

天津理工大学实验报告

学院（系）名称：计算机科学与工程学院

姓名		学号		专业	计算机科学与技术
班级		实验项目	JSP、Servlet 及 JDBC 应用开发		
课程名称		Web 应用程序设计与开发		课程代码	0668026
实验时间		2018 年 5 月 22 日 中午 5、6 节		实验地点	7-215
考核标准	实验准备（实验目的/工具熟悉情况）10 分	实验过程（实验方案可行性及步骤完整性）40 分	实验报告（实验内容丰富度与格式清晰度）30 分	实验结果（结论正确性以及分析合理性）20 分	成绩
考核内容	评价实验目的是否明确，实验工具是否清晰了解以及熟悉情况	<input type="radio"/> 可行，完整 <input type="radio"/> 可行，不完整 <input type="radio"/> 不可行，不完整	<input type="radio"/> 丰富，清晰 <input type="radio"/> 较丰富，较清晰 <input type="radio"/> 丰富，不清晰 <input type="radio"/> 不丰富，不清晰	<input type="radio"/> 结论正确，分析合理 <input type="radio"/> 结论正确，分析不充分 <input type="radio"/> 结论不正确，分析不合理	教师签字：

一、实验目的

掌握使用 JDBC 访问数据库，学会使用 JSP 的内置对象的使用，学会 Servlet 和 JavaBean 等技术的应用。

二、实验环境

Windows 操作系统，Tomcat，MyEclipse，HBuilder，记事本。

三、实验要求

1. 实现系统的用户登录，要求：

- （1）创建登录表，并利用此表实现系统的登录功能；
- （2）登录成功要求能够保存或维护用户信息，以便能够被其他界面使用用户信息。

2. 实现对单表的信息的添加功能，要求：

(1) 建立数据库的表结构，至少包含 4 个以上字段；

(2) 利用表单的提交功能。

(3) 利用 Servlet 处理，充分考虑编码及封装。

3. 对数据库的表的信息查询功能，要求：

(1) 利用 JavaBean 实现，并将表中的信息按行进行显示。

(2) 建立数据库的表结构，至少包含 4 个以上字段；表的信息可以是学生信息、图书信息、商品信息等。

(3) 实现时充分考虑代码的封装。

四、实验过程记录（源程序、测试用例、测试结果及心得体会等）

1. 实现系统的用户登录

(1) 创建登录表，并利用此表实现系统的登录功能；。

表结构如下：

名称	数据类型
userid	INT
username	VARCHAR
password	VARCHAR

(2) 登录成功要求能够保存或维护用户信息，以便能够被其他界面使用用户信息。

在 Servlet 中使用 Session 对象对登录状态进行存储，以便在其他界面调用验证
后端代码：

```
response.setContentType("text/html;charset=utf-8");
String username = WebUtil.getParameter(request,"username");
String password = WebUtil.getParameter(request,"password");
PrintWriter out = response.getWriter();

HashMap<String, String> hashMap = new HashMap<String, String>();
hashMap.put("username", username);
hashMap.put("password", password);

if(DBUtil.getDataCount("user",hashMap)==1) {
    HttpSession session = request.getSession();
    session.setAttribute("username",username); //用户名
    session.setAttribute("loginState","1"); //登录状态
    response.sendRedirect("./loginCheck.jsp");
}
else {
    HttpSession session = request.getSession();
```

```
session.setAttribute("loginState","0"); //登录状态
response.sendRedirect("./loginCheck.jsp");
```

}

演示：



图 1-1 登录前提示并自动跳转

2. 实现对单表的信息的添加功能

(1) 建立数据库的表结构

表结构如下：

名称	数据类型
sno	INT
sname	VARCHAR
age	VARCHAR
politicalstate	VARCHAR
birthday	VARCHAR
address	VARCHAR
phone	VARCHAR
institute	VARCHAR
demo	VARCHAR

(2) 利用表单的提交功能。

3. 对数据库的表的信息查询功能，要求：

- (1) 利用 JavaBean 实现，并将表中的信息按行进行显示。
- (2) 建立数据库的表结构，至少包含 4 个以上字段；表的信息可以是学生信息、图书信息、商品信息等。

(3) 实现时充分考虑代码的封装。

前端代码：

```
<form action="${pageContext.request.contextPath}/FormServlet" method="post">
```

```
    <table style="width: auto;" class=".table-bordered table table-striped table-  
hover table-bordered table table-condensed">
```

```
        <tr>
```

```
            <th>Key</th>
```

```
            <th>Value</th>
```

```
        </tr>
```

```
        <tr>
```

```
            <td>sno</td>
```

```
            <td>
```

```
                <input type="text" name="sno" id="sno">
```

```
            </td>
```

```
        </tr>
```

```
        <tr>
```

```
            <td>sname</td>
```

```
            <td>
```

```
                <input type="text" name="sname" id="sname">
```

```
            </td>
```

```
        </tr>
```

```
        <tr>
```

```
            <td>age</td>
```

```
            <td>
```

```
                <input type="text" name="age" id="age">
```

```
            </td>
```

```
        </tr>
```

```
        <tr>
```

```
            <td>politicalstate</td>
```

```
            <td>
```

```
                <input type="text" name="politicalstate" id="politicalstate">
```

```
            </td>
```

```
        </tr>
```

```
        <tr>
```

```
            <td>birthday</td>
```

```
            <td>
```

```
                <input type="text" name="birthday" id="birthday">
```

```
            </td>
```

```
        </tr>
```

```
        <tr>
```

```
            <td>address</td>
```

```
            <td>
```

```
                <input type="text" name="address" id="address">
```

```

        </td>
    </tr>
    <tr>
        <td>phone</td>
        <td>
            <input type="text" name="phone" id="phone">
        </td>
    </tr>
    <tr>
        <td>institute</td>
        <td>
            <input type="text" name="institute" id="institute">
        </td>
    </tr>
    <tr>
        <td>demo</td>
        <td>
            <input type="text" name="demo" id="demo">
        </td>
    </tr>
</table>
<div class="form-actions">
    <button type="submit" class="btn btn-primary">提交</button>
    <button type="button" class="btn">重置</button>
</div>
</form>

```

(3) 利用 Servlet 处理，充分考虑编码及封装。

后端代码：

```

request.setCharacterEncoding("UTF-8");
HttpSession session = request.getSession();
String op = request.getParameter("op");
System.out.print(op);
switch (op) {
    case "2Bean":
        response.setContentType("text/html;charset=utf-8");

        Student student_bean = new Student();
        student_bean.setAddress(request.getParameter("address"));
        student_bean.setAge(request.getParameter("age"));
        student_bean.setBirthday(request.getParameter("birthday"));
        student_bean.setDemo(request.getParameter("demo"));
        student_bean.setInstitute(request.getParameter("institute"));
        student_bean.setPhone(request.getParameter("phone"));

```

```
student_bean.setName(request.getParameter("sname"));
student_bean.setSno(Integer.parseInt(request.getParameter("sno")));
student_bean.setPoliticalstate(request.getParameter("politicalstate"));
```

```
session.setAttribute("studentBean", student_bean);
response.sendRedirect("./FormJavabean.jsp");
break;
```

case "2DB":

```
System.out.println("OK");
response.setContentType("text/html;charset=utf-8");
```

```
Student student_db = new Student();
System.out.println("OK2");
student_db.setAddress(request.getParameter("address"));
student_db.setAge(request.getParameter("age"));
student_db.setBirthday(request.getParameter("birthday"));
student_db.setDemo(request.getParameter("demo"));
student_db.setInstitute(request.getParameter("institute"));
student_db.setPhone(request.getParameter("phone"));
student_db.setName(request.getParameter("sname"));
student_db.setSno(Integer.parseInt(request.getParameter("sno")));
student_db.setPoliticalstate(request.getParameter("politicalstate"));
```

```
String sql = "INSERT INTO
student(sno,sname,age,politicalstate,birthday,address,phone,institute,demo) VALUES('";
sql+=student_db.getSno()+"', '"+student_db.getName()+"', '"+student_db.getAge()+"', '"+
student_db.getPoliticalstate()+"', '"+student_db.getBirthday()+"', '"+student_db.getAddre
ss()+"', '"+student_db.getPhone()+"', '"+student_db.getInstitute()+"', '"+student_db.getDem
o()+"'";
```

```
if(DBUtil.executeBatch(sql)){
    PrintWriter outPrintWriter = response.getWriter();
    outPrintWriter.println("表更新成功! ");
    System.out.println("表更新成功! ");
}
else{
    PrintWriter outPrintWriter = response.getWriter();
    outPrintWriter.println("表更新失败! ");
    System.out.println("表更新失败! ");
}
```

break;

default:

```
session.setAttribute("error", "参数错误");
```

```

        response.sendRedirect("./errorHandle.jsp");
        break;
    }
    private void insertStudentInfo(HttpServletRequest response, String sno) {
        try {
            String string = "";
            JSONObject jsonObject = new JSONObject();
            ArrayList<String> List = new ArrayList<>();
            String delete = "INSERT INTO student VALUES(?,?,?,?,?,?,?,?,?)";
            List.add(delete);
            boolean result = DBUtil.executeBatch(List);
            PrintWriter out = response.getWriter();
            response.setCharacterEncoding("utf-8");
            if(result){
                jsonObject.put("ret", "1");
            }
            else {
                jsonObject.put("ret", "0");
                jsonObject.put("reason", "数据库操作失败");
            }
            out.print(jsonObject);
            out.close();
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }
}

