武汉纺织大学

Web应用开发课程设计

**图书管理系统**

**学 院： 数学与计算机学院**

**班 级： 物联网11802**

**姓 名： 蒋光鑫**

**学 号： 1804240611**

**指导老师： 聂刚**

**成 绩：**

**完成日期： 2020年12月15日**

目 录

[1 需求分析 1](#_Toc59467665)

[1.1用户登陆 1](#_Toc59467666)

[1.2主页面 1](#_Toc59467667)

[1.3检索书籍 1](#_Toc59467668)

[1.4 还书 1](#_Toc59467669)

[1.5查询借阅历史 1](#_Toc59467670)

[1.6修改密码 1](#_Toc59467671)

[2 系统设计 2](#_Toc59467672)

[2.1 总体ER图 2](#_Toc59467673)

[2.2 UML时序图（Sequence Diagram） 2](#_Toc59467674)

[2.3 UML活动图（Activity Diagram） 2](#_Toc59467675)

[2.3.1 登录 2](#_Toc59467676)

[2.3.2 查书 3](#_Toc59467677)

[2.3.3 还书 4](#_Toc59467678)

[3 系统实现 4](#_Toc59467679)

[3.1 项目结构 4](#_Toc59467680)

[3.2 配置文件 5](#_Toc59467681)

[3.2.1 jdbc.properties文件 5](#_Toc59467682)

[*3.2.2* 包文件 6](#_Toc59467683)

[3.3 VO类User.java 6](#_Toc59467684)

[3.4 DAO接口类 6](#_Toc59467685)

[3.5 控制器Controller 7](#_Toc59467686)

[3.5.1 登录控制器 7](#_Toc59467687)

[3.5.2 借书功能控制器 8](#_Toc59467688)

[3.5.3修改密码控制器 9](#_Toc59467689)

[3.6过滤器filter文件 10](#_Toc59467690)

[3.7 前端界面Html、css、js文件 12](#_Toc59467691)

[4 系统测试 12](#_Toc59467692)

[5 系统总结 15](#_Toc59467693)

# 1 需求分析

设计一个图书馆图书管理系统，实现读者的“登录”、查询书籍、借阅书籍、查询历史、修改密码等功能，具体要求如下

设计一个超市管理系统，实现“登陆”、“收银”、“查询统计”、“商品维护”、“密码修改”、“数据导出”等功能，具体要求如下：

## 1.1用户登陆

读者用浏览器进入登录页面，输入学号密码和相应的验证码，信息和数据库内的信息不符时，跳转至error页面，显示出错误信息，10s后返回登录页面，再次等待用户输入提交信息；验证信息输入正确后进入图书管理主页面。

## 1.2主页面

显示图书馆的展示信息，主页面右上角显示出当前登录用户的信息，主页面用轮播图展示图书信息，页面下面设置检索书籍、还书、更多信息图块，当用户点击图块时，跳转至相应的页面；右上角设置安全退出按钮，当用户点击“安全退出”时，系统退出当前登录用户，清除cookier信息，跳转至登录页面。

## 1.3检索书籍

用户点击书籍检索，进入搜索页面，在搜索框内输入书籍名称或者ID查询数据库中的书籍，若无该书籍信息，则在搜索框下方显示出来，如果存在该书籍，跳转至结果页面，列表显示书籍的信息和借阅状态，支持模模糊查询。

用户点击查阅结果最后的“借阅”实现借阅该书籍功能。

## 1.4 还书

用户点击“还书”，进入还书页面，输入待归还书籍的信息和借书用户信息，后端从数据库中查取检验信息，如果无误，给数据库修改信息，页面给出提示“还书成功！”

