！');document.location='main/index.jsp';</script>");

}

}

}

}

}

数据库查询，与数据库存入的信息比对，插入数据信息。页面跳转，响应重定向。

**4.2.2.4 数据持久层**

注册模块的是数据持久层能对用户数据进行增删改查。注册模块数据持久层列表见表4-2-6所示。

表 4-2-6 注册模块数据持久层列表

|  |  |  |  |
| --- | --- | --- | --- |
| 事件 | DAO | 调用数据模型 | 说明 |
| 注册 | studentMreg.java  teacherMreg.java  userReg.java | 调用 Scourse.Java  Student.Java  Teacher.Java | 对用户信息进行增改查操作 |

超链接，点击进入相应处理页面(a)。修改注册信息，未填完整，提示信息未填完整，密码与确认密码不一致，提示确认密码输入错误，请重新输入，正确填写，则会提示修改成功（JavaScript）。提交页面交给Servlet处理，在其中调用JavaBean将数据插入数据库，并执行判断，成功返回主页面。

studentMreg.java

public class studentMreg extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doPost(request,response);

}

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

request.setCharacterEncoding("utf-8");

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

PrintWriter out = response.getWriter();

//接收数据

HttpSession session=request.getSession();

String strUserName=(String)session.getAttribute("name");

String specialty=(String)request.getParameter("specialty");

String classNum=(String)request.getParameter("classNum");

String strPassword=(String)request.getParameter("password");

String strrPassword=(String)request.getParameter("rpassword");

String strEmail=(String)request.getParameter("email");

//处理数据

Show show=new Show();

Str str=new Str();

Student student=new Student();

student.setSpecialty(specialty);

student.setClass\_num(classNum);

student.setPassword(str.dbEncode(strPassword));

student.setEmail(str.dbEncode(strEmail));

if(!strrPassword.equals(strPassword)) {

out.print(show.errorBox("新的密码与确认密码不符!","错误信息"));

}else {

if(student.getSpecialty().equals("")||student.getClass\_num().equals("")||student.getPassword().equals("")||student.getEmail().equals("")){

out.print(show.errorBox("请添写完整信息!","错误信息"));

return;

}else{

int intT=0;

Data data=new Data();

intT=data.insert("UPDATE tb\_student SET password='"+student.getPassword()+"',email='"+student.getEmail()+"',specialty='"+student.getSpecialty()+"',classnum='"+student.getClass\_num()+"' WHERE name='"+strUserName+"'");

if(intT<=0){

out.print(show.errorBox("修改失败，数据库错误!","错误信息"));

return;

}else{

out.print("<script>alert('修改成功！');document.location='student/reginfo.jsp';</script>");

}

}

}

}

}

teacherMreg.java

public class studentMreg extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doPost(request,response);

}

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

request.setCharacterEncoding("utf-8");

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

PrintWriter out = response.getWriter();

//接收数据

HttpSession session=request.getSession();

String strUserName=(String)session.getAttribute("name");

String specialty=(String)request.getParameter("specialty");

String classNum=(String)request.getParameter("classNum");

String strPassword=(String)request.getParameter("password");

String strrPassword=(String)request.getParameter("rpassword");

String strEmail=(String)request.getParameter("email");

//处理数据

Show show=new Show();

Str str=new Str();

Student student=new Student();

student.setSpecialty(specialty);

student.setClass\_num(classNum);

student.setPassword(str.dbEncode(strPassword));

student.setEmail(str.dbEncode(strEmail));

if(!strrPassword.equals(strPassword)) {

out.print(show.errorBox("新的密码与确认密码不符!","错误信息"));

}else {

if(student.getSpecialty().equals("")||student.getClass\_num().equals("")||student.getPassword().equals("")||student.getEmail().equals("")){

out.print(show.errorBox("请添写完整信息!","错误信息"));

return;

}else{

int intT=0;

Data data=new Data();

intT=data.insert("UPDATE tb\_student SET password='"+student.getPassword()+"',email='"+student.getEmail()+"',specialty='"+student.getSpecialty()+"',classnum='"+student.getClass\_num()+"' WHERE name='"+strUserName+"'");

if(intT<=0){

out.print(show.errorBox("修改失败，数据库错误!","错误信息"));

return;

}else{

out.print("<script>alert('修改成功！');document.location='student/reginfo.jsp';</script>");

}

}

}

}

}

userReg.java

public class userReg extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doPost(request,response);

}

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

request.setCharacterEncoding("utf-8");

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

String strSelect=(String)request.getParameter("select");

if(strSelect!=null&&strSelect.equals("teacher")){

PrintWriter out = response.getWriter();

//收集数据

String strName=(String)request.getParameter("name");

String strEmail=(String)request.getParameter("email");

String strPassword=(String)request.getParameter("password");

//处理数据

Str str=new Str();

Show show=new Show();

Teacher teacher=new Teacher();

teacher.setEmail(str.dbEncode(str.toChinese(strEmail)));

teacher.setName(str.dbEncode(str.toChinese(strName)));

teacher.setPassword(str.dbEncode(str.toChinese(strPassword)));

if(teacher.getName().equals("")||teacher.getPassword().equals("")||teacher.getEmail().equals("")){

out.print(show.errorBox("请添写完整信息！","错误信息"));

return;

}else{

int intT=0;

Data data=new Data();

intT=data.getRowCount("tb\_teacher WHERE name='"+teacher.getName()+"'");

if(intT>0){

out.print(show.errorBox("该用户名已被注册！","注册信息"));

return;

}else{

intT=data.insert("INSERT INTO tb\_teacher(password,name,email) VALUES('"+teacher.getPassword()+"','"+teacher.getName()+"','"+teacher.getEmail()+"')");

if(intT<=0){

out.print(show.errorBox("注册失败，数据库错误!","错误信息"));

return;

}else{

out.print("<script>alert('注册成功！')</script>");

response.sendRedirect("main/index.jsp");

}

}

}

}else {

request.setCharacterEncoding("utf-8");

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

PrintWriter out = response.getWriter();

//接收数据

String strEmail=(String)request.getParameter("email");

String strName=(String)request.getParameter("name");

String strPassword=(String)request.getParameter("password");

//处理数据

Str str=new Str();

Show show=new Show();

Student student=new Student();

student.setEmail(strEmail);

student.setName(strName);

student.setPassword(strPassword);

if(student.getName().equals("")||student.getPassword().equals("")||student.getEmail().equals("")){

out.print(show.errorBox("请添写完整信息！","错误信息"));

return;

}else{

int intT=0;

Data data=new Data();

intT=data.getRowCount("tb\_student WHERE name='"+student.getName()+"'");

if(intT>0){

out.print(show.errorBox("该用户名已被注册！","注册信息"));

return;

}else{

intT=data.insert("INSERT INTO tb\_student(password,name,email) VALUES('"+student.getPassword()+"','"+student.getName()+"','"+student.getEmail()+"')");

if(intT<=0){

out.print(show.errorBox("注册失败，数据库错误!","错误信息"));

return;

}else{

out.print("<script>alert('注册成功！');document.location='main/index.jsp';</script>");

}

}

}

}

}

}

**4.2.2.5 域模型层**

注册模块用到域模型层中User.java，User.java是一个公用域模型，在涉及到用户信息查询等操作时，就会调用到该模型，注册模块域模型层列表如表4-2-7所示。

表 4-2-7注册模块域模型层列表

|  |  |
| --- | --- |
| 域模型 | 描述 |
| Teacher.java  Student.java | 对用户信息的增、删、改、查操作 |

Teacher.java主要属性与方法：

public class Teacher {

private String tname;

private String password;

private String name;

private String email;

private String specialty;

private String cname;

