|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 题目名称 | 基于Java Web技术实现的一个图书商城 | | | | |
| **姓名** |  | **班级** | **软工191** | **学号** | **102** |
|  | | | | | |
| 具体要求 | | | | | |
| **1.功能要求**  **1）前端功能**  图书的分类展示;  图书详情  图书搜索  订单详情(购物车)  登录与注册  注意：注册用户可以购买商品，非注册用户仅可以浏览、查询商品  **2）后端功能**  用户管理：分普通用户和管理员两类；管理员可以禁用普通用户帐号、查看普通用户的订单信息；可以添加、修改和删除其他管理员帐号；  图书管理：管理员对图书信息的添加、修改、删除和查询功能，包括图书详细信息、图片上传等。  **2.技术要求**  1）基于Java Web开发技术，并使用MVC开发模式实现；数据库使用mysql；  2）知识点应用：[1]项目创建；代码调试;[2]JSP基础语法；[3]内置对象使用；[4]合理使用Javabean；[5]理解和掌握servlet；[6]理解和掌握MVC开发模式；[7]掌握在JSP中操作数据库的方法；[8]掌握JSTL核心标签库的使用；[9]前端基础知识的综合应用。  **3.设计文档**  1）内容  整体设计（功能描述、系统结构图、使用到的技术等）；数据库设计；详细设计(代码和界面)；小结；自我评价；  2）格式  参照任务书模版。要求格式良好。  提交：  1.项目源代码（包括数据库，导出数据库结构和数据，便于阅卷）；  2.本设计文档；  3.提交日期；2021-12-24 | | | | | |

目·录

[基于Java Web技术实现的一个图书商城 3](#_Toc90564311)

[一、 整体设计 3](#_Toc90564312)

[1.1系统结构 3](#_Toc90564313)

[1.2系统功能结构图 3](#_Toc90564314)

[1.2.1用户功能结构图 3](#_Toc90564315)

[1.2.2管理员功能结构图 4](#_Toc90564316)

[1.3数据JavaBean总体设计 4](#_Toc90564317)

[二、数据库设计 4](#_Toc90564318)

[2.1E-R图 4](#_Toc90564319)

[2.2设计原理 5](#_Toc90564320)

[三、详细设计 5](#_Toc90564321)

[3.1用户功能模块设计 5](#_Toc90564322)

[3.1.1主页设计 5](#_Toc90564323)

[3.1.2商品详情设计 8](#_Toc90564324)

[3.1.3购物车设计 11](#_Toc90564325)

[3.1.4订单展示设计 14](#_Toc90564326)

[3.2管理员功能模块设计 16](#_Toc90564327)

[3.2.1登陆模块设计 16](#_Toc90564328)

[3.2.2主页模块设计 18](#_Toc90564329)

[四、结果展示 18](#_Toc90564330)

[4.1用户展示 18](#_Toc90564331)

[4.2管理员展示 22](#_Toc90564332)

[五、设计小结 25](#_Toc90564333)

[六、自我评价 26](#_Toc90564334)

# 基于Java Web技术实现的一个图书商城

## 整体设计

### 1.1系统结构

系统采用jsp页面+servlet+JavaBean开发模式，采用MVC的开发模式，对每个模块都采用该种开发模式进行相应的设计。

### 1.2系统功能结构图

#### 1.2.1用户功能结构图



图1. 1用户系统功能图

#### 1.2.2管理员功能结构图



图1. 2管理员功能图

### 1.3数据JavaBean总体设计

采用javaBean的设计方案，其中包括用户类，管理员类，书籍类，书籍类型类，购物车类，订单类。

每个类的设计都是与数据库内的数据相对应，最后加上自己所需的数据。

## 二、数据库设计

### 2.1E-R图

数据库的整体E-R图



图2. 1整体E-R图

### 2.2设计原理

管理员表为登陆验证表，只需要账号和密码，书籍表包含多种属性，其中ID为主键，也是购物车表和订单表的外键；书籍类型的id是书籍表的外键；每个用户对自己的购物车可以进行操作。其中用户表的power是权限，由管理员可以设置。

## 三、详细设计

### 3.1用户功能模块设计

#### 3.1.1主页设计

当用户打开该网站的时候，无论用户是否登陆都可以进行预览，默认是展示所有的商品，提供按书名查询、按类型查询、按书名加类型查询。点击查询时，将请求信息提交到Main的servlet进行查询，并且返回当前页面。

当点击书籍时，将进入书籍的详细信息页面，实现方法是将书籍的ID提交到BookMsg的servlet进行处理。并且转发到详情页面，

main.jsp页面代码

<div id="main">

<h1>图书销售网</h1>

<br/>

<div id="inp">

<form action="Main" method="post">

书名：<input placeholder="输入书名搜索" name="findOfName" value="${findOfName }">

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

类型：

<select name="booktype">

<option value ="">全部</option>

<c:forEach items="${booktypes}" var="type">

<option value="${type.name }" ${booktype eq type.name ?'selected':''}>${type.name }</option>

