应用集成原理与工具集成实验报告

姓名： 刘开　　学号： 2013310200305　　班级： 计科 1403

实验环境：普通 PC 机，Windows 7 系统、MyEclipse10、jdk1.8.0_51、Apache Tomcat 7.0、MySQL Server 5.5、Navicat for MySQL。

实验目的：

采用 JSP+Servlet+JavaBean+JDBC 实现学生信息管理系统后台，实现管理员登陆对学生的基本信息和成绩信息进行管理，主要功能包括添加、修改和删除学生的基本信息及课程的基本信息；录入、修改和删除学生的成绩信息，对基本信息、成绩信息进行查询、排序及统计等操作，从而实现学生信息管理的自动化与计算机化。随着信息时代的到来，传统的手工登记管理的选课方式以及教务安排已经不能够适应现代高校的需求，更加及时、准确、全方位的网络化信息管理已经慢慢成为高校的必须工具。在管理以及业务日益复杂的现代生活，使用高科技、现代化的电脑自动化管理系统，来处理日益繁琐的学生信息管理，对现代高校已是必须具备的管理方式了。

采用 MVC 模式，目的就是实现 Web 系统的职能分工。

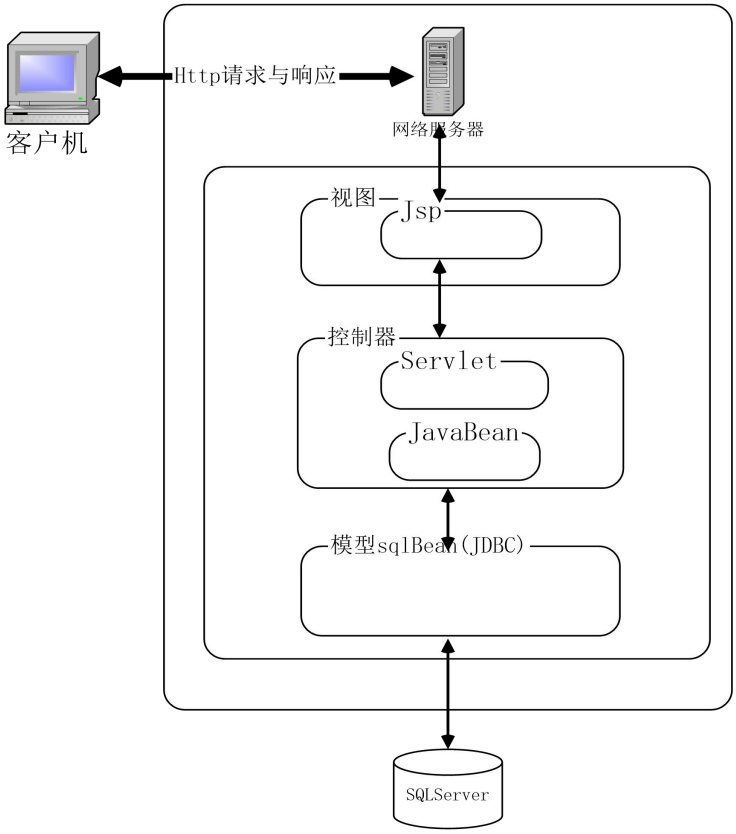
Model 层：实现系统的业务逻辑，即 javaBean 部分， JavaBean 可重复调用，需要接受用户的请求参数，进行相应的处理；

View 层：负责与用户交互，即在界面上展示数据对象给用户，即 html，jsp；

Control 层：Model 与 View 之间沟通的桥梁，它可以分派用户的请求并选择恰当的视图以用于显示，同时它也可以解释用户的输入并将它们映射为模型层可执行的操作，使用 Servlet 实现。

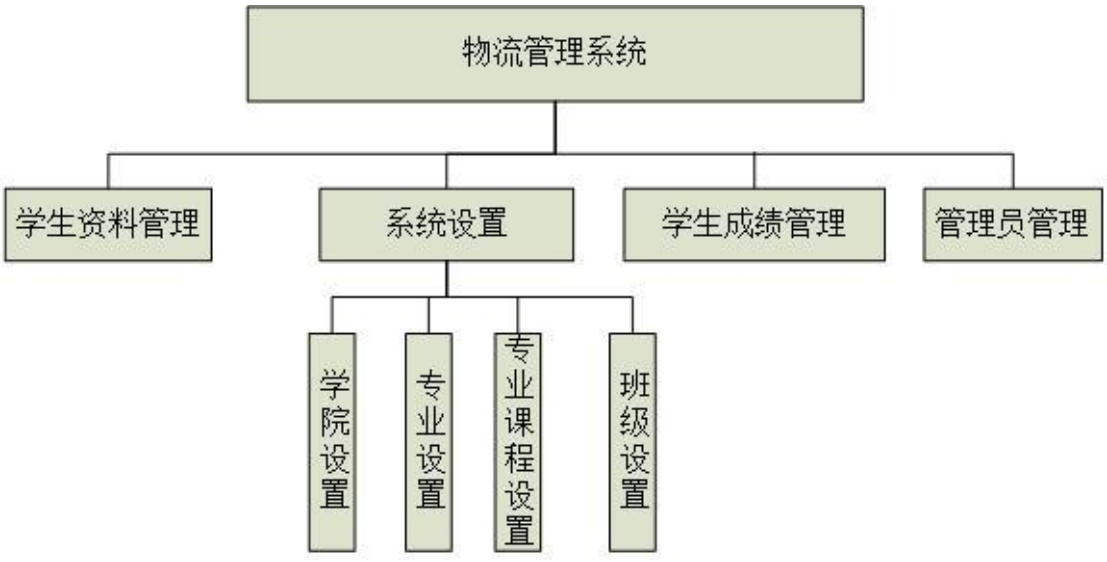
另外使用 DAO(数据库操作对象)是 JDBC 下常用的模式，保存数据时将 Java Bean 的属性拆分成正确的 SQL 语句，并保存到数据库中；读取数据时将数据从数据库中读取出来，并通过 setter 方法设置到 Java Bean 中。所有与数据库相关的操作都在 DAO 层实现，Servlet 或者 JSP 只操作 Java Bean 与 DAO 层，DAO

