

目 录

目 录.....	I
第一章 简介.....	2
1.1 项目内容.....	2
1.2 项目目标.....	2
第二章 功能需求分析.....	3
2.1 用户身份验证功能.....	3
2.2 图书管理功能.....	4
2.3 借阅和还书功能.....	4
第三章 系统结构设计.....	5
3.1 系统模块结构图.....	5
3.2 用户身份验证模块.....	5
3.3 图书管理模块.....	7
3.4 借阅和还书模块.....	9
第四章 系统实现.....	14
4.1 系统实现介绍.....	14
4.2 系统实现的不足.....	18
第五章 总结与展望.....	19
5.1 总结.....	19
5.2 展望.....	19

第一章 简介

1.1 项目内容

- 系统简介：** 本项目旨在开发一个图书管理系统。该系统旨在为图书馆或书店提供一种有效的方式来管理图书的存储、借阅和还书等操作。此系统旨在简化图书管理流程，提高效率，并提供用户友好的界面，以改善图书管理体验。
- 项目背景：** 图书馆、书店等场所需要高效地管理大量图书，包括记录图书信息、跟踪借阅记录、提供图书检索等功能。传统的纸质管理方式已经无法满足现代化的需求，因此需要一种自动化的、数字化的解决方案来管理图书。本项目的目标是为这些场所提供一种现代化、便捷的图书管理系统。
- 项目目标：**
 - 提供一个用户友好的图书管理界面，包括搜索、借阅、还书等功能。
 - 允许图书管理员管理图书、用户和借阅记录。
 - 提供图书信息的准确存储和检索。
 - 跟踪用户的借阅历史和图书库存。
 - 提高图书管理的效率和精确度。
- 项目意义：** 图书管理系统的开发有助于实现以下目标：
 - 提高图书管理的效率，减少纸质管理的工作量。
 - 提供更好的用户体验，使用户能够更轻松地查找和借阅图书。
 - 帮助图书管理员更好地管理图书馆资源。
 - 提供数据分析和报告功能，以便更好地了解图书馆的运营状况。
- 项目范围：** 本项目将实现基本的图书管理功能，包括用户登录、图书检索、借阅还书、图书管理等核心功能。

1.2 项目目标

- 用户管理：**
 - 实现用户登录和身份验证，包括普通用户和管理员。
 - 确保用户登录后只能访问其具有权限的功能。
- 图书管理：**
 - 提供管理员用户以及图书馆工作人员管理图书的功能，包括添加、删除、更新图书信息。
 - 确保图书信息的准确存储和检索，包括书名、作者、出版日期、ISBN 等。
 - 跟踪图书库存，确保库存数量的准确性。
- 借阅和还书：**
 - 实现用户借阅和还书图书的功能，包括借阅历史记录。
 - 跟踪用户的借阅历史，包括借阅时间和归还时间。
 - 提供用户借阅状态的可视化，以帮助用户了解借阅情况。
- 前台展示：**

- 提供用户友好的前台图书展示界面，包括图书列表和图书详情页面。
 - 允许用户搜索图书，查看图书的详细信息。
 - 用户可以轻松地查找和浏览图书。
5. 数据交互：
- 使用 A 标签或表单提交来实现前端与后端的数据交互。
 - A 标签用于传输 ID 和其他重要信息，表单用于更新或插入数据。
6. 系统框架：
- 使用 Java Bean、Servlet 和 JSP 开发后端。
 - 使用 JDBC 进行数据库连接。
 - 登录功能在 `com.gin.servlet.UserServlet` 中处理。
7. 安全性：
- 实现验证码功能，以增加系统的安全性。
 - 使用 Session 来存储和验证用户的登录状态，确保只有经过身份验证的用户能够执行敏感操作。
8. 数据分析与报告：
- 提供数据分析和报告功能，以便管理员更好地了解图书馆的运营状况，包括借阅趋势、最受欢迎的图书、用户统计等报告。
9. 易于扩展性：
- 设计系统以支持未来的需求扩展。例如，可以添加更多的图书类别、增加新的功能模块，或者支持不同的用户角色。

通过实现以上目标，图书管理系统将提供一种强大的工具，用于管理图书馆或书店的日常运营，提高效率，改善用户体验，并为决策者提供有价值的数据分析支持。

第二章 功能需求分析

2.1 用户身份验证功能

2.1.1 目标

- 确保只有经过身份验证的用户能够访问系统。
- 区分普通用户和管理员用户，并分配不同的权限。

2.1.2 功能描述

- 用户登录：用户输入用户名和密码进行登录，系统验证用户身份。
- 用户角色划分：系统区分普通用户和管理员用户，分配不同的权限。
- 登录状态维护：系统使用 Session 来维护用户的登录状态，以便后续操作需要验证用户身份。

2.2 图书管理功能

2.2.1 目标

- 提供管理员用户和图书馆工作人员管理图书的功能，包括图书的增删改查。
- 确保图书信息的准确性和完整性。

2.2.2 功能描述

- 添加图书：管理员用户可以添加新的图书信息，包括书名、作者、出版日期、ISBN等。
- 删除图书：管理员用户可以删除不需要的图书信息。
- 更新图书信息：管理员用户可以更新图书的相关信息，确保信息的准确性。
- 图书检索：用户可以根据关键字搜索图书，并查看图书的详细信息。

2.3 借阅和还书功能

2.3.1 目标

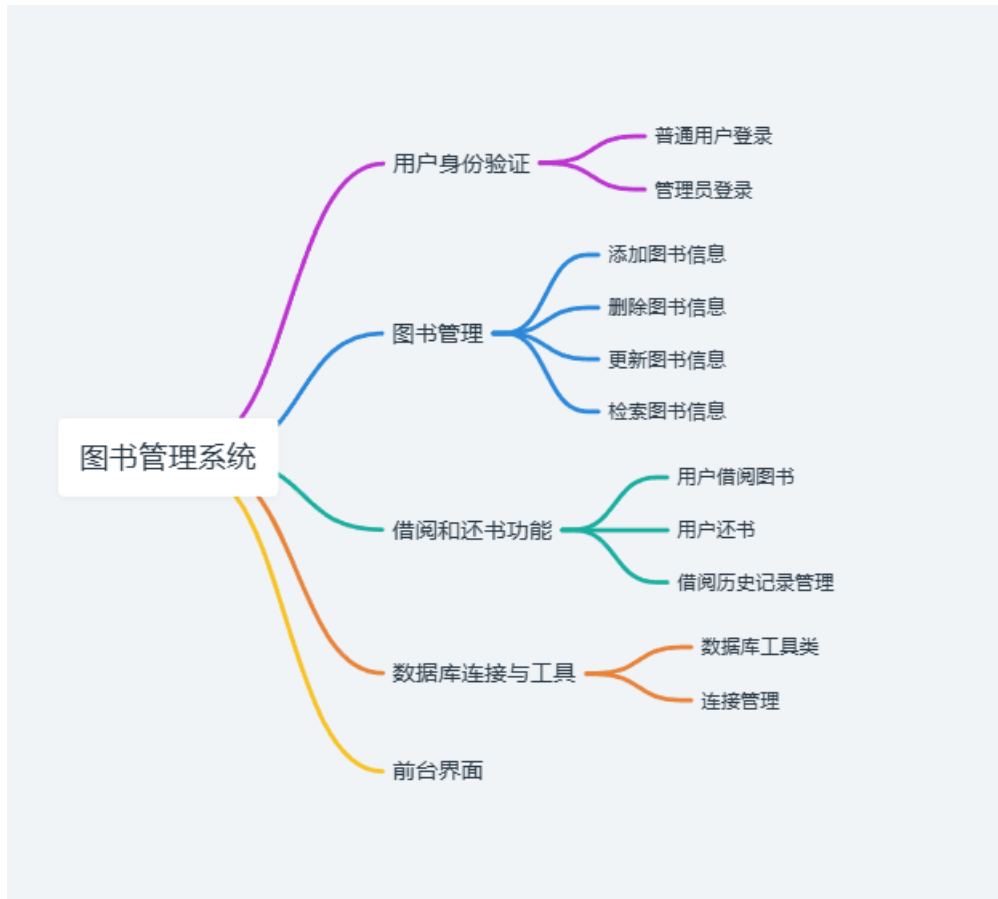
- 提供用户借阅和还书图书的功能，以及跟踪借阅历史记录。
- 提供用户友好的借阅状态显示。