## 1.5查询借阅历史

在用户信息页面时，点击借阅历史，显示出当前用户的借阅图书历史，包括借阅书籍名称、借阅时间、还书时间等

## 1.6修改密码

用户点击修改密码，系统跳转至修改密码页面，输入新密码确认提交后，系统后端验证当前用户身份，如果身份无误，调用方法修改用户数据库信息，完成用户密码的修改。

# 2 系统设计

## 2.1 总体ER图

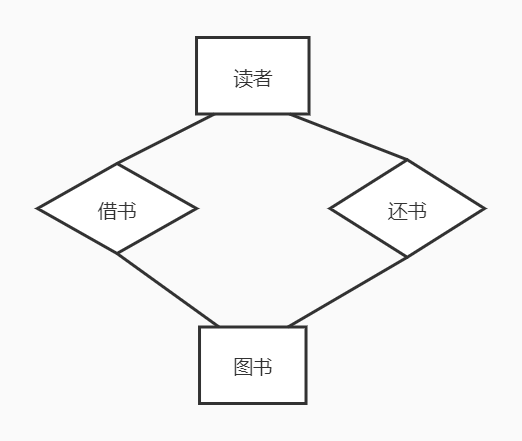


图 1

## 2.2 UML时序图（Sequence Diagram）

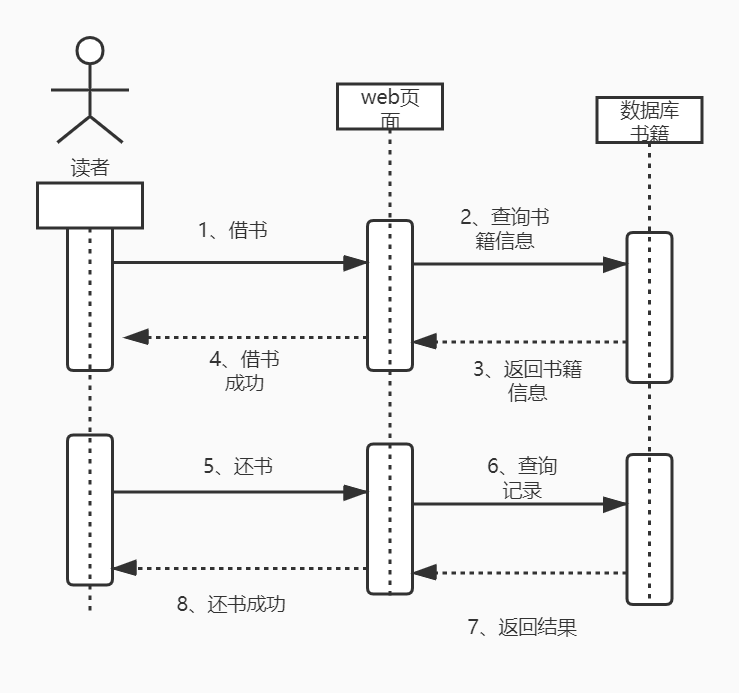


图 2

## 2.3 UML活动图（Activity Diagram）

### 2.3.1 登录

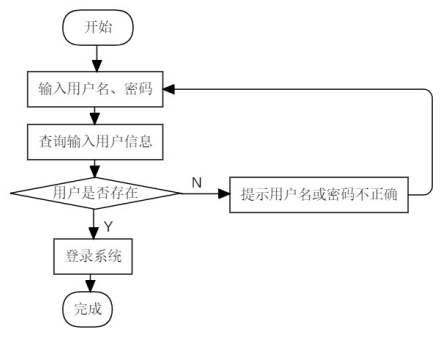


图 3

### 2.3.2 查书

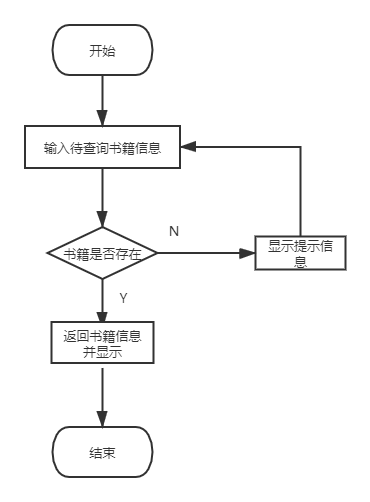


图 4

### 2.3.3 还书

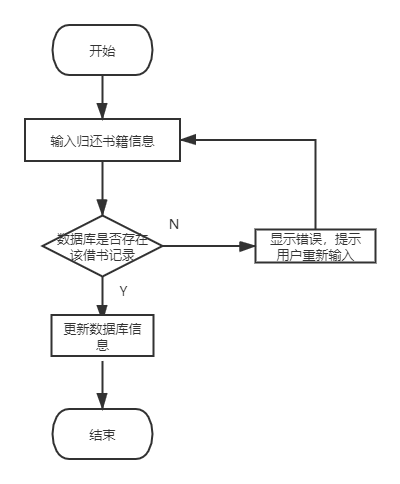


图 5

# 3 系统实现

## 3.1 项目结构

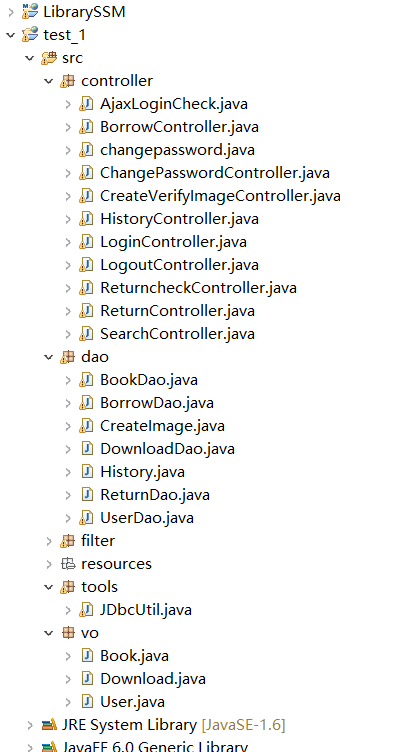


图 6（后端代码）

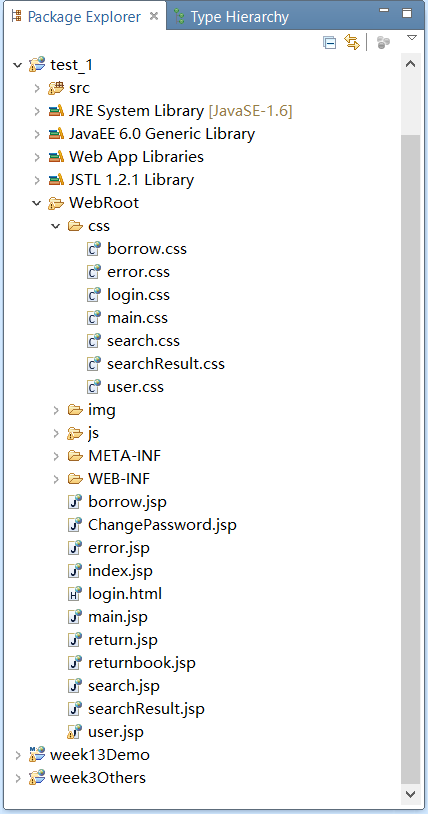


图 7（前端代码）

