《Web 技术》实验报告

院　　系	计算机院
专业班级	计算机科学与技术 1707
姓　　名	罗家璇
学　　号	0902170512
指导教师	钟萍

2019 年　12 月　20 日

摘　　要

这是《Web技术》课程的一次实验，要求用原生前后端语言实现一个全栈系统——学生信息管理系统，我做的是学生图书管理系统，前端用的是html+css+js，后端使用jsp和java搭建，前后端交互采用了ajax模式，数据传输格式为json，数据库用的是sql server，实现了在前端页面完成对数据库中数据的增删改查，完成高亮搜索，权限管理等额外功能。

关键词：jsp；学生图书管理系统；原生 js；

目　　录

1 实验概述

1.1 实验目的

通过设计并实现一个简单全栈系统的方式提升对 Web 相关技术的理解、掌握及应用。

1.2 实验任务

本次实验任务为实现一个信息管理系统，能够在前端页面中完成对数据库中 数据的查删改查。具体要求如下：

1）实现前端页面的基本布局。包含：

a) 顶部 LOGO 栏目：包含系统图标，系统名称以及*搜索框；

b) 左侧菜单栏：提供能够具备折叠展开的二级菜单功能。

c) 中部内容栏：用于内容的展示及相关操作。

d) 底部版权栏：一句话说明系统开发相关信息。

2）完成前后端数据交互（用 JSON 格式）

3）数据操作要求：

a) 数据统一存储在后端数据库中；

b) 前端能够展示从后端提取的数据；

c) 前端能够添加新的数据（增加一条新的记录）；

d) 添加新纪录过程中需对至少一个字段（如手机号码）做有效性验证。

e) 能够删除记录（同时删除前端页面中数据和后台数据库中数据）。删 除记录操作不会触发整个页面的刷新；

f) 能够根据某一个字段值的查询功能。

g) *能够对某一条记录内容进行修改

4）*顶部搜索框提供对当前页面所有内容搜索。对符合条件的文字进行高亮 显示。

5）*提供登录及权限管理功能。

6）后端实现技术不限、数据库系统不限

2 需求分析

审视实验任务书，我们发现，整个实验是要做一个简单的全栈系统，主要实现的功能是在前端页面中完成对数据库的数据操作，并将结果等显示给用户，我们大致可以把整个代码分成三个部分：数据库、前端界面、服务端代码。

数据库负责存入必要的数据比如用户信息和书籍信息，前端使用 html+css+js 方式实现界面布局，展示对用户友好的界面，服务端逻辑层代码，需要与前端数据进行处理和交互，同时在合适的时间访问数据库，对数据进行操作。

前端需要展示顶部 logo 栏，左侧菜单栏，中部内容栏和底部版权栏，其中顶部 logo 栏要有搜索框，并且搜索框输入关键字搜索后，符合条件的关键字会高亮；菜单栏需要放在左边，并且具有折叠展开的二级菜单功能。

用户能在前端进行的操作有：展示从后端提取的书籍信息，另外管理员用户能删除数据，能够点击添加书籍按钮添加书籍，能通过对某一条记录内容进行修改，而非管理员用户只能查看书籍、借书和还书等操作。