2.3.2 功能描述

- 借阅图书：用户可以借阅可用的图书，系统更新借阅记录和图书库存。
- 还书图书：用户可以归还借阅的图书，系统更新借阅状态和库存数量。
- 借阅历史记录：系统记录用户的借阅历史，包括借阅时间和归还时间。
- 借阅状态显示：用户可以在前台界面上轻松查看自己的借阅状态，以便了解借阅情况。

第三章 系统结构设计

3.1 系统模块结构图



3.2 用户身份验证模块

模块描述： 用户身份验证模块负责处理用户登录和身份验证的功能。这个模块的主要任务是确保只有经过身份验证的用户能够访问系统，并根据用户角色分配不同的权限。

模块结构：

- 登录页面：提供用户输入用户名和密码的界面。
- 用户管理：负责验证用户身份，根据用户角色划分权限。
- Session 管理：用于存储用户登录状态，以便后续操作需要验证用户身份。

交互关系：

- 用户通过登录页面输入用户名和密码。
- 用户管理模块验证用户身份，将用户分为普通用户和管理员用户。
- 根据用户角色，分配不同的权限。

- 登录状态通过 Session 管理，以确保只有经过身份验证的用户可以访问受限资源。

```

1 package com.gin.servlet;
2
3 import com.gin.dao.*;
4 import com.gin.entity.*;
5 import com.gin.service.*;
6 import com.gin.service.impl.*;
7 import com.gin.util.*;
8 import com.jntoo.db.*;
9 import java.io.IOException;
10 import javax.servlet.ServletException;
11 import javax.servlet.annotation.WebServlet;
12 import javax.servlet.http.HttpServlet;
13 import javax.servlet.http.HttpServletRequest;
14 import javax.servlet.http.HttpServletResponse;
15
16 @WebServlet("/user.do")
17 public class UserService extends BaseServlet {
18
19     @Override
20     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
21         doPost(request, response);
22     }
23
24     @Override
25     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
26         String ac = request.getParameter( "% "ac");
27         if (ac == null) ac = "login";
28
29         if (ac.equals("login")) {
30             doLogin(request, response); // 调用登录方法
31         } else if (ac.equals("updatePassword")) {
32             doUpdatePassword(request, response); // 调用修改密码方法
33         } else if (ac.equals("logout")) {
34             session.invalidate(); // 注销登录
35             showSuccess( message: "退出登录成功", url: "/");
36         } else {
37             response.sendError( 500);
38         }
39     }
40
41     /**
42      * 登录系统验证代码
43      * @param request
44      * @param response
45      * @throws ServletException
46      * @throws IOException
47      */

```

```

48     protected void doLogin(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
49         String username = request.getParameter( "% "username"); // 获取用户名
50         String password = request.getParameter( "% "pwd"); // 获取密码
51         String cx = request.getParameter( "% "cx"); // 获取当前登录类型
52         boolean isAdmin = request.getParameter( "% "admin") != null; // 是否为后台登录
53
54         /**
55          * 验证码
56          */
57         String pagerandom = request.getParameter( "% "pagerandom") == null ? "" : request.getParameter( "% "pagerandom");
58
59         if (username == null || "".equals(username)) {
60             showError("账号不允许为空");
61             return;
62         }
63         if (password == null || "".equals(password)) {
64             showError("密码不允许为空");
65             return;
66         }
67
68         /**
69          * 获取保存在session 中的验证码
70          */
71         String random = (String) request.getSession().getAttribute( "% "random");
72
73         if (!pagerandom.equals(random) && request.getParameter( "% "a") != null) {
74             request.setAttribute( "% "error", "% "验证码错误");
75             showError("验证码错误");
76             return;
77         }
78         if (cx.equals("用户")) {
79             YonghuService service = new YonghuServiceImpl();
80             Yonghu yonghu = service.login(username, password);
81             if (yonghu == null) {
82                 showError("用户名或密码错误");
83                 return;
84             }
85             // 设置当前登录会话
86             session.setAttribute( "% "id", yonghu.getId());
87             session.setAttribute( "% "login", cx);
88             session.setAttribute( "% "cx", yonghu.getCx());
89             session.setAttribute( "% "username", yonghu.getUsername());
90             session.setAttribute( "% "pwd", yonghu.getPwd());
91             session.setAttribute( "% "xingming", yonghu.getXingming());
92             session.setAttribute( "% "xingbie", yonghu.getXingbie());
93             session.setAttribute( "% "shouji", yonghu.getShouji());
94             session.setAttribute( "% "username", username);
95         }

```

```
String referer = request.getParameter("referer"); // 指定跳转位置
if (referer == null) {
    if (isAdmin) {
        referer = "/main.jsp";
    } else {
        referer = "/.";
    }
}
showSuccess("登录成功", referer);
}

/**
 * 修改密码
 * @param request
 * @param response
 * @throws ServletException
 * @throws IOException
 */
protected void doUpdatePassword(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    String username = request.getSession().getAttribute("username").toString();
    String cx = request.getSession().getAttribute("login").toString();
    String oldPassword = Request.get("oldPassword");
    String newPwd = Request.get("newPwd");
    String newPwd2 = Request.get("newPwd2");

    if (oldPassword.equals("")) {
        showError("请输入原密码");
        return;
    }

    if (newPwd.equals("")) {
        showError("请输入新密码");
        return;
    }

    if (!newPwd.equals(newPwd2)) {
        showError("两次密码不一致");
        return;
    }
}
```

```
if (cx.equals("用户")) {
    YonghuService service = new YonghuServiceImpl();
    Yonghu yonghu = service.login(username, oldPassword);
    if (yonghu == null) {
        showError("原密码不正确");
        return;
    }
    service.editPassword(yonghu.getId(), newPwd);
}
showSuccess("修改密码成功");
}
}
```