```

演示:

图 2-1 表单录入

Key	Value
sno	20152100
sname	王明d
age	21
politicalstate	共青团员
birthday	1997-04-18
address	天津理工大学
phone	13137127027
institute	计算机科学与工程学院
demo	学习委员

图 2-2 表单确认

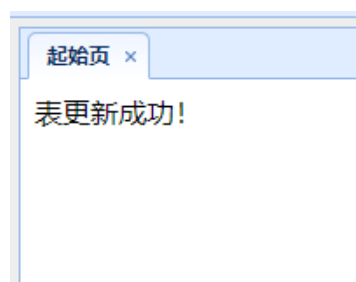


图 2-3 表单更新成功提醒

心得体会:

通过本次实验，我加深了对 JSP/Servlet 以及 JDBC 开发的理解。JSP 全名为 Java Server Pages，中文名叫 java 服务器页面，其根本是一个简化的 Servlet 设计。Servlet (Server Applet) 是 Java Servlet 的简称，称为小服务程序或服务连接器，用 Java 编写的服务器端程序，主要功能在于交互式地浏览和修改数据，生成动态 Web 内容。JDBC (Java DataBase Connectivity, java 数据库连接) 是一种用于执行 SQL 语句的 Java API，可以为多种关系数据库提供统一访问，它由一组用 Java 语言编写的类和接口组成。通过对以上技术的运用与实践，我可以初步做成基本的 WEB 前后端交互页面，并实现了与数据库的连接。在今后的学习中，我将继续对 WEB 应用程序进行了解与分析，实现更多功能。

天津理工大学实验报告

学院（系）名称：计算机科学与工程学院

姓名		学号		专业	计算机科学与技术
班级		实验项目	实验 3：扩展的 JavaScript 框架应用		
课程名称		Web 应用程序设计与开发		课程代码	0668026
实验时间		2018 年 5 月 25 日 1、2 节 2018 年 5 月 29 日 中午 5、6 节		实验地点	7-215
考核标准	实验准备（实验目的/工具熟悉情况）10 分	实验过程（实验方案可行性及步骤完整性）40 分	实验报告（实验内容丰富度与格式清晰度）30 分	实验结果（结论正确性以及分析合理性）20 分	成绩
考核内容	评价实验目的是否明确，实验工具是否清晰了解以及熟悉情况	<input type="radio"/> 可行，完整 <input type="radio"/> 可行，不完整 <input type="radio"/> 不可行，不完整	<input type="radio"/> 丰富，清晰 <input type="radio"/> 较丰富，较清晰 <input type="radio"/> 丰富，不清晰 <input type="radio"/> 不丰富，不清晰	<input type="radio"/> 结论正确，分析合理 <input type="radio"/> 结论正确，分析不充分 <input type="radio"/> 结论不正确，分析不合理	教师签字：

一、实验目的

- （1）掌握利用扩展的 JavaScript 框架构架 Web 应用程序。
- （2）掌握 EasyUI 中常用组件的使用方法。
- （3）掌握 jQuery 常用方法 \$.post() 的使用。
- （4）JSON 格式串和 Java 集合工具类的转换工具的使用。

二、实验环境

Windows 操作系统，Tomcat，MyEclipse，HBuilder，记事本。

三、实验要求

- 1、构建基于 Java 的 JDBC 访问数据库的类。具体要求：
 - （1）类的实现上应具有执行查询 SQL 语句的返回结果集的方法；
 - （2）能够实现执行 insert into、update 及 delete 方法；
 - （3）类应进行相关资源的释放。

(4) 构建时使用数据库连接池技术实现。

2、EasyUI 组件 ComboBox 的属性和使用，要求：

(1) 构建表，将表中的数据初始化到 ComboBox 中；

(2) 将 ComboBox 组件的值提交到 Servlet 中；

(3) 提交数据利用 jQuery 中的 \$.post() 方法。

3、EasyUI 组件数据表格组件：DataGrid 的属性和使用，具体要求：

(1) 初始化 DataGrid 组件：将表中的数据显示在 DataGrid 组件中；

(2) 获得选中 DataGrid 组件中某一行中的数据；

(3) DataGrid 控件中应带有分页功能；

(4) 当对数据库表中的数据进行修改时，能够自动刷新 DataGrid 中的数据。

四、实验过程记录（源程序、测试用例、测试结果及心得体会等）

1、构建基于 Java 的 JDBC 访问数据库的类。具体要求：

(1) 类的实现上应具有执行查询 SQL 语句的返回结果集的方法；

(2) 能够实现执行 insert into、update 及 delete 方法；

(3) 类应进行相关资源的释放。

(4) 构建时使用数据库连接池技术实现。

数据库连接池配置：

(1) 在项目：【WebRoot】-【META-INF】下：Context.xml 文件中加入如下内容：

<Context>

```
<Resource name="jdbc/DBPool" auth="Container"  
    type="javax.sql.DataSource"  
    factory="org.apache.commons.dbcp2.BasicDataSourceFactory"  
    username="用户名"  
    password="密码"  
    driverClassName="数据库驱动名"  
    url="数据库连接串"  
    maxTotal="100"  
    maxIdle="1000"  
    maxWaitMillis="5000" />
```

</Context>

(2) 在项目：【WebRoot】-【WEB-INF】下：web.xml 文件中加入如下内容：

```
<resource-ref>  
    <description>DB Connection</description>  
    <res-ref-name>jdbc/DBPool</res-ref-name>  
    <res-type>javax.sql.DataSource</res-type>  
    <res-auth>Container</res-auth>  
</resource-ref>
```

数据库工具类 (DBUtil.class)

```
/**
 * @ 函数名称: getConn
 * @ 功能描述: 获取数据库连接(通过连接池)
 * @ 传入参数: 无
 * @ 返回类型: Connection
 * @ 文件作者: DukeWF
 * @ 创建时间: 2018-04-30
 * @ 版本编号: 1.00
 */
public static Connection getConn(){
    try{
        Context ctx = new InitialContext();
        DataSource ds=(DataSource) ctx.lookup("java:comp/env/jdbc/DBPool");
        conn=ds.getConnection();
    }catch(Exception e){
        e.printStackTrace();
    }
    return conn;
}

/**
 * @ 函数名称: executeBatch
 * @ 功能描述: 根据查询SQL语句进行增删改操作。
 * @ 传入参数: 用于查询的SQL语句sql
 * @ 返回类型: boolean
 * @ 文件作者: DukeWF
 * @ 创建时间: 2018-04-30
 * @ 版本编号: 1.00
 */
public static boolean executeBatch(String sql) {
    boolean flag = true; // 返回值默认为true
    try {
        conn = getConn(); // 调用getConn()方法, 初始化数据库连接
        conn.setAutoCommit(false);
        st = conn.createStatement();
        st.addBatch(sql);
        st.executeBatch();
        conn.commit(); // 执行事务
        conn.setAutoCommit(true);
    } catch (Exception ex) {
        try {
            conn.rollback(); // 事务回滚
        } catch (SQLException e) {
        }
    }
}
```

```

        e.printStackTrace();
    }
    flag = false; // 执行失败，返回false
    ex.printStackTrace();
} finally {
    finallyHandle(conn, st, rs); // 关闭数据库连接
}
return flag;
}

/**
 * @ 函数名称: executeBatch
 * @ 功能描述: 根据查询SQL语句进行增删改操作。
 * @ 传入参数: 用于查询的SQL语句list (ArrayList<HashMap<String,Object>>)
 * @ 返回类型: boolean
 * @ 文件作者: DukeWF
 * @ 创建时间: 2018-04-30
 * @ 版本编号: 1.00
 */
public static boolean executeBatch(ArrayList<String> list) {
    boolean flag = true; // 返回值默认为true
    try {
        conn = getConn(); // 调用getConn()方法，初始化数据库连接
        conn.setAutoCommit(false);
        st = conn.createStatement();
        for (int i = 0; i < list.size(); i++) {
            st.addBatch(list.get(i));
        }
        st.executeBatch();
        conn.commit(); // 执行事务
        conn.setAutoCommit(true);

    } catch (Exception ex) {
        try {
            conn.rollback(); // 事务回滚
        } catch (SQLException e) {
            e.printStackTrace();
        }
        flag = false; // 执行失败，返回false
        ex.printStackTrace();
    } finally {
        finallyHandle(conn, st, rs); // 关闭数据库连接
    }
    return flag;
}
}