在实现删除操作时，要求删除记录时会同时删除前端界面中数据和后台数据库中的数据，并且不会触发整个页面的刷新，那么显然只能用 ajax 来处理这方面的需求。

3 程序设计

3.1 总体设计

如下图所示：

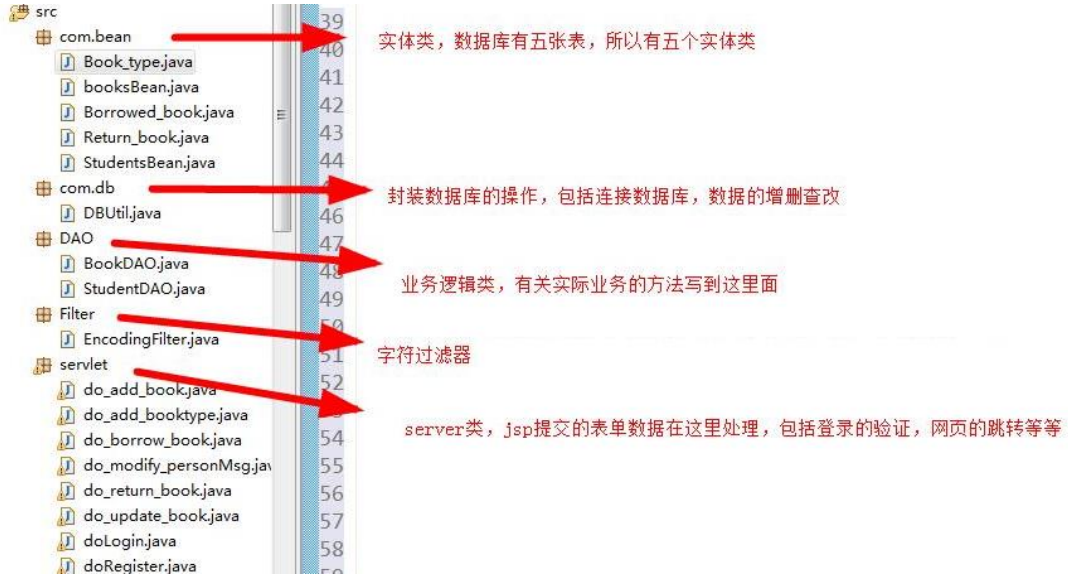

图 3-1 总体设计图

可以按需求分为如下五个包：

1. 实体类，里面放数据库存的数据表对应的 setter 和 getter 方法。

2. 数据库类，用于封装数据库操作，包括连接数据库、数据的增删查改等等。

3. DAO 业务逻辑类，也就是数据访问对象，提供了面向对象的数据库接口，可以通过数据库连接类操作数据库，完成逻辑转换或信息传递。

4. Filter 类，这里主要是字符过滤器，之前不加这个有时候在页面跳转后会出现乱码，加了之后就没问题。

5. server 类，jsp 提交的表单数据在这里处理，包括登陆验证、网页跳转、权限管理、用户操作等等。

3.2 详细设计

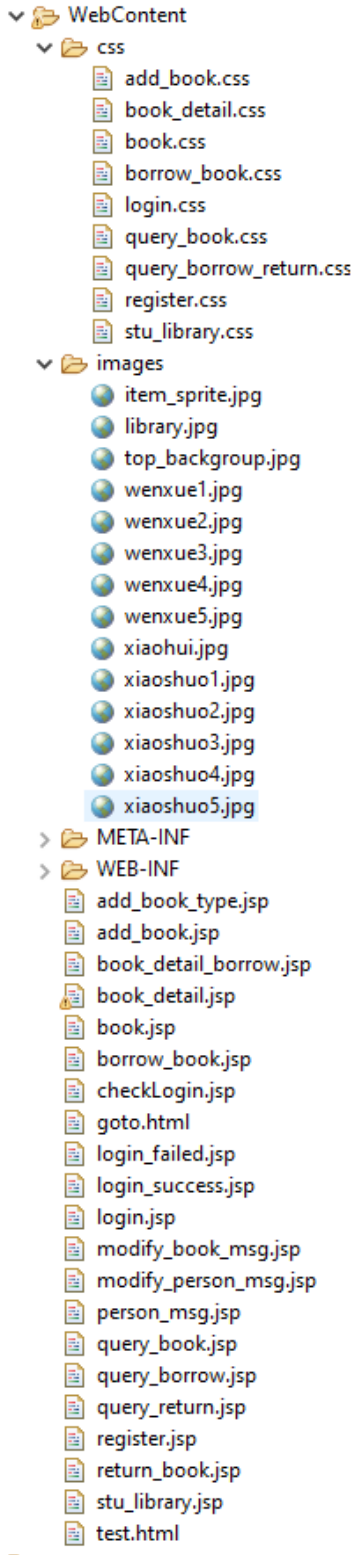

图 3-2 详细设计图

设计时是按实验书边写边改的，基本开发流程是按照顺序逻辑开发的，我先从前端界面开始写，先写注册和登陆，然后就发现需要存入用户数据，于是再去 sql

server 里建个用户表，把数据存进去，然后再返回前端写主页面，这样自己按照实验书提出需求，然后按需求顺序写代码，慢慢把整个全栈小系统补全，最后再优化界面和结构完成实验。

4 详细代码与解释

4.1 DAO 包

DAO 包里，我写了 BookDAO.java 和 StudentDAO.java 两个类，其中，BookDAO 类主要功能是将 book 数据表和 Book 类联系起来，把根据书籍名、书号等信息查询获得的 bookBean 对象写入 get 函数中，比如 `getAllBooksItem()`新建一个 ArrayList<booksBean>，然后调用数据库封装类查询操作，通过 Scard_no 借书号获得 Book 表的书籍数据填入列表中，然后就能在前端展现书库等操作。