3.3 图书管理模块

模块描述： 图书管理模块负责管理系统中的图书信息，包括图书的添加、删除、更新和检索。管理员用户和图书馆工作人员可以使用这个模块来管理图书库存。

模块结构：

- 添加图书：用于管理员添加新的图书信息。
- 删除图书：允许管理员删除不需要的图书信息。
- 更新图书信息：提供管理员更新图书信息的功能。
- 图书检索：用户可以根据关键字搜索图书，并查看图书的详细信息。

交互关系：

- 管理员用户通过相应的页面执行添加、删除、更新和检索图书的操作。
- 操作将涉及数据库的读写，以确保图书信息的准确性和完整性。

```

1 package com.gin.servlet;
2
3 import com.gin.dao.*;
4 import com.gin.entity.*;
5 import com.gin.service.*;
6 import com.gin.service.impl.*;
7 import com.gin.util.*;
8 import com.jntoo.db.*;
9 import java.io.IOException;
10 import java.util.*;
11 import javax.servlet.ServletException;
12 import javax.servlet.annotation.WebServlet;
13 import javax.servlet.http.HttpServletRequest;
14 import javax.servlet.http.HttpServletResponse;
15
16 /**
17  * 书籍 模块控制器
18  */
19 @WebServlet(value = { "/shuji.do" })
20 public class ShujiServlet extends BaseServlet {
21
22     @Override
23     protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
24         doPost(req, resp);
25     }
26
27     @Override
28     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
29         String ac = request.getParameter( S: "ac" );
30         if (ac == null) {
31             ac = "list";
32         }
33         ShujiService service = new ShujiServiceImpl();
34
35         // 查询数据
36         if (ac.indexOf("list") != -1) {
37             String orderby = Request.get( name: "orderby", def: "id" ); // 获取浏览器地址中的orderby 参数 默认按 发布时间 排序方式
38             String sort = Request.get( name: "sort", def: "DESC" ); // 获取浏览器地址中的sort参数 默认按 倒序排序
39             int pagesize = Request.getInt( name: "pagesize", def: 12 ); // 获取浏览器地址中的pagesize参数 默认按 12行每页显示
40             int page = Math.max(1, Request.getInt( name: "page", def: 1 )); // 获取当前页 默认 第一页
41
42             String where = "1=1"; // 防止sql 语句中where and 错误
43             // 以下是判断搜索框中是否有输入内容，判断是否满足是否有填写相关条件，符合则写入sql搜索语句
44             if (!Request.get("mingcheng").equals("")) {
45                 where += " AND mingcheng LIKE '%" + Request.get("mingcheng") + "%' ";
46             }
47             if (!Request.get("fenlei").equals("")) {
48                 where += " AND fenlei = '" + Request.get("fenlei") + "' ";
49             }
50
51             // 获取行数
52             long count = service.count(where);
53             // 创建分页信息
54             new Collect(count, pagesize, page);
55             // 分页并查询数据
56             List list = service.selectPages(pagesize, page, where, orderby: orderby + " " + sort);
57
58             request.setAttribute( S: "lists", list);
59
60             request.setAttribute( S: "orderby", orderby);
61             request.setAttribute( S: "sort", sort);
62             request.setAttribute( S: "pagesize", pagesize);
63
64             view( url: "/shuji/" + ac + ".jsp");
65             return;
66         } else if (ac.equals("add")) { // 书籍添加页面视图
67             if (ac.equals("add") && !checkLogin()) {
68                 showError("尚未登录");
69                 return;
70             }
71
72             view( url: "/shuji/" + ac + ".jsp");
73             return;
74         } else if (ac.equals("insert")) {
75             // 插入
76             Shuji post = new Shuji(); // 创建实体类
77             // 设置前台提交上来的数据到实体类中
78             post.setMingcheng(Request.get("mingcheng"));
79
80             post.setFenlei(Request.getInt( name: "fenlei"));
81
82             post.setZuozhe(Request.get("zuozhe"));
83
84             post.setChubanshe(Request.get("chubanshe"));
85
86             post.setIsbn(Request.get("isbn"));
87
88             post.setShuliang(Request.getInt( name: "shuliang"));
89
90             post.setTupian(Request.get("tupian"));
91
92             post.setJianjie(DownloadRemoteImage.run(Request.get("jianjie")));

```

```

50
51 // 获取行数
52 long count = service.count(where);
53 // 创建分页信息
54 new Collect(count, pagesize, page);
55 // 分页并查询数据
56 List list = service.selectPages(pagesize, page, where, orderby: orderby + " " + sort);
57
58 request.setAttribute( S: "lists", list);
59
60 request.setAttribute( S: "orderby", orderby);
61 request.setAttribute( S: "sort", sort);
62 request.setAttribute( S: "pagesize", pagesize);
63
64 view( url: "/shuji/" + ac + ".jsp");
65 return;
66 } else if (ac.equals("add")) { // 书籍添加页面视图
67     if (ac.equals("add") && !checkLogin()) {
68         showError("尚未登录");
69         return;
70     }
71
72     view( url: "/shuji/" + ac + ".jsp");
73     return;
74 } else if (ac.equals("insert")) {
75     // 插入
76     Shuji post = new Shuji(); // 创建实体类
77     // 设置前台提交上来的数据到实体类中
78     post.setMingcheng(Request.get("mingcheng"));
79
80     post.setFenlei(Request.getInt( name: "fenlei"));
81
82     post.setZuozhe(Request.get("zuozhe"));
83
84     post.setChubanshe(Request.get("chubanshe"));
85
86     post.setIsbn(Request.get("isbn"));
87
88     post.setShuliang(Request.getInt( name: "shuliang"));
89
90     post.setTupian(Request.get("tupian"));
91
92     post.setJianjie(DownloadRemoteImage.run(Request.get("jianjie")));