层至操作数据库。程序总体结构图如下：



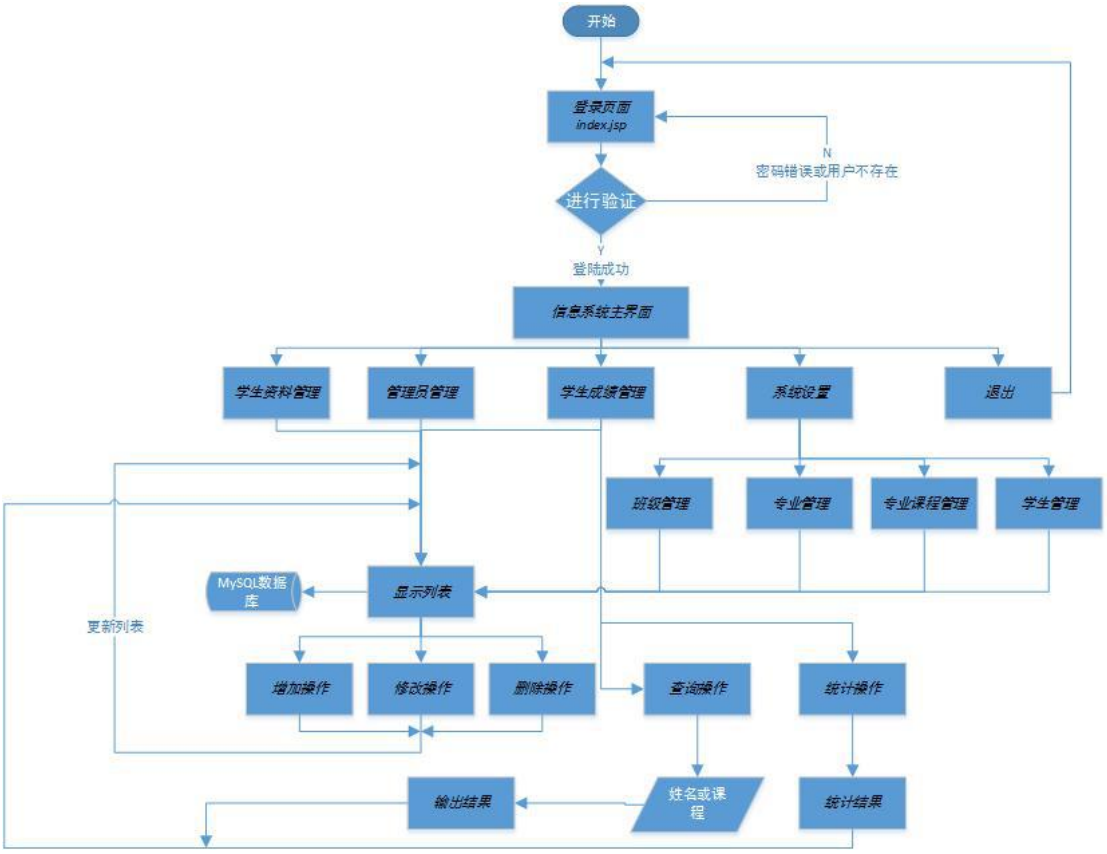
实验内容：

1. 进行系统模块设计



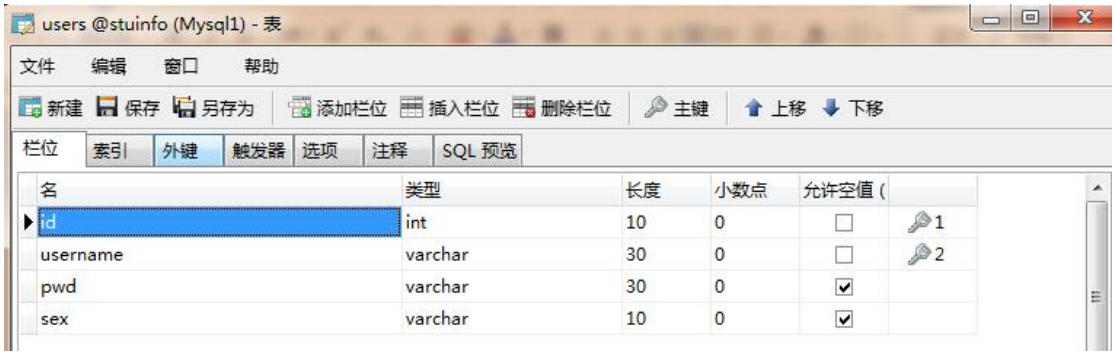
学生资料管理资料包括学生学号、学生所在院系、学生所在专业名称、学生姓名、学生所在班级、学生性别、学生年龄信息；管理员管理资料包括管理员ID、管理员名称管理员设置的登录密码、管理员的性别信息；学院管理资料包括学院编号、学院名称信息；专业管理资料包括专业编号、专业名称、专业所属学院信息；专业课程管理资料包括课程所属专业名称、课程名称、课程所属学院名称信息；学生成绩管理资料包括成绩编号、学生学号、学生姓名、课程名称以及课程分数信息，统计成绩资料包括课程名称、课程的最高分、课程的最低分以及、课程的平均分，查询成绩资料包括学生学号、学生姓名、课程名称以及课程分数信息；班级管理资料包括班级编号、班级名称、班级所属专业、班级所属学院信息；各类信息均使用 SQL 语句通过数据库以及统计计算得到。