## 3.2 配置文件

### 3.2.1 jdbc.properties文件

该配置文件主要是以文件形式保存数据库的驱动类名称，连接数据库的URL地址，访问数据库的用户名及对应的密码，程序运行时会读取该文件相关信息，避免硬编码，当相关信息发生变化时，只需修改配置文件而不用修改源代码，增加程序的可扩展型。

DBDRIVER=com.mysql.jdbc.Driver

DBURL=jdbc:mysql://localhost:3306/test1?useUnicode=true&characterEncoding=UTF-8

DBUSER=root

PASSWORD=149920

### *3.2.2* 包文件

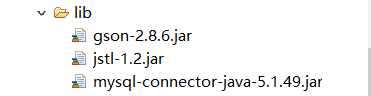
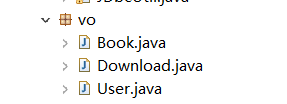


图 8

## 3.3 VO类User.java

用户vo类，书籍vo类和下载vo类



public class Book {

//书编号ID

private String bookID;

//书名，汉字

private String bookName;

//书籍分类

private String bookCategory;

//书籍简介

private String bookIntroduction;

public class User{

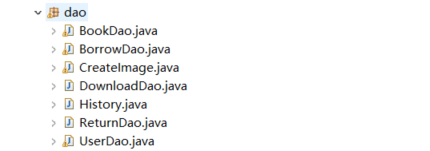
private String userName;

private String password;

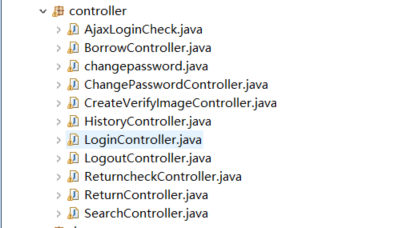
private String chrName;