```



```

93     service.insert(post); // 插入数据
94     int charuid = post.getId().intValue();
95
96     showSuccess( message: "保存成功", Request.getHeader( S: "referer") : Request.getHeader("referer"));
97 } else if (ac.equals("updt")) {
98     String id = Request.get("id");
99
100     Shuji mm = service.findWhere("id=" + id);
101     assign( name: "mm", mm);
102     view( url: "/shuji/updt.jsp");
103 } else if (ac.equals("update")) {
104     // 创建实体类
105     String charuid = request.getParameter( S: "id");
106     Shuji post = service.findWhere("id=" + charuid);
107     if (request.getParameter( S: "mingcheng") != null) post.setMingcheng(Request.get("mingcheng"));
108
109     if (request.getParameter( S: "fenlei") != null) post.setFenlei(Request.getInt( name: "fenlei"));
110
111     if (request.getParameter( S: "zuozhe") != null) post.setZuozhe(Request.get("zuozhe"));
112
113     if (request.getParameter( S: "chubanshe") != null) post.setChubanshe(Request.get("chubanshe"));
114
115     if (request.getParameter( S: "isbn") != null) post.setIsbn(Request.get("isbn"));
116
117     if (request.getParameter( S: "shuliang") != null) post.setShuliang(Request.getInt( name: "shuliang"));
118
119     if (request.getParameter( S: "tupian") != null) post.setTupian(Request.get("tupian"));
120
121     if (request.getParameter( S: "jianjie") != null) post.setJianjie(DownloadRemoteImage.run(Request.get("jianjie")));
122
123     service.update(post);
124
125     showSuccess( message: "保存成功", Request.getHeader("referer"));
126     // 更新
127 } else if (ac.equals("detail")) {
128     int id = Request.getInt( name: "id");
129     Shuji map = service.find(id); // 根据前台url 参数中的id获取行数据
130
131     request.setAttribute( S: "map", map); // 把数据写到前台
132     view( url: "/shuji/" + ac + ".jsp");
133 } else if (ac.equals("delete")) {
134     if (!checklogin()) {
135         showError("尚未登录");
136         return;
137     }
138
139     int id = Request.getInt( name: "id");
140
141     service.delete(id);
142     showSuccess( message: "删除成功", request.getHeader( S: "referer"));
143 } else {
144     response.sendError( 404);
145 }
146 }
147 }
148

```

3.4 借阅和还书模块

模块描述： 借阅和还书模块负责处理用户借阅和还书图书的功能，以及记录借阅历史。用户可以在这个模块中查看自己的借阅状态和历史记录。

模块结构：

- 借阅图书：用户可以借阅可用的图书，系统更新借阅记录和图书库存。
- 还书图书：用户可以归还借阅的图书，系统更新借阅状态和库存数量。
- 借阅历史记录：系统记录用户的借阅历史，包括借阅时间和归还时间。
- 借阅状态显示：用户可以在前台界面上轻松查看自己的借阅状态，以便了解借阅情况。

交互关系：

- 用户通过前台界面执行借阅和还书操作。
- 借阅和还书操作将涉及数据库的读写，以更新借阅记录和库存数量。
- 借阅历史记录将被记录，以供用户查阅。

```

package com.gin.servlet;

import com.gin.dao.*;
import com.gin.entity.*;
import com.gin.service.*;
import com.gin.service.impl.*;
import com.gin.util.*;
import com.jntoo.db.*;
import java.io.IOException;
import java.util.*;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * 借阅 模块控制器
 */
@WebServlet(value = { "/jibuyue.do" })
public class JibuyueServlet extends BaseServlet {

    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
        doPost(req, resp);
    }

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        String ac = request.getParameter( "ac" );
        if (ac == null) {
            ac = "list";
        }
        JibuyueService service = new JibuyueServiceImpl();

        // 查询数据
        if (ac.indexOf("list") != -1) {
            String orderby = Request.get( name: "orderby", def: "id" ); // 获取浏览器地址中的orderby 参数 默认按 发布时间 排序方式
            String sort = Request.get( name: "sort", def: "desc" ); // 获取浏览器地址中的sort参数 默认按 倒序排序
            int pagesize = Request.getInt( name: "pagesize", def: 12 ); // 获取浏览器地址中的sort参数 默认按 12行每页显示
            int page = Math.max(1, Request.getInt( name: "page", def: 1 )); // 获取当前页 默认 第一页

            String where = "1=1"; // 防止sql 语句中where 和 错误

            // 判断url 参数shujiid是否大于0
            if (Request.getInt( name: "shujiid" ) > 0) {
                // 大于0 则写入条件
                where += " AND shujiid=" + Request.getInt( name: "shujiid" ) + " ";
            }

            // 以下是判断搜索框中是否有输入内容，判断是否前台是否有填写相关条件，符合则写入sql搜索语句
            if (!Request.get("mingcheng").equals("")) {
                where += " AND mingcheng LIKE '%" + Request.get("mingcheng") + "%' ";
            }
            if (!Request.get("fenlei").equals("")) {
                where += " AND fenlei =" + Request.get("fenlei") + " ";
            }
            if (ac.equals("list_yonghu")) {
                where += " AND yonghu=" + session.getAttribute( "username" ) + " ";
            }

            // 获取行数
            long count = service.count(where);
            // 创建分页信息
            new Collect(count, pagesize, page);
            // 分页并查询数据
            List list = service.selectPages(pagesize, page, where, orderby: orderby + " " + sort);

            request.setAttribute( "lists", list );

            request.setAttribute( "orderby", orderby );
            request.setAttribute( "sort", sort );
            request.setAttribute( "pagesize", pagesize );

            view( url: "/jibuyue/" + ac + ".jsp" );
            return;
        } else if (ac.equals("add")) { // 借阅添加页面视图
            if (ac.equals("add") && !checkLogin()) {
                showError("尚未登录");
                return;
            }

            int id = Request.getInt( name: "id" ); // 根据id 获取书籍模块中的数据
            Shuji readMap = new ShujiServiceImpl().find(id);
            // 将数据行写入给前台jsp页面
            request.setAttribute( "readMap", readMap );
            view( url: "/jibuyue/" + ac + ".jsp" );
            return;
        } else if (ac.equals("insert")) {
            // 插入
            Jibuyue post = new Jibuyue(); // 创建实体类
            // 设置前台提交上来的数据到实体类中
            post.setShujiid(Request.getInt( name: "shujiid" ));
            post.setMingcheng(Request.get("mingcheng"));
            post.setFenlei(Request.getInt( name: "fenlei" ));
        }
    }
}

```

```

49 // 以下是判断搜索框中是否有输入内容，判断是否前台是否有填写相关条件，符合则写入sql搜索语句
50 if (!Request.get("mingcheng").equals("")) {
51     where += " AND mingcheng LIKE '%" + Request.get("mingcheng") + "%' ";
52 }
53 if (!Request.get("fenlei").equals("")) {
54     where += " AND fenlei =" + Request.get("fenlei") + " ";
55 }
56 if (ac.equals("list_yonghu")) {
57     where += " AND yonghu=" + session.getAttribute( "username" ) + " ";
58 }
59
60 // 获取行数
61 long count = service.count(where);
62 // 创建分页信息
63 new Collect(count, pagesize, page);
64 // 分页并查询数据
65 List list = service.selectPages(pagesize, page, where, orderby: orderby + " " + sort);
66
67 request.setAttribute( "lists", list );
68
69 request.setAttribute( "orderby", orderby );
70 request.setAttribute( "sort", sort );
71 request.setAttribute( "pagesize", pagesize );
72
73 view( url: "/jibuyue/" + ac + ".jsp" );
74 return;
75 } else if (ac.equals("add")) { // 借阅添加页面视图
76     if (ac.equals("add") && !checkLogin()) {
77         showError("尚未登录");
78         return;
79     }
80
81     int id = Request.getInt( name: "id" ); // 根据id 获取书籍模块中的数据
82     Shuji readMap = new ShujiServiceImpl().find(id);
83     // 将数据行写入给前台jsp页面
84     request.setAttribute( "readMap", readMap );
85     view( url: "/jibuyue/" + ac + ".jsp" );
86     return;
87 } else if (ac.equals("insert")) {
88     // 插入
89     Jibuyue post = new Jibuyue(); // 创建实体类
90     // 设置前台提交上来的数据到实体类中
91     post.setShujiid(Request.getInt( name: "shujiid" ));
92     post.setMingcheng(Request.get("mingcheng"));
93     post.setFenlei(Request.getInt( name: "fenlei" ));
94 }
95
96

