**《JAVA EE》课程设计**

**实验报告**

**题 目 ：**网上书店系统

**姓 名 ：**周亚亚

**学 号 ：**21160624

**学 院 ：**计算机科学与技术

**日 期 ：**2018.11.25

**目 录**

[1、开发相关技术及方法 3](#_Toc14091)

[2、系统架构及功能模块 3](#_Toc15715)

[3、数据库表的设计 4](#_Toc2561)

[4、测试效果及截图 5](#_Toc22948)

[5、部分代码附录 6](#_Toc32392)

[6、总结及心得体会 2](#_Toc18902)0

1. **开发相关技术及方法**

**1.1 开发环境**

（1）硬件环境：PC机

（2）软件环境：JDK-1.8，spring tool suite,Tomcat 9.0，Mysql

**1.2 相关技术及方法**

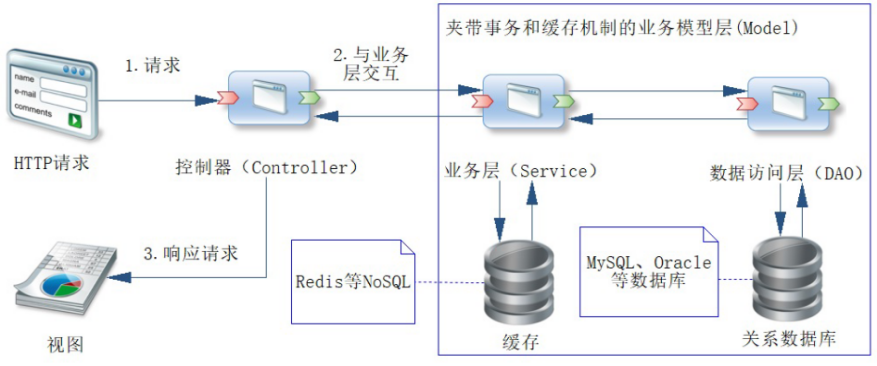
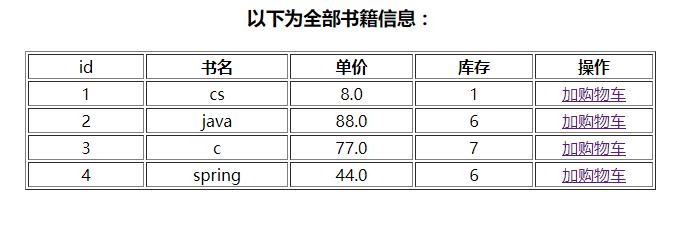
Jsp，servlet,spring MVC，spring boot，spring Data JPA，maven，Hibernate，数据库，服务器以及前端设计等

**2、系统架构及功能模块**

**2.1 系统架构**

本网上书店系统在整体开发时采用spring boot架构并通过注入依赖来自动配置Spring MVC框架，Spring MVC是Spring提供的一个实现了Web MVC设计模式的轻量级Web框架（图2.1.1），通过一套 MVC注解，让POJO（普通JavaBeans）成为处理请求的控制器，而无须实现任何接口。

对于数据库的操作，为了简化数据库的访问，使其访问变得简单快捷，本项目使用了Spring Data JPA，Spring Data JPA采用当前最为流行的Hibernate ORM框架，它允许开发者采用面向对象的方式来操作关系数据库，完成对象的持久化操作，减少了数据访问层 (DAO) 的开发量。