代码如下：

```
1. package DAO;
2.
3. import java.sql.ResultSet;
4. import java.sql.SQLException;
5. import java.util.ArrayList;
6.
7. import com.bean.Book_type;
8. import com.bean.Borrowed_book;
9. import com.bean.Return_book;
10.import com.bean.booksBean;
11.import com.db.DBUtil;
12./*
13. * BookDAO 类
14. * 将 Book 数据表与 Book 类联系起来
15. */
16.public class BookDAO {
17.
18.    //获得所有的书籍
19.    public ArrayList<booksBean> getAllBooksItem(){
20.        ArrayList<booksBean> list = new ArrayList<booksBean>();
21.
22.        DBUtil db = new DBUtil();
23.        String sql = "select * from Book";
24.        ResultSet rs = db.query(sql);
25.        try {
26.            //一次拿一行
27.            while (rs.next()) {
28.                booksBean book = new booksBean();
```

```
29.                 book.setBook_name(rs.getString("book_name"));
30.                 book.setPicture(rs.getString("picture"));
31.                 book.setBook_no(rs.getString("book_no"));
32. //              book.setBook_author(rs.getString("book_author"));
33. //              book.setBook_press(rs.getString("book_press"));
34. //              book.setBook_rest_num(rs.getInt("book_rest_num"));
35. //              book.setBook_type_no(rs.getString("book_type_no"));
36. //              book.setBook_details(rs.getString("book_details"));
37.                 list.add(book);
38.             }
39.             return list;
40.         } catch (SQLException e) {
41.             e.printStackTrace();
42.
43.         }
44.         return null;
45.     }
46.
47.     //通过 Book_no 获得书的实例
48.     public booksBean getBookByno(String Book_no){
49.         DBUtil db = new DBUtil();
50.         String sql = "select * from Book where book_no like '%"+Book_n
    o+"%'";
51.         ResultSet rs = db.query(sql);
52.             try {
53.                 if (rs.next()) {
54.                     booksBean book = new booksBean();
55.                     book.setBook_name(rs.getString("book_name"));
56.                     book.setPicture(rs.getString("picture"));
57.                     book.setBook_no(rs.getString("book_no"));
58.                     book.setBook_author(rs.getString("book_author"));

59.                     book.setBook_press(rs.getString("book_press"));
60.                     book.setBook_rest_num(rs.getInt("book_rest_num"));

61.                     book.setBook_type_no(rs.getString("book_type_no"))
    ;
62.                     book.setBook_details(rs.getString("book_details"))
    ;
63.                     return book;
64.                 }
65.                 else{
66.                     return null;
67.                 }
68.             } catch (SQLException e) {
```

```
69.                 // TODO Auto-generated catch block
70.                 e.printStackTrace();
71.
72.             }return null;
73.     }
74.     //通过书籍类型编号获得书记类型
75.     public Book_type getBookTypeByno(String book_type_no){
76.         DBUtil dbUtil = new DBUtil();
77.         String sql = "select * from Book_type where book_type_no like
   '%"+book_type_no+"%'";
78.         ResultSet rs= dbUtil.query(sql);
79.         try {
80.             if(rs.next()){
81.                 Book_type book_type=new Book_type();
82.                 book_type.setBook_type_name(rs.getString("book_type_na
   me"));
83.                 return book_type;
84.             }
85.             else{
86.                 return null;
87.             }
88.         } catch (SQLException e) {
89.             // TODO Auto-generated catch block
90.             e.printStackTrace();
91.
92.         }
93.         return null;
94.     }
95.     //根据书籍作者获得相应的书籍列表
96.     public ArrayList<booksBean> getBooksByAuthor(String book_author){

97.         ArrayList<booksBean> list = new ArrayList<booksBean>();
98.
99.         DBUtil db = new DBUtil();
100.         String sql = "select * from Book where book_author like '%"+b
   ook_author+"%'";
101.         ResultSet rs = db.query(sql);
102.         try {
103.             while (rs.next()) {
104.                 booksBean book = new booksBean();
105.                 book.setBook_name(rs.getString("book_name"));
106.                 book.setBook_no(rs.getString("book_no"));
107.                 book.setBook_author(rs.getString("book_author"));
108.                 book.setBook_press(rs.getString("book_press"));
109.                 book.setBook_rest_num(rs.getInt("book_rest_num"));
```

```
110.                 book.setBook_type_no(rs.getString("book_type_no"));
111.                 list.add(book);
112.             }
113.             return list;
114.         } catch (SQLException e) {
115.             e.printStackTrace();
116.
117.         }
118.         return null;
119.     }
120.     //根据出版社获得相应的书籍列表
121.     public ArrayList<booksBean> getBooksByPress(String book_press){
122.         ArrayList<booksBean> list = new ArrayList<booksBean>();
123.         DBUtil db = new DBUtil();
124.         String sql = "select * from Book where book_press like '%"+bo
    ok_press+"%'";
125.         ResultSet rs = db.query(sql);
126.         try {
127.             while (rs.next()) {
128.                 booksBean book = new booksBean();
129.                 book.setBook_name(rs.getString("book_name"));
130.                 book.setBook_no(rs.getString("book_no"));
131.                 book.setBook_author(rs.getString("book_author"));
132.                 book.setBook_press(rs.getString("book_press"));
133.                 book.setBook_rest_num(rs.getInt("book_rest_num"));
134.                 book.setBook_type_no(rs.getString("book_type_no"));
135.                 list.add(book);
136.             }
137.             return list;
138.         } catch (SQLException e) {
139.             e.printStackTrace();
140.
141.         }
142.         return null;
143.     }
144.
145.     //根据书籍名称获得相应的书籍类型编号
146.     public Book_type getBookTypenoByBookType(String book_type_name){

147.         DBUtil db=new DBUtil();
148.         String sql="select * from Book_type where book_type_name like
     '%"+book_type_name+"%'";
149.         ResultSet rs=db.query(sql);
150.         try {
151.             while(rs.next()){
```

```
152.                 Book_type book_type=new Book_type();
153.                 book_type.setBook_type_no(rs.getString("book_type_no"
    ));
154.                 book_type.setBook_type_name(book_type_name);
155.                 return book_type;
156.             }
157.         } catch (SQLException e) {
158.             e.printStackTrace();
159.
160.         }
161.         return null;
162.     }
163.     //根据书籍类型编号获得相应的书籍列表
164.     public ArrayList<booksBean>   getBookByBookTypeno(String book_typ
    e_no){
165.         ArrayList<booksBean> list=new ArrayList<booksBean>();
166.         DBUtil db=new DBUtil();
167.         String sql="select * from Book where book_type_no like '%"+bo
    ok_type_no+"%'";
168.         ResultSet rs=db.query(sql);
169.             try {
170.                 while(rs.next()){
171.                     booksBean book= new booksBean();
172.                     book.setBook_name(rs.getString("book_name"));
173.                     book.setBook_no(rs.getString("book_no"));
174.                     book.setBook_author(rs.getString("book_author"));

175.                     book.setBook_press(rs.getString("book_press"));
176.                     book.setBook_rest_num(rs.getInt("book_rest_num"))
    ;
177.                     book.setBook_type_no(rs.getString("book_type_no")
    );
178.                     list.add(book);
179.                 }
180.                 return list;
181.             } catch (SQLException e) {
182.                 // TODO Auto-generated catch block
183.                 e.printStackTrace();
184.         }
185.         return null;
186.     }
187.     //根据书名获得相应的书籍
188.     public booksBean getBookByname(String Book_name){
189.         DBUtil db = new DBUtil();
```

```
190.         String sql = "select * from Book where book_name like '%"+Boo
    k_name+"%'";
191.         ResultSet rs = db.query(sql);
192.             try {
193.                 if(rs.next()) {
194.                     booksBean book= new booksBean();
195.                     book.setBook_name(rs.getString("book_name"));
196.                     book.setBook_no(rs.getString("book_no"));
197.                     book.setBook_author(rs.getString("book_author"));

198.                     book.setBook_press(rs.getString("book_press"));
199.                     book.setBook_rest_num(rs.getInt("book_rest_num"))
    ;
200.                     book.setBook_type_no(rs.getString("book_type_no")
    );
201.                     return book;
202.                 }
203.
204.             } catch (SQLException e) {
205.                 // TODO Auto-generated catch block
206.                 e.printStackTrace();
207.
208.             }
209.             return null;
210.     }
211.
212.     //根据借书证获得借书信息
213.     public ArrayList<Borrowed_book> getBorrowMsgByScardNo(String Scar
    d_no){
214.         ArrayList<Borrowed_book> borrow_list=new ArrayList<Borrowed_b
    ook>();
215.         DBUtil db=new DBUtil();
216.         String sql="select * from borrowed_book where Scard_no like '
    %"+Scard_no+"%'";
217.         ResultSet rs=db.query(sql);
218.         try {
219.             while(rs.next()){
220.                 Borrowed_book borrow=new Borrowed_book();
221.                 borrow.setBook_no(rs.getString("book_no"));
222.                 borrow.setBorrowed_book_time(rs.getString("borrowed_b
    ook_time"));
223.                 borrow.setScard_no(rs.getString("Scard_no"));
224.                 borrow.setBorrowed_book_num(rs.getString("borrowed_bo
    ok_num"));
225.                 borrow_list.add(borrow);
```

```
226.             }
227.             return borrow_list;
228.         } catch (SQLException e) {
229.             e.printStackTrace();
230.         }
231.
232.
233.         return null;
234.     }
235.     //根据书籍编号获得借书信息
236.     public ArrayList<Borrowed_book> getBorrowMsgByBookNo(String book_
    no){
237.         ArrayList<Borrowed_book> borrow_list=new ArrayList<Borrowed_b
    ook>();
238.         DBUtil db=new DBUtil();
239.         String sql="select * from borrowed_book where book_no like '%
    "+book_no+"%'";
240.         ResultSet rs=db.query(sql);
241.         try {
242.             while(rs.next()){
243.                 Borrowed_book borrow=new Borrowed_book();
244.                 borrow.setBook_no(book_no);
245.                 borrow.setBorrowed_book_time(rs.getString("borrowed_b
    ook_time"));
246.                 borrow.setScard_no(rs.getString("Scard_no"));
247.                 borrow.setBorrowed_book_num(rs.getString("borrowed_bo
    ok_num"));
248.                 borrow_list.add(borrow);
249.             }
250.             return borrow_list;
251.         } catch (SQLException e) {
252.             e.printStackTrace();
253.         }
254.         return null;
255.     }
256.     //根据借书证获得还书信息
257.         public ArrayList<Return_book> getReturnMsgByScardNo(String Sc
    ard_no){
258.             ArrayList<Return_book> return_list=new ArrayList<Return_b
    ook>();
259.             DBUtil db=new DBUtil();
260.             String sql="select * from return_book where Scard_no like
     '%"+Scard_no+"%'";
261.             ResultSet rs=db.query(sql);
262.             try {
```

```
263.                 while(rs.next()){
264.                     Return_book returnBook=new Return_book();
265.                     returnBook.setBook_no(rs.getString("book_no"));
266.                     returnBook.setScard_no(rs.getString("Scard_no"));
267.                     returnBook.setReturn_book_num(rs.getString("retur
    n_book_num"));
268.                     returnBook.setReturn_book_time(rs.getString("retu
    rn_book_time"));
269.                     return_list.add(returnBook);
270.                 }
271.                 return return_list;
272.             } catch (SQLException e) {
273.                 e.printStackTrace();
274.             }
275.             return null;
276.         }
277.
278.         //根据书籍编号获得还书信息
279.         public ArrayList<Return_book> getReturnMsgByBookNo(String boo
    k_no){
280.             ArrayList<Return_book> return_list=new ArrayList<Return_b
    ook>();
281.             DBUtil db=new DBUtil();
282.             String sql="select * from return_book where book_no like
    '%"+book_no+"%'";
283.             ResultSet rs=db.query(sql);
284.             try {
285.                 while(rs.next()){
286.                     Return_book returnBook=new Return_book();
287.                     returnBook.setBook_no(rs.getString("book_no"));
288.                     returnBook.setScard_no(rs.getString("Scard_no"));
289.                     returnBook.setReturn_book_num(rs.getString("retur
    n_book_num"));
290.                     returnBook.setReturn_book_time(rs.getString("retu
    rn_book_time"));
291.                     return_list.add(returnBook);
292.                 }
293.                 return return_list;
294.             } catch (SQLException e) {
295.                 e.printStackTrace();
296.             }
297.             return null;
298.         }
```

```
299. }
```