</c:forEach>

</select>

&nbsp;&nbsp;&nbsp;&nbsp;

<button style="width: 80px">查找</button>

</form>

</div>

<br/>

<br/>

<ul id="book\_ul">

<c:forEach items="${books}" var="book">

<li id="book" onclick="window.location.href='BookMsg?bookId='+${book.bookId}">

<img src="${book.photo }">

<span style="left:10px; font-size:17px;">${book.bookname}</span>

<br/>

<br/>

<span style="left:10px; font-size:19px; color: red">￥${book.price}元</span>

</li>

</c:forEach>

</ul>

</div>

Servlet的设计

Main.java

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

request.setCharacterEncoding("utf-8");

response.setContentType("text/html;charSet=utf-8");

Connection conn = DBConn.getConn();

String sql=null;

//获取类型列表

try {

sql = "select \* from booktype";

PreparedStatement pst = conn.prepareStatement(sql);

ResultSet rs = pst.executeQuery();

List<BookType> booktypes = new ArrayList<BookType>();

while(rs.next()) {

BookType booktype = new BookType();

booktype.id=rs.getInt("id");

booktype.name=rs.getString("name");

booktype.msg = rs.getString("msg");

booktypes.add(booktype);

}

request.setAttribute("booktypes", booktypes);

}catch (Exception e) {

// TODO: handle exception

e.printStackTrace();

}

//获取书籍信息

List<Book> books = new ArrayList<Book>();

try {

String name = request.getParameter("findOfName");

name=name==null?"":name;

String typename = request.getParameter("booktype");

typename=typename==null?"":typename;

//保存搜索内容，返回时依旧显示

request.setAttribute("findOfName", name);

request.setAttribute("booktype", typename);

sql = "select \* from booklist A join booktype B "

+ "on A.typeid = B.id where A.bookname like ? and B.name like ?";

PreparedStatement pst = conn.prepareStatement(sql);

pst.setString(1,"%" + name + "%");

pst.setString(2,"%" + typename + "%");

ResultSet rs = pst.executeQuery();

while(rs.next()){

Book book = new Book();

book.bookId = rs.getInt("bookId");

book.bookname = rs.getString("bookname");

book.price = rs.getDouble("price");

book.typename = rs.getString("name");

book.photo = rs.getString("photo");

book.bookmsg = rs.getString("bookmsg");

books.add(book);

}

request.setAttribute("books", books);

//转发到成功的页面

request.getRequestDispatcher("main.jsp").forward(request, response);

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

BookMsg.java

String bookid = request.getParameter("bookId");

Connection conn = DBConn.getConn();

String sql = null;

PreparedStatement pst = null;

ResultSet rs = null;

try {

sql = "select \* from booklist where bookid = ? ";

pst = conn.prepareStatement(sql);

pst.setString(1, bookid);

rs = pst.executeQuery();

rs.next();

Book book = new Book();

book.bookId = rs.getInt("bookid");

book.bookname = rs.getString("bookname");

book.price = rs.getDouble("price");

book.photo = rs.getString("photo");

book.bookmsg = rs.getString("bookmsg");

sql = "select \* from booktype where id = ? ";

pst = conn.prepareStatement(sql);

pst.setInt(1, rs.getInt("typeid"));

rs = pst.executeQuery();

if(rs.next()) {

book.typename = rs.getString("name");

}

request.setAttribute("book", book);

rs.close();

pst.close();

conn.close();

request.getRequestDispatcher("bookmsg.jsp").forward(request, response);

#### 3.1.2商品详情设计

商品详情处展示商品的详细信息，其中包括商品的名称、价格、类型、详细介绍，提供添加购物车的功能。

点击添加购物车时，若未登陆，则跳转至登陆页面；否则添加入购物车。

点击左上角的返回可以返回主页。

Bookmsg.jsp

<a href="Main">返回主页</a>

<jsp:include page="loginTag.jsp"/>

<h1>图 书 详 情</h1>

<div id="main">

<div id="img"><img src="${book.photo}"></div>

<div id="bookmsg1">

<p>书名: ${book.bookname}</p>

<p>价格: ￥${book.price}元</p>

<p>类型: ${book.typename}</p>

<br/>

<button id='bt' onclick="window.location.href='CartUp?bookId=${book.bookId}&source=BookMsg'">加入购物车</button>

</div>

<h2>详情</h2>

<div id="bookmsg2">

<p style="margin:30px">${book.bookmsg}</p>

</div>

</div>

提交的servlet

CartUp.java

String bookId = request.getParameter("bookId");//获取书籍信息

String loginTag = (String)((request.getSession()).getAttribute("loginTag"));//获取登陆用户名

//未登录

if(loginTag==null) {

response.sendRedirect("login.jsp");

return;

}

Connection conn=DBConn.getConn();

PreparedStatement pst=null;

String sql =null;

ResultSet rs = null;

int userid=0;