```

```

/**
 * @ 函数名称: getDataSet
 * @ 功能描述: 根据查询SQL语句进行查询操作。
 * @ 传入参数: 用于查询的SQL语句sql
 * @ 返回类型: (ArrayList<HashMap<String, String>>)
 * @ 文件作者: DukeWF
 * @ 创建时间: 2018-04-30
 * @ 版本编号: 1.00
 */
public static ArrayList<HashMap<String, String>> getDataSet(String sql) {
    HashMap<String, String> hash = null;
    ArrayList<HashMap<String, String>> list = new ArrayList<>();
    ResultSetMetaData rsma = null;
    int columncount = 0;

    try {
        conn = DBUtil.getConn();
        st = conn.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY);
        rs = st.executeQuery(sql);
        rsma = rs.getMetaData();
        while (rs.next()) {
            hash = new HashMap<>();
            columncount = rsma.getColumnCount();
            for (int i = 1; i <= columncount; i++) {
                hash.put(rsma.getColumnName(i), rs.getString(i));
            }
            list.add(hash);
        }
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        finallyHandle(conn, st, rs);
    }
    return list;
}

/**
 * @ 函数名称: getDataSetInfoByCon
 * @ 功能描述: 根据查询SQL语句、页码及页数返回部分多条记录。
 * @ 传入参数: 用于查询的SQL语句、页码、页数
 * @ 返回类型: (ArrayList<HashMap<String, Object>>)
 * @ 文件作者: DukeWF
 * @ 创建时间: 2018-05-06
 * @ 版本编号: 1.00
 */

```

```

public static ArrayList<HashMap<String, String>> getDataSetInfoByCon(String sql, int
rowCount, int page) {
    Connection conn = null;
    ArrayList<HashMap<String, String>> result = null;
    Statement st = null;
    ResultSet rs = null;
    ResultSetMetaData rsmd = null;
    try {
        conn = getConn();
        st = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_READ_ONLY);
        if (rowCount > 0)
            st.setMaxRows(page * rowCount);
        rs = st.executeQuery(sql);
        if (page >= 0 && rowCount > 0)
            rs.absolute((page - 1) * rowCount);
        rsmd = rs.getMetaData();
        result = new ArrayList<HashMap<String, String>>();
        while (rs.next()) {
            int columnCount = rsmd.getColumnCount();
            HashMap<String, String> record = new HashMap<String, String>();
            for (int i = 1; i <= columnCount; i++) {
                record.put(rsmd.getColumnName(i), rs.getString(i));
            }
            result.add(record);
        }
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        finallyHandle(conn, st, rs);
    }
    return result;
}

/**
 * @ 函数名称: finallyHandle
 * @ 功能描述: 对数据库操作结束进行资源释放工作。
 * @ 传入参数: 当前连接conn、状态st、结果集rs
 * @ 返回类型: void
 * @ 文件作者: DukeWF
 * @ 创建时间: 2018-04-30
 * @ 版本编号: 1.00
 */
private static void finallyHandle(Connection conn, Statement st, ResultSet rs) {
    try {
        if (rs != null) {

```

```

        rs.close();
        rs = null;
    }
    if (st != null) {
        st.close();
        st = null;
    }
    if (conn != null) {
        conn.close();
        conn = null;
    }
} catch (Exception ex) {
    ex.printStackTrace();
}
}
}

```

演示：

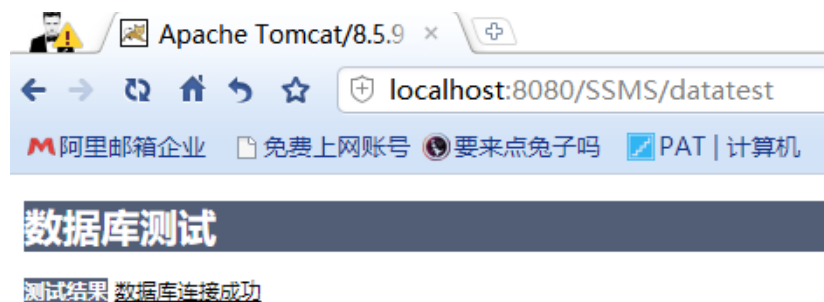


图 1 数据库测试结果

2、EasyUI 组件 ComboBox 的属性和使用，要求：

- (1) 构建表，将表中的数据初始化到 ComboBox 中；
- (2) 将 ComboBox 组件的值提交到 Servlet 中；
- (3) 提交数据利用 jQuery 中的 \$.post() 方法。

前端代码：

```

<div>
    &nbsp;姓名: <input class="easyui-combobox" id="div_name"
        style="width:150px"></input> &nbsp;所在学院: <input
        class="easyui-combobox" id="div_institute" style="width:150px"></input>
    &nbsp;<a id="query" class="easyui-linkbutton" iconCls="icon-search"
        plain="true">查 询</a>
</div>

```

JS 代码：

```

$(document).ready(function() {
    $("#div_institute").combobox({

```

```

        url : "SystemStudentService?op=institute",
        valueField : "institute_id",
        textField : "institute_name",
        panelHeight : 'auto'
    });
}

```

后端代码:

```

private void getComboBox(HttpServletResponse response, String type) {
    String result = "";
    ArrayList<HashMap<String, String>> dt = null;
    String sql;
    int rowcount = 0;
    try {
        sql = "SELECT * FROM " + type;
        dt = DBUtil.getDataSet(sql);
        result = JSON.toJSONString(dt);
        System.out.println(result);
        response.setCharacterEncoding("utf-8");
        PrintWriter out = response.getWriter();

        out.print(result);
        out.close();
    } catch (Exception ex) {
        ex.printStackTrace();
    }
}

```

演示:

The screenshot shows a web application interface with a table of student records and a dropdown menu for selecting a college. The dropdown is highlighted with a red box.

姓名:	学号	所在学院:	政治面貌
王帆	1	计算机科学与工程学院	共青团员
李妹洁	7	机械工程学院	共青团员
陈国栋	8	管理学院	共青团员
陈国栋	15	海运学院	共青团员
陈国栋	16	艺术学院	共青团员
陈国栋	18	软件学院	共青团员
陈国栋	25	外国语学院	共青团员
陈国栋	26	汉语言文化学院	共青团员
陈国栋	27	社会发展学院	共青团员
陈国栋	20152100	环境科学与安全学院	共青团员
陈国栋		理学院	共青团员
陈国栋		电气电子工程学院	共青团员

图 2 ComboBox 控件使用演示

3、EasyUI 组件数据表格组件：DataGrid 的属性和使用，具体要求：

- (1) 初始化 DataGrid 组件：将表中的数据显示在 DataGrid 组件中；
- (2) 获得选中 DataGrid 组件中某一行中的数据；
- (3) DataGrid 控件中应带有分页功能；
- (4) 当对数据库表中的数据进行修改时，能够自动刷新 DataGrid 中的数据。

前端代码：

```
<div id="dlgStudentInfo" class="easyui-dialog" style="width:360px;height:450px;
padding:10px 10px" closed="true" buttons="#dlg-buttons">
    <div class="fitem">学生信息</div>
    <form id="fm" method="post" novalidate>

        <table class="fitem">
            <tr>
                <td><label class="fitem">学 号</label></td>
                <td><input type="text" id="sno" name="sno"></td>
            </tr>
            <tr>
                <td><label class="fitem">姓名</label></td>
                <td><input type="text" id="sname" name="sname"></td>
            </tr>
            <tr>
                <td><label class="fitem">年 龄</label></td>
                <td><input type="text" id="sname" name="sname"></td>
            </tr>
            <tr>
                <td><label class="fitem">政治面貌</label></td>
                <td><input class="easyui-combobox" type="text"
                    name="politicalstate" id="politicalstate" /></td>
            </tr>
            <tr>
                <td><label class="fitem">出生日期</label></td>
                <td><input type="text" id="birthday" name="birthday" /></td>
            </tr>
            <tr>
                <td><label class="fitem">地址</label></td>
                <td><input type="text" id="address" name="address" /></td>
            </tr>
            <tr>
                <td><label>联系方式</label></td>
                <td><input type="text" id="phone" name="phone" /></td>
            </tr>
            <tr>
                <td><label>学院</label></td>
```

```

        <td><input class="easyui-combobox" type="text"
            id="dlg_institute" name="dlg_institute" /></td>
    </tr>
    <tr>
        <td><label class="fitem">备注</label></td>
        <td><input type="text" id="demo" name="demo" /></td>
    </tr>
</table>
</form>
</div>