}

## 3.4 DAO接口类



## 3.5 控制器Controller



### 3.5.1 登录控制器

public class LoginController extends HttpServlet {

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

request.setCharacterEncoding("utf-8");

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

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

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

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

HttpSession session = request.getSession();

String saveVcode = (String) session.getAttribute("verifyCode");

String forwardPath = "";

//比较输入的验证码和生成的验证码是否一样

if(!vcode.equalsIgnoreCase(saveVcode)){

request.setAttribute("info", "验证码不正确！");

forwardPath = "/error.jsp";

}

else{//验证码正确

UserDao userDao = new UserDao();

User user = userDao.get(userName);

if(user==null){//用户名不存在！

request.setAttribute("info", "你输入的用户名不存在！");

forwardPath="error.jsp";

}

else{

if(!user.getPassword().equals(password)){

request.setAttribute("info", "你输入的密码不正确！");

forwardPath="error.jsp";

}

else{

if("on".equalsIgnoreCase(autoLogin)){

Cookie cookie=new Cookie("username",userName);

cookie.setPath("/");

cookie.setMaxAge(60\*60\*24\*7);

response.addCookie(cookie);

}

session.setAttribute("currentUser", user);

forwardPath="/main.jsp";

}

}

}

RequestDispatcher rd = request.getRequestDispatcher(forwardPath);

rd.forward(request,response);

}

### 3.5.2 借书功能控制器

@WebServlet(urlPatterns ="/borrowbook.do")

public class BorrowController extends HttpServlet {

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

request.setCharacterEncoding("utf-8");

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

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

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

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

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

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

//HttpSession session = request.getSession();

String forwardPath = "";

//User user = (User) request.getSession().getAttribute("userName");

//Book book = (Book) request.getSession().getAttribute("bookName");

if(borrowbookName!=null){

if(borrowbookID!=null){

BorrowDao borrowDao = new BorrowDao();

boolean flag =borrowDao.borrow(borrowbookID,

borrowbookName,

borrowuserName,

borrowchaName,

borrowtime,

returntime);

if(flag==true){

request.setAttribute("info", "借书成功!");

request.setAttribute("path", "searchResult.jsp");

forwardPath = "searchResult.jsp";

}

}

}

RequestDispatcher rd = request.getRequestDispatcher(forwardPath);

rd.forward(request,response);

}

}

### 3.5.3修改密码控制器

@WebServlet(urlPatterns ="/changepassword.do")

public class ChangePasswordController extends HttpServlet {

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

request.setCharacterEncoding("utf-8");

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

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

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

HttpSession session = request.getSession();

String forwardPath = "";

//HttpSession session = request.getSession();

//User user = (User) request.getSession().getAttribute("userName");

if(oldpassword.equals(newpassword)){

request.setAttribute("info", "新旧密码相同，不能修改");

request.setAttribute("path", "ChangePassword.jsp");

forwardPath = "ChangePassword.jsp";

}

else{

UserDao userDao = new UserDao();

boolean flag =userDao.changepassword(newpassword,userName);

if(flag==true){

request.setAttribute("info", "修改成功!");

request.setAttribute("path", "ChangePassword.jsp");

forwardPath = "ChangePassword.jsp";

}

else{

request.setAttribute("info", "修改失败，请重试!");

request.setAttribute("path", "ChangePassword.jsp");

forwardPath = "ChangePassword.jsp";

}

}

RequestDispatcher rd = request.getRequestDispatcher(forwardPath);

rd.forward(request,response);

}

## 3.6过滤器filter文件

public class PermissionFilter implements Filter {

private String notCheckPath;//不需要过滤额请求地址，从web.xml文件中读取

@Override

public void destroy() {

// TODO Auto-generated method stub

}

@Override

public void doFilter(ServletRequest req, ServletResponse resp,

FilterChain chain) throws IOException, ServletException {

// TODO Auto-generated method stub

HttpServletRequest request = (HttpServletRequest) req;

String path = request.getServletPath();

System.out.println("请求地址url-pattern:"+path);

if(notCheckPath.indexOf(path) == -1){

HttpSession session = request.getSession();

if(session.getAttribute("currentUse") == null){

request.setAttribute("info", "没有权限访问");

request.getRequestDispatcher("/error.jsp").forward(request,

resp);