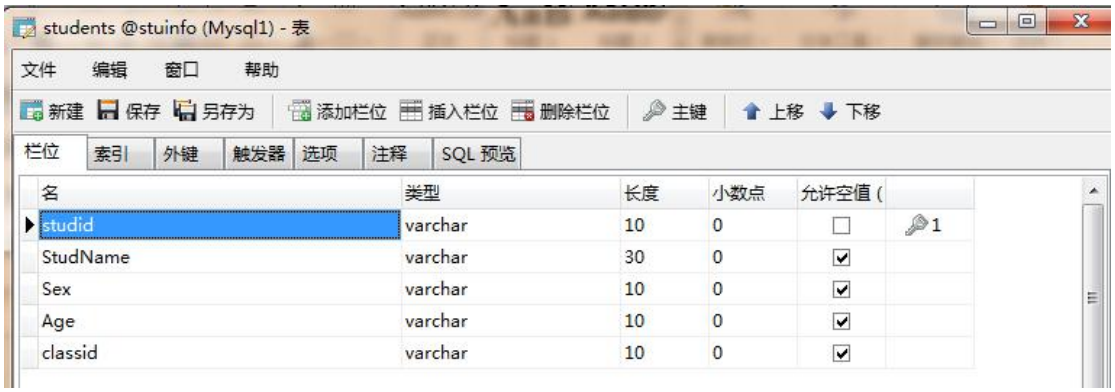
系统设计的大致流程图如下：

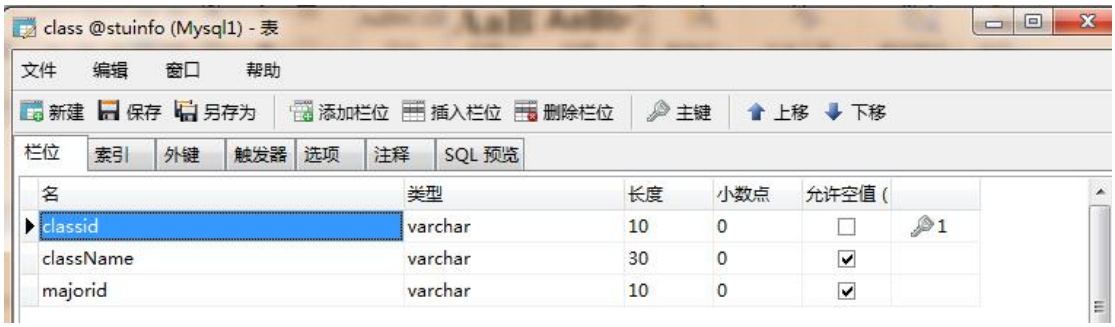


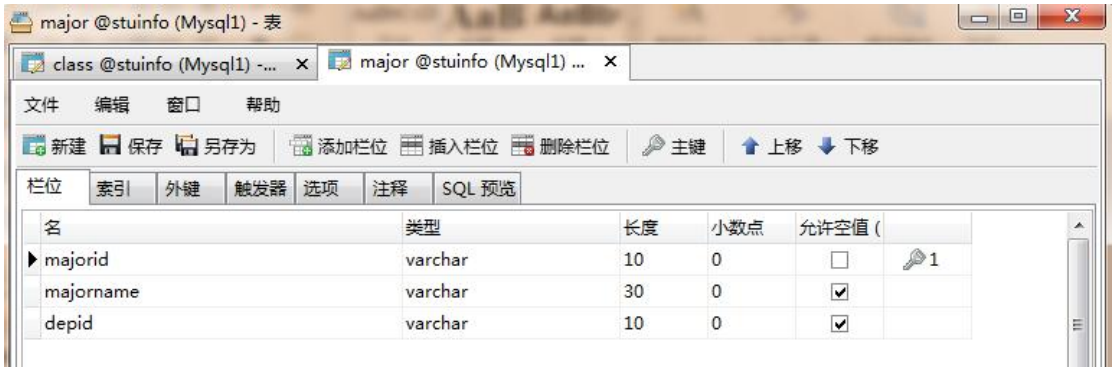
2.数据库设计

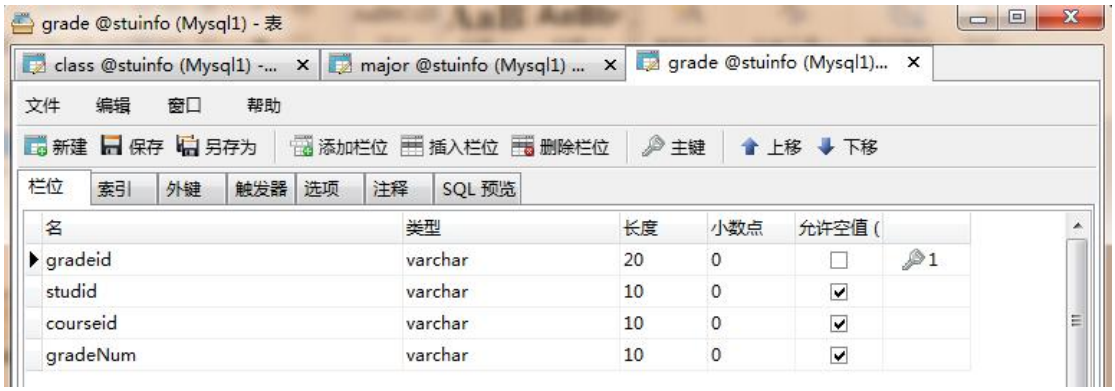
设计数据库名称为 stuinfo，包括 uesrs 表，students 表，class 表，major 表，grade 表，department 表，majorcourse 表，course 表，各表的设计信息如下，包括所有的属性以及主键等。

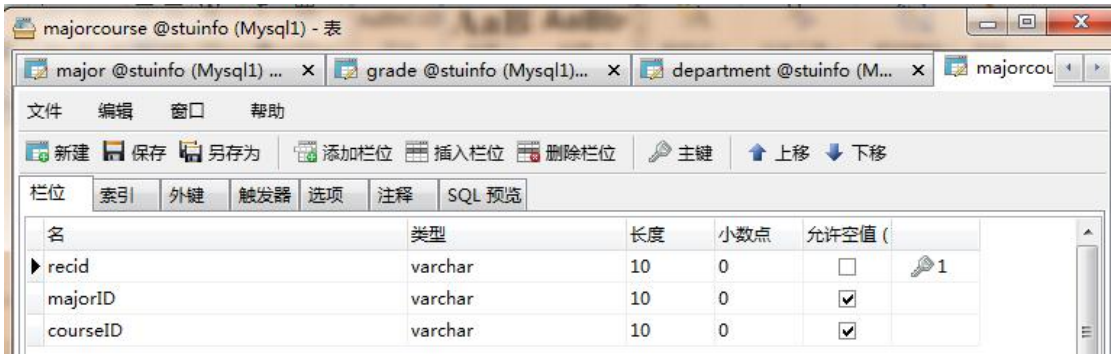


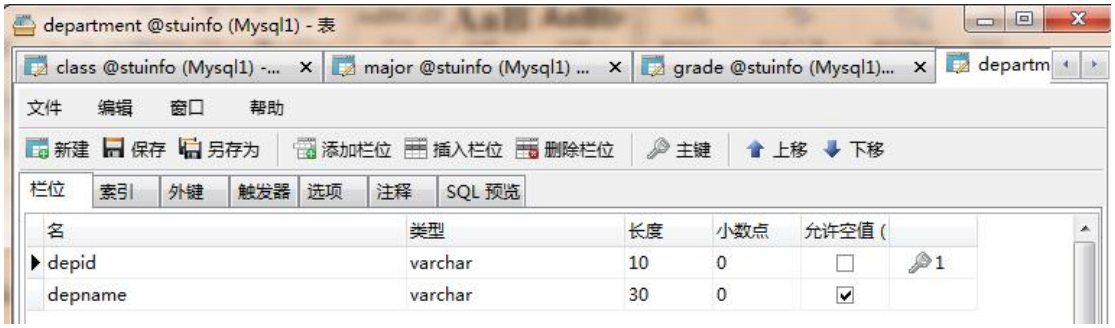


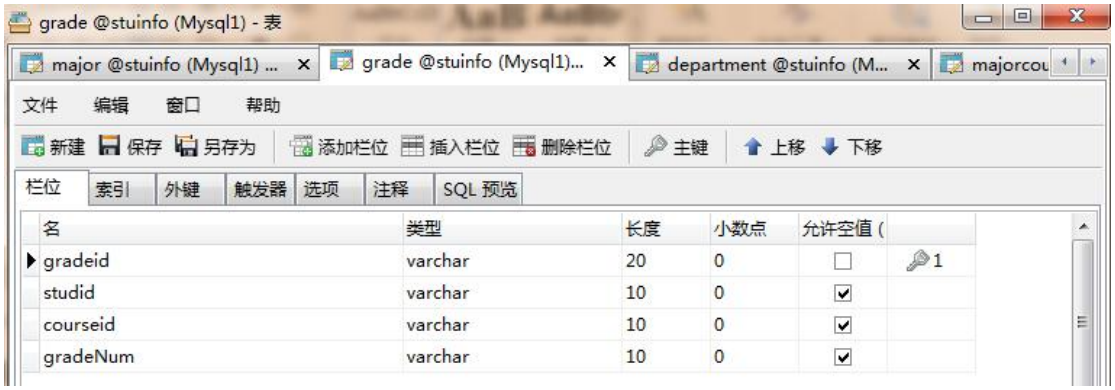












3. 系统主程序的设计

采用 JSP+Servlet+JavaBean 设计程序，其中包括 DAO 模式。

具体步骤：

1.设计包 testJdbc，DbManager.java 实现数据库连接部分；Pagination.java 实现分页功能，避免数据过多列举不方便；JDBC(Java 数据基础连接，即 Java Database Connectivity)是标准的 Java 访问数据库的 API。JDBC 定义了数据库的连接，SQL 语句的执行以及查询结果集的遍历等。将于数据库连接的部分封装起来，有助于其他程序的调用。通过该类可以大大的简化开发，在需要进行数据库连接时，只需常见该类的实例，并调用其中的方法就可以获得数据库连接对象和关闭数据库，不必再进行重复操作。另外，数据量大时需要进行分页显示，分页显示时只从数据库取出本页需要显示的记录，而不需要把所有记录都取出来。Pagination.java 实现方法为利用 LIMIT，先查询记录的总数，然后计算总页数，本页第一条记录在数据库中的行数，当前的页数将作为地址栏参数传递给服务器，还需要生成导航的信息，如第一页、上一页、下一页、最后一页等。