```

JS 代码:

```

$(function() {
    $('#birthday').datebox({
        required : false
    });

    $("#info").datagrid({
        loadMsg : "数据加载中, 请等待...",
        iconCls : 'icon-issue',
        nowrap : false,
        striped : true,
        collapsible : true,
        rownumbers : true,
        pagination : true,
        singleSelect : true,
        autoRowHeight : true,
        fitColumns : false,
        pageSize : 10,
        pageList : [ 10, 20, 30, 40 ],
        cache : false,
        url : "SystemStudentService?op=init"
    });

    $("#add").click(function() {
        operation = "add";
        $('#dlgStudentInfo').dialog('open').dialog('center').dialog('setTitle', '添加学生信息');
        $('#fm').form('clear');
        $("#dlg_institute").combobox({
            url : "SystemStudentService?op=institute",
            valueField : "institute_id",
            textField : "institute_name",
            panelHeight : 'auto'
        });
    });
}

```

```

        $("#politicalstate").combobox({
            url : "SystemStudentService?op=politicalstate",
            valueField : "politicalstate_id",
            textField : "politicalstate_name",
            panelHeight : 'auto'
        });
    });

    $("#edit").click(function() {
        var row = $('#info').datagrid('getSelected');
        if (row) {
            sno = row.sno;
            $('#fm').form('clear');
            operation = "edit";
            $('#dlgStudentInfo').dialog('open').dialog('center').dialog('setTitle', '编辑
学生信息');
            $("#dlg_institute").combobox({
                url : "SystemStudentService?op=institute",
                valueField : "institute_id",
                textField : "institute_name",
                panelHeight : 'auto'
            });
            $("#politicalstate").combobox({
                url : "SystemStudentService?op=politicalstate",
                valueField : "politicalstate_id",
                textField : "politicalstate_name",
                panelHeight : 'auto'
            });
            $("#sno").val(row.sno);
            $("#sname").val(row.sname);
            $("#age").val(row.age);
            $("#politicalstate").combobox("setValue", row.politicalstate);
            $("#birthday").datebox("setValue", row.birthday);
            $("#address").val(row.address);
            $("#phone").val(row.phone);
            $("#dlg_institute").combobox("setValue", row.institute);
            $("#demo").val(row.demo);
        } else {
            $.messager.alert("提示", "请选择要编辑的学生信息!", 'info');
            return;
        }
    });

    //对话框中的保存按钮操作:完成添加和编辑操作
    $("#save").click(function() {

```

```

    if ($("#sno").val() == "") {
        $.messager.alert("提示", "学生编码不能为空!", 'info');
        return;
    }

    if (operation == "add") {
        $.post("SystemStudentService?op=add",
            $("#fm").serialize(),
            function(data) {
                var value = eval('(' + data + ')');
                if (value.ret == "0") {
                    $.messager.alert("提示", "添加成功!", 'info');
                    initstudentinfo();
                    return;
                } else {
                    $.messager.alert("提示", value.reason, 'info');
                    return;
                }
            });
    } else if (operation == "edit") {
        $.post("SystemStudentService?op=edit&oldsno=" + sno,
            $("#fm").serialize(),
            function(data) {
                var value = eval("(" + data + ")");
                if (value.ret == "0") {
                    $.messager.alert("提示", "修改成功!", 'info');
                    initstudentinfo();
                    return;
                } else {
                    $.messager.alert("提示", value.reason, 'info');
                    return;
                }
            });
    }
});

$("#delete").click(function() {
    var row = $('#info').datagrid('getSelected');
    if (row == "" || row == null) {
        $.messager.alert("提示", "请选择要删除的信息!", 'info');
        return;
    }
    $.post("SystemStudentService?op=delete&sno=" + row.sno,

    function(data) {

```

```

        var value = eval("(" + data + ")")
        if (value.ret == "1") {
            $.messenger.alert("提示", "删除成功!", 'info');
            initstudentinfo();
            return;
        } else {
            $.messenger.alert("提示", "删除失败! " + value.reason, 'info');
            return;
        }
    });
});

$("#close").click(function() {
    $('#dlgStudentInfo').dialog('close');
    return;
});

function initstudentinfo() {
    $("#studentinfo").datagrid({
        loadMsg : "数据加载中, 请等待...",
        url : 'SystemStudentService?op=init&con='
    });
}
})

```

后端代码:

```

protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    // TODO Auto-generated method stub
    String op = request.getParameter("op");
    System.out.println(op);

    switch (op) {
        case "politicalstate":
            getComboBox(response, "politicalstate");
            break;

        case "institute":
            getComboBox(response, "institute");
            break;

        case "student":
            getComboBox(response, "student_simple");
            break;
    }
}

```

```

        case "init":
            String row = request.getParameter("rows");
            String page = request.getParameter("page");
            getStudentInfo(response, "", page, row);
            break;
        default:
            break;
    }
}

private static String getStudentInfo(HttpServletResponse response, String con, String page,
String row) {
    String result = "";
    Map<String, Object> map = new HashMap<String, Object>();
    ArrayList<HashMap<String, String>> dt = null;
    String sql;

    int rowcount = 0;
    if (con == null)
        con = "";
    if (row == null)
        row = "0";
    if (page == null)
        page = "0";
    try {
        int r = Integer.parseInt(row);
        int p = Integer.parseInt(page);
        if (!con.equals("")) {
            sql = "select * from student where " + con;
        } else {
            sql = "select * from student";
        }
        dt = DBUtil.getDataSetInfoByCon(sql, r, p);
        rowcount = DBUtil.getRowCount(sql);
        map.put("total", rowcount);
        map.put("rows", dt);
        result = JSON.toJSONString(map);
        response.setCharacterEncoding("utf-8");
        PrintWriter out = response.getWriter();

        out.print(result);
        out.close();
    } catch (Exception ex) {
        ex.printStackTrace();
    }
    return result;
}