在 StudentDAO 类中，我主要就写了一个函数

`getStudentByScardNo(String Scard_no)`

顾名思义，这个函数的作用就是通过 Scard_no 参数，调用数据库封装类进行查询操作，将各字段数据放入 StudentBean 实例对象中，然后返回 StudentBean 类型。

代码如下：

```
1. package DAO;
2.
3. import java.sql.ResultSet;
4. import java.sql.SQLException;
5.
6. import com.bean.StudentsBean;
7. import com.db.DBUtil;
8.
9. /*
10. * StudentDAO 类
11. * 通过此类将 Students 数据表与 web 的 Student 类联系起来
12. */
13.public class StudentDAO {
14.    //通过借书卡号获得学生实例
15.    public StudentsBean getStudentByScardNo(String Scard_no){
16.        DBUtil db=new DBUtil();
17.        String sql="select * from Students where Scard_no like '%"+
   Scard_no+"%'";
18.        ResultSet rs=db.query(sql);
19.        try {
20.            if(rs.next()){
21.                StudentsBean stu=new StudentsBean();
22.                stu.setSname(rs.getString("Sname"));
23.                stu.setSsex(rs.getString("Ssex"));
24.                stu.setGrade(rs.getString("grade"));
25.                stu.setSpwd(rs.getString("Spwd"));
26.                return stu;
27.            }
28.        } catch (SQLException e) {
29.            // TODO Auto-generated catch block
30.            e.printStackTrace();
```

```
31.         }
32.         return null;
33.     }
34.}
```

4.2 Servlet 包

如图所示：

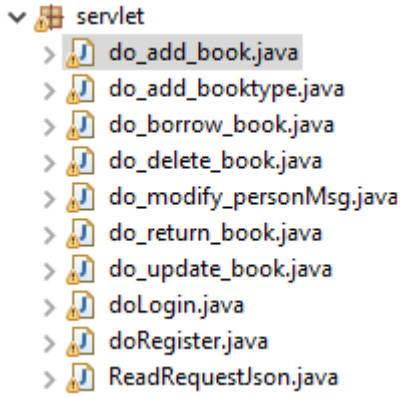

图 4-2 servlet 包

Servlet 包里的 java 代码主要是配合 jsp 实现逻辑控制，比如 jsp 的 login. jsp 通过表单提交方式，页面跳转到 do_login. java 去，然后在此 java 文件中进行集中地处理逻辑，整个 servlet 包主要有登陆、注册、读 Json（因为原生 java 是不支持 json 的，所以要导入 Microsoft 的 gson 包同时自己写个处理 json 函数），修改个人信息，更新书本信息，增加书本、删除书本、借阅、归还书籍等等功能。

每个 java 代码都是继承 HttpServlet 类，然后重写 doGet()和 doPost()方法。

主要代码如下所示：

doLogin. java:

```
1. package servlet;
2.
3. import java.io.IOException;
4. import java.sql.ResultSet;
5. import java.sql.SQLException;
6. import java.util.HashMap;
7. import java.util.Map;
8.
9. import javax.servlet.ServletException;
```

```
10. import javax.servlet.http.HttpServlet;
11. import javax.servlet.http.HttpServletRequest;
12. import javax.servlet.http.HttpServletResponse;
13. 
14. import com.db.DBUtil;
15. 
16. /*
17.  * 读者登陆操作
18.  */
19.     public class doLogin extends HttpServlet {
20. 
21.         public void doGet(HttpServletRequest request, HttpServletRe
    sponse response)
22.                 throws ServletException, IOException {
23.             String Scard_no = request.getParameter("Scard_no");//获
    取借书卡编号
24.             String Spwd = request.getParameter("Spwd");//获取密码
25.             String adminflag="false";
26.             //System.out.println(Scard_no);
27.             //System.out.println(Spwd);
28.             DBUtil db = new DBUtil();
29.             String sql ="select * from Students where Scard_no='"+S
    card_no+"' and Spwd ='"+Spwd+"'";
30.             ResultSet rs = db.query(sql);
31.             try {
32.                 if(rs.next()){
33.                     adminflag = rs.getString("adminflag").toString(
    );
34.                     //System.out.print(adminflag); true
35.                     request.getSession().setAttribute("adminflag",a
    dminflag);
36.                     request.getSession().setAttribute("Scard_no", S
    card_no);
37.                     request.getRequestDispatcher("stu_library.jsp")
    .forward(request, response);
38.                 }else{
39.                     //StudentLogin.setLogined(false);
40.                     response.sendRedirect("login_failed.jsp");
41.                 }
42.             } catch (SQLException e) {
43.                 // TODO Auto-generated catch block
44.                 e.printStackTrace();
45.             }
46. 
47.         }
```

```
48.         public void doPost(HttpServletRequest request, HttpServletR
    esponse response)
49.                 throws ServletException, IOException {
50.
51.             doGet(request,response);
52.         }
53.
54.}
```

do_add_book. java:

```
1. package servlet;
2.
3. import java.io.IOException;
4. import java.io.PrintWriter;
5. import java.util.regex.Matcher;
6. import java.util.regex.Pattern;
7.
8. import javax.servlet.ServletException;
9. import javax.servlet.http.HttpServlet;
10.import javax.servlet.http.HttpServletRequest;
11.import javax.servlet.http.HttpServletResponse;
12.
13.import com.db.DBUtil;
14./*
15. * 增加书本
16. */
17.public class do_add_book extends HttpServlet {
18.
19.
20.    public void doGet(HttpServletRequest request, HttpServletRespon
    se response)
21.             throws ServletException, IOException {
22.
23.         response.setContentType("text/html");
24.         PrintWriter out = response.getWriter();
25.         response.setCharacterEncoding("utf-8");
26.
27.         String book_no=request.getParameter("book_no");
28.         String book_name=request.getParameter("book_name");
29.         String book_author=request.getParameter("book_author");
30.         String book_press=request.getParameter("book_press");
31.         String book_rest_num=request.getParameter("book_rest_num");
```

```
32.         String book_type_no = request.getParameter("book_type_no");

33.         String book_details=request.getParameter("book_details");
34.         String picture = request.getParameter("picture");
35.
36.         DBUtil db=new DBUtil();
37.         String sql = "insert into Book values ('"+book_no+"','"+boo
    k_name+"','"+book_author+"'," +
38.                 "'"+book_press+"','"+Integer.parseInt(book_rest_num
    )+"','"+book_type_no+"','"+book_details+"','"+picture+"')";
39.         db.update(sql);
40.
41.         String loginURL = "http://localhost:8080/library/book.jsp";

42.         String setTime = "3";
43.         String say = "添加书籍成功！正在跳转到书籍页面";
44.         response.sendRedirect("goto.html?gotoURL=" + loginURL + "&s
    etTime=" + setTime + "&say=" + java.net.URLEncoder.encode(say,"utf8
    "));
45.     }
46.
47.
48.     public void doPost(HttpServletRequest request, HttpServletRespo
    nse response)
49.             throws ServletException, IOException {
50.
51.         doGet(request, response);
52.
53.     }
54.
55.}
```