包含的主要代码

```
public class DbManager {

    /**
     * 获取默认数据库连接
     */
    public static Connection getConnection() throws SQLException {
        return getConnection("stuinfo", "root", "123456");
    }

    public static Connection getConnection(String dbName, String userName,
            String password) throws SQLException{
        String url = "jdbc:mysql://localhost:3306/" + dbName
                + "?characterEncoding=utf-8";
            try {
                Class.forName("com.mysql.jdbc.Driver");
                return DriverManager.getConnection(url, userName, password);
            } catch (Exception e) {}
             return null;
    }
    /* 带参数执行 SQL，返回影响的行数  */
    public static int executeUpdate(String sql, Object... params)
            throws SQLException {
        Connection conn = null;
```

```
            PreparedStatement preStmt = null;
            try {
                conn = getConnection();
                preStmt = conn.prepareStatement(sql);
                setParams(preStmt, params);
                return preStmt.executeUpdate();
            } finally {
                if (preStmt != null)
                    preStmt.close();
                if (conn != null)
                    conn.close();
            }
        }

}
    public static String getPagination(int pageNum, int pageCount,
                int recordCount, String pageUrl) {
            String url = pageUrl.contains("?") ? pageUrl : pageUrl + "?";
            if(!url.endsWith("?") && !url.endsWith("&")){   url += "&";}
            StringBuffer buffer = new StringBuffer();
            buffer.append("第 " + pageNum + "/" + pageCount + " 页 共 " + recordCount
                    + " 记录 ");
            buffer.append(pageNum == 1 ? " 第一页 " : " <a href='" + url
                    + "pageNum=1'>第一页</a> ");
            buffer.append(pageNum == 1 ? " 上一页 " : " <a href='" + url + "pageNum="
                    + (pageNum - 1) + "'>上一页</a> ");
            buffer.append(pageNum == pageCount ? " 下一页 " : " <a href='" + url
                    + "pageNum=" + (pageNum + 1) + "'>下一页</a> ");
            buffer.append(pageNum == pageCount ? " 最后一页 " : " <a href='" + url
                    + "pageNum=" + pageCount + "'>最后一页</a> ");
            buffer.append(" 到 <input type='text' ");
            buffer.append("  name='goto_input' ");
            buffer.append("  style='width:25px; text-align:center; '> 页 ");
            buffer.append(" <input type='button'");
            buffer.append("  name='goto_button' value='Go'>");
            buffer.append("<script language='javascript'>");
            buffer.append("function _enter(){");
            buffer.append("    if(event.keyCode == 13){");
            buffer.append("        _goto();");
            buffer.append("        return false;");
            buffer.append("    }");
            buffer.append("    return true;");
            buffer.append("} ");
            buffer.append("function _goto(){");
```

```
            buffer.append("    var numText = document.getElementsByName('goto_input')[0].value;");
            buffer.append("    var num = parseInt(numText, 10);");
            buffer.append("    if(!num){");
            buffer.append("         alert('页数必须为数字'); ");
            buffer.append("         return;");
            buffer.append("    }");
            buffer.append("    if(num<1 || num>" + pageCount + "){");
            buffer.append("         alert('页数必须大于 1，且小于总页数 " + pageCount + " ');   ");
            buffer.append("         return;");
            buffer.append("    }");
            buffer.append("    location='" + url + "pageNum=' + num;");
            buffer.append("}");
            buffer.append("document.getElementsByName('goto_input')[0].onkeypress = _enter;");
            buffer.append("document.getElementsByName('goto_button')[0].onclick = _goto;");
            buffer.append("</script>");
            return buffer.toString();
        }
```

2.设计包 myBean，包括 javaBean 类有 Users.java,Students.java,Class.java,Major.java,Department.java,MajorCourse.java。JavaBean 是使用 Java 语言开发的一个可重用的组件，在 JSP 的开发中可以使用 JavaBean 减少重复代码，使整个 JSP 代码的开发更简洁，另外 JavaBean 本身就是一个类，属于 Java 的面向对象编程。JSP 搭配 JavaBean 来使用，有很多的优点，包括可将 HTML 和 Java 代码分离，这主要是为了日后维护的方便。如果把所有的程序代码（HTML 和 Java）写到 JSP 页面中，会使整个程序代码又多又复杂，造成日后维护上的困难。可利用 JavaBean 的优点将日常用到的程序写成 JavaBean 组件，当在 JSP 要使用时，只要调用 JavaBean 组件来执行用户所要的功能，不用再重复写相同的程序，这样以来也可以节省开发所需的时间。

部分主要代码（省略 getter 与 setter 方法）

```
public class Users {
    private Integer id;
    private String name;
    private String pwd;
    private String sex;
}
public class Studens {
```

```
    private String id;
    private String age;
    private String sex;
    private String studName;
    private String className;
}
public class Class {
    private String classID;
    private String className;
    private String majorName;
    private String majorId;
}
```

3.设计包 myDAO，采用 DAO(数据库操作对象)模式对 myBean 进行全部的数据库操作。其中包括的类有 UsersDAO.java,StudentsDAO.java,ClassDAO.java, MajorDAO.java,DepartmentDAO.java,MajorCourseDAO.java。DAO 实现了所有的用户操作，如添加记录、删除记录、更新记录以及查询记录等。比如外部可以使用 UsersDAO 直接操作 Users 对象，JavaBean 与 DAO 建立成功后，Servlet 与 JSP 的编程就会非常方便，它们只与 avaBean 与 DAO 进行交流，而不会有 JDBC 层的 Connection、Statement、ResultSet、SQL 语句等。

部分主要代码，其他类似；

```
public class UsersDAO {
    /**
     * 插入一条 users 记录
     */
    public static int insert(Users users) throws Exception {
        String sql = "insert users (username,pwd,sex) values(?,?,?)";
        return DbManager.executeUpdate(sql, users.getName(),users.getPwd(),users.getSex());
    }
    /**
     * 检查 users 登陆
     */
    public static boolean check(Users users) throws Exception {
        String sql = "select * from users where username=? and pwd=?";
        Connection conn = null;
        PreparedStatement preStmt = null;
        ResultSet rs = null;
        try {
            conn = DbManager.getConnection();
            preStmt = conn.prepareStatement(sql);
            preStmt.setString(1, users.getName());
```

```
            preStmt.setString(2, users.getPwd());
            rs = preStmt.executeQuery();
            if (rs.next()) {//存在该用户
                return true;
            } else {  return false;}
        } finally {
            if (rs != null)    rs.close();
            if (preStmt != null)         preStmt.close();
            if (conn != null)       conn.close();
        }
    }
    /**
     * 更新密码
     */
    public static int updatePwd(Users users) throws Exception {
        String sql = "update users set pwd=? where id=?";
        return DbManager.executeUpdate(sql, users.getPwd(), users
                .getId());
    }

    /**
     * 删除 users
     *
     */
    public static void delete(String id) throws Exception {
        Connection conn = DbManager.getConnection();
        String sql = "delete from users where WHERE id in ("+id+")";
        PreparedStatement pst = conn.prepareStatement(sql);
        pst.executeUpdate();
        pst.close();
        conn.close();
    }
    /**
     * 计算总数
     */
    public static int total() throws Exception {
        String sql = "SELECT count(*) FROM users";
        return  DbManager.getCount(sql);
    }
    /**
     * 列出所有的 users
     */
    public static List<Users> listUsers(int startRecord,int pageSize) throws Exception {
        List<Users> list = new ArrayList<Users>();
```

```
        String sql = null;
        Connection conn = null;
        PreparedStatement preStmt = null;
        ResultSet rs = null;
        try {
             sql = "SELECT * FROM users LIMIT ?, ? ";
        conn = DbManager.getConnection();
        preStmt = conn.prepareStatement(sql);
        DbManager.setParams(preStmt, startRecord, pageSize);
        rs = preStmt.executeQuery();
        while (rs.next()) {
                  Users users = new Users();
                  users.setId(rs.getInt("id"));
                  users.setName(rs.getString("username"));
                  users.setPwd(rs.getString("pwd"));
                  users.setSex(rs.getString("sex"));
                  list.add(users);
             }
        } finally {
             if (rs != null)    rs.close();
             if (preStmt != null)    preStmt.close();
             if (conn != null)       conn.close();
        }
        return list;
    }
}
public class StudentsDAO {

    /**
     * 插入一条 studens 记录
     */
    public static int insert(Studens stu) throws Exception {
        String sql = "insert students values(?,?,?,?,?)";
        return DbManager.executeUpdate(sql, stu.getId(),stu.getStudName(),
stu.getSex(),stu.getAge(),stu.getClassId());
    }


    /**
     * 删除 studens
     *
     */
    public static int delete(String studId) throws Exception {
```