```

}

演示：

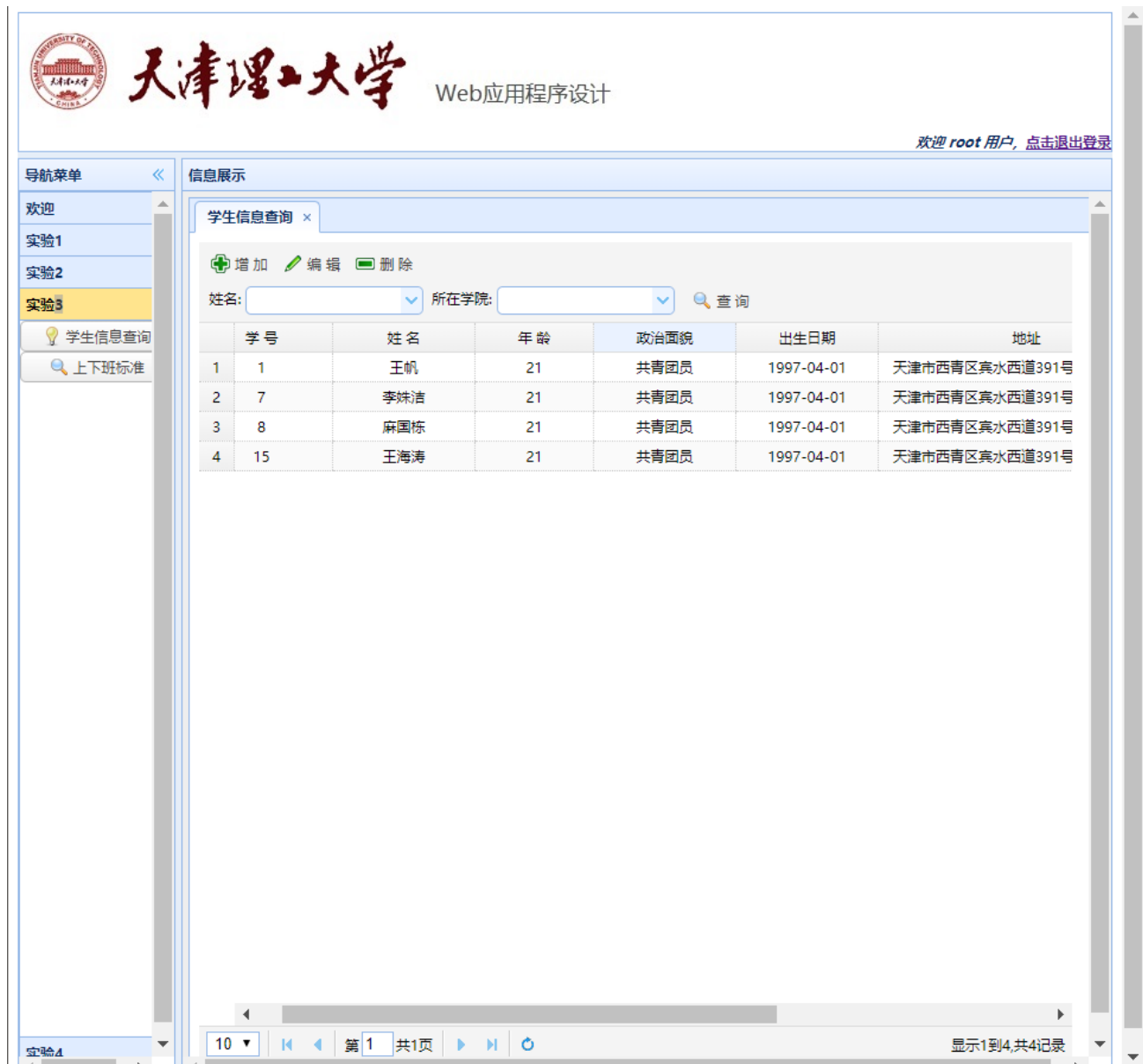


图 3 DataGrid 控件使用演示

心得体会：

通过本次实验，我初步了解了利用扩展的 JavaScript 框架构架 Web 应用程序的方法，并学习掌握了 EasyUI 中常用组件的使用方法，包括下拉列表（ComboBox）与表格（DataGrid）控件。此外我也掌握 jQuery 常用方法的使用，并对 JSON 格式串和 Java 集合工具类的转换工具的使用有了一定的了解。Web 开发包括前后端与数据库等方面，通过前三个实验的学习，我能够初步实现一个有较好交互界面的 Web 应用程序。虽然还有很多问题亟需解决，但在学习的过程中我的编程能力有了较好的提升。这对于日后的编程课程与其他专业课程的学习，乃至日后的工作与科研，都有着较为关键的帮助。

天津理工大学实验报告

学院（系）名称：计算机科学与工程学院

姓名		学号		专业	计算机科学与技术
班级		实验项目	实验 4: Web 应用综合设计		
课程名称		Web 应用程序设计与开发		课程代码	0668026
实验时间		2018 年 6 月 1 日 1、2 节 2018 年 6 月 5 日 中午 5、6 节 2018 年 6 月 8 日 1、2 节 2018 年 6 月 12 日 5、6 节		实验地点	7-215
考核标准	实验准备（实验目的/工具熟悉情况）10 分	实验过程（实验方案可行性及步骤完整性）40 分	实验报告（实验内容丰富度与格式清晰度）30 分	实验结果（结论正确性以及分析合理性）20 分	成绩
考核内容	评价实验目的是否明确，实验工具是否清晰了解以及熟悉情况	<input type="radio"/> 可行，完整 <input type="radio"/> 可行，不完整 <input type="radio"/> 不可行，不完整	<input type="radio"/> 丰富，清晰 <input type="radio"/> 较丰富，较清晰 <input type="radio"/> 丰富，不清晰 <input type="radio"/> 不丰富，不清晰	<input type="radio"/> 结论正确，分析合理 <input type="radio"/> 结论正确，分析不充分 <input type="radio"/> 结论不正确，分析不合理	教师签字：

一、实验目的

- （1）综合利用 JDBC、JSP、Servlet、EasyUI 及 jQuery 等技术实现一个小型 Web 应用系统；
- （2）系统应具有用户登录功能；
- （3）实现登录后的主界面；
- （4）至少实现 2 个具体的业务相关的功能模块（包括对表的增删改查基本操作）；
- （5）系统应能够对登录用户的会话的跟踪（可利用 Session 方式）。

二、实验环境

Windows 操作系统，Tomcat，MyEclipse，Dreamweaver，记事本。

三、实验要求

1、构建基于 SQL Server2008R 数据库相关的表。要求：

- (1) 详细列写说明各个表的结构；
- (2) 指出并说明各个表的作用。

2、构建系统对数据库的通用访问类，具体要求：

- (1) 类的实现上应具有执行查询 SQL 语句的返回结果集的方法；
- (2) 能够实现执行 insert into、update 及 delete 方法；
- (3) 类应进行相关资源的释放。

3、实现系统的登录功能，要求：

- (1) 实现用户名和密码到数据库表中的验证；
- (2) 用户名和密码错误的提示。

4、主界面的设计及实现，要求：

- (1) 应显示出当前登录用户的信息；
- (2) 具有打开某个功能模块的链接。

5、具体模块的功能实现，要求：

- (1) 设计和模块相应的数据库表；
- (2) 实现对模块对应表的增删改查操作。

6、选做内容：

实现统计报表功能：可利用 poi 技术将数据导入 EXCEL 文件方式实现。

四、实验过程记录（源程序、测试用例、测试结果及心得体会等）

1、构建基于 SQL Server2008R 数据库相关的表。

由于我主要使用的数据库是 MariaDB，因此我将使用 HeidiSQL 与 Navcat 作为数据库管理软件，对本实验数据库进行管理，并形成数据库 E-R 图。

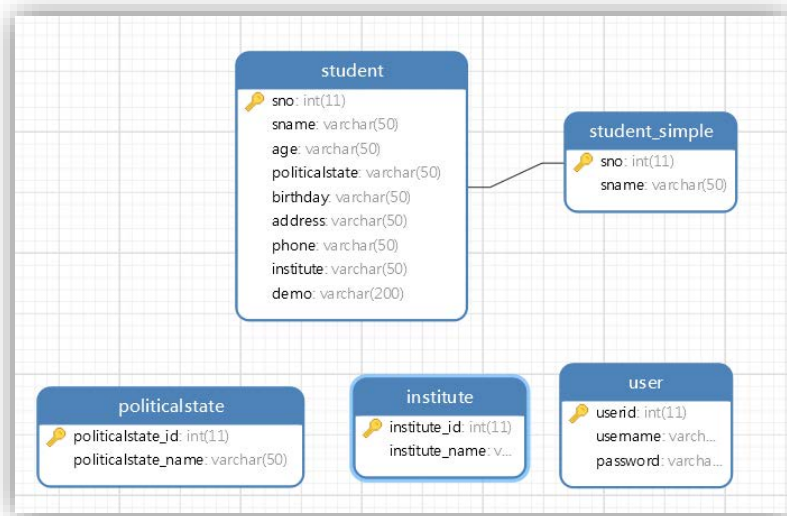


图 1 数据库 E-R 图

表 1 数据表结构（用户信息）

用户登录注册信息管理	
名称	数据类型
userid	INT
username	VARCHAR
password	VARCHAR

表 2 数据表结构（学生信息）

学生详情信息管理	
名称	数据类型
sno	INT
sname	VARCHAR
age	VARCHAR
politicalstate	VARCHAR
birthday	VARCHAR
address	VARCHAR
phone	VARCHAR
institute	VARCHAR
demo	VARCHAR

表 3 数据表结构（学生简要信息）

学生简要信息管理	
名称	数据类型
sno	INT
sname	VARCHAR

表 4 数据表结构（政治面貌）

政治面貌类型管理	
名称	数据类型
politicalstate_id	INT
politicalstate_name	VARCHAR

表 5 数据表结构（院系）

院系信息管理	
名称	数据类型
institute_id	INT
institute_name	VARCHAR

2、构建系统对数据库的通用访问类。

数据库连接池配置：

(1) 在项目：【WebRoot】 - 【META-INF】 下：Context.xml 文件中加入如下内容：

```
<Context>
<Resource name="jdbc/DBPool" auth="Container"
    type="javax.sql.DataSource"
    factory="org.apache.commons.dbcp2.BasicDataSourceFactory"
    username="用户名"
    password="密码"
    driverClassName="数据库驱动名"
    url="数据库连接串"
    maxTotal="100"
    maxIdle="1000"
    maxWaitMillis="5000" />
</Context>
```

(2) 在项目：【WebRoot】 - 【WEB-INF】 下：web.xml 文件中加入如下内容：

```
<resource-ref>
    <description>DB Connection</description>
    <res-ref-name>jdbc/DBPool</res-ref-name>
    <res-type>javax.sql.DataSource</res-type>
    <res-auth>Container</res-auth>
</resource-ref>
```