Student.java主要属性与方法：

public class Student {

private String name;

private String password;

private String email;

private String specialty;

private String class\_num;

**4.3、上传/下载模块**

在上传/下载模块时，系统内部的相应响应操作示意图如图4-3-1所示。

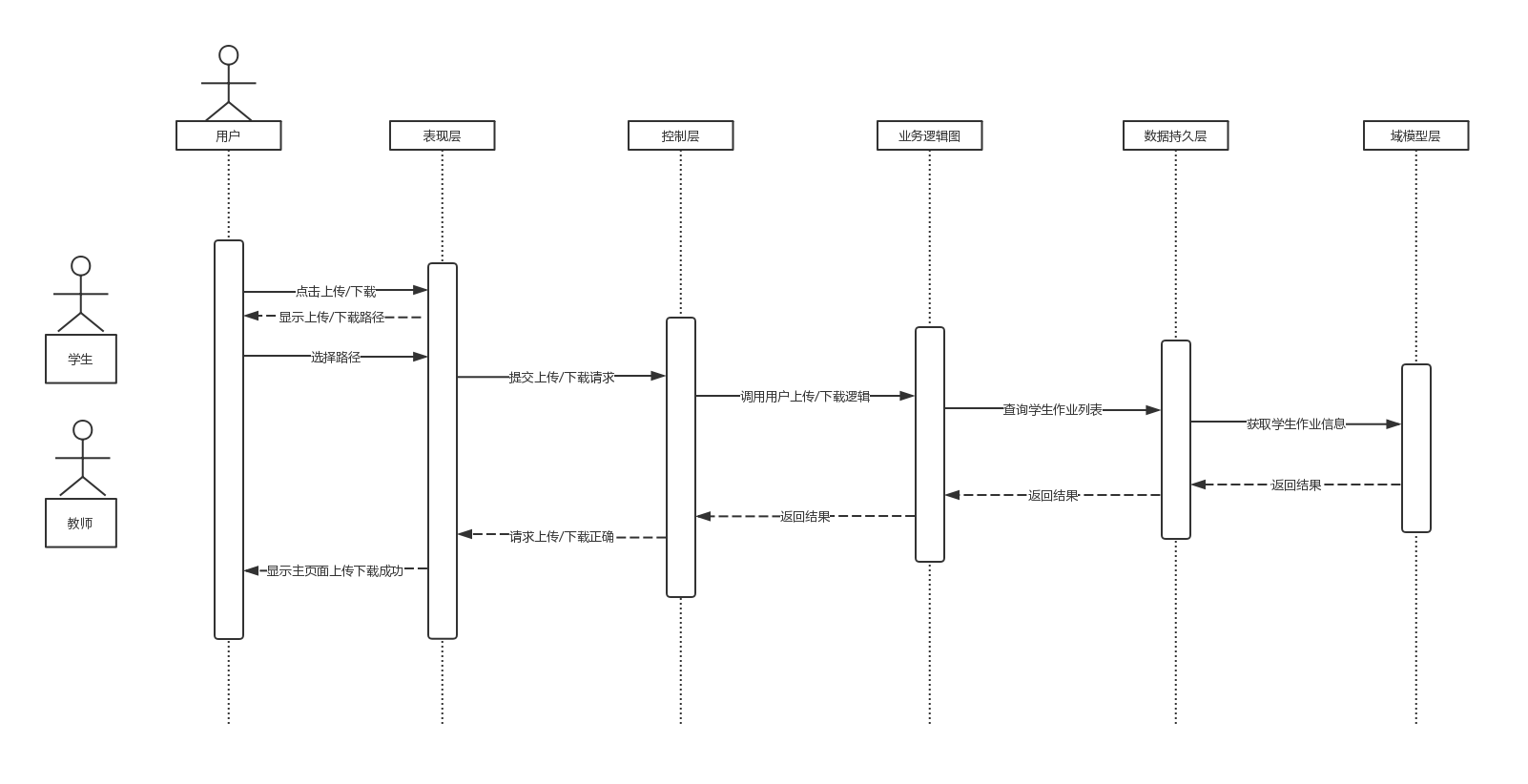


图 4-3-1 系统响应示意图

**4.3.1 表现层**

上传/下载的表现层主要完成用户的上传/下载功能，上传/下载页面是通过学生作业是否上传，显示操作应该是上传还是下载，选择上传操作就会弹出选择文件页面，提示上传成功或失败的提示。选择下载操作就会弹出选择下载路径页面，提示下载成功或失败的提示。表现层对应的JSP页面列表见表4-3-2所示。

表 4-3-2 上传/下载模块表现层JSP列表

|  |  |  |
| --- | --- | --- |
| 界面 | JSP | 功能描述 |
| 上传/下载页面 | vcinfo.jsp | 用户（教师、学生、）下载/上传功能。 |

vcinfo.jsp的流程图如图4-3-3所示。

进入游首页

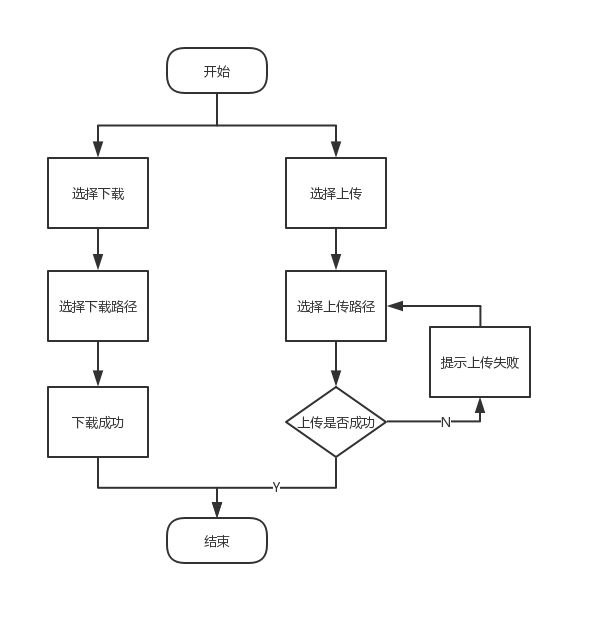


图 4-3-3 上传/下载模块详细设计流程图

**4.3.2 控制层**

上传/下载模块的控制层负责接受来自vcinfo.jsp的用户信息，同时调用模块的业务逻辑接口，将学生作业信息传递到业务逻辑层进行判定。等到业务逻辑处理完成之后，将来自业务逻辑层的相应信息传到表现层，并决定显示页面。上传/下载模块控制层列表见表4-3-4所示。

表 4-3-4 上传/下载模块控制层列表

|  |  |  |  |
| --- | --- | --- | --- |
| 事件 | Action | 转移说明 | 出口 |
| 上传/下载 | servletdown.java  servletUpload.java | SUCCESS | student/download.jsp  student/selectfile.jsp  teacher/ download.jsp  —上传/下载成功，显示提示窗口。 |
| ERROR | vcinfo.jsp  ---上传/下载失败，显示提示窗口。 |

在控制层中servletdown.java的描述如下所示：

public class servletdown extends HttpServlet{

public servletdown() {

}

//处理GET请求

public void doGet(HttpServletRequest request,HttpServletResponse response)throws ServletException,IOException{

PrintWriter out=response.getWriter();

out.println("这个方法必须用post");

}

//相应POST请求

protected void doPost(HttpServletRequest request,HttpServletResponse response)throws ServletException,IOException{

PageContext pageContext = JspFactory.getDefaultFactory().getPageContext(this,request,

response,null,true,8192,true);

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

// PrintWriter out=response.getWriter();

//定义变量

int count=0;