```

```

96
97     post.setZuozhe(Request.get("zuozhe"));
98
99     post.setYonghu(Request.get("yonghu"));
100
101     post.setKaishi(Request.get("kaishi"));
102
103     post.setJieshu(Request.get("jieshu"));
104
105     post.setShizhang(Request.getInt( name: "shizhang"));
106
107     post.setZhuangtai(Request.get("zhuangtai"));
108
109     post.setShujiid(Request.getInt( name: "shujiid"));
110
111     service.insert(post); // 插入数据
112     int charuid = post.getId().intValue();
113     DB.execute( sql: "UPDATE jiejue SET shizhang=(TIMESTAMPDIFF(DAY , kaishi,jieshu)) WHERE id='"+ charuid + "'");
114
115     DB.execute( sql: "UPDATE shuji SET shuliang=shuliang-1 WHERE id='"+ request.getParameter( $: "shujiid") + "'");
116
117     showSuccess( message: "保存成功", Request.get("referer").equals("") ? request.getHeader( $: "referer") : Request.get("referer"));
118 } else if (ac.equals("updt")) {
119     String id = Request.get("id");
120
121     Jiejue mmm = service.findWhere("id=" + id);
122     // 根据mmm.Shujiid 获取书籍模块中的数据
123     Shuji readMap = new ShujiServiceImpl().find(mmm.getShujiid());
124     // 将数据写入给前台jsp页面
125     request.setAttribute( $: "readMap", readMap);
126     assign( name: "mmm", mmm);
127     view( url: "/jiejue/updt.jsp");
128 } else if (ac.equals("update")) {
129     // 创建实体类
130     String charuid = request.getParameter( $: "id");
131     Jiejue post = service.findWhere("id=" + charuid);
132     if (request.getParameter( $: "shujiid") != null) post.setShujiid(Request.getInt( name: "shujiid"));
133
134     if (request.getParameter( $: "mingcheng") != null) post.setMingcheng(Request.get("mingcheng"));
135
136     if (request.getParameter( $: "fenlei") != null) post.setFenlei(Request.getInt( name: "fenlei"));
137
138     if (request.getParameter( $: "zuozhe") != null) post.setZuozhe(Request.get("zuozhe"));
139
140     if (request.getParameter( $: "yonghu") != null) post.setYonghu(Request.get("yonghu"));
141
142     JiejueServlet.doPost()

```

```

140     if (request.getParameter( $: "yonghu") != null) post.setYonghu(Request.get("yonghu"));
141
142     if (request.getParameter( $: "kaishi") != null) post.setKaishi(Request.get("kaishi"));
143
144     if (request.getParameter( $: "jieshu") != null) post.setJieshu(Request.get("jieshu"));
145
146     if (request.getParameter( $: "shizhang") != null) post.setShizhang(Request.getInt( name: "shizhang"));
147
148     if (request.getParameter( $: "zhuangtai") != null) post.setZhuangtai(Request.get("zhuangtai"));
149
150     service.update(post);
151
152     DB.execute( sql: "UPDATE jiejue SET shizhang=(TIMESTAMPDIFF(DAY , kaishi,jieshu)) WHERE id='"+ request.getParameter( $: "id") + "'");
153
154     showSuccess( message: "保存成功", Request.get("referer"));
155     // 更新
156 } else if (ac.equals("delete")) {
157     if (!checkLogin()) {
158         showError("尚未登录");
159         return;
160     }
161     int id = Request.getInt( name: "id");
162
163     service.delete(id);
164     showSuccess( message: "删除成功", request.getHeader( $: "referer"));
165 } else {
166     response.sendError( $: 404);
167 }
168
169 }
170

```

归还:

```

1 package com.gin.servlet;
2
3 import com.gin.dao.*;
4 import com.gin.entity.*;
5 import com.gin.service.*;
6 import com.gin.service.impl.*;
7 import com.gin.util.*;
8 import com.jntoo.db.*;
9 import java.io.IOException;
10 import java.util.*;
11 import javax.servlet.ServletException;
12 import javax.servlet.annotation.WebServlet;
13 import javax.servlet.http.HttpServletRequest;
14 import javax.servlet.http.HttpServletResponse;
15
16 /**
17  * 归还 模块控制器
18  */
19 @WebServlet(value = { "/guihuan.do" })
20 public class GuihuanServlet extends BaseServlet {
21
22     @Override
23     protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
24         doPost(req, resp);
25     }
26
27     @Override
28     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
29         String ac = request.getParameter( "S:"ac");
30         if (ac == null) {
31             ac = "list";
32         }
33         GuihuanService service = new GuihuanServiceImpl();
34
35         // 查询数据
36         if (ac.indexOf("list") != -1) {
37             String orderby = Request.get( name: "orderby", def: "id"); // 获取浏览器地址中的orderby 参数 默认按 发布时间 排序方式
38             String sort = Request.get( name: "sort", def: "DESC"); // 获取浏览器地址中的sort参数 默认按 倒序排序
39             int pagesize = Request.getInt( name: "pagesize", def: 12); // 获取浏览器地址中的sort参数 默认按 12行每页显示
40             int page = Math.max(1, Request.getInt( name: "page", def: 1)); // 获取当前页 默认 第一页
41
42             String where = "1=1"; // 防止sql 语句中where and 错误
43
44             // 判断URL 参数jieyueid是否大于0
45             if (Request.getInt( name: "jieyueid") > 0) {
46                 // 大于0 则写入条件
47                 where += " AND jieyueid=" + Request.getInt( name: "jieyueid") + " ";
48             }
49
50             // 以下是判断搜索框中是否有输入内容，判断是否前台是否有填写相关条件，符合则写入sql搜索语句
51             if (!Request.get("mingcheng").equals("")) {
52                 where += " AND mingcheng LIKE '%" + Request.get("mingcheng") + "%' ";
53             }
54             if (!Request.get("fenlei").equals("")) {
55                 where += " AND fenlei ='" + Request.get("fenlei") + "' ";
56             }
57             if (ac.equals("list_yonghu")) {
58                 where += " AND yonghu=" + session.getAttribute( "S:"username") + " ";
59             }
60
61             // 获取行数
62             long count = service.count(where);
63             // 创建分页信息
64             new Collect(count, pagesize, page);
65             // 分页并查询数据
66             List list = service.selectPages(pagesize, page, where, "orderby: orderby + " " + sort);
67
68             request.setAttribute( "S:"lists", list);
69
70             request.setAttribute( "S:"orderby", orderby);
71             request.setAttribute( "S:"sort", sort);
72             request.setAttribute( "S:"pagesize", pagesize);
73
74             view( url: "/guihuan/" + ac + ".jsp");
75             return;
76         } else if (ac.equals("add")) { // 归还添加页面视图
77             if (ac.equals("add") && !checkLogin()) {
78                 showError("尚未登录");
79                 return;
80             }
81
82             int id = Request.getInt( name: "id"); // 根据id 获取借阅模块中的数据
83             Jieyue readMap = new JieyueServiceImpl().find(id);
84             // 将数据行写入给前台jsp页面
85             request.setAttribute( "S:"readMap", readMap);
86             view( url: "/guihuan/" + ac + ".jsp");
87             return;
88         } else if (ac.equals("insert")) {
89             // 插入
90             Guihuan post = new Guihuan(); // 创建实体类
91             // 设置前台提交上来的数据到实体类中
92             post.setJieyueid(Request.getInt( name: "jieyueid"));
93             post.setShujiid(Request.getInt( name: "shujiid"));
94             post.setMingcheng(Request.get("mingcheng"));
95