数据库工具类（DBUtil.class）

```
/**
 * @ 函数名称： getConn
 * @ 功能描述： 获取数据库连接(通过连接池)
 * @ 传入参数： 无
 * @ 返回类型： Connection
 * @ 文件作者： DukeWF
 * @ 创建时间： 2018-04-30
 * @ 版本编号： 1.00
 */
public static Connection getConn(){
    try{
        Context ctx = new InitialContext();
        DataSource ds=(DataSource) ctx.lookup("java:comp/env/jdbc/DBPool");
        conn=ds.getConnection();
    }catch(Exception e){
        e.printStackTrace();
    }
    return conn;
}
/**
 * @ 函数名称： executeBatch
```

```

* @ 功能描述: 根据查询SQL语句进行增删改操作。
* @ 传入参数: 用于查询的SQL语句sql
* @ 返回类型: boolean
* @ 文件作者: DukeWF
* @ 创建时间: 2018-04-30
* @ 版本编号: 1.00
**/
public static boolean executeBatch(String sql) {
    boolean flag = true; // 返回值默认为true
    try {
        conn = getConn(); // 调用getConn()方法, 初始化数据库连接
        conn.setAutoCommit(false);
        st = conn.createStatement();
        st.addBatch(sql);
        st.executeBatch();
        conn.commit(); // 执行事务
        conn.setAutoCommit(true);

    } catch (Exception ex) {
        try {
            conn.rollback(); // 事务回滚
        } catch (SQLException e) {
            e.printStackTrace();
        }
        flag = false; // 执行失败, 返回false
        ex.printStackTrace();
    } finally {
        finallyHandle(conn, st, rs); // 关闭数据库连接
    }
    return flag;
}
/**
* @ 函数名称: executeBatch
* @ 功能描述: 根据查询SQL语句进行增删改操作。
* @ 传入参数: 用于查询的SQL语句list (ArrayList<HashMap<String,Object>>)
* @ 返回类型: boolean
* @ 文件作者: DukeWF
* @ 创建时间: 2018-04-30
* @ 版本编号: 1.00
**/
public static boolean executeBatch(ArrayList<String> list) {
    boolean flag = true; // 返回值默认为true
    try {
        conn = getConn(); // 调用getConn()方法, 初始化数据库连接
        conn.setAutoCommit(false);

```

```

        st = conn.createStatement();
        for (int i = 0; i < list.size(); i++) {
            st.addBatch(list.get(i));
        }
        st.executeBatch();
        conn.commit();// 执行事务
        conn.setAutoCommit(true);

    } catch (Exception ex) {
        try {
            conn.rollback();// 事务回滚
        } catch (SQLException e) {
            e.printStackTrace();
        }
        flag = false;// 执行失败, 返回false
        ex.printStackTrace();
    } finally {
        finallyHandle(conn, st, rs);// 关闭数据库连接
    }
    return flag;
}

/**
 * @ 函数名称: getDataSet
 * @ 功能描述: 根据查询SQL语句进行查询操作。
 * @ 传入参数: 用于查询的SQL语句sql
 * @ 返回类型: (ArrayList<HashMap<String, String>>)
 * @ 文件作者: DukeWF
 * @ 创建时间: 2018-04-30
 * @ 版本编号: 1.00
 */
public static ArrayList<HashMap<String, String>> getDataSet(String sql) {
    HashMap<String, String> hash = null;
    ArrayList<HashMap<String, String>> list = new ArrayList<>();
    ResultSetMetaData rsma = null;
    int columncount = 0;

    try {
        conn = DBUtil.getConn();
        st = conn.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY);
        rs = st.executeQuery(sql);
        rsma = rs.getMetaData();
        while (rs.next()) {
            hash = new HashMap<>();
            columncount = rsma.getColumnCount();

```

```

        for (int i = 1; i <= columncount; i++) {
            hash.put(rsma.getColumnName(i), rs.getString(i));
        }
        list.add(hash);
    }
} catch (SQLException e) {
    e.printStackTrace();
} finally {
    finallyHandle(conn, st, rs);
}
return list;
}
/**
 * @ 函数名称: getDataSetInfoByCon
 * @ 功能描述: 根据查询SQL语句、页码及页数返回部分多条记录。
 * @ 传入参数: 用于查询的SQL语句、页码、页数
 * @ 返回类型: (ArrayList<HashMap<String, Object>>)
 * @ 文件作者: DukeWF
 * @ 创建时间: 2018-05-06
 * @ 版本编号: 1.00
 */
public static ArrayList<HashMap<String, String>> getDataSetInfoByCon(String sql, int
rowCount, int page) {
    Connection conn = null;
    ArrayList<HashMap<String, String>> result = null;
    Statement st = null;
    ResultSet rs = null;
    ResultSetMetaData rsmd = null;
    try {
        conn = getConn();
        st = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_READ_ONLY);
        if (rowCount > 0)
            st.setMaxRows(page * rowCount);
        rs = st.executeQuery(sql);
        if (page >= 0 && rowCount > 0)
            rs.absolute((page - 1) * rowCount);
        rsmd = rs.getMetaData();
        result = new ArrayList<HashMap<String, String>>();
        while (rs.next()) {
            int columnCount = rsmd.getColumnCount();
            HashMap<String, String> record = new HashMap<String, String>();
            for (int i = 1; i <= columnCount; i++) {
                record.put(rsmd.getColumnName(i), rs.getString(i));
            }

```

```

        result.add(record);
    }
} catch (Exception e) {
    e.printStackTrace();
} finally {
    finallyHandle(conn, st, rs);
}
return result;
}
/**
 * @ 函数名称: finallyHandle
 * @ 功能描述: 对数据库操作结束进行资源释放工作。
 * @ 传入参数: 当前连接conn、状态st、结果集rs
 * @ 返回类型: void
 * @ 文件作者: DukeWF
 * @ 创建时间: 2018-04-30
 * @ 版本编号: 1.00
 */
private static void finallyHandle(Connection conn, Statement st, ResultSet rs) {
    try {
        if (rs != null) {
            rs.close();
            rs = null;
        }
        if (st != null) {
            st.close();
            st = null;
        }
        if (conn != null) {
            conn.close();
            conn = null;
        }
    } catch (Exception ex) {
        ex.printStackTrace();
    }
}

```

演示:



图 2 数据库测试结果

3、实现系统的登录功能

前端代码：

登录页

```
<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<title>登录</title>
<link rel="stylesheet"
      href="https://maxcdn.bootstrapcdn.com/font-awesome/4.5.0/css/font-awesome.min.css">
<link rel="stylesheet"
      href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css">
<link rel="stylesheet" type="text/css"
      href="${pageContext.request.contextPath}/css/Lab2_Login.css">
</head>
<body>
  <div class="container">
    <div class="row">
      <div class="col-md-offset-3 col-md-6">
        <form class="form-horizontal"
              action="${pageContext.request.contextPath}/LoginServlet"
              method="post">
          <span class="heading">用户登录</span>
          <div class="form-group">
            <input type="text" class="form-control" id="username"
name="username"
                    placeholder="用户名"> <i class="fa fa-user"></i>
          </div>
          <div class="form-group help">
            <input type="password" class="form-control" id="password"
name="password"
```



```

placeholder="密 码"><i class="fa fa-lock"></i> <a href="#"
class="fa fa-question-circle"></a>
</div>
<div class="form-group">
  <div class="main-checkbox">
    <input type="checkbox" value="None" id="checkbox1"
name="check" />
    <label for="checkbox1"></label>
  </div>
  <span class="text">Remember me</span>
  <button type="submit" class="btn btn-default">登录</button>

  <button type="button" class="btn btn-default"
onclick="location.href='reg.jsp'">注册</button>
  </div>
</form>
</div>
</div>
</div>
<p>${msg}</p>
</body>
登录状态页
</html>
<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
<%
String path = request.getContextPath();
String basePath =
request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";
%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <base href="<%=basePath%>">

    <title>登录状态页</title>

  </head>

  <body>
    <%
      if(session.getAttribute("loginState")==="1"){
        response.sendRedirect(".\\index.jsp");
      }
    else{

```

```

        response.sendRedirect("./\\login.jsp");
    }
    %>
</body>
</html>

```

后端实现:

```

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
    // TODO Auto-generated method stub
    //doGet(request, response);
    response.setContentType("text/html;charset=utf-8");
    String username = WebUtil.getParameter(request,"username");
    String password = WebUtil.getParameter(request,"password");
    PrintWriter out = response.getWriter();

    //System.out.println(username+","+password);

    HashMap<String, String> hashMap = new HashMap<String, String>();
    hashMap.put("username", username);
    hashMap.put("password", password);

    if(DBUtil.getDataCount("user",hashMap)==1) {
        HttpSession session = request.getSession();
        session.setAttribute("username",username); //用户名
        session.setAttribute("loginState","1"); //登录状态
        response.sendRedirect("./loginStateHandle.jsp");
    }
    else {
        HttpSession session = request.getSession();
        session.setAttribute("loginState","0"); //登录状态
        response.sendRedirect("./loginCheck.jsp");
    }
}
}

```

演示

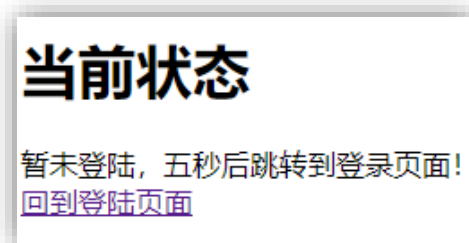


图 3-1 用户登录



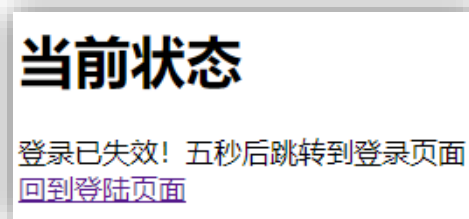
The image shows a web form titled "新用户注册" (New User Registration). It contains three input fields: "用户名" (Username), "密码" (Password), and "重复密码" (Repeat Password). Each field has a small icon on the left and a question mark icon on the right. Below the fields is a "Remember Me" checkbox and a blue "注册" (Register) button.