SmartUpload mySmartUpload=new SmartUpload();

//String file=request.getParameter(arg0)

try{

//初始化

mySmartUpload.initialize(pageContext);

//下载

mySmartUpload.setContentDisposition(null);

//保存下载文件到指定目录

//PATH为form表单发过来的

HttpSession s=request.getSession();

String file=(String)s.getAttribute("cname");

mySmartUpload.downloadFile("C:/Users/qny/Desktop/上传/"+file+"");

count=mySmartUpload.save(mySmartUpload.getRequest().getParameter("path"));

System.out.println(mySmartUpload.getRequest().getParameter("path"));

//显示处理结果

// out.println(count+" file download");

}

catch(Exception e)

{

// out.println("unable to download the file");

// out.println(e.toString()+"");

}

}

}

在控制层中servletUpload.java的描述如下所示：

public class servletUpload extends HttpServlet{

public servletUpload() {

}

//处理GET请求

public void doGet(HttpServletRequest request,HttpServletResponse response)throws ServletException,IOException{

PrintWriter out=response.getWriter();

out.println("这个方法必须用post");

}

//相应POST请求

protected void doPost(HttpServletRequest request,HttpServletResponse response)throws ServletException,IOException{

request.setCharacterEncoding("gb2312");

response.setContentType("text/html;charset=gb2312");

Show show=new Show();

PageContext pageContext = JspFactory.getDefaultFactory().getPageContext(this,request, response,null,true,8192,true);

PrintWriter out=response.getWriter();

//定义变量

int count=0;

HttpSession s=request.getSession();

String strUserName=(String)s.getAttribute("name");

SmartUpload mySmartUpload=new SmartUpload();

try{

//初始化

mySmartUpload.initialize(pageContext);

//上传

mySmartUpload.upload();

String file=mySmartUpload.getFiles().getFile(0).getFileName();

if(file.equals("")||file==null||file.equals("null")) {

out.print(show.errorBox("未选择文件或文件名为空!","错误信息"));

return;

}

s.setAttribute("file",file);

//保存上传文件到指定目录

//PATH为form表单发过来的

count=mySmartUpload.save("C://Users//qny//Desktop//上传");

//显示处理结果

// out.println(count+" file upload");

//JavaBean处理

Conn con=new Conn();

Data data=new Data();

Str str=new Str();

Time time=new Time();

Scourse course=new Scourse();

course.setSname(strUserName);

course.setCname(file);

course.setPtime(time.getYMD());

int intT=0;

intT=data.insert("INSERT INTO tb\_work\_completed(cname,sname) VALUES('"+course.getCname()+"','"+course.getSname()+"')");

if(intT<=0){

out.print(show.errorBox("上传失败，数据库错误!","错误信息"));

return;

}else{

out.print("<script>alert('上传成功！');document.location='student/vcinfo.jsp';</script>");

}

//response.sendRedirect("student/acinfo.jsp");

}

catch(Exception e)

{

out.println("unable to upload the file");

out.println(e.toString()+"");

}

}

}

**4.3.3 业务逻辑层**

上传/下载模块的业务逻辑层主要是使用超链接，点击进入相应处理页面。上传/下载功能，用到Smartupload组件，超链接的参数传递。下载功能的实现通过Servlet。比如：上传/下载模块业务逻辑层列表如图4-3-5所示。

表 4-3-5上传/下载模块业务逻辑层列表

|  |  |  |  |
| --- | --- | --- | --- |
| 事件 | Service | 调用说明 | 出口 |
| 上传/下载 | servletdown.java  servletUpload.java | 调用JavaBean将数据插入数据库 | 返回给Action |

servletdown.java方法：

public class servletdown extends HttpServlet{

public servletdown() {

}

//处理GET请求

public void doGet(HttpServletRequest request,HttpServletResponse response)throws ServletException,IOException{

PrintWriter out=response.getWriter();

out.println("这个方法必须用post");

}

//相应POST请求

protected void doPost(HttpServletRequest request,HttpServletResponse response)throws ServletException,IOException{

PageContext pageContext = JspFactory.getDefaultFactory().getPageContext(this,request,

response,null,true,8192,true);

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

// PrintWriter out=response.getWriter();

//定义变量

int count=0;

SmartUpload mySmartUpload=new SmartUpload();

//String file=request.getParameter(arg0)

try{

//初始化

mySmartUpload.initialize(pageContext);

//下载

mySmartUpload.setContentDisposition(null);

//保存下载文件到指定目录

//PATH为form表单发过来的

HttpSession s=request.getSession();

String file=(String)s.getAttribute("cname");

mySmartUpload.downloadFile("C:/Users/qny/Desktop/上传/"+file+"");

count=mySmartUpload.save(mySmartUpload.getRequest().getParameter("path"));

System.out.println(mySmartUpload.getRequest().getParameter("path"));

//显示处理结果

// out.println(count+" file download");

}

catch(Exception e)

{

// out.println("unable to download the file");

// out.println(e.toString()+"");

}

}

}

数据库查询，查看学生作业信息，显示是上传还是下载。

servletUpload.java主要实现方法:

public class servletUpload extends HttpServlet{

public servletUpload() {

}

//处理GET请求

public void doGet(HttpServletRequest request,HttpServletResponse response)throws ServletException,IOException{

PrintWriter out=response.getWriter();

out.println("这个方法必须用post");

}

//相应POST请求

protected void doPost(HttpServletRequest request,HttpServletResponse response)throws ServletException,IOException{

request.setCharacterEncoding("gb2312");

response.setContentType("text/html;charset=gb2312");

Show show=new Show();

PageContext pageContext = JspFactory.getDefaultFactory().getPageContext(this,request, response,null,true,8192,true);

PrintWriter out=response.getWriter();

//定义变量

int count=0;

HttpSession s=request.getSession();

String strUserName=(String)s.getAttribute("name");

SmartUpload mySmartUpload=new SmartUpload();

try{

//初始化

mySmartUpload.initialize(pageContext);

//上传

mySmartUpload.upload();

String file=mySmartUpload.getFiles().getFile(0).getFileName();

if(file.equals("")||file==null||file.equals("null")) {

out.print(show.errorBox("未选择文件或文件名为空!","错误信息"));

return;

}

s.setAttribute("file",file);

//保存上传文件到指定目录

//PATH为form表单发过来的

count=mySmartUpload.save("C://Users//qny//Desktop//上传");

//显示处理结果

// out.println(count+" file upload");

//JavaBean处理

Conn con=new Conn();

Data data=new Data();

Str str=new Str();

Time time=new Time();

Scourse course=new Scourse();

course.setSname(strUserName);

course.setCname(file);

course.setPtime(time.getYMD());

int intT=0;

intT=data.insert("INSERT INTO tb\_work\_completed(cname,sname) VALUES('"+course.getCname()+"','"+course.getSname()+"')");

if(intT<=0){

out.print(show.errorBox("上传失败，数据库错误!","错误信息"));

return;

}else{

out.print("<script>alert('上传成功！');document.location='student/vcinfo.jsp';</script>");

}

//response.sendRedirect("student/acinfo.jsp");

}

catch(Exception e)

{

out.println("unable to upload the file");

out.println(e.toString()+"");

}

}

}

**4.3.4 数据持久层**

上传/下载模块的是数据持久层能对学生作业数据进行增删改查。上传/下载模块数据持久层列表见表4-3-6所示。

表 4-3-6 上传/下载模块数据持久层列表

|  |  |  |  |
| --- | --- | --- | --- |
| 事件 | DAO | 调用数据模型 | 说明 |
| 上传/下载 | servletdown.java  servletUpload.java | 调用 Scourse.Java | 对学生作业信息进行增改查操作 |