//获取登陆人id

{

sql="select \* from User where username = ?";

try {

pst = conn.prepareStatement(sql);

pst.setString(1, loginTag);

rs = pst.executeQuery();

rs.next();

userid = rs.getInt("id");

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

//新增购物车或商品+1

sql = "select \* from Cart where bookId= ? and userid =?";

try {

pst= conn.prepareStatement(sql);

pst.setString(1, bookId);

pst.setInt(2,userid);

rs = pst.executeQuery();

//该用户存在该商品

if(rs.next()) {

int num = rs.getInt("num");

sql = "update Cart set num = ? where bookId= ? and userid =?";

pst= conn.prepareStatement(sql);

pst.setInt(1, ++num);

pst.setString(2, bookId);

pst.setInt(3,userid);

pst.execute();

}else {

//不存在

sql = "insert into Cart values(?,?,1)";

pst = conn.prepareStatement(sql);

pst.setString(1, bookId);

pst.setInt(2,userid );

pst.execute();

}

rs.close();

pst.close();

conn.close();

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

source= source.equals("BookMsg")?"BookMsg?bookId="+bookId:source;

response.sendRedirect(source);

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

#### 3.1.3购物车设计

在首页内，用户可以点击右上角的购物车标志，从而进入购物车，在购物车内，用户可以看到商品的详细信息，并且可以对商品进行数量的增加以及删除，实现方法是每次点击，向后端提交一次请求，并且存储到数据库，再转发到当前页面。

可以全选或者全不选，或者选择一部分，最后进行模拟提交，形成相应的订单。

cart.jsp

<jsp:include page="loginTag.jsp" />

<a href="Main">返回主页</a>

<div id="main">

<h1>购物车</h1>

<table border="1">

<tr>

<th style="width: 50px">选择</th>

<th>图片</th>

<th>名称</th>

<th>价格</th>

<th>数量</th>

</tr>

<c:forEach items="${carts}" var="cart">

<tr>

<!-- 复选框 -->

<td><input type="checkbox" name="select" value="${cart.book.bookId}"></td>

<td><img id="cart\_img" src="${cart.book.photo}"/></td>

<td>${cart.book.bookname}</td>

<td>${cart.book.price}</td>

<td>

<button onclick="window.location.

href='CartDown?bookId=${cart.book.bookId}'">-</button>

${cart.num}

<button onclick="window.location.

href='CartUp?bookId=${cart.book.bookId}&source=MyCart'">+</button>

</td>

</tr>

</c:forEach>

<tr>

<td><button onclick="selectAll()">全选</button></td>

<td colspan="4"><button onclick="submit()">提交订单</button></td>

</tr>

</table>

</div>

</body>

<script>

function selectAll() {

var select = document.getElementsByName("select");

var tag = true;

for(var i=0;i<select.length;i++){

if(select[i].checked==false){

tag=false;

}

}

for(var i=0;i<select.length;i++){

if(tag==false)

select[i].checked = true;

else

select[i].checked = !select[i].checked;

}

}

//提交

function submit() {

var s = document.getElementsByName("select");

var select='';

for(var i=0;i<s.length;i++){

if(s[i].checked==true){

select += s[i].value+"\*";

}

}

window.location.href='CartSubmit?select='+select;

}

</script>

提交订单的servlet

CartSubmit.java

//用户获取

HttpSession se = request.getSession();

//未登录

if(se.getAttribute("loginTag")==null) {

response.sendRedirect("login.jsp");

return;

}

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

if(select==""||select==null) {

response.sendRedirect("MyCart");

return;

}

//书籍Id获取

String[] bookids = select.split("\\\*");

Connection conn=DBConn.getConn();

PreparedStatement pst=null;

String sql =null;

ResultSet rs = null;

//用户id获取

int userid = -1;

try {

sql = "select \*from user where username = ?";

pst = conn.prepareStatement(sql);

pst.setString(1, (String)se.getAttribute("loginTag"));

rs = pst.executeQuery();

rs.next();

userid = rs.getInt("id");

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

for(String bookid :bookids) {

Goods good = new Goods();

good.bookid = Integer.valueOf(bookid);

good.userid = userid;

double price = 0;

//查询购物车的数量

try {

sql = "select \* from cart A join booklist B on A.bookid=B.bookid "

+ " where A.bookid = ? and userid = ?";

pst = conn.prepareStatement(sql);

pst.setString(1, bookid);

pst.setInt(2, userid);

rs = pst.executeQuery();

rs.next();

good.num = rs.getInt("num");

price = rs.getDouble("price");

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

//执行数据库的提交

try {

Timestamp time = new Timestamp(System.currentTimeMillis());

sql="insert into goods value(null,?,?,?,?,?,?)";

pst = conn.prepareStatement(sql);

pst.setInt(1, good.bookid);

pst.setInt(2, good.userid);

pst.setInt(3, good.num);

pst.setString(4, "已提交");

pst.setTimestamp(5, time);

pst.setDouble(6, good.num\*price);

pst.execute();