图 3-2 新用户注册



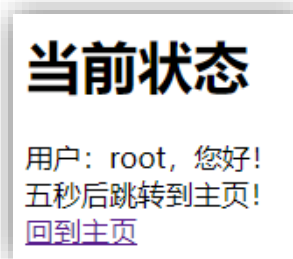
The image shows a message box titled "当前状态" (Current Status). The text inside says "暂未登陆, 五秒后跳转到登录页面!" (Not logged in, will redirect to login page in 5 seconds!) and includes a purple link "回到登陆页面" (Return to login page).

图 3-3 用户未登录



The image shows a message box titled "当前状态" (Current Status). The text inside says "登录已失效! 五秒后跳转到登录页面!" (Login is invalid, will redirect to login page in 5 seconds!) and includes a purple link "回到登陆页面" (Return to login page).

图 3-4 用户退出登录



The image shows a message box titled "当前状态" (Current Status). The text inside says "用户: root, 您好!" (User: root, Hello!) and "五秒后跳转到主页!" (Will redirect to homepage in 5 seconds!). It includes a purple link "回到主页" (Return to homepage).

图 3-5 用户登录成功提示

4、主界面的设计及实现

前端实现:

```
<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
<%
String path = request.getContextPath();
String basePath =
request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";
```

```

%>

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">

<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>main</title>

  <!-- BOOTSTRAP STYLES-->
  <link href="assets/css/bootstrap.css" rel="stylesheet" />
  <!-- FONTAWESOME STYLES-->
  <link href="assets/css/font-awesome.css" rel="stylesheet" />
  <!--CUSTOM BASIC STYLES-->
  <link href="assets/css/basic.css" rel="stylesheet" />
  <!--CUSTOM MAIN STYLES-->
  <link href="assets/css/custom.css" rel="stylesheet" />
  <!-- GOOGLE FONTS-->
  <link href='http://fonts.googleapis.com/css?family=Open+Sans' rel='stylesheet'
type='text/css' />
</head>

<body>
  <div id="page-inner">
    <div class="row">
      <div class="col-md-12">
        <h1 class="page-head-line">欢迎使用</h1>
        <h1 class="page-subhead-line" style="font-style: normal">在使用过程中，如有疑问，请联系管理员。</h1>

      </div>
    </div>
    <!--/.ROW-->
    <div class="row">
      <div class="col-md-12">
        <div class="panel panel-default">
          <div class="panel-heading">
            功能选项
          </div>
          <div class="panel-body">
            <div class="row">
              <div class="col-md-3" onclick="location='welcome.jsp'">
                <div class="alert alert-info text-center">
                  <i class="fa fa-desktop fa-5x"></i>

```

```

        <h3>实验1</i>
    </h3>
    实现HTML静态界面与数据库连接操作<br>
</div>
</div>
<div class="col-md-3 " onclick="location='Form.jsp'">
    <div class="alert alert-success text-center">
        <i class="fa fa-bars fa-5x"></i>
        <h3>实验2</h3>
        实现登录、表单数据的提交，并使用JavaBean对数据进行封装
    </div>
</div>
<div class="col-md-3 " onclick="location='DataGrid.jsp'">
    <div class="alert alert-warning text-center">
        <i class="fa fa-fax fa-5x"></i>
        <h3>实验3</h3>
        <br> 实现easyUI框架DataGrid的使用
    </div>
</div>
<div class="col-md-3 " onclick="location='welcome.jsp'">
    <div class="alert alert-danger text-center">
        <i class="fa fa-bomb fa-5x"></i>
        <h3>实验4</h3>
        <br> 实现SSMS管理系统
    </div>
</div>
</div>
</div>
</div>
</div>
</div>
<!--/.ROW-->
</div>
</body>

```

演示：



图 4-1 用户信息提示模块

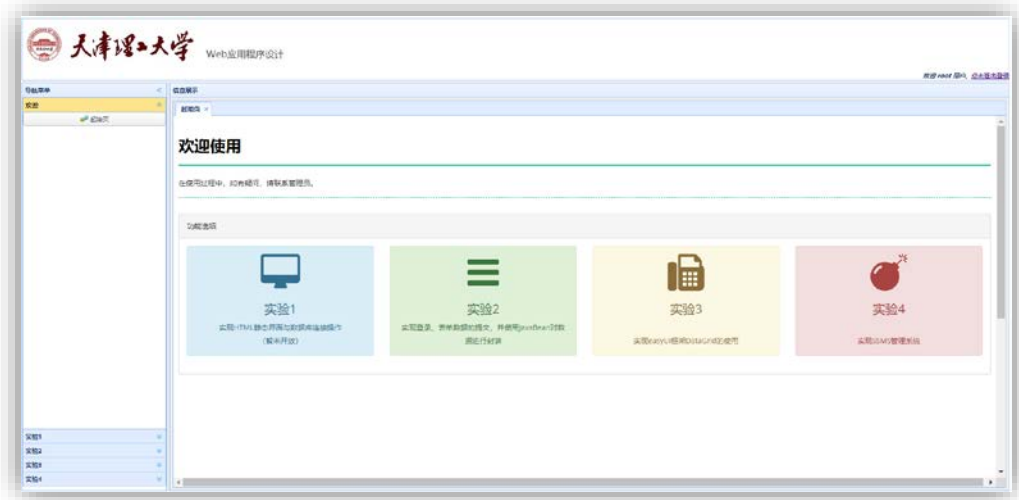


图 4-2 主界面（1920*1080 分辨率下）



图 4-3 主界面（分屏模式下）

5、具体模块的功能实现

增加信息：

后端：

case "add":

```
Student student_insert = new Student();
student_insert.setAddress(request.getParameter("address"));
student_insert.setAge(request.getParameter("age"));
student_insert.setBirthday(request.getParameter("birthday"));
student_insert.setDemo(request.getParameter("demo"));
student_insert.setInstitute(request.getParameter("institute"));
student_insert.setPhone(request.getParameter("phone"));
student_insert.setSname(request.getParameter("sname"));
student_insert.setSno(Integer.parseInt(request.getParameter("sno")));
student_insert.setPoliticalstate(request.getParameter("politicalstate"));

InsertStudentInfo(response, student_insert);
break;
private void InsertStudentInfo(HttpServletResponse response, Student student) {
    try {
        JSONObject jsonObject = new JSONObject();
        ArrayList<String> List = new ArrayList<>();
        String insert = "INSERT INTO
student(sno,sname,age,politicalstate,birthday,address,phone,institute,demo) VALUES('";
        insert+=student.getSno()+"', '"+student.getSname()+"', '"+student.getAge()+"', '"+student.getPoliticalstate()+"', '"+student.getBirthday()+"', '"+student.getAddress()+"', '"+student.getPhone()+"', '"+student.getInstitute()+"', '"+student.getDemo()+"')";
        System.out.println(insert);
        boolean result = DBUtil.executeBatch(insert);
        PrintWriter out = response.getWriter();
        response.setCharacterEncoding("utf-8");
        if(result){
            jsonObject.put("ret", "1");
            System.out.println("插入成功");
        }
        else {
            jsonObject.put("ret", "0");
            jsonObject.put("reason", "数据库操作失败");
            System.out.println("插入失败");
        }
        out.print(jsonObject);
        out.close();
    } catch (Exception ex) {
        ex.printStackTrace();
    }
}
```