超链接，点击进入相应处理页面(a)。指定文件下载功能，用到Smartupload组件，超链接的参数传递。下载功能的实现通过Servlet。在其中调用JavaBean将数据插入数据库，并执行判断，成功返回主页面。

SmartUpload mySmartUpload=new SmartUpload();

//String file=request.getParameter(arg0)

try{

//初始化

mySmartUpload.initialize(pageContext);

//下载

mySmartUpload.setContentDisposition(null);

//保存下载文件到指定目录

//PATH为form表单发过来的

HttpSession s=request.getSession();

String file=(String)s.getAttribute("cname");

mySmartUpload.downloadFile("C:/Users/qny/Desktop/上传/"+file+"");

count=mySmartUpload.save(mySmartUpload.getRequest().getParameter("path"));

**4.3.5 域模型层**

上传/下载模块用到域模型层中Scourse.java，Scourse.java是一个公用域模型，在涉及到用户信息查询等操作时，就会调用到该模型，登录模块域模型层列表如表4-7所示。

表 4-7上传/下载模块域模型层列表

|  |  |
| --- | --- |
| 域模型 | 描述 |
| Scourse.java | 对学生作业信息的增、改、查操作 |

Scourse.java主要属性与方法：

public class Scourse {

private String cname;

private String sname;

private String tname;

private int time;

private int score;

private String goal;

private String require;

private String other;

private String ptime;

private String etime;

**4.4、修改/布置模块**

在修改/布置模块时，系统内部的相应响应操作示意图如图4-4-1所示。

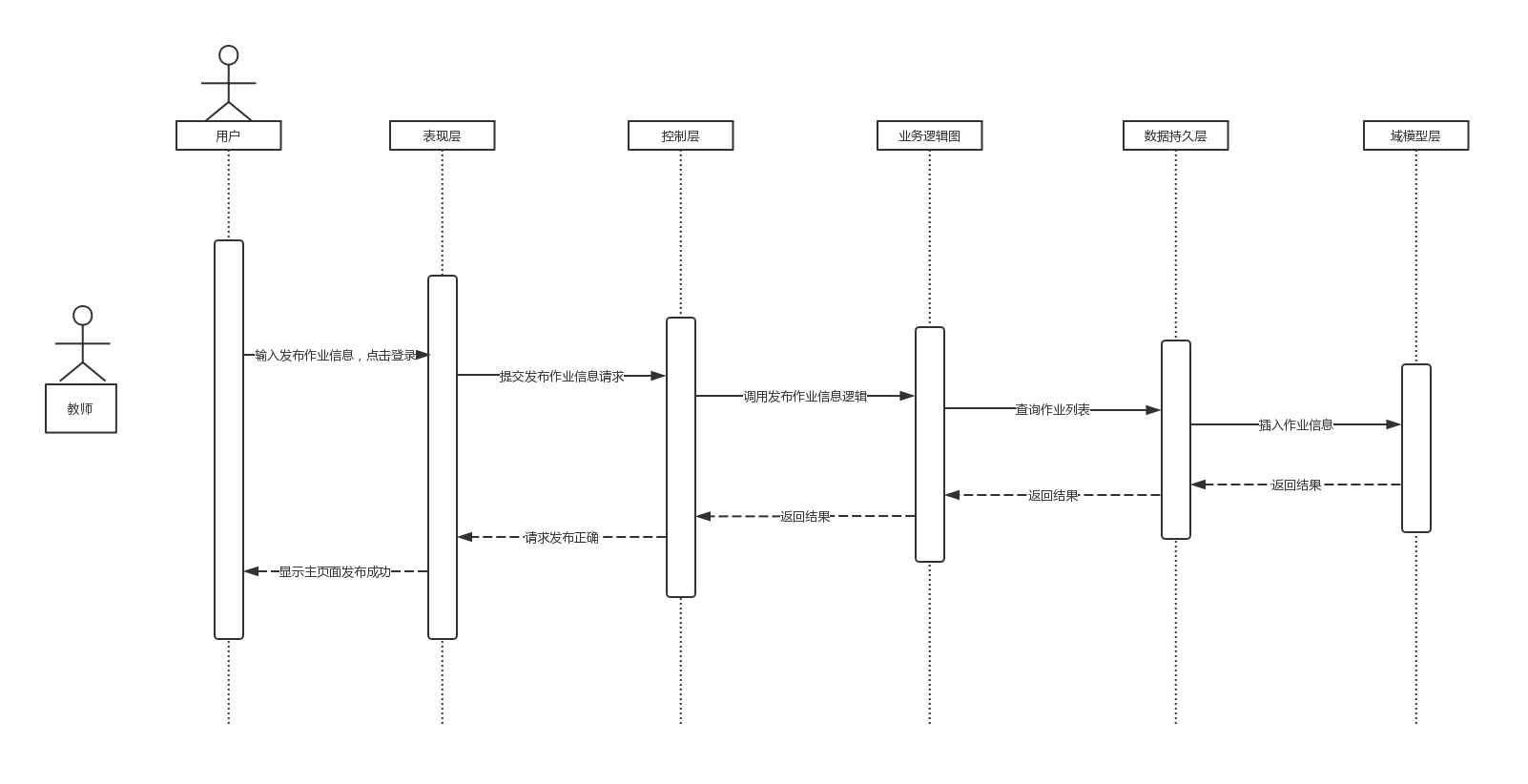


图 4-4-1 系统响应示意图

**4.4.1 表现层**

修改/布置的表现层主要完成教师用户的修改/布置功能，修改/布置页面时要求教师用户输入作业的基本信息，确认后页面给出响应消息，提示修改/布置成功。表现层对应的JSP页面列表见表4-4-2所示。

表 4-4-2 修改/布置模块表现层JSP列表

|  |  |  |
| --- | --- | --- |
| 界面 | JSP | 功能描述 |
| 修改/布置页面 | acinfo.jsp  editscore.jsp | 用户（教师）修改/布置功能。 |

acinfo.jsp、editscore.jsp的流程图如图4-4-3所示。

进入游首页

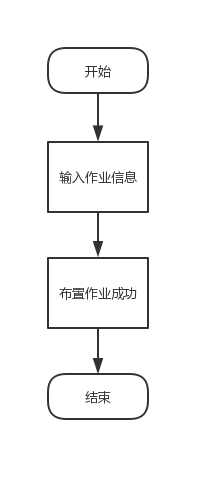
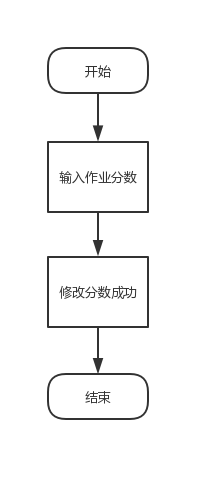
 

图 4-4-3 修改/布置模块详细设计流程图

**4.4.2 控制层**

修改/布置模块的控制层负责接受来自acinfo.jsp、editscore.jsp的作业信息，同时调用模块的业务逻辑接口，将作业信息传递到业务逻辑层进行判定。等到业务逻辑处理完成之后，将来自业务逻辑层的相应信息传到表现层，并决定显示页面。修改/布置模块控制层列表见表4-4-4所示。

表 4-4-4 修改/布置模块控制层列表

|  |  |  |  |
| --- | --- | --- | --- |
| 事件 | Action | 转移说明 | 出口 |
| 修改/布置 | teacherAcinfo.java  editscore.jsp | SUCCESS | acinfo.jsp  editscore.jsp  —修改/布置成功，显示提示窗口。 |
| ERROR | acinfo.jsp  editscore.jsp  ---修改/布置失败，显示提示窗口。 |