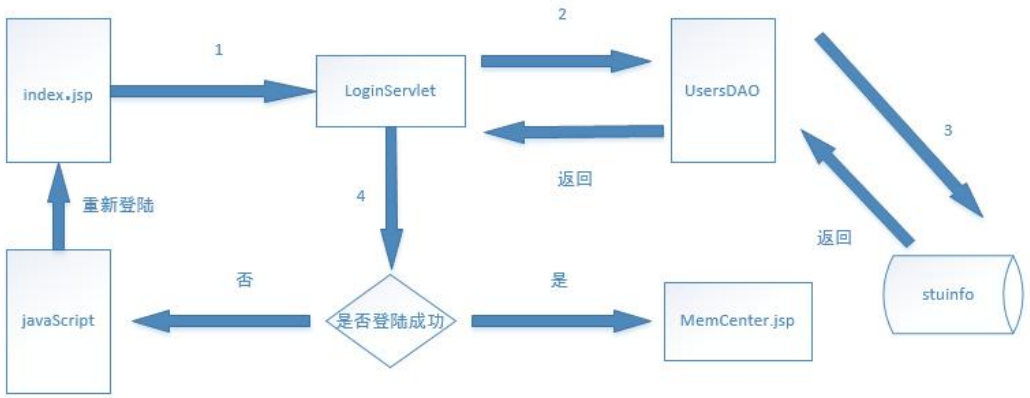
```
        String sql = "delete from students where studid=? ";
        return DbManager.executeUpdate(sql, studId);
    }

    /**
     * 按 id 查找是否存在
     */
    public static boolean find(String studId) throws Exception {
        String sql = "select * from students where studid= ? ";
        Connection conn = null;
        PreparedStatement preStmt = null;
        ResultSet rs = null;
        try {
            conn = DbManager.getConnection();
            preStmt = conn.prepareStatement(sql);
            preStmt.setString(1, studId);
            rs = preStmt.executeQuery();
            if (rs.next()) {// 存在该用户
                return true;
            } else {  return false;  }
        } finally {
            if (rs != null)    rs.close();
            if (preStmt != null)    preStmt.close();
            if (conn != null)conn.close();
        }
    }

    /**
     * 计算总数
     */
    public static int total() throws Exception {
        String sql = "SELECT count(*) from students s left outer join class c on s.classid=c.classid
left outer join major m on m.majorid=c.majorid left outer join department d on d.depid=m.depid";
        return DbManager.getCount(sql);
    }

    /**
     * 列出所有的 studens 信息
     */
    public static List<Studens> liststudens(int startRecord, int pageSize)
            throws Exception {
        List<Studens> list = new ArrayList<Studens>();
        String sql = null;
        Connection conn = null;
```

```
        PreparedStatement preStmt = null;
        ResultSet rs = null;
        try {
            sql = "select s.*,c.classname,m.majorname,d.depname from students s left outer join
class c on s.classid=c.classid left outer join major m on m.majorid=c.majorid left outer join
department d on d.depid=m.depid LIMIT ?, ? ";
            conn = DbManager.getConnection();
            preStmt = conn.prepareStatement(sql);
            DbManager.setParams(preStmt, startRecord, pageSize);
            rs = preStmt.executeQuery();
            while (rs.next()) {
                Studens studens = new Studens();
                studens.setId(rs.getString("StudID"));
                studens.setStudName(rs.getString("StudName"));
                studens.setSex(rs.getString("Sex"));
                studens.setAge(rs.getString("Age"));
                studens.setClassName(rs.getString("ClassName"));
                studens.setMajorname(rs.getString("majorname"));
                studens.setDepname(rs.getString("depname"));
                list.add(studens);
            }

        } finally {
            if (rs != null)    rs.close();
            if (preStmt != null) preStmt.close();
            if (conn != null) conn.close();
        }    return list;
    }
    /**
     * 按照 id 列出所有的 studens 个人信息
     */
    public static List<Studens> liststudens2(String stuID)
            throws Exception {
        List<Studens> list = new ArrayList<Studens>();
        String sql = "select * from students where StudID=?";
        Connection conn = null;
        PreparedStatement preStmt = null;
        ResultSet rs = null;
        try {
            conn = DbManager.getConnection();
            preStmt = conn.prepareStatement(sql);
            preStmt.setString(1, stuID);
            rs = preStmt.executeQuery();
            while (rs.next()) {
```

```
                Studens studens = new Studens();
                studens.setId(rs.getString("StudID"));
                studens.setStudName(rs.getString("StudName"));
                studens.setSex(rs.getString("Sex"));
                studens.setAge(rs.getString("Age"));
                studens.setClassId(rs.getString("classid"));
                list.add(studens);
            }

        } finally {
            if (rs != null)    rs.close();
            if (preStmt != null)    preStmt.close();
            if (conn != null)  conn.close();
        }return list;
    }
}
```

4.设计包 myServlet 包括所有需要的 Servlet，其中包括的类有 LoginServlet.java,UserDeleteServlet.java,UserEditServlet.java, UserAddServlet.java,StudentEditServlet.java,StudentDeleteServlet.java, StudentAddServlet.java,ClassAddServlet.java,ClassEditServlet.java,ClassDeleteServlet.java,GradeStatServlet.java,ConditionServlet.java 等。Servlet 是 MVC 模式里面的 C(控制器)，控制器接受用户的输入并调用模型和视图去完成用户的需求。所以当单击 Web 页面中的超链接和发送 HTML 表单时，控制器(比如 servlet)本身不输出任何东西和做任何处理。它只是接收请求并决定调用哪个模型构件去处理请求，然后确定用哪个视图来显示模型处理返回的数据。比如实现登陆功能时使用 LoginServlet.java，主要修改 loginServlet.java 文件里面的 doPost 方法。

页面流向图为



部分主要代码，其他类似；

```
public class LoginServlet extends HttpServlet {
        private static final long serialVersionUID = 73811691340165566647L;
         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");
            //设置响应所采用的编码方式
            response.setCharacterEncoding("utf-8");
            String uname=request.getParameter("userID");
            String passwd=request.getParameter("pwd");
            PrintWriter out = response.getWriter();
            Users user=new Users();
              user.setName(uname);
              user.setPwd(passwd);
            boolean bool;
              try {
                   bool = UsersDAO.check(user);
                 if(bool){
                    RequestDispatcher rd=request.getRequestDispatcher("MemCenter.jsp");
                    rd.forward(request,response);
                     }else{
                          out.println("<script>");//输出 script 标签
                          out.println("alert('用户名或密码错误');");//js 语句：输出 alert 语句
                          out.println("history.back();");//js 语句：输出网页回退语句
                          out.println("</script>");//输出 script 结尾标签
                    }

                } catch (Exception e) {
                    e.printStackTrace();
                }
         }
    }
public class UserDeleteServlet extends HttpServlet {
          private static final long serialVersionUID = 1L;

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