图2.1.1

**2.2 功能模块**

1、**登录**：提供注册界面和登录界面，注册则链接数据库后创建新的用户，登陆时会判断是否为管理员，若为管理员登录，则跳向后台管理界面，否则直接跳到书店首页。

2、**管理员操作菜单**：管理员可分别对图书进行操作（包括添加书籍、删除书籍、查看全部库存）及对用户进行操作（包括注销账户、查看全部账户）。

3、**购书主页**：此页面显示库存全部图书，进入该页后，用户可点击按钮将书籍信息通过get请求方式加入到购物车对象中)。

4、**购物车**：获取此用户对象的购物车并显示出来，计算当前购物车内图书的总价，用户在点击支付按钮前需提交个人信息（收件人、手机号、地址等）。

**3、数据库表的设计**

在数据库表的设计方面，本项目通过Hibernate持久化类自动创建了admin、reader和book三个数据库表。

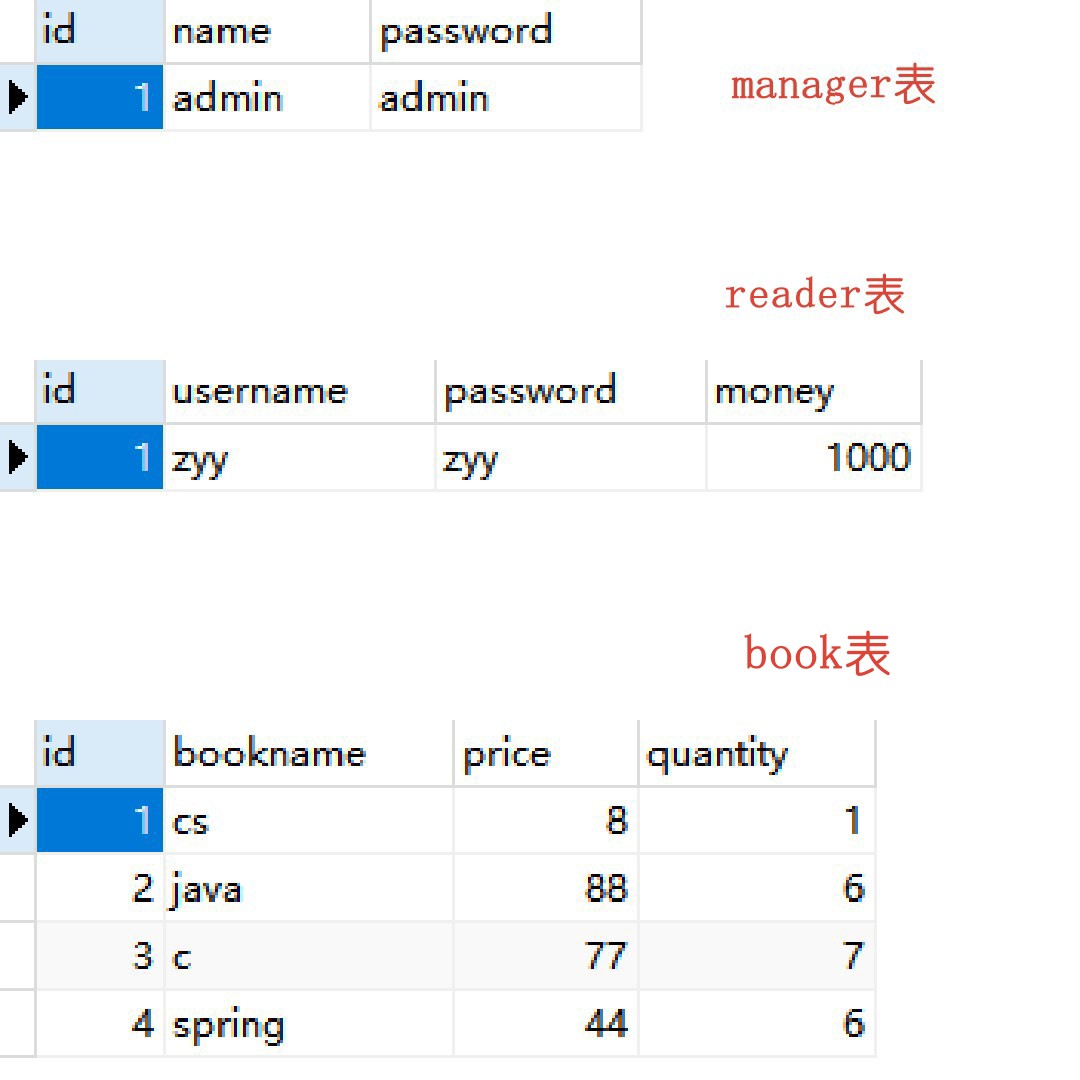
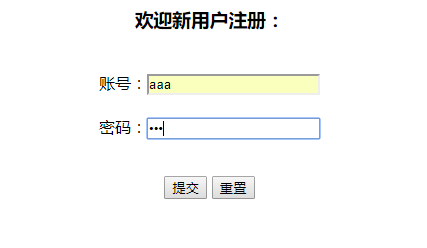
其中admin表保存了管理员的信息，在以管理员身份登陆时通过静态方式进行判断；reader表保存了该系统下的用户信息，可通过用户注册和管理员的注销账户操作进行动态更新；book表则保存当前系统库存全部书籍信息，可通过管理员的移除书目操作进行更新（具体内容参考图3.1）。