在控制层中teacherAcinfo.java的描述如下所示：

public class teacherAcinfo extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doPost(request,response);

}

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

request.setCharacterEncoding("utf-8");

PrintWriter out = response.getWriter();

//收集数据

HttpSession session=request.getSession();

String strName=(String)session.getAttribute("name");

String strCname=(String)request.getParameter("cname");

String strGoal=(String)request.getParameter("goal");

String strRequire=(String)request.getParameter("require");

String eTime=(String)request.getParameter("eTime");

// DateFormat df = new SimpleDateFormat("yyyy-MM-dd");

// Date data2 = df.parse(eTime);

int Itime=0;

Itime= Integer.valueOf(request.getParameter("time"));

//处理数据

Show show=new Show();

Data data=new Data();

Str str=new Str();

Time time=new Time();

Scourse course=new Scourse();

int intT=0;

course.setCname(strCname);

course.setGoal(strGoal);

course.setRequire(strRequire);

course.setEtime(eTime);

course.setTime(Itime);

course.setPtime(str.dbEncode(time.getYMD()));

if(course.getCname().equals("")||course.getGoal().equals("")||course.getRequire().equals("")||course.getTime()==0||course.getEtime().equals("")){

out.print(show.errorBox("请添写完整信息!","错误信息"));

return;

}else{

// intT=data.insert("INSERT INTO tb\_course(cname,goal,require,other,tname,time,ptime) VALUES('"+course.getCname()+"','"+

// course.getGoal()+"','"+course.getRequire()+"','"+course.getOther()+"','"+strName+"','"+course.getTime()+"','"+time.getYMD()+"')");

intT=data.insert("insert into tb\_course (cname,goal,requirement,tname,time,ptime,etime) values('"+course.getCname()+"','"+course.getGoal()+"','"+course.getRequire()+"','"+strName+"','"+course.getTime()+"','"+course.getPtime()+"','"+course.getEtime()+"')");

if(intT<=0){

out.print(show.errorBox("发布失败，数据库错误!","错误信息"));

return;

}else{

out.print("<script>alert('发布成功！');document.location='teacher/acinfo.html';</script>");

}

}

}

editscore.jsp

<%

Conn con=new Conn();

Show show=new Show();

Str str=new Str();

String strChecked="checked";

String strUserName=(String)session.getAttribute("name");

String strCname=(String)request.getParameter("cName");

String sName=(String)request.getParameter("sName");

session.setAttribute("cname",strCname);

ResultSet rs=con.getRs("SELECT \* FROM tb\_work\_completed WHERE sname='"+sName+"'");

while(rs.next()){

if(strCname.equals(rs.getString(1))){

%>

**4.4.3 业务逻辑层**

修改/布置模块的业务逻辑层主要是使用超链接，点击进入相应处理页面。Html/Servlet，完成功能。比如：修改/布置模块业务逻辑层列表如图4-4-5所示。

表 4-4-5修改/布置模块业务逻辑层列表

|  |  |  |  |
| --- | --- | --- | --- |
| 事件 | Service | 调用说明 | 出口 |
| 修改/布置 | teacherAcinfo.java  editscore.jsp | 调用JavaBean将数据插入数据库 | 返回给Action |

servletdown.java方法：

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

request.setCharacterEncoding("utf-8");

PrintWriter out = response.getWriter();

//收集数据

HttpSession session=request.getSession();

String strName=(String)session.getAttribute("name");

String strCname=(String)request.getParameter("cname");

String strGoal=(String)request.getParameter("goal");

String strRequire=(String)request.getParameter("require");

String eTime=(String)request.getParameter("eTime");

editscore.jsp

<%

Conn con=new Conn();

Show show=new Show();

Str str=new Str();

String strChecked="checked";

String strUserName=(String)session.getAttribute("name");

String strCname=(String)request.getParameter("cName");

String sName=(String)request.getParameter("sName");

session.setAttribute("cname",strCname);

ResultSet rs=con.getRs("SELECT \* FROM tb\_work\_completed WHERE sname='"+sName+"'");

while(rs.next()){

if(strCname.equals(rs.getString(1))){

%>

**4.4.4 数据持久层**

修改/布置模块的是数据持久层能对作业数据进行增加。修改/布置模块数据持久层列表见表4-4-6所示。

表 4-4-6 修改/布置模块数据持久层列表

|  |  |  |  |
| --- | --- | --- | --- |
| 事件 | DAO | 调用数据模型 | 说明 |
| 修改/布置 | teacherAcinfo.java  editscore.jsp | 调用 Scourse.Java | 对作业信息进行操作 |

超链接，点击进入相应处理页面。Html/Servlet，完成功能。下载功能的实现通过Servlet。在其中调用JavaBean将数据插入数据库，并执行判断，成功返回主页面。

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

request.setCharacterEncoding("utf-8");

PrintWriter out = response.getWriter();

//收集数据

HttpSession session=request.getSession();

String strName=(String)session.getAttribute("name");

String strCname=(String)request.getParameter("cname");

String strGoal=(String)request.getParameter("goal");

String strRequire=(String)request.getParameter("require");

String eTime=(String)request.getParameter("eTime");

editscore.jsp

<%

Conn con=new Conn();

Show show=new Show();

Str str=new Str();

String strChecked="checked";

String strUserName=(String)session.getAttribute("name");

String strCname=(String)request.getParameter("cName");

String sName=(String)request.getParameter("sName");

session.setAttribute("cname",strCname);

ResultSet rs=con.getRs("SELECT \* FROM tb\_work\_completed WHERE sname='"+sName+"'");

while(rs.next()){

if(strCname.equals(rs.getString(1))){

%>

**4.4.5 域模型层**

修改/布置模块用到域模型层中Scourse.java，Scourse.java是一个公用域模型，在涉及到用户信息查询等操作时，就会调用到该模型，修改/布置模块域模型层列表如表4-4-7所示。