Do_delete_book.java:

```
1. package servlet;
2.
3. import java.io.IOException;
4. import java.sql.ResultSet;
5. import java.sql.SQLException;
6.
7. import javax.servlet.ServletException;
8. import javax.servlet.http.HttpServlet;
9. import javax.servlet.http.HttpServletRequest;
10.import javax.servlet.http.HttpServletResponse;
11.import javax.swing.JOptionPane;
```

```
12.
13.import java.io.IOException;
14.import com.db.DBUtil;
15.import com.google.gson.JsonObject;
16.import com.google.gson.JsonParser;
17.
18.public class do_delete_book extends HttpServlet {
19.    public void doGet(HttpServletRequest request, HttpServletRespon
   se response)
20.            throws ServletException, IOException {
21.        //int res = JOptionPane.showConfirmDialog(null, "是否继续操
   作", "是否继续", JOptionPane.YES_NO_OPTION);
22.        //System.out.println("进来");
23.        String json = ReadRequestJson.readJSONString(request);
24.        JsonObject object = new JsonParser().parse(json).getAsJsonO
   bject();
25.        String book_no = object.get("book_no").getAsString();
26.
27.        DBUtil db = new DBUtil();
28.        String sql="select * from borrowed_book where book_no='"+bo
   ok_no+"'";
29.        if(db.query(sql)!=null) {
30.            //说明有外借，不能直接删除
31.            sql="delete from borrowed_book where book_no='"+book_no
   +"'";
32.            db.delete(sql);//先删除在借表记录
33.            sql="delete from Book where book_no='"+book_no+"'";
34.            db.delete(sql);//再删 book 表
35.        }
36.        else {
37.            //直接删除 book 数据
38.            sql ="delete from Book where book_no='"+book_no+"'";
39.            db.delete(sql);
40.        }
41.    }
42.    public void doPost(HttpServletRequest request, HttpServletRespo
   nse response)
43.            throws ServletException, IOException {
44.
45.        doGet(request,response);
46.    }
47.}
```

Do_borrow_book.java:

```
1. package servlet;
2. 
3. import java.io.IOException;
4. import java.io.PrintWriter;
5. import java.sql.ResultSet;
6. import java.sql.SQLException;
7. import java.text.SimpleDateFormat;
8. import java.util.Date;
9. 
10.import javax.servlet.ServletException;
11.import javax.servlet.http.HttpServlet;
12.import javax.servlet.http.HttpServletRequest;
13.import javax.servlet.http.HttpServletResponse;
14.
15.import com.db.DBUtil;
16./*
17. * 借书操作
18. */
19.public class do_borrow_book extends HttpServlet {
20.
21.
22.    public void doGet(HttpServletRequest request, HttpServletRespon
   se response)
23.            throws ServletException, IOException {
24.
25.        response.setContentType("text/html");
26.
27.        //设置日期格式
28.        SimpleDateFormat sdf=new SimpleDateFormat("yyyy-MM-
   dd HH:mm:ss");
29.        Date date=new Date();
30.
31.        //获取表单信息
32.        String Scard_no = request.getParameter("Scard_no");
33.        String book_no=request.getParameter("book_no");
34.        String borrowed_book_num=request.getParameter("borrowed_boo
   k_num");
35.
36.        DBUtil db=new DBUtil();
37.        String sql = "insert into borrowed_book values ('"+book_no+
   "','"+Scard_no+"','"+
38.        Integer.parseInt(borrowed_book_num)+"','"+sdf.format(date)+
   "')";
39.
40.
```

```
41.         String sql1="select book_rest_num from Book where book_no='
    "+book_no+"'";//查询该书编号对应的书的剩余数量
42.         ResultSet rs=db.query(sql1);
43.         int book_rest_num = 0;
44.         try {
45.             if(rs.next()){
46.                     book_rest_num=rs.getInt("book_rest_num");
47.             }
48.         } catch (SQLException e) {
49.             // TODO Auto-generated catch block
50.             e.printStackTrace();
51.         }
52.         int book_num=book_rest_num-
    Integer.parseInt(borrowed_book_num);//剩余书数量-借走的数量
53.         String sql2="update Book set book_rest_num='"+book_num+"'wh
    ere book_no='"+book_no+"'";
54.
55.         String loginURL="";
56.         String setTime = "";
57.         String say="";
58.         //判断 book_num 是否大于 0，以此来决定是否将书借出
59.         if (book_num>=0) {
60.             db.update(sql1);
61.             db.update(sql2);
62.             loginURL = "http://localhost:8080/library/stu_library.j
    sp";
63.             say = "恭喜"+Scard_no+"借书成功！";
64.             response.sendRedirect("goto.html?gotoURL=" + loginURL +
     "&setTime=" + setTime +
65.                     "&say=" + java.net.URLEncoder.encode(say,"utf8"
    ));
66.         }else{//要借的书数目大于剩余数量
67.             loginURL = "http://localhost:8080/library/borrow_book.j
    sp";
68.             setTime = "10";
69.             say = "很抱歉，"+Scard_no+"该书剩余数目不足，请重新借书！
    ";
70.             response.sendRedirect("goto.html?gotoURL=" + loginURL +
     "&setTime=" + setTime +
71.                     "&say=" + java.net.URLEncoder.encode(say,"utf8"
    ));
72.         }
73.
74.
75.     }
```

```
76.
77.
78.     public void doPost(HttpServletRequest request, HttpServletRespo
    nse response)
79.             throws ServletException, IOException {
80.         doGet(request, response);
81.     }
82.
83.}
```

4.3 Jsp 代码

Jsp 实质是 servlet，而 servlet 是服务端的，所以 jsp 是属于后端的，但在 mvc 分层中，jsp 属于 view 层，现在 jsp 更多的作用变成了数据显示和页面显示。

在本次作业中，我 jsp 主要是数据显示和页面显示，但同时为了更好的和 java 端契合，我也在 jsp 中通过注入 java 代码的形式进行了一些逻辑业务处理。