```

```

95     post.setMingcheng(Request.get("mingcheng"));
96
97     post.setFenlei(Request.getInt( name: "fenlei"));
98
99     post.setZuozhe(Request.get("zuozhe"));
100
101     post.setYonghu(Request.get("yonghu"));
102
103     post.setAddtime(Info.getDateStr());
104
105     post.setJieyueid(Request.getInt( name: "jieyueid"));
106
107     service.insert(post); // 插入数据
108     int charuid = post.getId().intValue();
109     DB.execute( sql: "UPDATE shuji SET shuliang=shuliang+1 WHERE id='" + request.getParameter( $: "shujiid") + "'");
110
111     DB.execute( sql: "UPDATE jieyue SET zhuangtai='已归还' WHERE id='" + request.getParameter( $: "jieyueid") + "'");
112
113     showSuccess( message: "保存成功", Request.get("referer").equals("") ? request.getHeader( $: "referer") : Request.get("referer"));
114 } else if (ac.equals("updt")) {
115     String id = Request.get("id");
116
117     Guihuan mmm = service.findWhere("id=" + id);
118     // 根据mmm.jieyueid 获取借阅模块中的数据
119     Jieyue readMap = new JieyueServiceImpl().find(mmm.getJieyueid());
120     // 将数据行写入给前台jsp页面
121     request.setAttribute( $: "readMap", readMap);
122     assign( name: "mmm", mmm);
123     view( url: "/guihuan/updt.jsp");
124 } else if (ac.equals("update")) {
125     // 创建实体类
126     String charuid = request.getParameter( $: "id");
127     Guihuan post = service.findWhere("id=" + charuid);
128     if (request.getParameter( $: "jieyueid") != null) post.setJieyueid(Request.getInt( name: "jieyueid"));
129
130     if (request.getParameter( $: "shujiid") != null) post.setShujiid(Request.getInt( name: "shujiid"));
131
132     if (request.getParameter( $: "mingcheng") != null) post.setMingcheng(Request.get("mingcheng"));
133
134     if (request.getParameter( $: "fenlei") != null) post.setFenlei(Request.getInt( name: "fenlei"));
135
136     if (request.getParameter( $: "zuozhe") != null) post.setZuozhe(Request.get("zuozhe"));
137
138     if (request.getParameter( $: "yonghu") != null) post.setYonghu(Request.get("yonghu"));
139
140     if (request.getParameter( $: "addtime") != null) post.setAddtime(Request.get("addtime"));
141
142     service.update(post);
143
144     showSuccess( message: "保存成功", Request.get("referer"));
145     // 更新
146 } else if (ac.equals("delete")) {
147     if (!checkLogin()) {
148         showError("尚未登录");
149         return;
150     }
151     int id = Request.getInt( name: "id");
152
153     service.delete(id);
154     showSuccess( message: "删除成功", request.getHeader( $: "referer"));
155 } else {
156     response.sendError( $: 404);
157 }
158 }
159 }

```

第四章 系统实现

4.1 系统实现介绍

本章将介绍图书管理系统的实际实现情况，包括关键功能、技术选型、技术难点的解决以及实现效果的简要说明。以下是系统实现的主要要点：

关键功能实现：

- 用户身份验证：实现了用户登录和身份验证功能，通过 Session 管理登录状态，确保只有经过验证的用户可以访问系统。
- 图书管理：管理员用户可以执行添加、删除、更新和检索图书信息的操作，确保图书信息的准确性和完整性。
- 借阅和还书：用户可以借阅和还书图书，系统会记录借阅历史，用户可以查看借阅状态。
- 前台展示：前台界面提供用户友好的图书列表、图书详情和用户借阅状态的显示。

技术选型：

- 后端开发采用 Java 语言，使用 Servlet 和 JSP 进行 Web 开发。
- 数据库连接使用 JDBC，与 MySQL 数据库进行交互。
- 前端界面使用 Bootstrap CSS 框架，JavaScript 和 jQuery 进行开发，以实现响应式设计。
- 用户身份验证和登录状态管理使用 Session 技术。
- 图书信息的存储和检索使用数据库。

技术难点与解决：

- 身份验证和权限管理：实现了基于角色的权限管理，通过 Session 管理用户登录状态，以确保只有具备权限的用户可以执行特定操作。
- 数据库操作：实现了对数据库的数据增删改查操作，确保图书信息的准确存储和检索。
- 图书借阅历史记录：设计了合适的数据结构和数据库表结构来记录用户的借阅历史，以使用户查看。