表 4-4-7修改/布置模块域模型层列表

|  |  |
| --- | --- |
| 域模型 | 描述 |
| Scourse.java | 对学生作业信息的增、改、查操作 |

Scourse.java主要属性与方法：

**public** **class** Scourse {

**private** String cname;

**private** String sname;

**private** String tname;

**private** **int** time;

**private** **int** score;

**private** String goal;

**private** String require;

**private** String other;

**private** String ptime;

**private** String etime;

# 5、管理端模块详细设计

## 5.1登陆模块

在登录模块时，系统内部的相应响应操作示意图如图5-1-1所示。

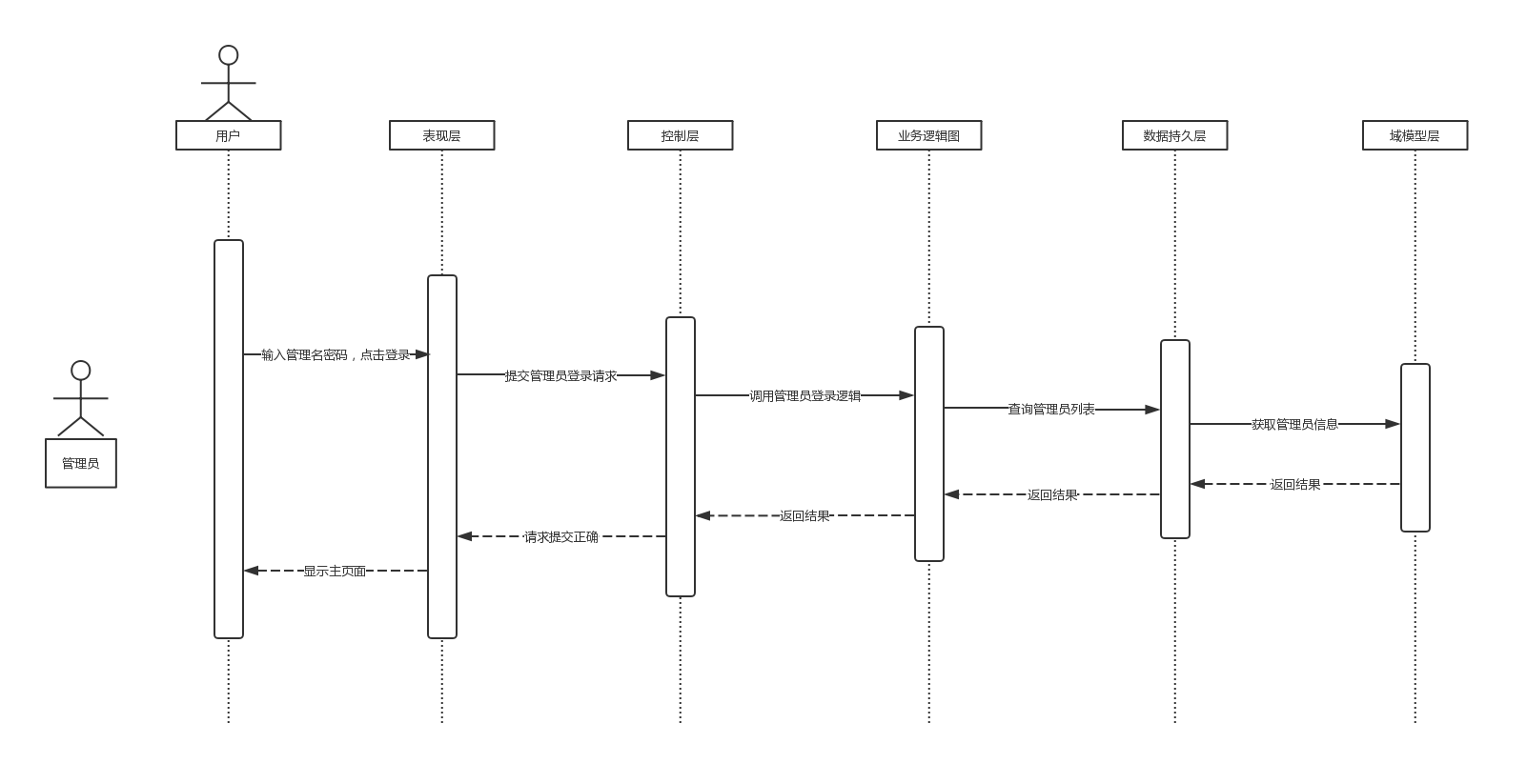


图 5-1-1 系统响应示意图

#### 5.1.1 表现层

登录模块的表现层主要完成不同用户的登录功能，登录页面时要求用户输入账号密码的基本信息，确认后页面给出响应消息，提示登录成功或失败的提示。表现层对应的JSP页面列表见表5-1-2所示。

表 5-1-2 登录模块表现层JSP列表

|  |  |  |
| --- | --- | --- |
| 界面 | JSP | 功能描述 |
| 登录页面 | main/index.jsp | 用户（教师、学生、管理员）登入功能，当登入出错时给出提示。 |

main/index.jsp的流程图如图5-1-3所示。

进入游首页

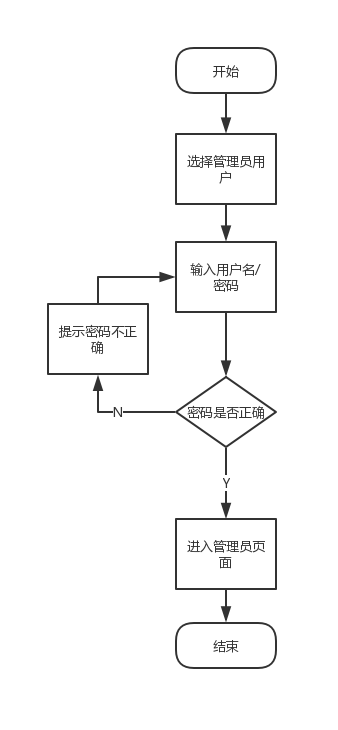


图 5-1-3登录模块详细设计流程图

#### 5.1.2 控制层

登录模块的控制层负责接受来自main/index.jsp的用户输入，同时调用登录模块的业务逻辑接口，将用户名与密码等用户关键信息传递到业务逻辑层进行判定。等到业务逻辑处理完成之后，将来自业务逻辑层的相应信息传到表现层，并决定显示页面。登录模块控制层列表见表5-1-4所示。

表 5-1-4 登录模块控制层列表

|  |  |  |  |
| --- | --- | --- | --- |
| 事件 | Action | 转移说明 | 出口 |
| 登录 | servletMain.java  login.java | SUCCESS | teacher/adver.jsp  student/adver.jsp  admin/admin.jsp  —登录成功，显示提示窗口。 |
| ERROR | servletMain.jsp  login.jsp  ---登录失败，显示提示窗口。 |

在控制层中servletMain.java的描述如下所示：

public class servletMain extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doPost(request,response);

}

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

request.setCharacterEncoding("utf-8");

//Servlet接收

PrintWriter out = response.getWriter();

HttpSession session=request.getSession();

String strUserName=(String)request.getParameter("username");

String strPassword=(String)request.getParameter("password");

String strSelect=(String)request.getParameter("select");

//JavaBean处理

Data data=new Data();

Show show=new Show();

int intT=0;

if(strSelect!=null&&strSelect.equals("student")){

intT=data.getRowCount("tb\_student WHERE name='"+strUserName+"' AND password='"+strPassword+"'");

if(intT>0){

session.setAttribute("name",strUserName);

session.setAttribute("use","student");

response.sendRedirect("student/login.jsp");

}else{

out.print(show.errorBox("检查你的用户名或密码！","验证错误"));

return;

}

}else if(strSelect!=null&&strSelect.equals("teacher")){

intT=data.getRowCount("tb\_teacher WHERE name='"+strUserName+"' AND password='"+strPassword+"'");

if(intT>0){

session.setAttribute("name",strUserName);

session.setAttribute("use","teacher");

response.sendRedirect("teacher/login.jsp");

}else{

out.print(show.errorBox("检查你的用户名或密码！","验证错误"));

return;

}

}else{

out.print(show.errorBox("检查你的用户名或密码！","验证错误"));

return;

}

}

}

Login.java

public class login extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doPost(request,response);

}

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

//Servlet接收

PrintWriter out=response.getWriter();

HttpSession session=request.getSession();

String strName=(String)request.getParameter("name");

String strPassword=(String)request.getParameter("password");

//JavaBean处理

Data data=new Data();

Show show=new Show();

int intT=0;

if(strName!=null||strPassword!=null){

intT=data.getRowCount("tb\_admin WHERE name='"+strName+"' AND password='"+strPassword+"'");

if(intT>0){

session.setAttribute("admin",strName);

response.sendRedirect("admin/admin.jsp");

}else{

out.print(show.errorBox("请检查你的用户名密码！","验证信息"));

return;

}

}else{

out.print(show.errorBox("请检查你的用户名密码！","验证信息"));

return;

}

}

}

#### 5.1.3 业务逻辑层

登录模块的业务逻辑层主要完成对用户登录逻辑的判定，Servlet响应单选按钮，判断进入不同用户验证。数据库查询，与数据库存入的信息比对，验证身份。页面跳转，响应重定向。比如：登录模块业务逻辑层列表如图5-1-5所示。