```
            response.setContentType("text/html;charset=UTF-8");
            PrintWriter out = response.getWriter();
            System.out.println(request.getParameterValues("info"));
            String[] selectdelete = request.getParameterValues("info");
            String ids = "";
            for (int i = 0; i < selectdelete.length; i++) {
                ids += "'" + selectdelete[i] + "'";
                if (i != selectdelete.length - 1)
                    ids += ",";
                try {
                    UsersDAO.delete(ids);
                    out.print("<script language='javascript'>alert('删除成功!
');window.location.href='UserList.jsp';</script>");
                    out.flush();
                    out.close();
                } catch (Exception e) { }
            }
        }

        protected void doGet(HttpServletRequest request,
                HttpServletResponse response) throws ServletException, IOException {
            processRequest(request, response);
        }

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

    }
    public class UserEditServlet extends HttpServlet {
        private static final long serialVersionUID = 7381169134016556647L;

        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");
            // 设置响应所采用的编码方式
            response.setCharacterEncoding("utf-8");
            PrintWriter out = response.getWriter();
```

```
            try {
                System.out.println(request.getParameter("id") + ""
                        + request.getParameter("oldPwd"));
                if ((UsersDAO.find(new Integer(request.getParameter("id")),
                        request.getParameter("oldPwd")))) {
                    Users user = new Users();// 实例化一个管理员用户
                    user.setId(new Integer(request.getParameter("id")));
                    user.setPwd(request.getParameter("pwd"));
                    UsersDAO.updatePwd(user);// 更新
                    RequestDispatcher rd = request
                            .getRequestDispatcher("UserList.jsp");
                    rd.forward(request, response);
                } else {
                    out.println("<script>");// 输出 script 标签
                    out.println("alert('旧密码错误');");// js 语句：输出 alert 语句
                    out.println("history.back();");// js 语句：输出网页回退语句
                    out.println("</script>");// 输出 script 结尾标签

                }

            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }
```

注意需要在 Web.xml 中配置 Servlet

部分配置代码

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="3.0"
    xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd">
  <display-name></display-name>
  <servlet>
   <servlet-name>LoginServlet</servlet-name>
   <servlet-class>myServlet.LoginServlet</servlet-class>
 </servlet>

 <servlet-mapping>
   <servlet-name>LoginServlet</servlet-name>
   <url-pattern>/LoginServlet</url-pattern>
 </servlet-mapping>
   <servlet>
```

```
  <servlet-name>UserEditServlet</servlet-name>
  <servlet-class>myServlet.UserEditServlet</servlet-class>
 </servlet>

 <servlet-mapping>
  <servlet-name>UserEditServlet</servlet-name>
  <url-pattern>/UserEditServlet</url-pattern>
 </servlet-mapping>
```

5. jsp 代码编写，设计 head. jsp 与 bottom. jsp，方便其他 jsp 文件进行添加，统一页面的头与尾，head. jsp 主要包含系统的导航信息，点击进行模块的选择，链接到相应页面。

head. jsp 代码如下

```
<%@ page contentType="text/html; charset=utf-8" %>
<html>
<head>
<link href="css/mycss.css" rel="stylesheet" type="text/css">

<title>
学生信息管理系统
</title>
</head>
<body bgcolor="#ffffff">
<div id="head">
<table id="headTable">
  <tr>
    <td width="128"><img src="images/top.jpg" width="241" height="73"></td>
    <td width="619"><table id="innerTable">
      <tr>
        <td width="101"><div align="center"><a href="StudList.jsp"> [学生资料管理]</a></div></td>
        <td width="100"><div align="center"><a href="GradeList.jsp"> [学生成绩管理]</a></div></td>
        <td width="75"><div align="center"><a href="SysIndex.jsp">[系统设置]</a></div></td>
        <td width="84"><div align="center"><a href="UserList.jsp">[管理员管理]</a></div></td>
        <td width="47"><div align="center"><a href="index.jsp">[退出]</a></div></td>
      </tr>
    </table></td>
  </tr>
</table>
</div>
<br />
<hr width="768" size="1">
</body>
```

```
</html>
```

举例管理员管理代码

UserList.jsp 代码

```
<%@ page contentType="text/html; charset=utf-8" language="java"
    errorPage=""%>
<jsp:directive.page import="testJdbc.Pagination" />
<jsp:directive.page import="myBean.Users" />
<jsp:directive.page import="myDAO.UsersDAO" />
<jsp:directive.page import="java.util.List" />
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<link href="css/mycss.css" rel="stylesheet" type="text/css">
</head>

<body>
    <%
        if (session.getAttribute("username") == null
        || session.getAttribute("username").equals("")) {
            response.sendRedirect("index.jsp");    }
    %>
    <jsp:include flush="true" page="head.jsp" />
    <br />
    <%
        final int pageSize = 8; // 一页显示 8 条记录
        int pageNum = 1; // 当前页数
        int pageCount = 1; // 总页数
        int recordCount = 0; // 总记录数
            try {
            // 从地址栏参数取当前页数
            pageNum = Integer.parseInt(request.getParameter("pageNum"));
        } catch (Exception e) {
        }
        try {
            recordCount =UsersDAO.total();
            // 计算总页数
            pageCount = (recordCount + pageSize - 1) / pageSize;
            // 本页从 startRecord 行开始
            int startRecord = (pageNum - 1) * pageSize;
             List<Users> listUsers=UsersDAO.listUsers(startRecord, pageSize);
    %>
    <p class="measureList">管理员资料列表</p>
    <div class="list">
```

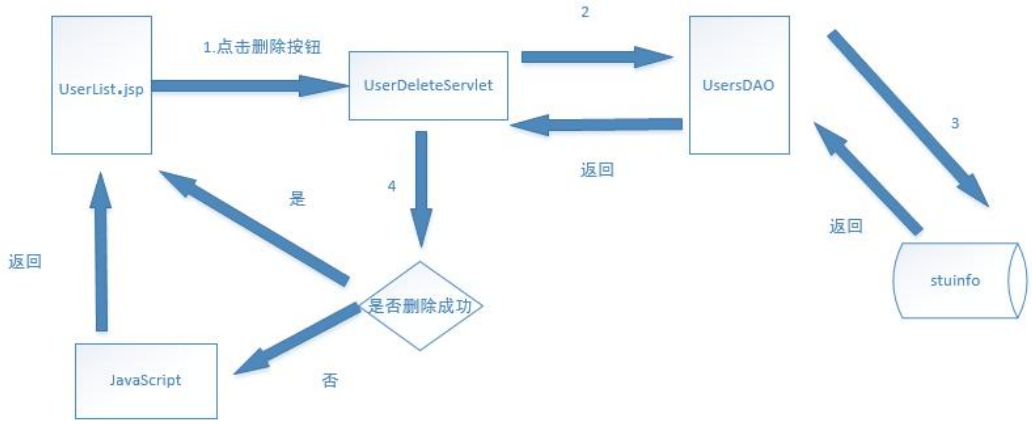
```
        <table>
            <tr>
                <th>
             <form action="<%= request.getContextPath() %>/UserDeleteServlet" method=post>
                        全选/反选<br /><input type="checkbox" id="checkall" name="checkall"
onclick="checkAll(checkall)" /><br /> <input type="submit" value="删除"  onclick="return confirm('
确定删除选择项？')" align="left">
                        </form>
                    </th>
                <th>ID</th>
                <th>管理员名称</th>
                <th>登录密码</th>
                <th>性别</th>
                <th>修改密码</th>
                <th>删除资料</th>
            </tr>
            <%
                String user_name = "";
                    String id = "";
                    for(Users user:listUsers){
                        user_name = user.getName();
                        id=user.getId()+"";
            %>
            <tr>
<td align ="center"><input type="checkbox"  name="info" value= " +id+ " /></td>

                <td><%=id%></td>
                <td><%=user_name%></td>
                <td><%=user.getPwd()%></td>
                <td><%=user.getSex()%></td>
                <td><a href="ChangePwd.jsp?id=<%=id%>">修改密码</a></td>
                <td><a href="UserDel.jsp?id=<%=id%>" onclick="return check();">删除资料</a>
                </td>
            </tr>
            <%
                }
            %>
        </table>
    </div>

    <br />
    <div class="add">
        <a href="UserAdd.jsp">新增管理员</a>
    </div>
```

```
    <table align=right>
        <tr>
            <td><%=Pagination.getPagination(pageNum, pageCount, recordCount,
request.getRequestURI())%>
            </td>
        </tr>
    </table>

    <%
        }catch(Exception e){}
    %>
    <p> </p>
    <p> </p>
    <jsp:include flush="true" page="bottom.jsp" />
</body>
</html>
<script type="text/javascript">
    function check() {
        if (confirm("删除确认")) {
            return true;
        }
        return false;
    }
</script>
<script type="text/javascript">
        function checkAll(checkall) {
            arr = document.getElementsByName("info");
            if (checkall.checked == true) {
                for(i=0;i<arr.length;i++){
                    arr[i].checked = true;
                }
                }else{
                    for(i=0;i<arr.length;i++){
                        if((arr[i]).checked==false){
                            arr[i].checked = true;
                        }else
                        {arr[i].checked = false; }
                    }
            }
    }

</script>
```