}

else {//已经登录，直接放行

String value=null;

Boolean flag=false;

Cookie[] cookie=request.getCookies();

for(Cookie a:cookie){

if(a.getName().equals("username")){

flag=true;

value=a.getValue();

break;

}

}

System.out.println(value);

if(flag){

User user=null;

try {

//user=dao.getUser(value);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

System.out.println(user);

session.removeAttribute("currentUser");

String chr=user.getChrName();

session.setAttribute("currentUser", chr);

}

else{

request.setAttribute("info", "没有权限访问");

request.getRequestDispatcher("/error.jsp").forward(request, resp);

}

chain.doFilter(req, resp);

}

}

else{//地址不需要过滤的，直接放行

chain.doFilter(req, resp);

}

}

@Override

public void init(FilterConfig config) throws ServletException {

// 从web.xml配置文件中的filter中读取名为notCheckPath的初始值

notCheckPath = config.getInitParameter("notCheckPath");

}

}

## 3.7 前端界面Html、css、js文件

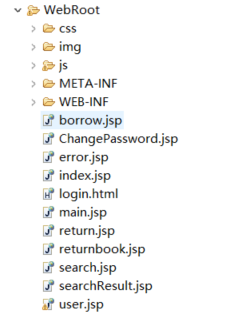


图 9

# 4 系统测试

（1）登录页面，输入学号、密码和验证码正确后进入主页面，支持7天免登录