表 5-1-5登录模块业务逻辑层列表

|  |  |  |  |
| --- | --- | --- | --- |
| 事件 | Service | 调用说明 | 出口 |
| 登录 | servletMain.java  login.java | 调用JavaBean将数据插入数据库 | 返回给Action |

servletMain.java方法：

public class servletMain extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doPost(request,response);

}

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

request.setCharacterEncoding("utf-8");

//Servlet接收

PrintWriter out = response.getWriter();

HttpSession session=request.getSession();

String strUserName=(String)request.getParameter("username");

String strPassword=(String)request.getParameter("password");

String strSelect=(String)request.getParameter("select");

//JavaBean处理

Data data=new Data();

Show show=new Show();

int intT=0;

if(strSelect!=null&&strSelect.equals("student")){

intT=data.getRowCount("tb\_student WHERE name='"+strUserName+"' AND password='"+strPassword+"'");

if(intT>0){

session.setAttribute("name",strUserName);

session.setAttribute("use","student");

response.sendRedirect("student/login.jsp");

}else{

out.print(show.errorBox("检查你的用户名或密码！","验证错误"));

return;

}

}else if(strSelect!=null&&strSelect.equals("teacher")){

intT=data.getRowCount("tb\_teacher WHERE name='"+strUserName+"' AND password='"+strPassword+"'");

if(intT>0){

session.setAttribute("name",strUserName);

session.setAttribute("use","teacher");

response.sendRedirect("teacher/login.jsp");

}else{

out.print(show.errorBox("检查你的用户名或密码！","验证错误"));

return;

}

}else{

out.print(show.errorBox("检查你的用户名或密码！","验证错误"));

return;

}

}

}

Servlet响应单选按钮，判断进入不同用户验证。数据库查询，与数据库存入的信息比对，验证身份。页面跳转，响应重定向。

login.java主要实现方法:

public class login extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doPost(request,response);

}

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

//Servlet接收

PrintWriter out=response.getWriter();

HttpSession session=request.getSession();

String strName=(String)request.getParameter("name");

String strPassword=(String)request.getParameter("password");

//JavaBean处理

Data data=new Data();

Show show=new Show();

int intT=0;

if(strName!=null||strPassword!=null){

intT=data.getRowCount("tb\_admin WHERE name='"+strName+"' AND password='"+strPassword+"'");

if(intT>0){

session.setAttribute("admin",strName);

response.sendRedirect("admin/admin.jsp");

}else{

out.print(show.errorBox("请检查你的用户名密码！","验证信息"));

return;

}

}else{

out.print(show.errorBox("请检查你的用户名密码！","验证信息"));

return;

}

}

}

，成功返回主页面。

#### 5.1.5 域模型层

登录模块用到域模型层中User.java，User.java是一个公用域模型，在涉及到用户信息查询等操作时，就会调用到该模型，登录模块域模型层列表如表5-1-7所示。

表 5-1-7登录模块域模型层列表

|  |  |
| --- | --- |
| 域模型 | 描述 |
| Teacher.java  Student.java | 对用户信息的增、删、改、查操作 |

Teacher.java主要属性与方法：

public class Teacher {

private String tname;

private String password;

private String name;

private String email;

private String specialty;

private String cname;

Student.java主要属性与方法：

public class Student {

private String name;

private String password;

private String email;

private String specialty;

private String class\_num;

**5.2管理信息模块**

在管理信息模块时，系统内部的相应响应操作示意图如图5-2-1所示。

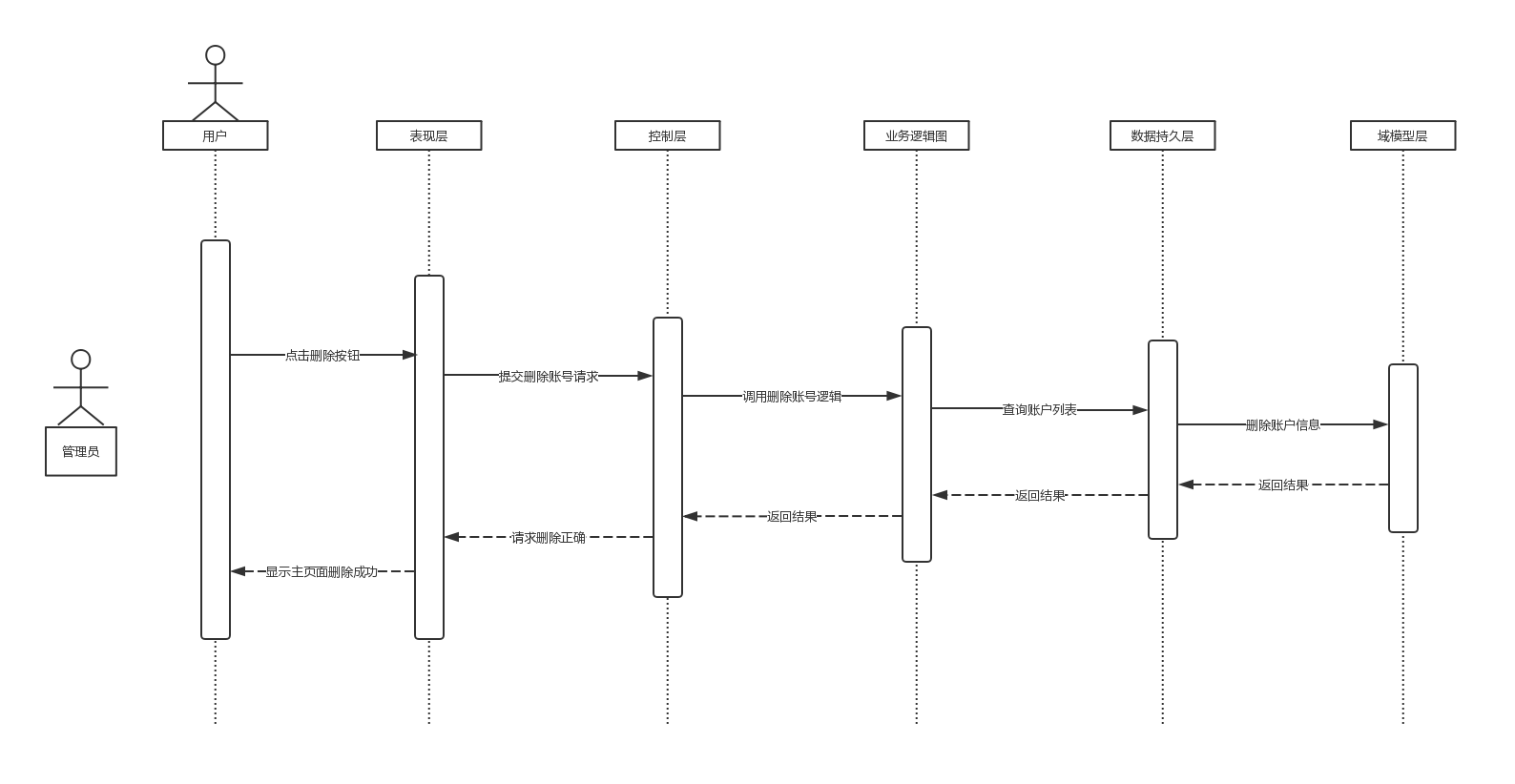


图 5-2-1 系统响应示意图

**5.2.1 表现层**

管理信息的表现层主要完成学生教师账号的删除功能，管理信息页面时要求管理员点击删除学生或教师，确认后页面给出响应消息，提示管理信息成功。表现层对应的JSP页面列表见表5-2-2所示。