//删除购物车

sql = "delete from cart where bookid = ? and userid = ?";

pst = conn.prepareStatement(sql);

pst.setInt(1,good.bookid);

pst.setInt(2,good.userid);

pst.execute();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

try {

rs.close();

pst.close();

conn.close();

response.sendRedirect("MyCart");

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

#### 3.1.4订单展示设计

点击该订单，页面对servlet请求的数据进行展示，对展示的数据可以按照时间顺序进行排序，提供按照商品名称进行查询

订单展示页面代码

goodslist.jsp

<div id="main">

<h1>订 单 中 心</h1>

<form action="GoodsList" method="post">

<div id="inp">

<input placeholder="输入商品名称搜索" name="findOfName" value="${findOfName }">

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

状态：

<select name="state">

<option value ="">全部</option>

<option value ="已提交" ${state eq "已提交" ?'selected':''}> 已提交</option>

<option value ="已发货" ${state eq "已发货" ?'selected':''}> 已发货</option>

<option value ="已完成" ${state eq "已完成" ?'selected':''}> 已完成</option>

</select>

&nbsp;&nbsp;&nbsp;&nbsp;

<button>查找</button>

</div>

</form>

<br/>

<br/>

<table border="1">

<tr>

<th>订单号</th>

<th style="width:100px;height: 70px;">图片</th>

<th>商品名称</th>

<th>数量</th>

<th>金额</th>

<th style="width:200px">时间</th>

<th>状态</th>

</tr>

<c:forEach items="${goodslist}" var="goods">

<tr>

<td>${goods.id }</td>

<td><img style="width:60px;height: 60px;" src="${goods.photo}"/></td>

<td>${goods.bookname }</td>

<td>${goods.num }</td>

<td>${goods.sumprice }</td>

<td>${goods.time }</td>

<td><div style="color: blue">${goods.state }</div></td>

</tr>

</c:forEach>

</table>

</div>

### 3.2管理员功能模块设计

管理员的功能和用户的不完全相同，因此登陆的系统也不完全一样，其可以禁用用户的登陆功能，以及增删改查图书。

#### 3.2.1登陆模块设计

管理员进行登陆，和用户登陆的模块基本一样，因此可以进行复用。登陆成功才允许进行后台进行管理。

提交到后端servlet会进行数据的整理，对于各种问题都会进行转发回，成功之后转发到admain页面。

登陆页面设计

<div id="login">

<h1>管 理 员 登 录</h1>

<form action="AdLoginServlet" method="post">

<br/>

&nbsp;&nbsp;&nbsp;

AdminUser:<input type="text" name="userName"/>

<br/>

<br/>

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

Password:<input type="password" name="password"/>

<br/>

<br/>

<br/>

<div id="button">

<button>登录</button>

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

<a href="adregister.jsp" >注 册</a>

<br/>

<p>${msg}</p>

</div>

<br/>

<div id="adin"><a href="login.jsp">用户登录</a></div>

</form>

</div>

提交到后端处理AdLoginServlet.java

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

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

if(userName==null) {

request.getRequestDispatcher("adlogin.jsp").forward(request, response);

return;

}

if(userName.equals("")) {

request.setAttribute("msg", "用户名为空");

request.getRequestDispatcher("adlogin.jsp").forward(request, response);

return;

}

if(password.equals("")) {

request.setAttribute("msg", "密码为空");

request.getRequestDispatcher("adlogin.jsp").forward(request, response);

return;

}

Connection conn=DBConn.getConn();

String sql = null;

try {

sql = "select \* from admin where name=?";

PreparedStatement pst = conn.prepareStatement(sql);

pst.setString(1, userName);

ResultSet rs = pst.executeQuery();

if(rs.next()) {//存在该管理员

//判断密码

if(!rs.getString("password").equals(password)) {

request.setAttribute("msg", "密码错误，请重新输入");

request.getRequestDispatcher("adlogin.jsp").forward(request, response);

return;

}

//登录成功

{

//转发到成功的页面

HttpSession se = request.getSession();

se.setAttribute("adlogintag", userName);

response.sendRedirect("admain.jsp");

}

}else {

request.setAttribute("msg", "用户不存在");

request.getRequestDispatcher("adlogin.jsp").forward(request, response);