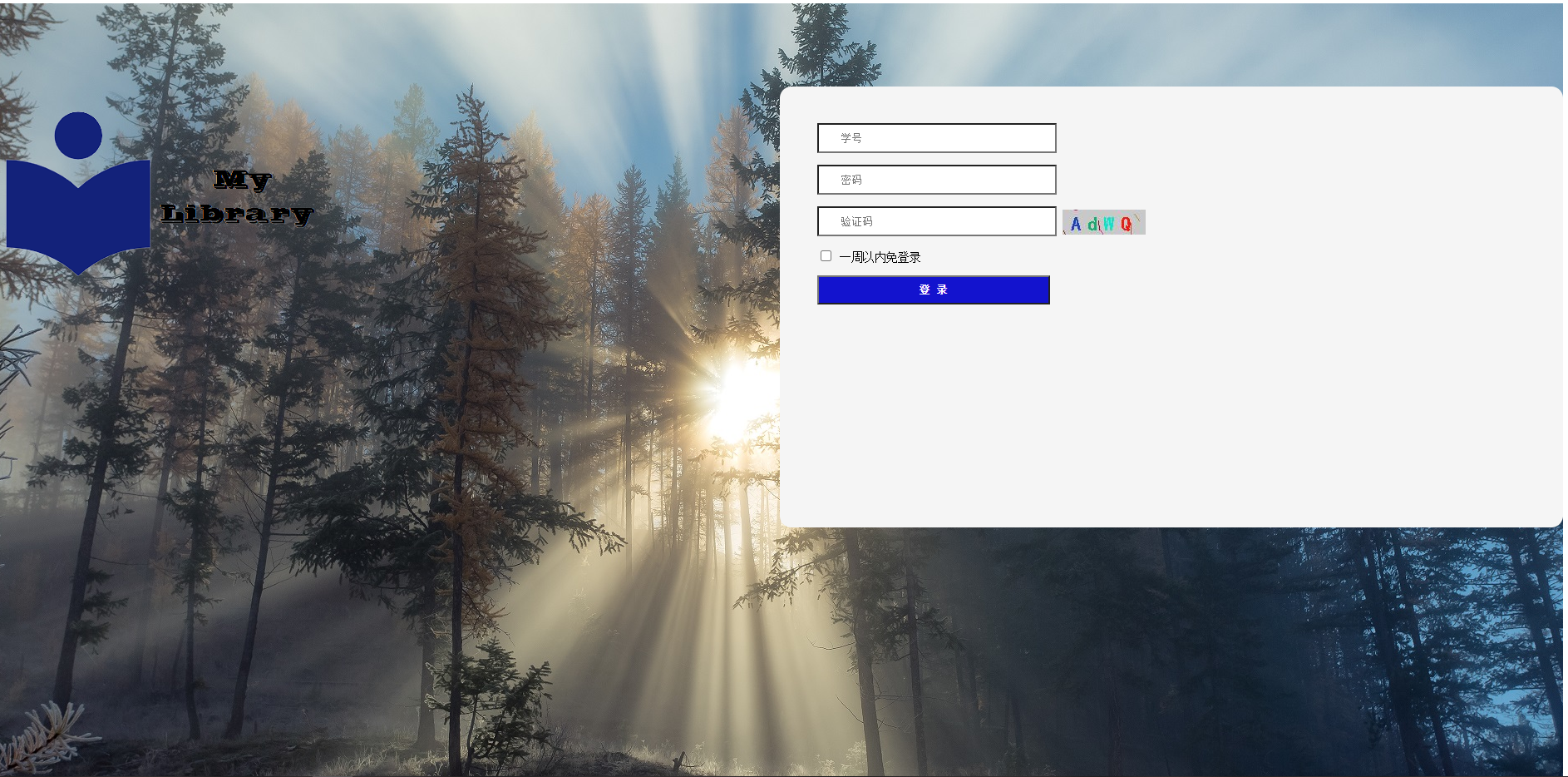


图 10

（2）主页面，上方导航栏，页面主题使用轮播图显示图书馆信息，下方有检索、还书、信息展示栏

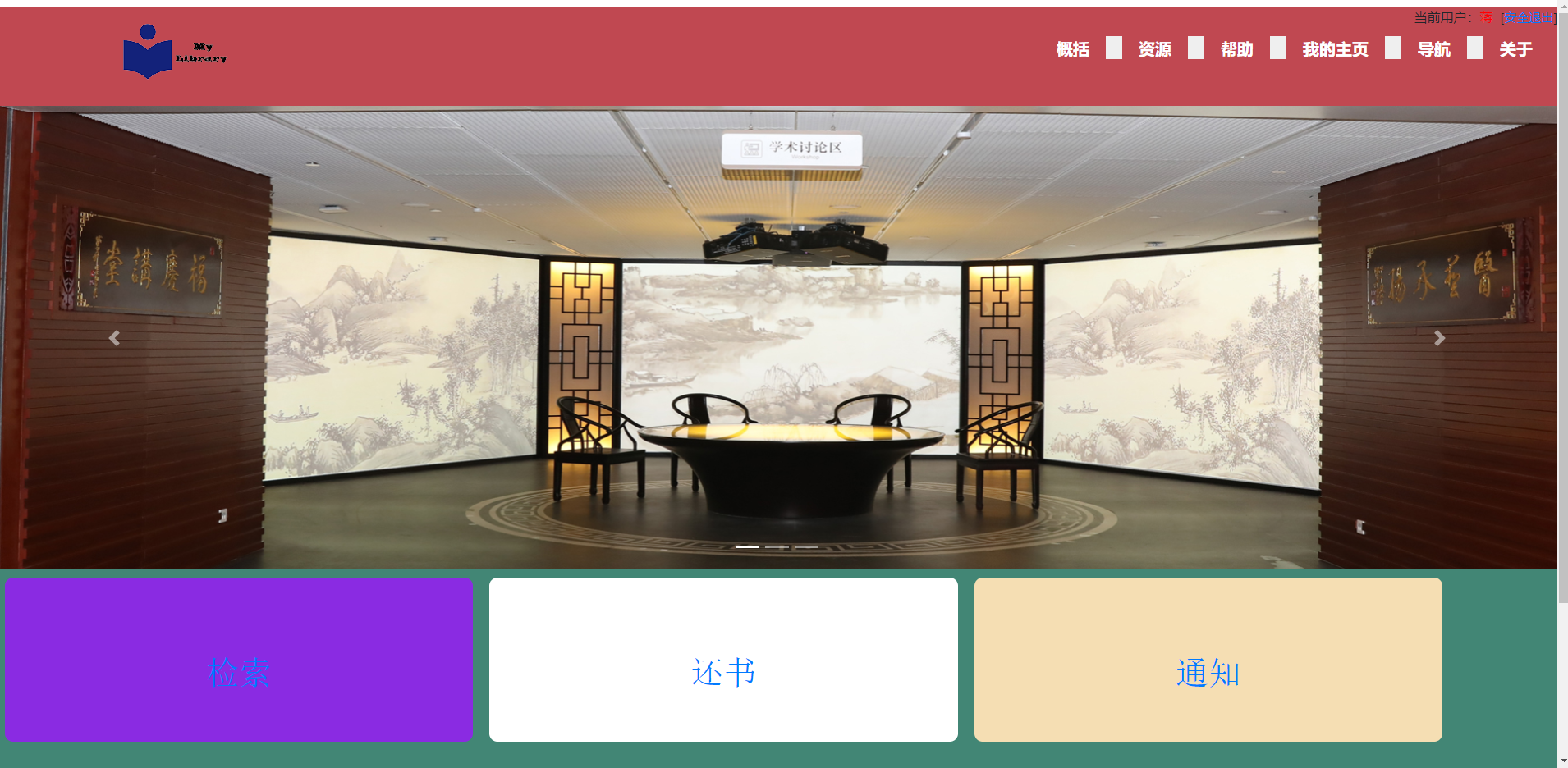


图 11

（3）检索书籍页面，使用书籍ID作为检索关键词

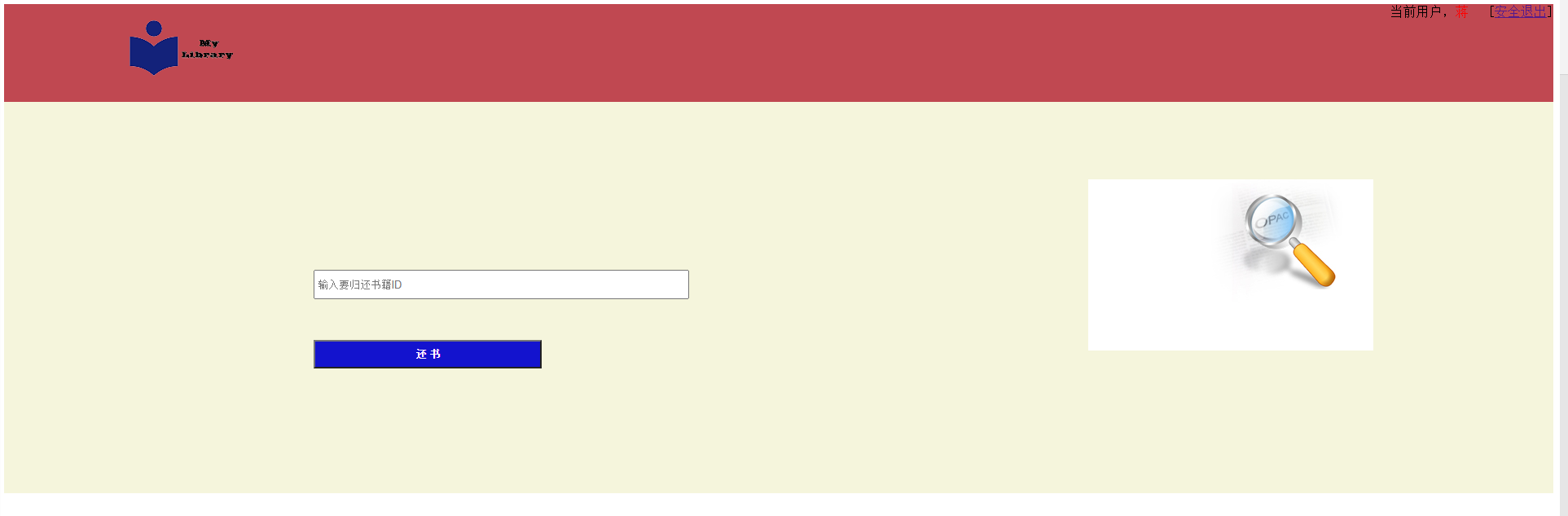


图 12

（4）检索结果显示，table显示，最后“借阅”可直接跳转至借书页面

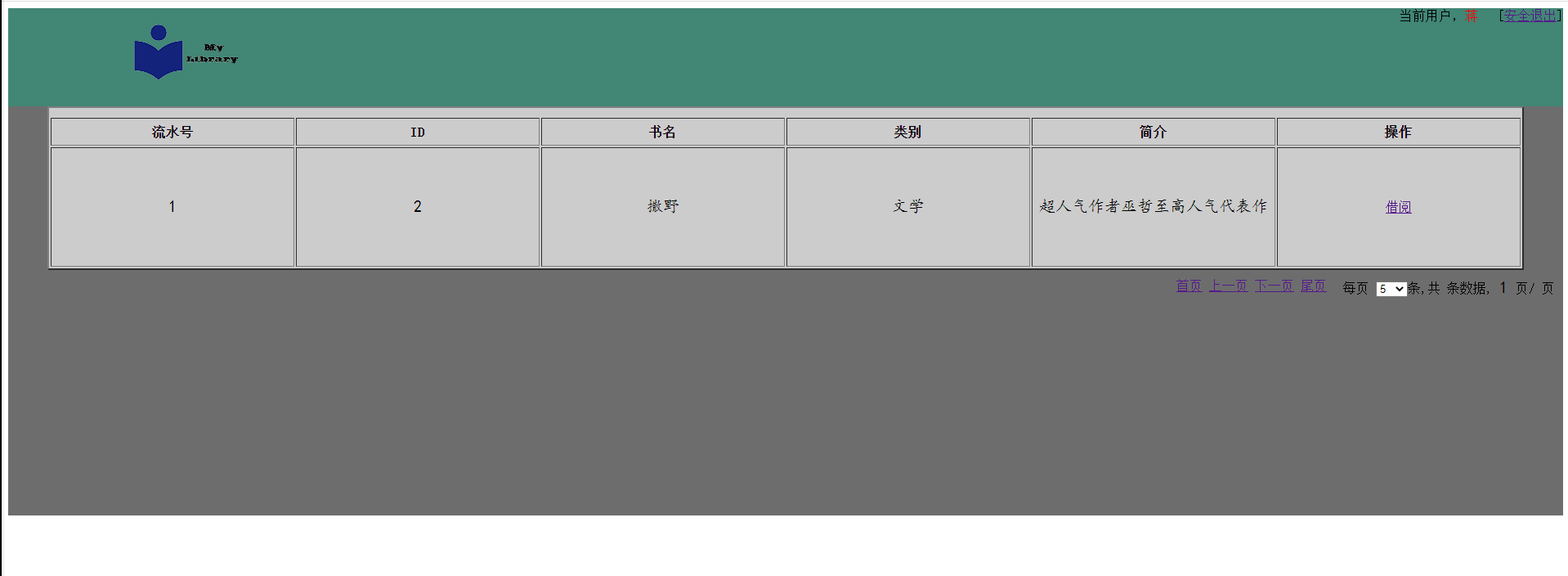


图 13

（5）借书页面，自动填入所查询书籍信息，提交后给出成功提示



图 14

（6）还书页面，输入书籍ID和用户ID即可查询借阅信息，完成还书



图 15

（7）用户页面，页面可以再完善，目前可以实现修改密码和查询历史功能



图 16

（8）修改密码页面，填入密码信息后系统自动检验用户身份，完成密码修改



图 17