演示：

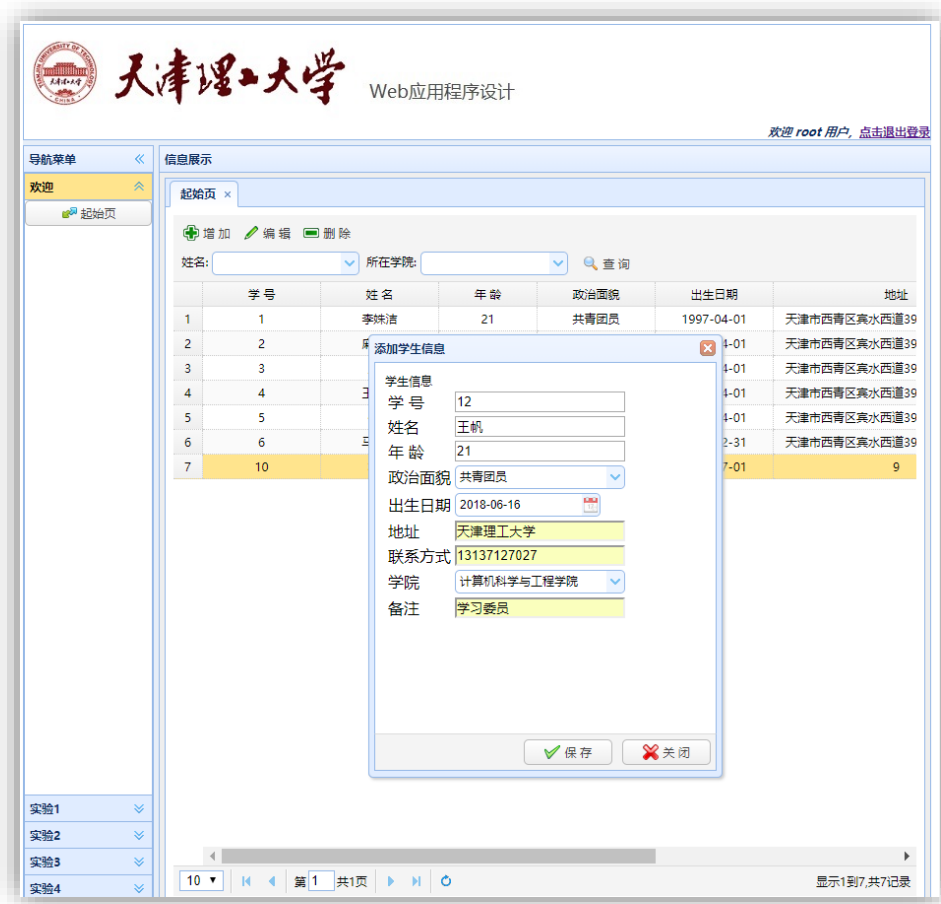


图 5-1-1 添加学生信息

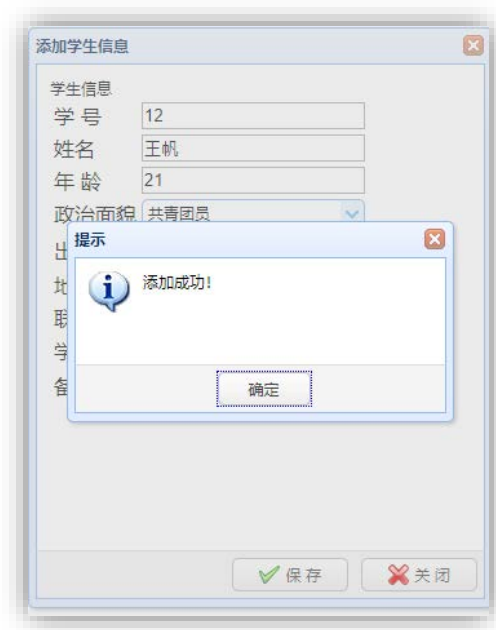


图 5-1-2 添加成功提示

修改信息

后端:

case "edit":

```
int oldsno = Integer.parseInt(request.getParameter("oldsno"));
Student student_update = new Student();

student_update.setAddress(request.getParameter("address"));
student_update.setAge(request.getParameter("age"));
student_update.setBirthday(request.getParameter("birthday"));
student_update.setDemo(request.getParameter("demo"));
student_update.setInstitute(request.getParameter("institute"));
student_update.setPhone(request.getParameter("phone"));
student_update.setSname(request.getParameter("sname"));
student_update.setPoliticalstate(request.getParameter("politicalstate"));
```

```
UpdateStudentInfo(response, student_update, oldsno);
```

```
break;
```

private void UpdateStudentInfo(HttpServletResponse response, Student student, int oldsno) {

try {

```
JSONObject jsonObject = new JSONObject();
```

```
ArrayList<String> List = new ArrayList<>();
```

```
String update = "UPDATE student SET ";
```

```
update+="sname = '"+student.getSname()+"',";
```

```
update+="birthday = '"+student.getBirthday()+"',";
```

```
update+="age = '"+student.getAge()+"',";
```

```
update+="politicalstate = '"+student.getPoliticalstate()+"',";
```

```
update+="address = '"+student.getAddress()+"',";
```

```
update+="phone = '"+student.getPhone()+"',";
```

```
update+="institute = '"+student.getInstitute()+"',";
```

```
update+="demo = '"+student.getDemo()+"'";
```

```
update+=" WHERE sno='"+oldsno+"'";
```

```
boolean result = DBUtil.executeBatch(update);
```

```
PrintWriter out = response.getWriter();
```

```
response.setCharacterEncoding("utf-8");
```

```
if(result){
```

```
    jsonObject.put("ret", "1");
```

```
    System.out.println("更新成功");
```

```
}
```

```
else {
```

```
    jsonObject.put("ret", "0");
```

```
    jsonObject.put("reason", "数据库操作失败");
```

```
    System.out.println("更新失败");
```

```

    }
    out.print(jsonObject);
    out.close();
} catch (Exception ex) {
    ex.printStackTrace();
}
}

```

演示：



图 5-2-1 修改学生信息



图 5-2-2 修改成功提示

删除信息:

后端代码:

```
private void deleteStudentInfo(HttpServletRequest response, String sno) {  
    try {  
        JSONObject jsonObject = new JSONObject();  
        ArrayList<String> List = new ArrayList<>();  
        String delete = "DELETE FROM student WHERE sno='" + sno + "'";  
        List.add(delete);  
        boolean result = DBUtil.executeBatch(List);  
        PrintWriter out = response.getWriter();  
        response.setCharacterEncoding("utf-8");  
        if(result){  
            jsonObject.put("ret", "1");  
            System.out.println("删除成功");  
        }  
        else {  
            jsonObject.put("ret", "0");  
            jsonObject.put("reason", "数据库操作失败");  
            System.out.println("删除失败");  
        }  
        out.print(jsonObject);  
        out.close();  
    } catch (Exception ex) {  
        ex.printStackTrace();  
    }  
}
```

演示:

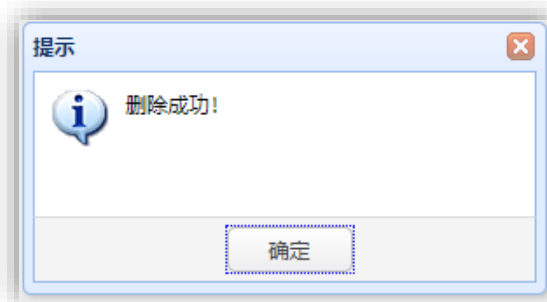


图 5-3 删除成功提示

查找特定信息:

后端代码:

```
private static String getStudentInfo(HttpServletRequest response, String con, String  
page, String row) {  
    String result = "";
```

```
Map<String, Object> map = new HashMap<String, Object>();
ArrayList<HashMap<String, String>> dt = null;
String sql;
int rowcount = 0;
if (con == null)
    con = "";
if (row == null)
    row = "0";
if (page == null)
    page = "0";
try {
    int r = Integer.parseInt(row);
    int p = Integer.parseInt(page);
    if (!con.equals("")) {
        sql = "select * from student where " + con;
    } else {
        sql = "select * from student";
    }
    dt = DBUtil.getDataSetInfoByCon(sql, r, p);
    rowcount = DBUtil.getRowCount(sql);
    map.put("total", rowcount);
    map.put("rows", dt);
    result = JSON.toJSONString(map);
    response.setCharacterEncoding("utf-8");
    PrintWriter out = response.getWriter();

    out.print(result);
    out.close();
} catch (Exception ex) {
    ex.printStackTrace();
}
return result;
```

1
演示:

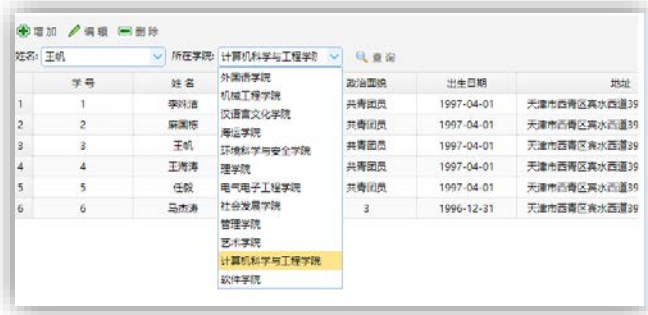


图 5-4 查询选项

心得体会：

通过本次实验，我初步了解了利用扩展的 JavaScript 框架构架 Web 应用程序的方法，并学习掌握了 EasyUI 中常用组件的使用方法，包括下拉列表（ComboBox）与表格（DataGrid）控件。此外我也掌握 jQuery 常用方法的使用，并对 JSON 格式串和 Java 集合工具类的转换工具的使用有了一定的了解。Web 开发包括前后端与数据库等方面，通过前三个实验的学习，我能够初步实现一个有较好交互界面的 Web 应用程序。虽然还有很多问题亟需解决，但在学习的过程中我的编程能力有了较好的提升。这对于日后的编程课程与其他专业课程的学习，乃至日后的工作与科研，都有着较为关键的帮助。