}

rs.close();

pst.close();

conn.close();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

#### 3.2.2主页模块设计

主页包含左侧的导航栏，导航栏的设计可以满足到相应的模块进行操作。

主页展示代码：

<body>

<jsp:include page="adloginTag.jsp"/>

</body>

其中adloginTag为：

<div id="main">

<ul id="tagul">

<li style="height: 100px;">

欢迎你:<br/>

<span style="color: black;">${adlogintag}</span><br/>

<a href="AdExit">退出登录</a>

</li>

<li style="line-height: 70px;"><a href="AdUser">用户管理</a></li>

<li style="line-height: 70px;"><a href="AdAdmin">管理员管理</a></li>

<li style="line-height: 70px;"><a href="AdGoods">订单管理</a></li>

<li style="line-height: 70px;"><a href="AdBook">书籍管理</a></li>

</ul>

</div>

## 四、结果展示

### 4.1用户展示

浏览商品时：

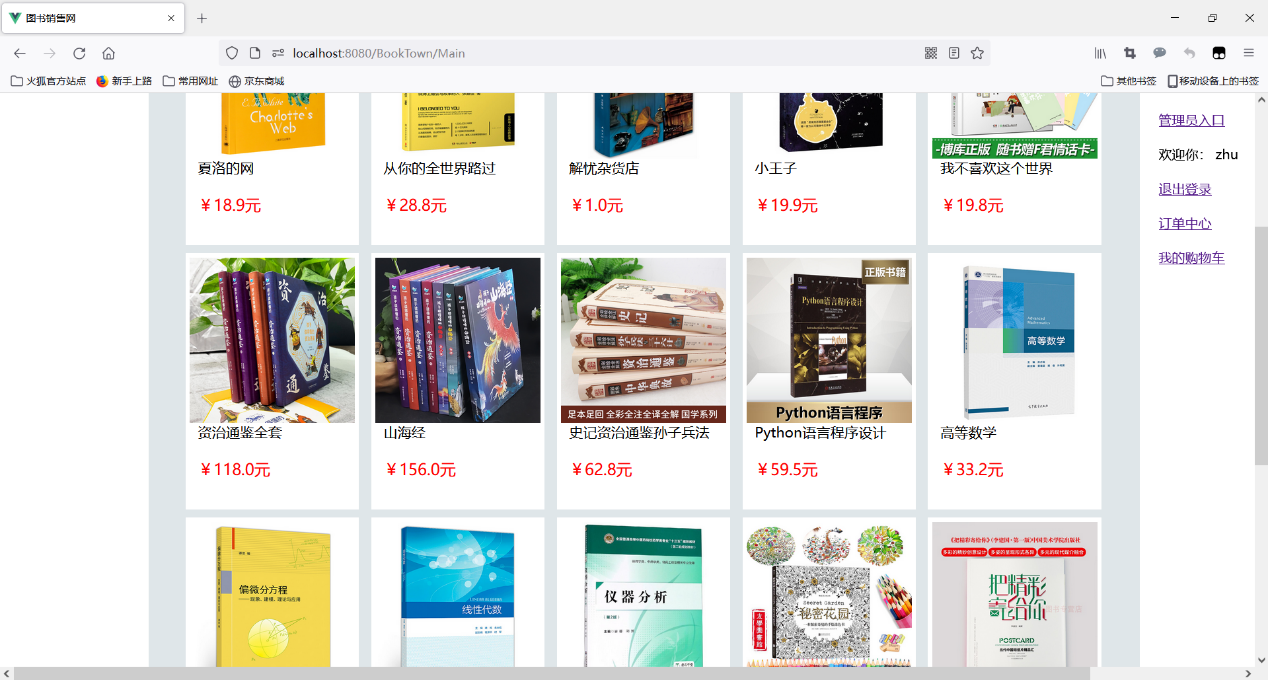


图4. 1浏览商品效果图

可以对商品进行筛选，对名称和状态进行筛选

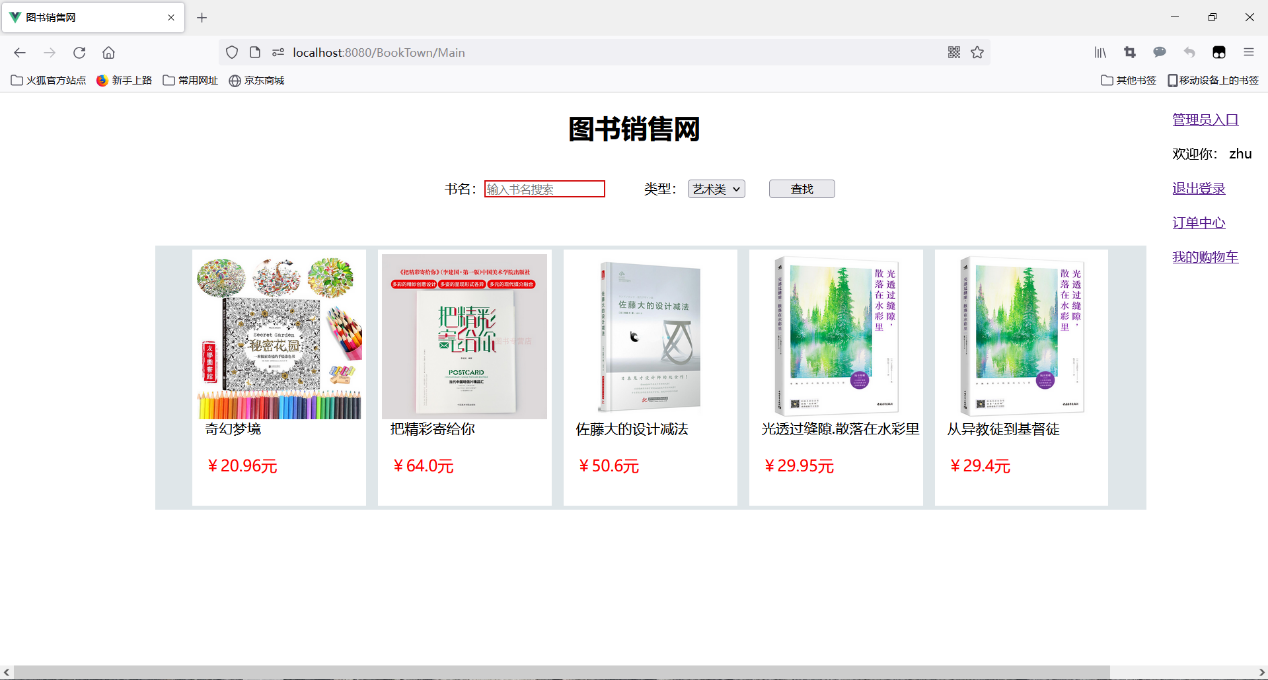


图4. 2商品筛选效果图

点击商品进入详情



图4. 3商品详情效果图

可以选择加入购物车

若未登录，跳转到登陆页面



图4. 4登陆页面

已登陆的状态，右上角会展示自己的信息，以及必要的订单按钮

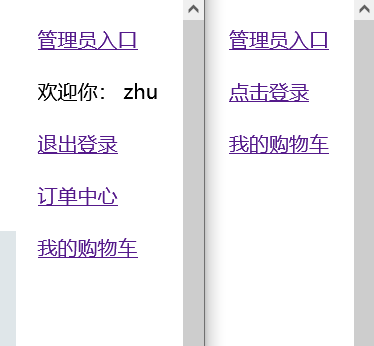


图4. 5登陆状态图

进入购物车，可以展示自己选择的商品

点击+-按钮，可以将商品进行修改