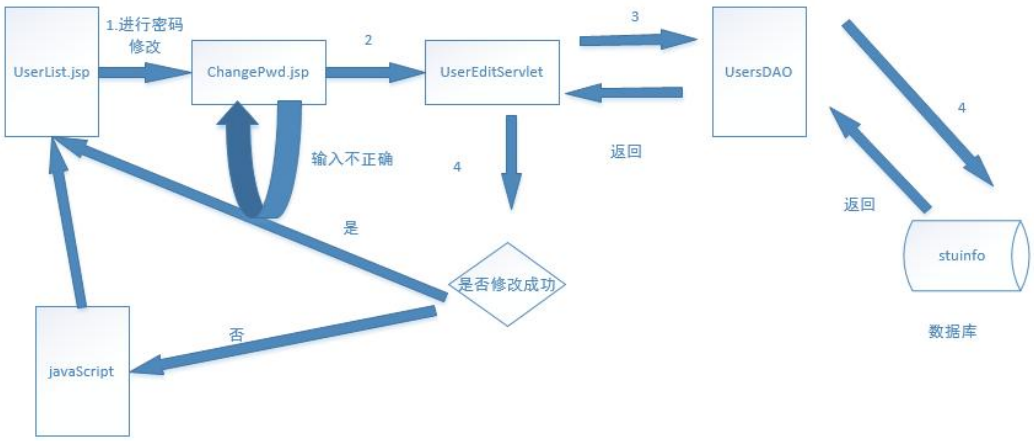
其他部分页面流向图如下

管理员管理操作

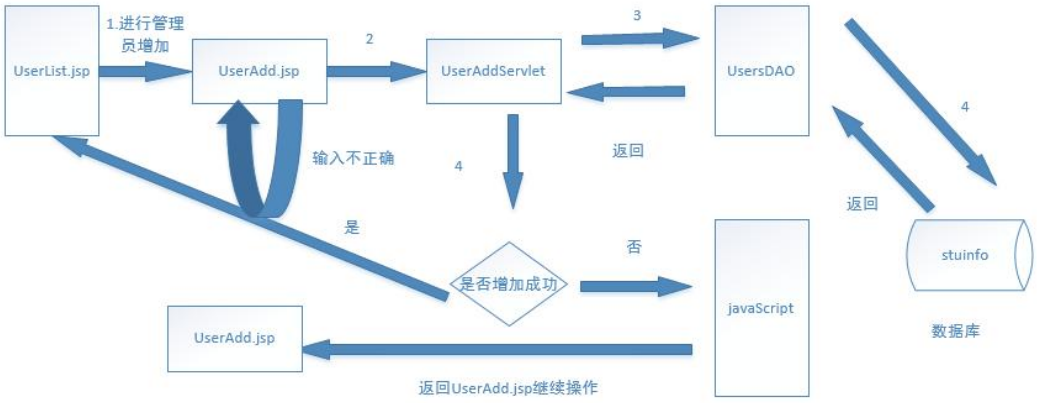
进行管理员删除操作时，页面流向图为



进行管理员密码修改操作时，页面流向图为

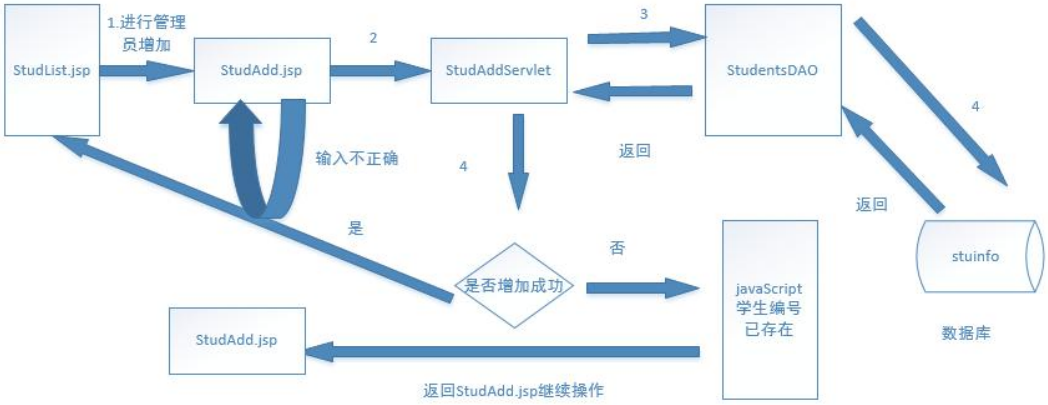


进行增加管理员操作时，页面流向图为



学生资料管理操作

进行增加学生操作时，页面流向图为

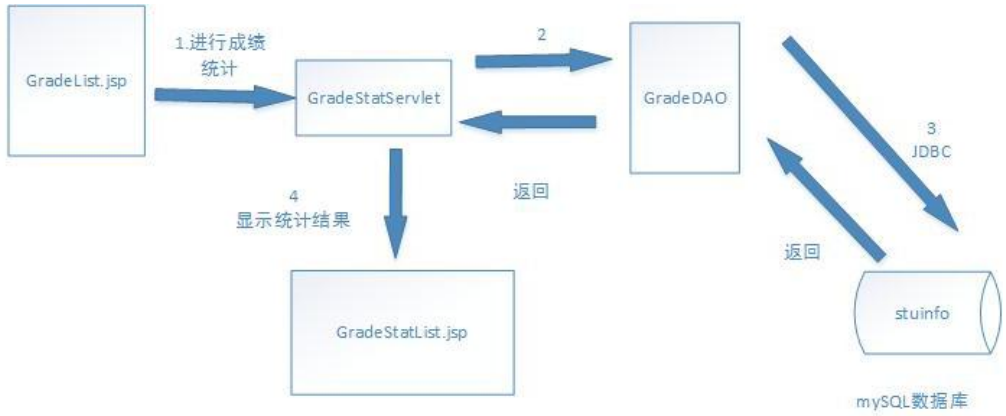


进行修改学生资料以及删除学生操作时，页面流向图类似上图，注意删除学生资料时应先删除有关的成绩等信息。

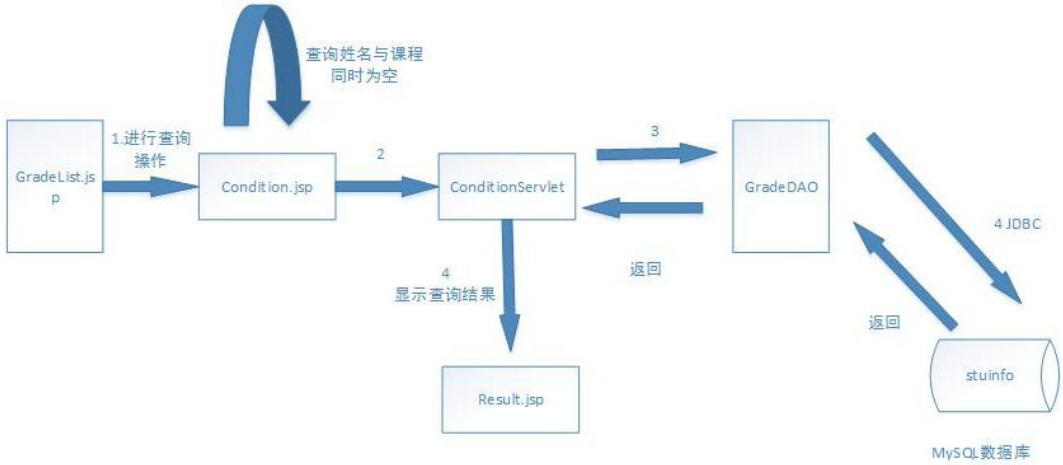
系统设置中的学院设置、专业设置、专业课程设置以及班级设置均可进行增加、修改、删除操作，页面流向图均类似上图。

学生成绩资料管理操作

进行统计成绩操作时，页面流向图为



进行查询成绩操作时，页面流向图为



系统中成绩管理操作结果

点击查询得到查询结果如下

查询结果如下

学生学号	学生姓名	课程名称	课程分数
2014001	小赵	C语言	89

学生信息管理系统

当查询条件中课程项选择空时

点击查询得到查询结果如下

查询结果如下

学生学号	学生姓名	课程名称	课程分数
2014001	小赵	C语言	89
2014001	小赵	Java语言	88
2014001	小赵	数据结构	78
2014001	小赵	算法分析	66
2014001	小赵	操作系统	78

实验结果：

登陆页面，管理员登陆输入注册名以及密码进行登录，登陆失败会弹窗提醒。

登陆成功进入主界面，可以点击导航进行各类信息的管理

管理员资料页面

修改管理员密码页面

增加管理员页面 ，注意管理员名称不能重复，重复会弹窗提醒。

打开学生资料管理

对学生资料进行基本操作，新增和修改资料

打开学生成绩资料管理

点击成绩统计

系统设置页面，其中包括学院设置、专业设置、专业课程设置以及班级设置，进行点击选择相应的设置。

系统设置主界面

- 学院设置：
- 专业设置：
- 专业课程设置：
- 班级设置：

学生信息管理系统

打开专业资料管理

专业资料列表

全选 反选 删除	专业编号	专业名称	所属学院	修改资料	删除资料
	1	计算机科学与技术	信息学院	修改资料	删除资料
	2	生物信息	信息学院	修改资料	删除资料
	3	信息科学与技术	理学院	修改资料	删除资料
	6	应用化学	理学院	修改资料	删除资料

新增专业

打开专业课程资料管理

专业课程资料列表

全选 反选 删除	专业名称	课程名称	所属学院	修改资料	删除资料
	计算机科学与技术	C语言	信息学院	修改资料	删除资料