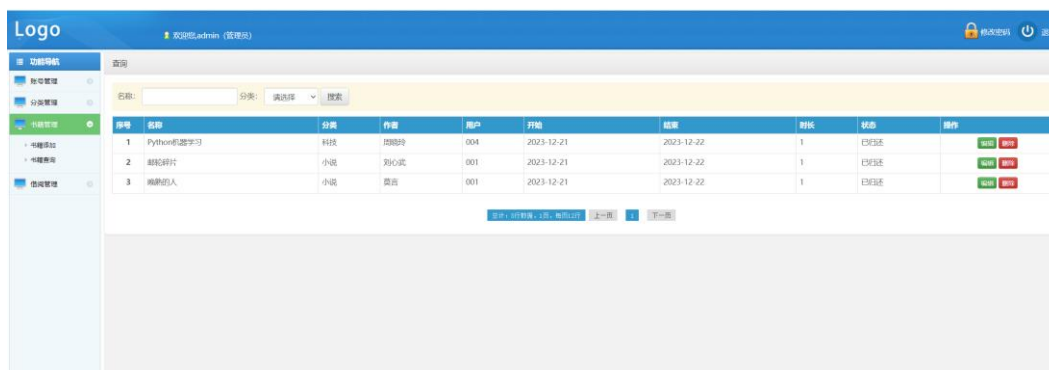
实现效果：

登录界面：



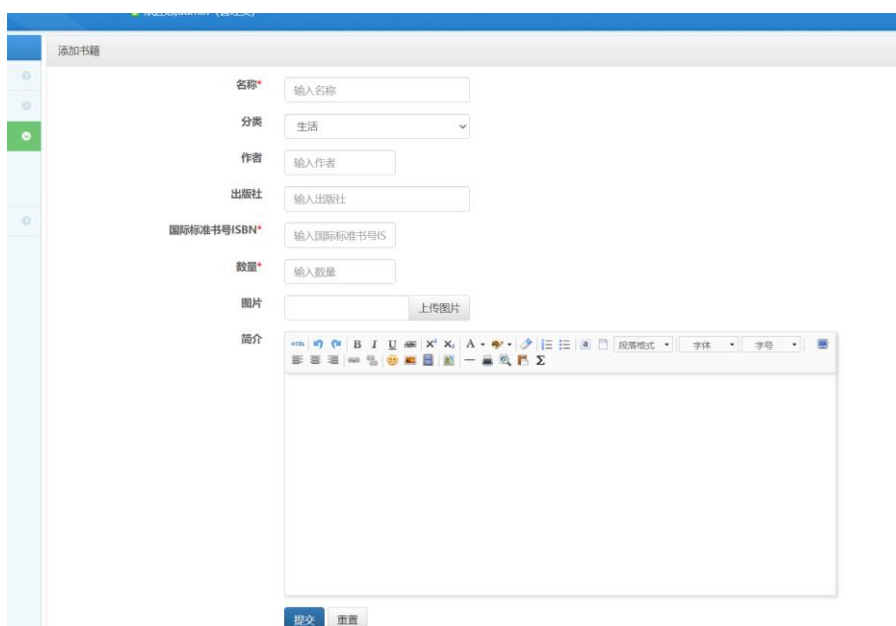
The login interface for the Library Management System. It features a dark blue header with the title '图书管理系统' (Library Management System). Below the header is a white login box with a blue title bar '系统登录'. The box contains three input fields: '用户名' (Username), '密码' (Password), and '验证码' (Captcha). The captcha is a green box with the number '3481'. At the bottom of the box are two blue buttons: '登录' (Login) and '注册' (Register). The background of the interface shows a book cover with the title '我循着火光而来' (I follow the fire light) by Zhang Yueran.

管理员权限登录后：



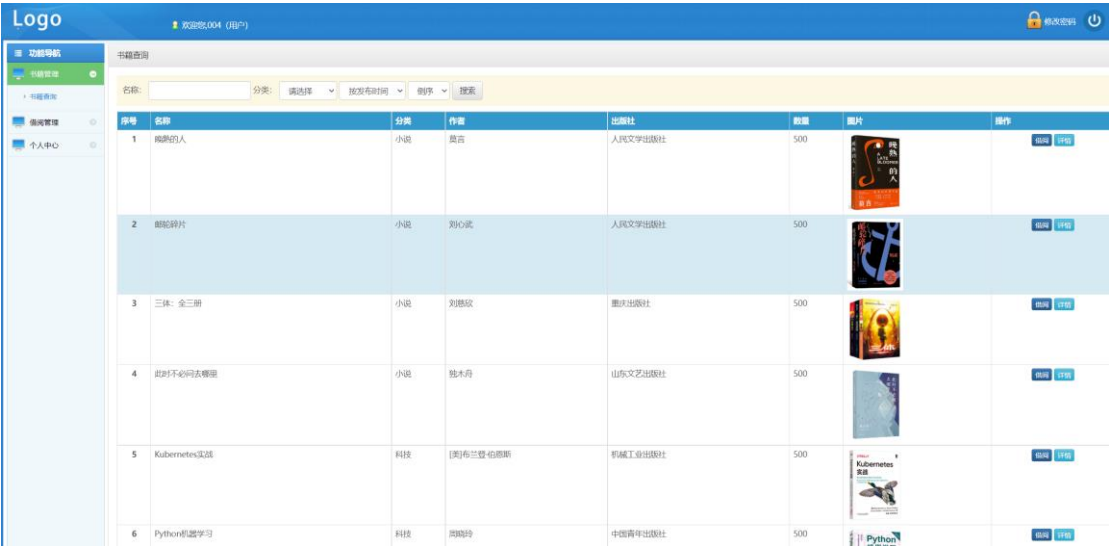
The admin dashboard of the Library Management System. It has a blue header with the 'Logo' and a user profile '张悦然 (管理员)'. The left sidebar contains navigation links: '功能导航', '账号管理', '分类管理', '书籍管理', '书籍添加', '书籍查询', and '借阅管理'. The main content area shows a table of books with columns: '序号' (Serial Number), '名称' (Name), '分类' (Category), '作者' (Author), '用户' (User), '开始' (Start), '结束' (End), '时长' (Duration), '状态' (Status), and '操作' (Action). The table contains three rows of data.

序号	名称	分类	作者	用户	开始	结束	时长	状态	操作
1	Python机器学习	科技	周志华	004	2023-12-21	2023-12-22	1	已归还	删除 重置
2	解忧杂货店	小说	东野圭吾	001	2023-12-21	2023-12-22	1	已归还	删除 重置
3	活着的人	小说	余华	001	2023-12-21	2023-12-22	1	已归还	删除 重置