图4. 6购物车效果图

点击订单中心，可以展示自己的订单



图4. 7订单中心效果图

可以在上边对订单进行筛选，最后进行展示

### 4.2管理员展示

管理员先需要进行登陆才允许进入

管理页面

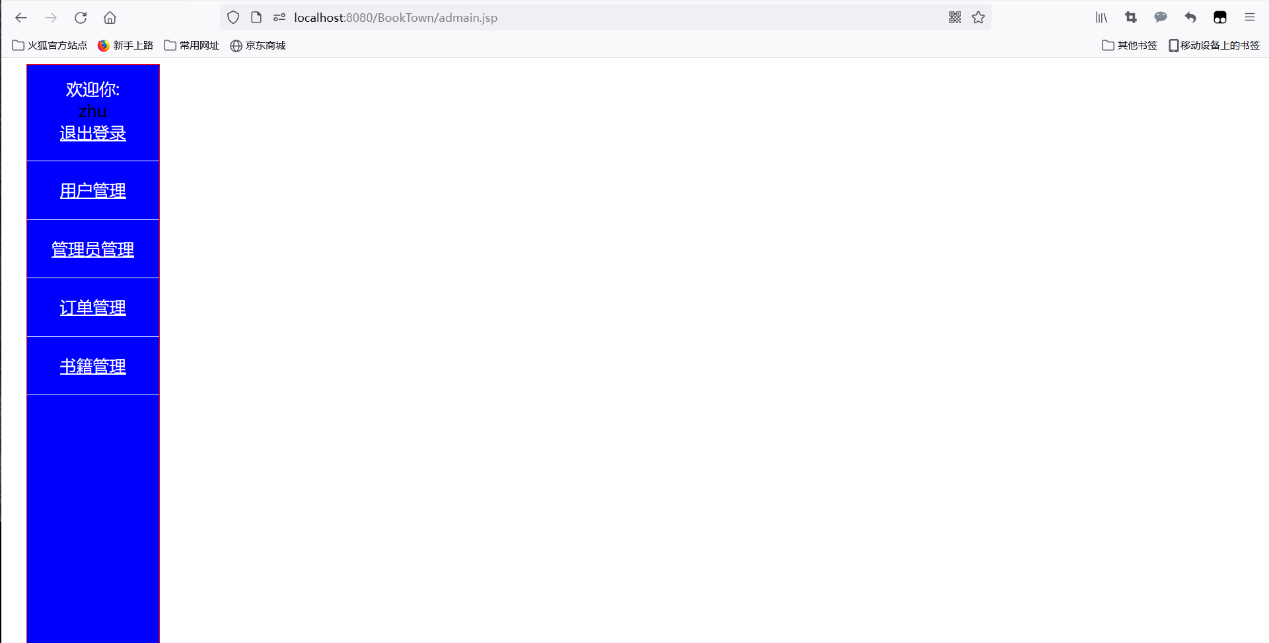


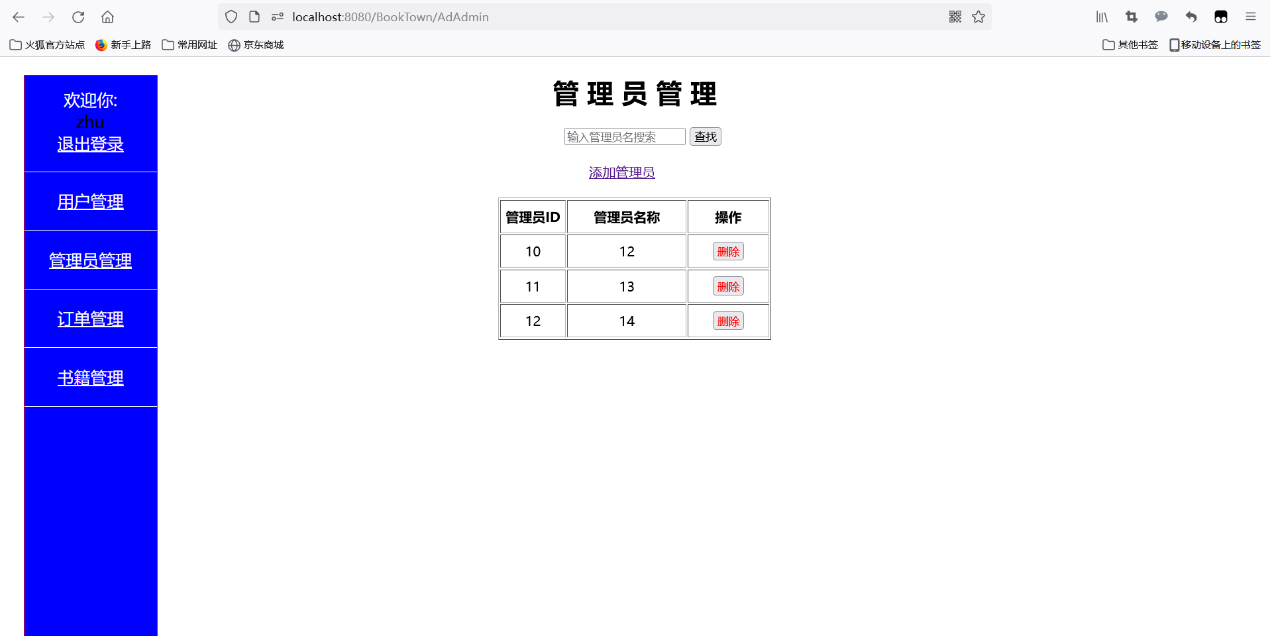
图4. 8管理页面

选择用户管理时，会展示所有的用户，并且提供用户的禁用/启用功能



图4. 9用户管理

管理员管理可以进行添加管理员，展示其他管理员和删除管理员的功能。



订单管理

可以对订单进行查询，展示每个用户的相应订单。



图4. 10订单管理

书籍管理

可以对书籍的信息进行增删改查

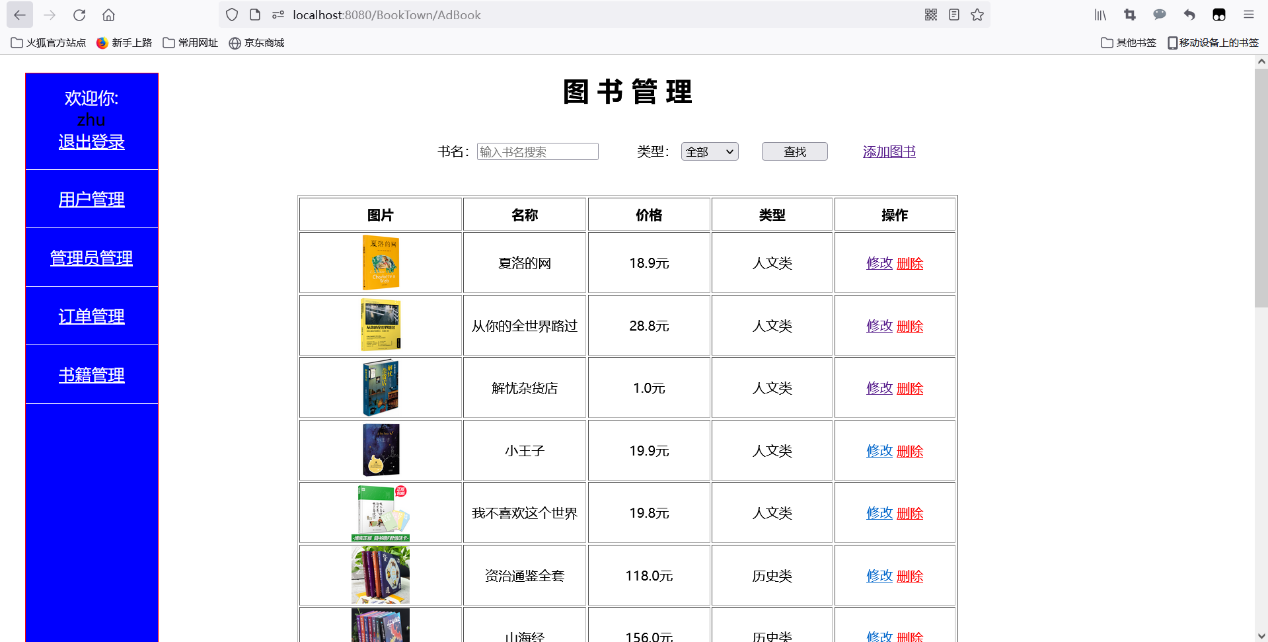


图4. 11书籍管理

书籍信息进行修改，修改页面

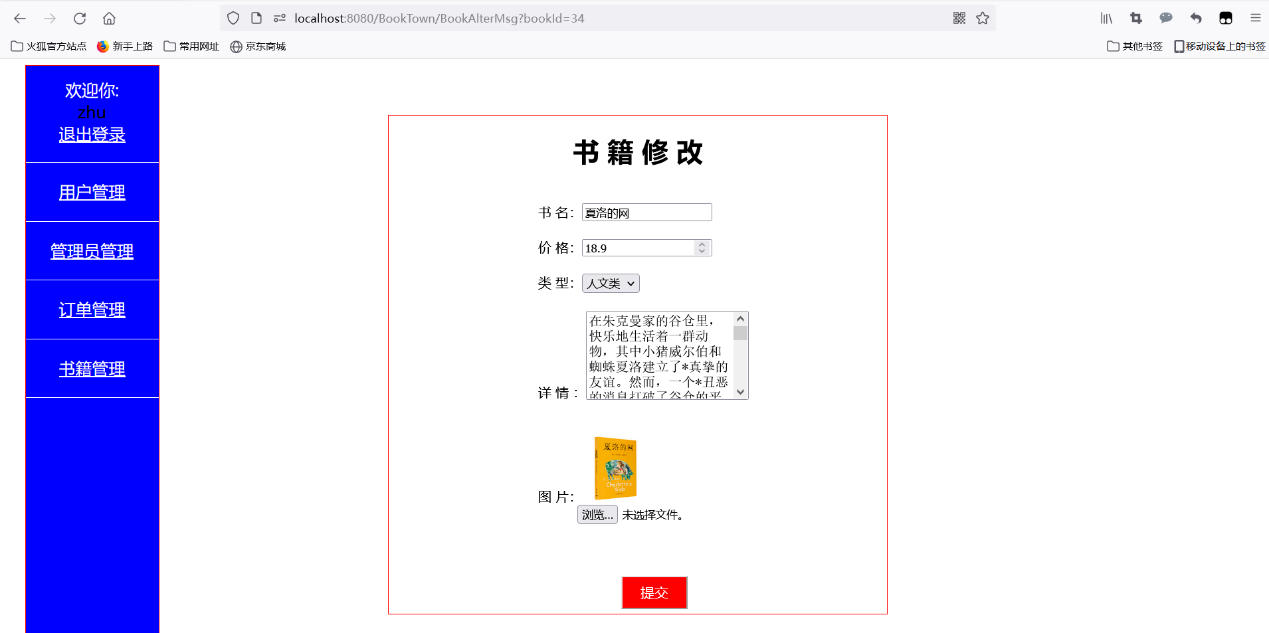


图4. 12书籍修改

书籍添加



图4. 13书籍添加