主要代码如下：

Register.jsp:

```
1. <%@ page language="java" import="java.util.*" contentType="text/html; charset=utf-8"%>
2. <%
3. String path = request.getContextPath();
4. String basePath = request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";
5. %>
6. <!-- 使用过渡型文档 -->
7. <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
8. <html>
9.   <head>
10.    <base href="<%=basePath%>">
11.
12.    <title>注册</title>
13.    <!-- 拒绝缓存，让页面每次进入都重新加载 -->
14.    <meta http-equiv="pragma" content="no-cache">
15.    <meta http-equiv="cache-control" content="no-cache">
16.    <meta http-equiv="expires" content="0">
17.    <meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
18.    <meta http-equiv="description" content="This is my page">
19.    <!-- 调用 register css 样式表 -->
20.    <link rel="stylesheet" type="text/css" href="./css/register.css">
21.
22.
23.  </head>
24.  <%
25.   request.setCharacterEncoding("utf-8");
26.   %>
27.  <body>
28.   <main>
29.        <div class="container">
30.        <div class="logo"></div>
31.            <fieldset>
32.            <form action="doRegister" method="post">
33.
34.            <label for="Scard_no">借书证编号：</label>
35.            <br>
36.            <input type="text" placeholder="请输入借书证编号" name="Scard_no" pattern="[0-9]{10}" title="A credit
37.card number is 10 digits with no spaces or dashes" required>
38.            <br>
39.
40.            <label for="Spwd">密码：</label>
```

```
            <br>
            <input type="password" placeholder="请输入密码
" name="Spwd" required>
            <br>

            <label for="Sname">姓名：</label>
                <br>
            <input type="text" placeholder="请输入姓名
" name="Sname" required>
            <br>

            <label for="Ssex">性别：</label>
            <input type="radio" name="Ssex" value="男" >男
            <input type="radio" name="Ssex" value="女">女
            <br>

            <label for="grade">年级：</label>
            <!-- 复选框 -->
            <select  name="grade">
                <option value="大一">大一</option>
                <option value="大二">大二</option>
                <option value="大三">大三</option>
                <option value="大四">大四</option>
            </select>
            <br>
            <!-- 表单提交 -->
            <input type="submit" class="register" value="注册
"></a>
    </form>
    </fieldset>
        </div>
    </main>
    <footer></footer>
  </body>
</html>
```

Stu_library.jsp:
主页面：

```
<%@ page language="java" import="java.util.*" contentType="text/htm
l; charset=utf-8"%>
<%
String path = request.getContextPath();
String basePath = request.getScheme()+"://"+request.getServerName()
+":"+request.getServerPort()+path+"/";
```

```
5. %>
6.
7. <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
8. <html>
9.   <head>
10.    <base href="<%=basePath%>">
11.
12.    <title>学生图书管理系统</title>
13.
14.    <meta http-equiv="pragma" content="no-cache">
15.    <meta http-equiv="cache-control" content="no-cache">
16.    <meta http-equiv="expires" content="0">
17.    <meta http-
   equiv="keywords" content="keyword1,keyword2,keyword3">
18.    <meta http-equiv="description" content="This is my page">
19.
20.    <link rel="stylesheet" type="text/css" href="./css/stu_library.
   css">
21.  </head>
22.
23.  <body>
24.   <header>
25.   <img src="./images/top_backgroup.jpg" alt="无法加载图片资源">
26.    </header>
27.    <div class="mydiv">
28.    <ul>
29.        <li>
30.        <a href="stu_library.jsp">首页</a>
31.            <ul class="second-ul">
32.                <li><a href="register.jsp">注册</a></li>
33.            </ul>
34.        </li>
35.        <li>
36.        <a href="stu_library.jsp">书籍借阅</a>
37.            <ul class="second-ul">
38.                <li><a href="book.jsp">查看书库</a></li>
39.                <li><a href="query_borrow.jsp">借书查询</a></li>
40.                <li><a href="query_return.jsp">还书查询
   </a></li>
41.            </ul>
42.        </li>
43.        <li>
44.        <a href="stu_library.jsp">书籍管理</a>
45.            <ul class="second-ul">
46.                <li><a href="query_book.jsp">书籍查询</a></li>
```

```
47.                 <li id="admin1"><a href="add_book.jsp">添加书籍
   </a></li>
48.                 <li id="admin2"><a href="add_book_type.jsp">添加书类
   别</a></li>
49.             </ul>
50.         </li>
51.         <span class="exit">
52.         <a href="person_msg.jsp?Scard_no=<%=request.getSession().ge
   tAttribute("Scard_no") %>" title="查看个人资料
   ">(<%=request.getSession().getAttribute("Scard_no")%>)</a>
53.         <a href="login.jsp">退出</a>
54.         </span>
55.     </ul>
56.     </div>
57.     <main>
58.         <div class="container">
59.             <div class="logo"></div>
60.             <div class="content">
61.                 <div class="say">欢迎来到学生图书馆，我要
   <a href="borrow_book.jsp">借书</a>或<a href="return_book.jsp">还书
   </a>! </div>
62.             </div>
63.         </div>
64.     </main>
65.     <footer>
66.     <p>计科 1707 罗家璇版权所有</p>
67.     </footer>
68.   </body>
69.
70.</html>
71.<script>
72.     <%String adminflag=request.getSession().getAttribute("adminflag
   ").toString();%>//通过 session 拿到 adminflag 的值
73.     //如果不是管理员，那么不能添加书籍也不能添加书籍类型，也不能修改书库
   里书的信息和删除列表中的书籍
74.     var adminflag = <%=adminflag%>;
75.     adminflag=adminflag.toString();
76.     //alert(adminflag);
77.     if(adminflag!='true')//非管理员账号
78.         {
79.         //清除掉添加书籍按钮
80.             document.getElementById("admin1").innerHTML="";
81.             document.getElementById("admin2").innerHTML="";
82.         }
83.</script>
```