图3.1

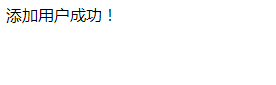
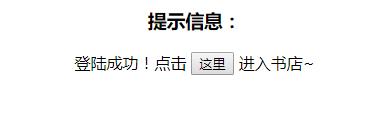
（注：图中数据库表中数据为初始数据）

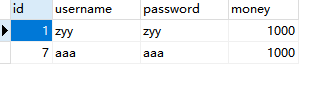
**4、测试效果及截图**

1、欢迎登陆 2、点击立即注册按钮

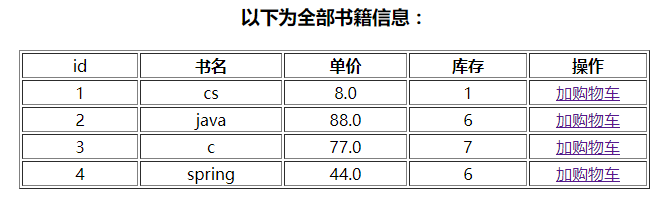


3、注册成功 4、用注册的账号进行登录



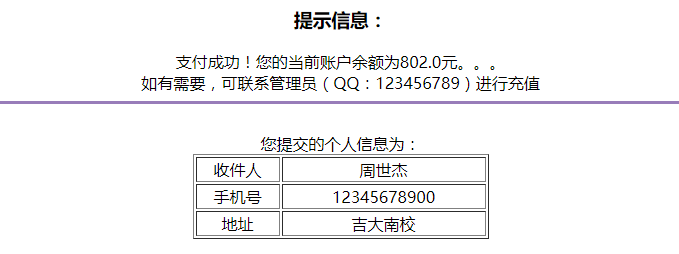


5、进入书店 6、加入多本图书至购物车

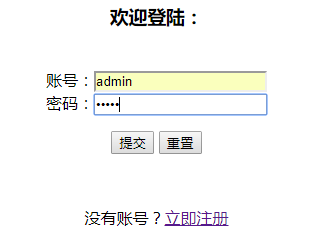


7、填写信息并点击支付 8、支付成功

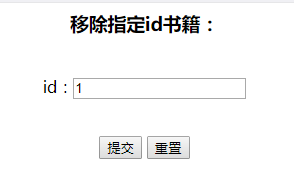
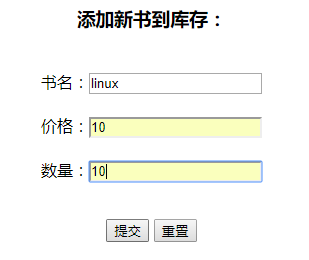