表 5-5-2 管理信息模块表现层JSP列表

|  |  |  |
| --- | --- | --- |
| 界面 | JSP | 功能描述 |
| 管理信息页面 | dcourse.jsp  dstudent.jsp  dteacher.jsp | 管理员删除用户（学生、教师）功能。 |

dcourse.jsp、dstudent.jsp、dteacher.jsp的流程图如图5-2-3所示。

进入游首页

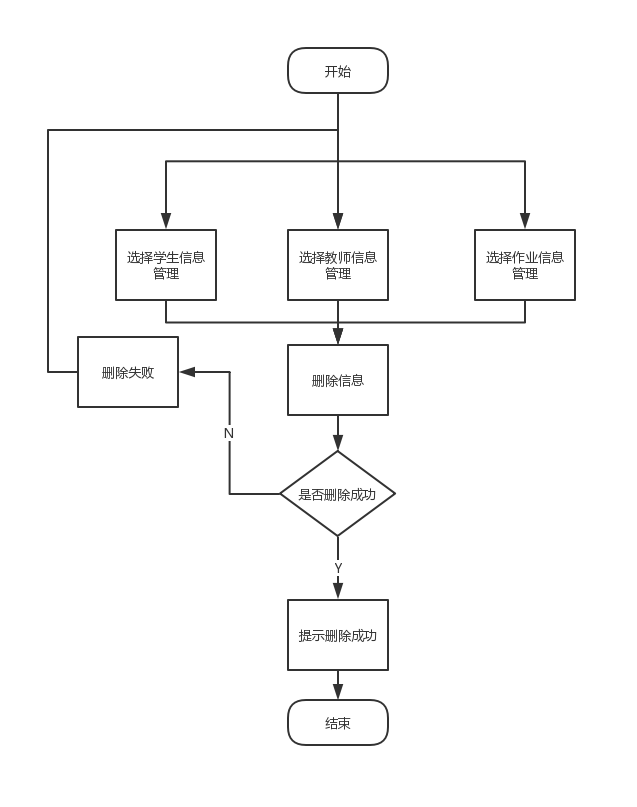


图 5-2-3 管理信息模块详细设计流程图

**5.2.2 控制层**

管理信息模块的控制层负责接受来自dcourse.jsp、dstudent.jsp、dteacher.jsp的信息，数据库操作（查询、删除），管理信息模块控制层列表见表5-2-4所示。

表 5-2-4 管理信息模块控制层列表

|  |  |  |  |
| --- | --- | --- | --- |
| 事件 | Action | 转移说明 | 出口 |
| 管理信息 | dcourse.jsp  dstudent.jsp  dteacher.jsp | SUCCESS | dcourse.jsp  dstudent.jsp  dteacher.jsp—删除成功，显示提示窗口。 |
| ERROR | dcourse.jsp  dstudent.jsp  dteacher.jsp ---删除失败，显示提示窗口。 |

dcourse.jsp的描述如下所示：

<%

String cName=(String)request.getParameter("cName");

Data data=new Data();

int intT=data.delete("DELETE FROM tb\_course WHERE cname='"+cName+"'");

if(intT>0){

out.print("<script>alert('删除成功！');document.location='course.jsp';</script>");

}else{

new Show().errorBox("删除失败！","错误信息");

}

%>

dstudent.jsp的描述如下所示：

<%

request.setCharacterEncoding("utf-8");

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

String strSname=(String)request.getParameter("sname");

System.out.println(strSname);

Data data=new Data();

int intT=data.delete("DELETE FROM tb\_student WHERE name='"+strSname+"'");

if(intT>0){

out.print("<script>alert('删除成功！');document.location='student.jsp';</script>");

}else{

new Show().errorBox("删除失败！","错误信息");

}

%>

dteacher.jsp的描述如下所示：

<%

request.setCharacterEncoding("utf-8");

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

String tName=(String)request.getParameter("tName");

System.out.println(tName);

Data data=new Data();

int intT=data.delete("DELETE FROM tb\_teacher WHERE name='"+tName+"'");

if(intT>0){

out.print("<script>alert('删除成功！');document.location='teacher.jsp';</script>");

}else{

new Show().errorBox("删除失败！","错误信息");

}

%>

**5.2.3 业务逻辑层**

管理信息模块的业务逻辑层主要是使用超链接，数据库操作（查询、删除），其中删除需页面传递参数。比如：管理信息模块业务逻辑层列表如图5-2-5所示。

表 5-2-5管理信息模块业务逻辑层列表

|  |  |  |  |
| --- | --- | --- | --- |
| 事件 | Service | 调用说明 | 出口 |
| 管理信息 | dcourse.jsp  dstudent.jsp  dteacher.jsp | 数据库操作 | 返回给Action |

dcourse.jsp的描述如下所示：

<%

String cName=(String)request.getParameter("cName");

Data data=new Data();

int intT=data.delete("DELETE FROM tb\_course WHERE cname='"+cName+"'");

if(intT>0){

out.print("<script>alert('删除成功！');document.location='course.jsp';</script>");

}else{

new Show().errorBox("删除失败！","错误信息");

}

%>

dstudent.jsp的描述如下所示：

<%

request.setCharacterEncoding("utf-8");

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

String strSname=(String)request.getParameter("sname");

System.out.println(strSname);

Data data=new Data();

int intT=data.delete("DELETE FROM tb\_student WHERE name='"+strSname+"'");

if(intT>0){

out.print("<script>alert('删除成功！');document.location='student.jsp';</script>");

}else{

new Show().errorBox("删除失败！","错误信息");

}

%>

dteacher.jsp的描述如下所示：

<%

request.setCharacterEncoding("utf-8");

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

String tName=(String)request.getParameter("tName");

System.out.println(tName);

Data data=new Data();

int intT=data.delete("DELETE FROM tb\_teacher WHERE name='"+tName+"'");

if(intT>0){

out.print("<script>alert('删除成功！');document.location='teacher.jsp';</script>");

}else{

new Show().errorBox("删除失败！","错误信息");

}

%>

**5.2.4 数据持久层**

管理信息模块的是数据持久层能对作业数据进行增加。管理信息模块数据持久层列表见表5-2-6所示。

表 5-2-6 管理信息模块数据持久层列表

|  |  |  |  |
| --- | --- | --- | --- |
| 事件 | DAO | 调用数据模型 | 说明 |
| 管理信息 | dcourse.jsp  dstudent.jsp  dteacher.jsp | 调用Student.java  Steacher.java  Scourse.Java | 对作业、学生、教师信息进行删除 |

**5.2.5 域模型层**

管理信息模块用到域模型层中Student.java、Steacher.java 、Scourse.Java，Student.java、Steacher.java 、Scourse.Java是一个公用域模型，在涉及到用户信息查询等操作时，就会调用到该模型，管理信息模块域模型层列表如表5-2-7所示。

表 5-2-7管理信息模块域模型层列表

|  |  |
| --- | --- |
| 域模型 | 描述 |
| Student.java  Steacher.java  Scourse.Java | 对学生、教师、作业信息的删除操作 |

Teacher.java主要属性与方法：

public class Teacher {

private String tname;

private String password;

private String name;

private String email;

private String specialty;

private String cname;

Student.java主要属性与方法：

public class Student {

private String name;

private String password;

private String email;

private String specialty;

private String class\_num;

Scourse.java主要属性与方法：

public class Scourse {

private String cname;

private String sname;

private String tname;

private int time;

private int score;

private String goal;

private String require;

private String other;

private String ptime;

private String etime;

# 6、公共部分模块详细设计

## 6.1 公共页面

每个用户登陆的界面顶端都是一致的。

|  |  |  |
| --- | --- | --- |
| 界面 | JSP | 功能描述 |
| 登陆进入界面顶端 | top.html | 登陆的进入界面的顶端导航栏 |