Book.jsp:

```
<%@ page language="java" import="java.util.*" contentType="text/html; charset=utf-8"%>
<%@ page import="DAO.BookDAO" %>
<%@ page import="com.bean.booksBean" %>

<%
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>书库</title>

    <meta http-equiv="pragma" content="no-cache">
    <meta http-equiv="cache-control" content="no-cache">
    <meta http-equiv="expires" content="0">
    <meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
    <meta http-equiv="description" content="This is my page">

    <link rel="stylesheet" type="text/css" href="./css/book.css">


  </head>

  <body>
   <header>

    </header>
    <div style="background:url('./images/top_backgroup.jpg');width:76.47%;height:300px;">
            <input type="text" placeholder="请输入关键词" name="key" value="" id="key">
            <input type="button" value="高亮" name="button" id="button">
    </div>
    <div class="mydiv">
```

```
37.     <ul>
38.         <li>
39.         <a href="stu_library.jsp">首页</a>
40.             <ul class="second-ul">
41.                 <li><a href="register.jsp">注册</a></li>
42.             </ul>
43.         </li>
44.         <li>
45.         <a href="stu_library.jsp">书籍借阅</a>
46.             <ul class="second-ul">
47.                 <li><a href="book.jsp">查看书库</a></li>
48.                 <li><a href="query_borrow.jsp">借书查询</a></li>
49.                 <li><a href="query_return.jsp">还书查询
   </a></li>
50.             </ul>
51.         </li>
52.         <li>
53.         <a href="stu_library.jsp">书籍管理</a>
54.             <ul class="second-ul">
55.                 <li><a href="query_book.jsp">书籍查询</a></li>
56.                 <li id="admin1"><a href="add_book.jsp">添加书籍
   </a></li>
57.                 <li id="admin2"><a href="add_book_type.jsp">添加书类
   别</a></li>
58.             </ul>
59.         </li>
60.         <span class="exit">
61.         <a href="person_msg.jsp?Scard_no=<%=request.getSession().ge
   tAttribute("Scard_no") %>" title="查看个人资料
   ">(<%=request.getSession().getAttribute("Scard_no")%>)</a>
62.         <a href="login.jsp">退出</a>
63.         </span>
64.     </ul>
65.     </div>
66.     <main>
67.         <div class="container">
68.
69.             <table>
70.             <tr>
71.
72.              <td>
73.             <!-- 商品循环开始 -->
74.                 <%
75.              BookDAO book = new BookDAO();
```

```
76.             ArrayList<booksBean> booklist = book.getAllBooksItem()
    ;
77.             if(booklist!=null&&booklist.size()>0){
78.                for(int i=0;i<booklist.size();i++){
79.                booksBean booksBean=booklist.get(i);
80.             %>
81.                <div class="bookitem">
82.                    <dl>
83.                        <dt><a href="book_detail.jsp?book_no=<%=boo
    ksBean.getBook_no()%>"><img src="./images/<%=booksBean.getPicture()
     %>" alt="无法加载图片资源"></a></dt>
84.                        <dd class="content"><a href="book_detail.js
    p?book_no=<%=booksBean.getBook_no()%>"><%=booksBean.getBook_name()
    %></a></dd>
85.                    </dl>
86.                </div>
87.                <%
88.                }
89.            }
90.             %>
91.            </td>
92.            <!-- 商品循环结束 -->
93.
94.        </tr>
95.    </table>
96.<script>
97.                var contentarr=document.getElementsByTagName('dd');

98.                var aArr=[];
99.                //console.log(contentarr);
100.                //alert(contentarr);
101.                //alert("0");
102.                for(var i=0;i<contentarr.length;i++){
103.                    var content=contentarr[i].getElementsByTagName
    ('a')[0];//获取 a 标签内的对象
104.                    aArr.push(contentarr[i].getElementsByTagName('
    a')[0]);//存入数组中
105.                    //alert(contentarr[i].getElementsByTagName('a'
    )[0].innerHTML);
106.                }
107.                var key = document.getElementById("key");//获取关键
    字
108.                var button = document.getElementById("button");//
    获取 button
109.                button.onclick=function(){
```

```
                    var value=key.value;//获得关键词文本
                    //alert("关键词："+value);
                    for(var j=0;j<aArr.length;j++){
                        var content=aArr[j].innerHTML;//获得书籍书名内容
                        //alert("书籍名："+content);
                        var values = content.split(value);//按关键词分词
                        //alert(values);
                        aArr[j].innerHTML=values.join('<span style="background:red;">'+value+'</span>');
                    }
                }
            </script>
        </div>
    </main>
  </body>
  <footer>
    <p>计科1707罗家璇版权所有</p>
    </footer>
</html>
<script>
    <%String adminflag=request.getSession().getAttribute("adminflag").toString();%>//通过session拿到adminflag的值
    //如果不是管理员，那么不能添加书籍也不能添加书籍类型，也不能修改书库里书的信息和删除列表中的书籍
    var adminflag = "<%=adminflag%>";
    //alert(adminflag);
    if(adminflag!="true")//非管理员账号
        {
        //清除掉添加书籍按钮
            document.getElementById("admin1").innerHTML="";
            document.getElementById("admin2").innerHTML="";
        }
</script>
```