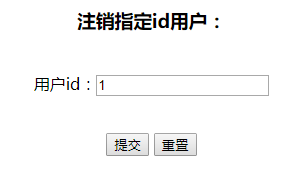
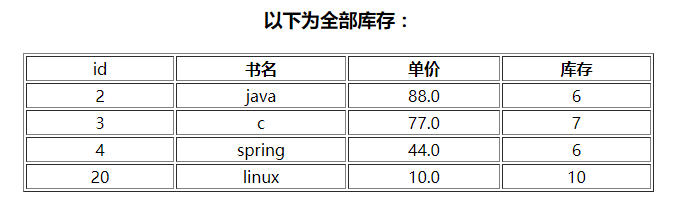
9、管理员登陆 10、管理员操作菜单



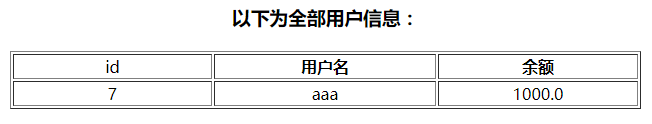
11、添加书籍 12、移除书籍



13、查看库存 14、注销账号

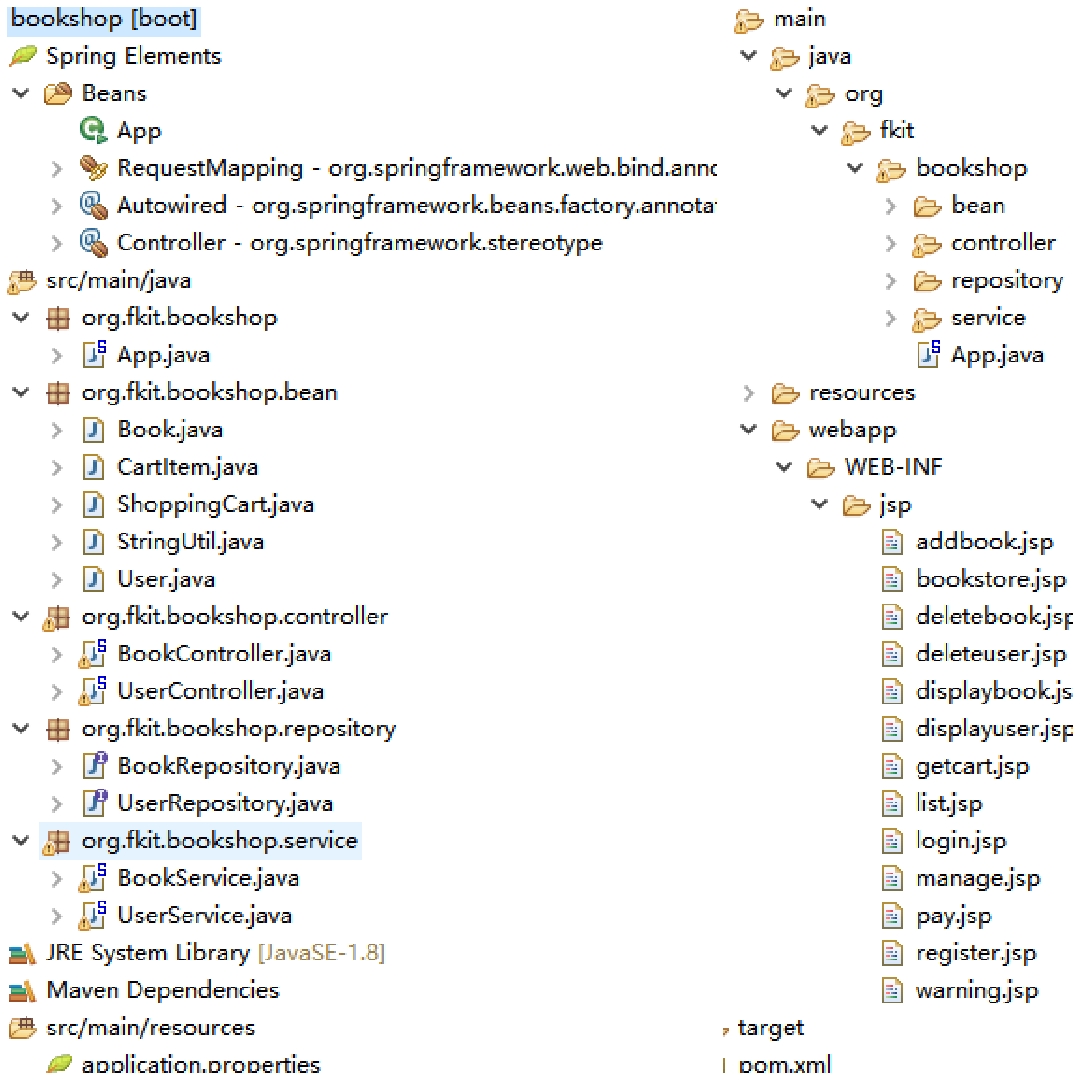


15、查看全部用户



**5、部分代码附录**

该web项目文件目录如下图所示，由于代码量较大，此处只附录部分代码。具体使用效果可参考上述第4部分-测试效果及截图...



**5.1 配置文件代码**

1、src/main/resources/application.properties：

spring.datasource.url=jdbc:mysql://localhost:3306/bookshop

spring.datasource.username=root

spring.datasource.password=root

spring.datasource.driverClassName=com.mysql.jdbc.Driver

spring.datasource.max-active=20

spring.datasource.max-idle=8

spring.datasource.min-idle=8

spring.datasource.initial-size=10

spring.jpa.database=MYSQL

spring.jpa.show-sql=true

spring.jpa.hibernate.ddl-auto=update

spring.mvc.view.prefix=/WEB-INF/jsp/

spring.mvc.view.suffix=.jsp

2、pom.xml：

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>org.fkit</groupId>

<artifactId>bookshop</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>bookshop</name>

<url>http://maven.apache.org</url>

<!--

spring-boot-starter-parent是Spring Boot的核心启动器，

包含了自动配置、日志和YAML等大量默认的配置，大大简化了我们的开发。

引入之后相关的starter引入就不需要添加version配置，

spring boot会自动选择最合适的版本进行添加。

-->

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.0.2.RELEASE</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncodin <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>

<java.version>1.8</java.version>

</properties>

<dependencies>

<!-- 添加spring-boot-starter-web模块依赖 -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

<!-- 添加MySQL依赖 -->

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