# 5 系统总结

做这个图书管理系统时遇到的主要问题是浏览器请求的转发处理，在映射地址正确的情况下，浏览器访问客户端时总会出现请求不存在的问题，通过网上查询资料，我通过多次重启tomcat服务器解决了问题，可能是服务器对于项目部署没有更新的问题。

目前这个图书馆管理系统还有很大的不足，查询功能目前只能使用图书完整的ID查询，不能使用模糊查询，返回结果不能完整显示，系统用户交互页面设计的不够合理，页面和后端处理都还有待完善的部分，整个项目我没有使用框架结构，所以做起来的进度也是比较慢；另外完整的图书管理系统还应该有验证用户身份的功能，设置管理员、普通用户和游客三种不同的身份，由于时间和个人能力的原因，暂时还没有实现，后期会慢慢完善该系统功能。因为我只熟悉掌握了前面的内容，后面的框架知识我还没有掌握，所以我自评80分。

通过学习Web应用开发这门课程，我对于Web有了更加全面系统的认识，对于web前端设计 ，后端处理以及前后端的交互过程有了全面的了解，使用java语言对于上学期的java学习进行了巩固，整体来说，课程具有综合性，我从这门课中学到了很多计算机web方面的知识，最后感谢聂刚老师这学期对于我们学习的指导，感谢在Web学习帮助我的同学！