	计算机科学与技术	Java语言	信息学院	修改资料	删除资料
	信息科学与技术	操作系统	理学院	修改资料	删除资料
	信息科学与技术	Java语言	理学院	修改资料	删除资料
	计算机科学与技术	数据结构	信息学院	修改资料	删除资料
	计算机科学与技术	算法分析	信息学院	修改资料	删除资料
	计算机科学与技术	操作系统	信息学院	修改资料	删除资料
	生物信息	生物	信息学院	修改资料	删除资料

新增专业课程

第 1/2 页 共 11 记录 第一页 上一页 下一页 最后一页 到 页 Go

学生信息管理系统

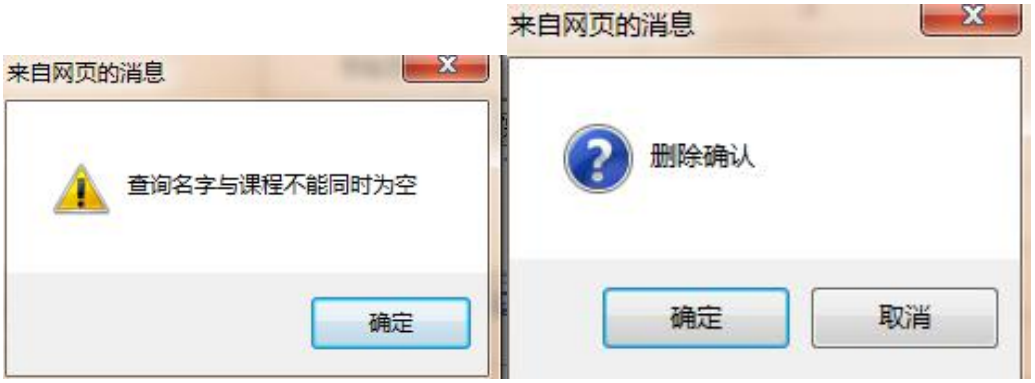
打开班级资料管理

打开学院资料管理

对于学生专业资料以及专业课程进行新增操作。

部分 JavaScript 提示信息


来自网页的消息
学生编号已存在，请选择其它编号
确定
来自网页的消息
旧密码错误
确定


来自网页的消息
查询名字与课程不能同时为空
确定
来自网页的消息
删除确认
确定
取消

实验总结：

通过本次实验加深了对 java web 编程开发的认识与了解，对于 MVC 模式有了更深刻的理解，并且通过实验提高了自己分析问题以及解决问题的能力。经过这段时间的实验，过程中查询了许多相关资料，最终实现了预计系统的一些基本功能，实验过程中不但加深了理论知识，而且通过把理论应用于实践，在实践中发现并解决问题，真正做到了学以致用。我在课程设计期间学到了很多知识，比如 jsp 知识以及网页制作方面的知识。实验过程中也遇到了很多的错误，比如对于数据库部分，操作 MySQL 时遇到乱码问题，有时显示时出现乱码，也有时进行添加用户时数据库中出现乱码，最终查询了资料才得以解决，解决办法为统一使用 UTF-8 编码，在 JDBC 连接中，对于 URL 指定 UTF-8 编码，另外对 request 也指定编码方式，使 JSP 页面、request 编码、response 编码、数据库编码保持一致。另外，实验一开始时并没有使用 JSP+Servlet+JavaBean+JDBC 方式，直接使用 JSP，实验过程听到老师讲解才发现需要改正，花了一段时间进行更改，但是过程中收获颇多，对于 MVC 模式有了很深的了解，这对以后会有很大的帮助，还有由于对 javascript 的使用不够熟练，还有很多地方不够了解，但也有所应用，对于前台的美工 CSS 样式运用较少，系统界面也不是太美观。最后虽然大致已经按照要求将系统实现，但还是存在一些问题，系统实用性并不强，功能太弱，只允许管理员进行登录，学生以及老师不能登录，对于学院以及专业管理功能过于冗余，学生成绩管理信息查询等功能实现也不是很好，功能不灵活。另外编写的程序代码过于分散，比如对于一个管理员把所有的操作全部分开实现，可以使用一个文件进行操作，因此还需要后期进行改进。