## 五、设计小结

本次设计使用了本学期所学习的所有知识，包括MVC的设计方法，和文件的上传等等概念，其中包括数据库设计，请求的转发和重定向等等。

在设计项目之初，已经在脑海里构思出了各种各样的逻辑结构和页面布局，最后敲定下来的一种，正所谓“磨刀不误砍柴功”，因为这种思想，在设计的时候，感觉事半功倍。其中在设计订单的时候，遇到一些数据库的设计问题，但是通过查阅资料以及翻阅老师上课的PPT，最后还是将其解决了。

请求的servlet会有多种转发途径，因此我选择将来源通过请求进行提交，最后按照一定的规定进行转发，从而实现了代码的复用。

在登陆注册会向servlet进行提交，并且对数据进行审查，对不符合要求的进行打回，并且提示用户，以完成引导用户的使用。

因此本次的作业给我的感触还是很多的，除了可以对上课的短板进行补充，还可以对所学的知识进行巩固，到目前为止，所学的jsp知识可以很好的和之前所学的知识进行结合。以后在学习的过程中会更加注重这种设计理念。

## 六、自我评价

我觉得本次作业的设计还是可以完全按照要求进行编写的，要求都可以实现了，虽然界面布局不太完美，介于单调，但是我觉得还是可以拿到良好~优秀的成绩的。