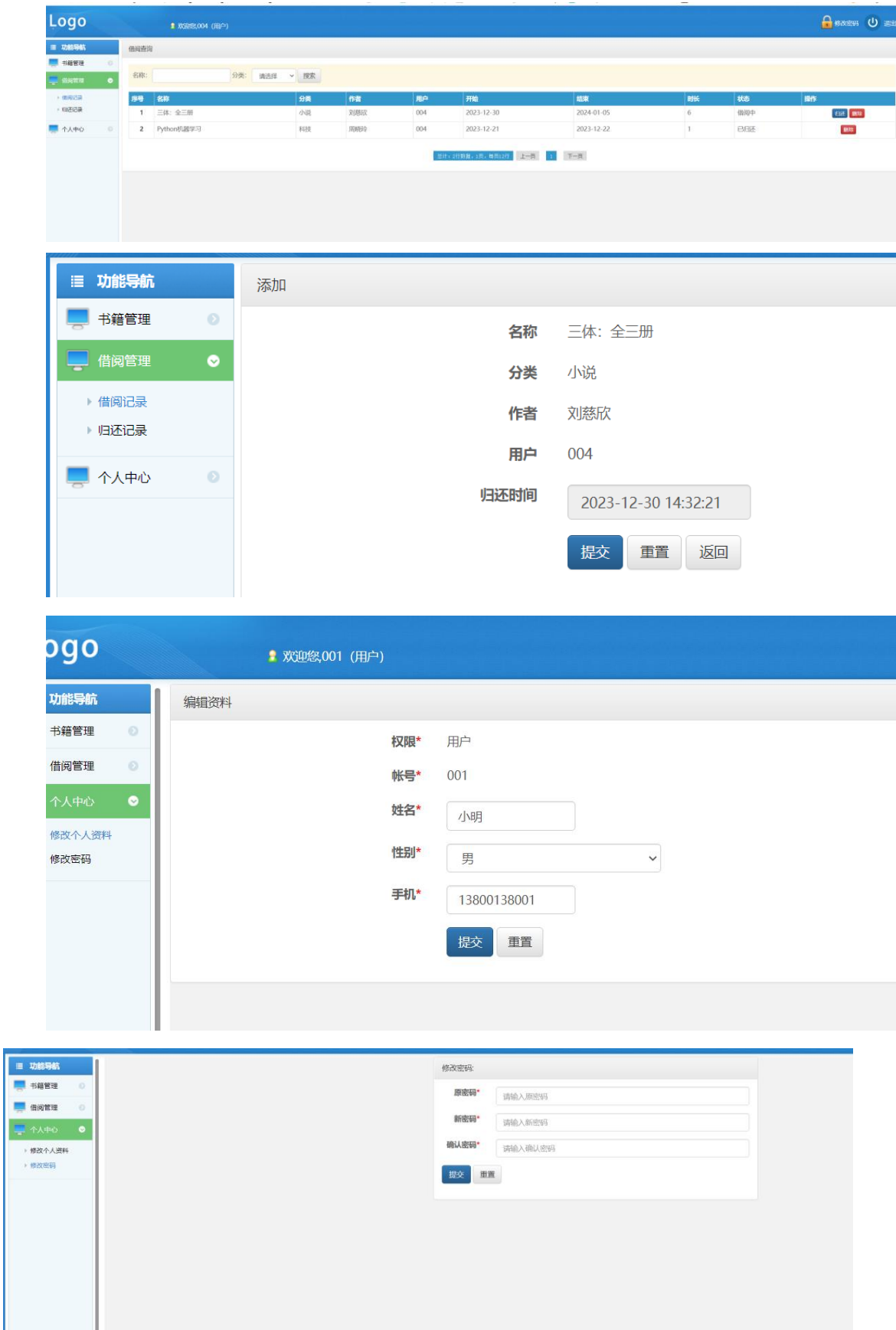
The '添加书籍' (Add Book) form in the Library Management System. It has a blue header with the title '添加书籍'. The form contains several input fields: '名称*' (Name), '分类' (Category), '作者' (Author), '出版社' (Publisher), '国际标准书号ISBN*' (ISBN), '数量*' (Quantity), and '图片' (Image). There is also a '简介' (Introduction) text area. At the bottom are two buttons: '提交' (Submit) and '重置' (Reset).

用户登录后：



借阅/归还书籍：





- 用户友好的界面：前台界面采用 Bootstrap 和响应式设计，提供了良好的用户体验。
- 身份验证：用户可以安全地登录系统，管理员和普通用户分别拥有不同的权限。

- 图书管理：管理员可以轻松管理图书信息，包括添加、删除和更新操作。
- 借阅和还书：用户可以借阅和还书图书，系统会记录借阅历史，用户可以查看借阅状态。

系统实现过程中，充分考虑了用户需求和系统的稳定性，确保了图书管理系统的顺利运行和用户满意度。在接下来的章节中，将进一步详细说明各个关键功能的实现细节。

4.2 系统实现的不足

系统实现的不足主要包括以下几个方面：

1. **用户体验方面的改进：** 尽管系统提供了基本的用户界面，但仍有改进的空间，以提高用户体验。例如，可以进一步改进前端界面的设计，增加交互性元素，使用户更容易使用系统。
2. **系统安全性：** 虽然系统实现了基本的用户身份验证和 Session 管理，但对于安全性还有进一步加强的空间。需要更全面的安全性措施，如输入验证、防止 SQL 注入等，以防止潜在的安全威胁。
3. **性能优化：** 随着系统使用的增加，性能可能会成为一个问题。需要考虑优化数据库查询、使用缓存等方式来提高系统的响应速度，特别是在大规模数据操作时。
4. **错误处理和日志记录：** 在系统中加强错误处理和日志记录，以便更容易追踪和排查问题。用户应该能够获得有用的错误信息，而开发人员应该能够访问系统日志以监视系统运行。
5. **前端与后端的分离：** 目前系统的前端和后端代码可能存在耦合，不易维护和扩展。考虑将前端与后端进行更好的分离，采用前后端分离的开发方式，以提高可维护性和扩展性。

第五章 总结与展望

5.1 总结

在本项目中，我们成功地设计和实现了一个 Java MVCS 架构的图书管理系统。该系统具备了基本的用户身份验证、图书管理、借阅和还书等核心功能。系统实现了以下关键点：

- 用户身份验证：**实现了用户登录和身份验证功能，通过 Session 管理登录状态，确保只有经过验证的用户可以访问系统。
- 图书管理：**管理员用户可以执行添加、删除、更新和检索图书信息的操作，确保图书信息的准确性和完整性。
- 借阅和还书：**用户可以借阅和还书图书，系统会记录借阅历史，用户可以查看借阅状态。
- 前台展示：**前台界面采用 Bootstrap 和响应式设计，提供了良好的用户体验。

在系统的实际实现过程中，面临了一些挑战，包括用户体验的改进、安全性的加强、性能优化、错误处理和日志记录等方面的改进空间。然而，通过不断改进和优化，系统已经能够满足基本的图书管理需求，并为用户和管理员提供了一种有效的方式来管理图书。

5.2 展望

尽管已经实现了基本的功能，但图书管理系统仍有进一步的发展空间和改进机会。以下是一些可能的展望和改进方向：

- 安全性增强：**可以进一步加强系统的安全性，包括数据加密、输入验证、防止 SQL 注入等方面的增强。
- 性能优化：**随着系统的使用增加，可能需要进一步优化性能，包括数据库查询的优化、缓存的使用等。
- 功能扩展：**可以考虑添加更多的功能，如图书推荐、在线支付、电子书管理等，以满足不同用户的需求。
- 用户文档和培训：**提供详细的用户文档和培训材料，以帮助用户更好地使用系统。
- 前后端分离：**考虑采用前后端分离的开发方式，提高系统的可维护性和扩展性。
- 自动化测试：**增加自动化测试覆盖率，以确保系统的稳定性和质量。

-
- **响应式设计：** 进一步改进前台界面的响应式设计，以适应不同设备和屏幕尺寸。

总之，图书管理系统在不断发展和完善中，可以根据实际需求和用户反馈进行进一步的改进和扩展，以提供更好的服务和体验。系统的未来发展将取决于需求和技术的发展趋势。