5 实验运行结果

5.1 登陆界面

如下图所示：

图 5-1 登陆界面

5.2 主界面

图 5-2 library 主界面

可以看到，左侧为菜单栏，将鼠标移到一级菜单上会出现二级菜单：

图 5-2-1 library 主界面菜单

5.3 书籍管理之查看书库

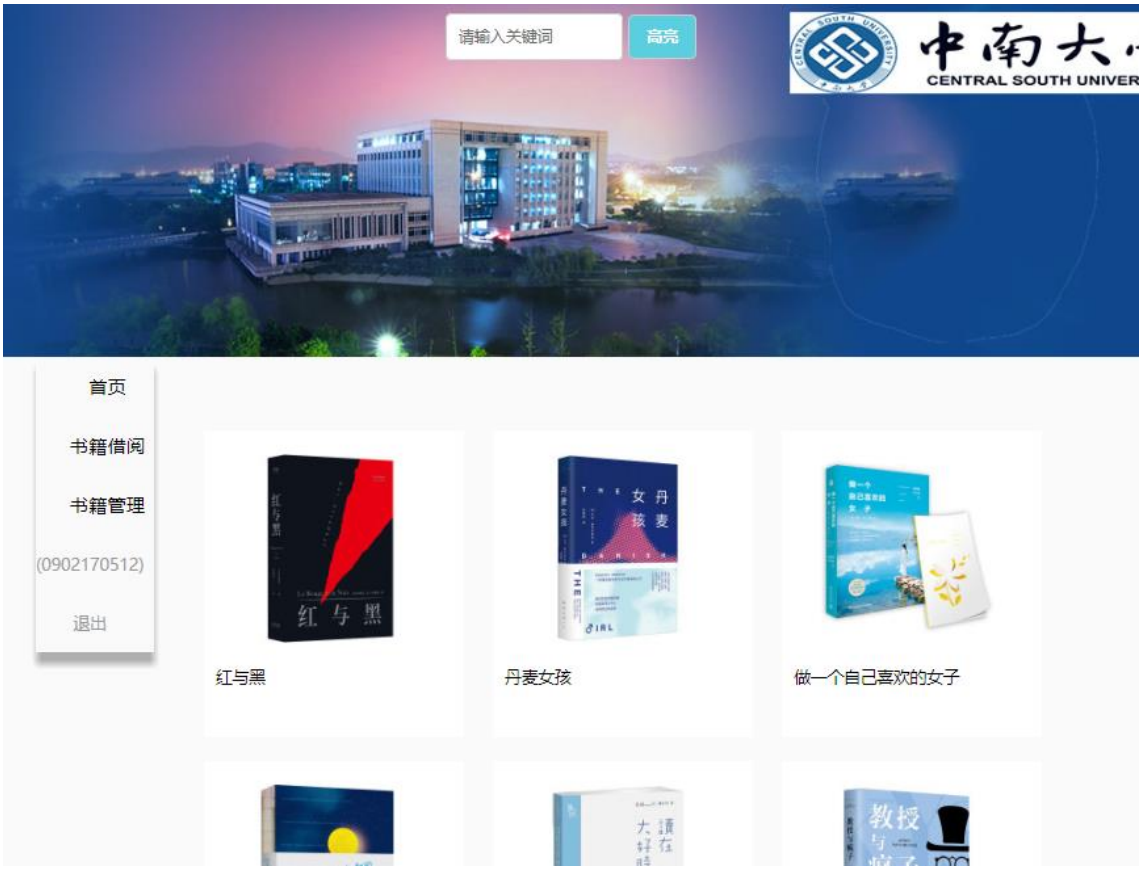

图 5-3 library 书库界面展示

可以看到，书库内展示了所有的书籍，顶部栏有搜索框，可以输入关键字高亮所示：

图 5-3-1 library 书库高亮搜索

5.4 权限切换

图 5-4-1 管理员界面

图 5-4-2　普通用户界面

可以看到，目前该用户是管理员，管理员可以添加书籍和添加书类别，也可以删除书籍，而非管理员只能借阅归还书籍不能对书本进行修改等操作。

5.5　书籍删除（需要管理员权限）

图 5-5-2　管理员按钮

如图所示，当点击删除该书时，会删除数据库的该书数据，前端也同时会删除展示内容。

5.6 书籍模糊查询

图 5-6 书籍模糊查询

可以看到，我仅仅在出版社那一栏输入了“人民”两个字，但是照样检索出了“红与黑”这本书，出版社为“人民出版社”，其实原理很简单，就是在数据库查询时使用‘where XXX like %’这种形式即可。

6 实验心得

这次 web 实验并不简单，而且我出于兴趣做了许多额外的功能，完善了借阅书籍，归还书籍等等操作，所以碰到了许许多多的问题，最后所幸都解决了。

其中有几个我印象比较深刻的：

6.1 在使用 ajax 进行删除书籍操作时

我发现明明写法跟 w3school 等网站上的 ajax 获取对象然后 send 等操作一模一样，但是就是一直 404 错误，找不到我 post 传过去的 url，我一开始怀疑是 web.xml 文件没有把 do_delete_book.java 写入 servlet-mapping，但是后来检查 web.xml 发现我写进去了，而且也没有写错，搞了半天一直 404，于是我尝试不用映射，直接传绝对地址进去，发现没有 404 了，这次链接服务器成功了，我仔细检查了 ajax 实例对象的格式，发现如果 url 把'/do_delete_book'改成 'do_delete_book'就可以连接上，但是我同学就是加了'/'也能访问，我想大概是我用的 jsp 他用的 servlet 的原因，或者就是版本不同。

```
function mydelete(){
    var bookno="<%=book.getBook_no()%>";//获取编号
    //alert(bookno);
    if(confirm("请确认删除!")){
        var request = ajaxFunction();
        //request.open("GET","/do_delete_book?book_no="+bookno,true);
        request.open("POST","do_delete_book",true);
        request.setRequestHeader('Content-type','application/x-www/form-urlencoded;charset=utf-8');
        var data={
            book_no:bookno
        };
        //alert(data["book_no"]);
        var json = JSON.stringify(data);
        //alert(json);
        //request.send();
        request.send(json);
        request.onreadystatechange=function(){
            //alert(1);
            if(request.readyState==4&&request.status==200){
                var lURL = "http://localhost:8080/library/book.jsp";
                window.location.href=lURL;
            }
        }
    }
}
```

图 6-1 ajax 实例

6.2 关键字高亮技术

之前我用搜索框搜完后，想要加个高亮，但第一个问题就是，我的书库里书为了美观都是带有图片的一个一个有序排列的对象，这导致我在获得图片下面的文字关键字时需要链式调用 getElementsByTagName 或者 getElementsByClass 等几次，写着写着就把自己弄晕了，我把关键字找到，然后写完高亮代码后，运行发现并没有高亮出现，定睛找了一会才发现，我要改变的样式应该要赋值给 content.innerHTML，但我之前为了简便，用了 var 变量代替了 content.innerHTML，导致 var 变量失去了 content 最源头的地址信息，导致我最后把新样式赋给 innerHTML 时，document 并未发生改变。

```
<script>
        var contentarr=document.getElementsByTagName('dd');
        var aArr=[];
        //console.log(contentarr);
        //alert(contentarr);
        //alert("0");
        for(var i=0;i<contentarr.length;i++){
            var content=contentarr[i].getElementsByTagName('a')[0];//获取a标签内的对象
            aArr.push(contentarr[i].getElementsByTagName('a')[0]);//存入数组中
            //alert(contentarr[i].getElementsByTagName('a')[0].innerHTML);
        }
        var key = document.getElementById("key");//获取关键字
        var button = document.getElementById("button");//获取button
        button.onclick=function(){
            var value=key.value;//获得关键词文本
            //alert("关键词："+value);
            for(var j=0;j<aArr.length;j++){
                var content=aArr[j].innerHTML;//获得书籍书名内容
                //alert("书籍名："+content);
                var values = content.split(value);//按关键词分词
                //alert(values);
                aArr[j].innerHTML=values.join('<span style="background:red;">'+value+'</span>');
            }
        }
    </script>
```

图 6-2 关键字高亮技术

结束语

这次 web 实验虽然时间不长，但我巩固了之前所学的知识，知道了前后端如何交互，如何使用 ajax，它们的原理是什么，我实验后端用的是 jsp 和 java 代码，由于 jsp 可以在前端嵌入 java 代码，导致我 jsp 代码一度很乱，里面既有 html 还有 js 还有导入的 java 语句，这虽然将前后端揉在一起不用切来切去，写代码写的很快，但是逻辑出错调试时非常麻烦，我完全不知道到底是哪部分出了问题，在实验中途曾经一度想直接把 jsp 全删了，用 php 来写，把前端和后台分离来写，但后来写顺手了，觉得 jsp 也挺好用的，只要注意格式，不要让 java 代码导入注入，在规定的模块写 java 代码，规定的模块写 js 代码，同时写好注释，其实也是可以很清晰而且高效的。

总的来说，这次 web 实验还算成功，我学到了一些代码知识的同时，也启发我对规范代码的思考，以后一定要规范写代码，不要自己给自己埋坑，留下隐患 bug。

参考文献

[1]李俊青，季文天.高职 WEB 程序设计实训课程探索与实践[J].职业教育研究，2009（9）.

[2]唐灿.下一代 Web 界面前端技术综述[J].重庆工商大学学报：自然科学版，2009（8）.

[3]曹刘阳.编写高质量代码——Web 前端开发修炼之道[M].北京：机械工业出版社，2010.