</dependency>

<!-- 添加Spring Data JPA依赖 -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<!-- 添加 JSTL（JSP Standard Tag Library，JSP标准标签库) -->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

</dependency>

<!-- Jasper是tomcat中使用的JSP引擎，运用tomcat-embed-jasper可以将项目与tomcat分开 -->

<dependency>

<groupId>org.apache.tomcat.embed</groupId>

<artifactId>tomcat-embed-jasper</artifactId>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

<configuration><fork>true</fork></configuration>

</plugin>

</plugins>

</build>

</project>

**5.2 src/main/java/org.fkit.bookshop**

1、App.java：

package org.fkit.bookshop;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class App

{

public static void main( String[] args )

{

SpringApplication.run(App.class, args);

}

}

2、User.java

package org.fkit.bookshop.bean;

import java.io.Serializable;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

// 用于标记持久化类,Spring Boot项目加载后会自动根据持久化类建表

@Table(name="reader")

public class User implements Serializable{

private static final long serialVersionUID = 1L;

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;// 主键.

private String username;// 姓名

private String password;// 密码

private float money; // 余额

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public float getMoney() {

return money;

}

public void setMoney(float money) {

this.money = money;

}

}

3、Book.java，CartItem.java：略

4、UserController.java：

package org.fkit.bookshop.controller;

import java.util.ArrayList;

import java.util.List;

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import org.fkit.bookshop.bean.User;

import org.fkit.bookshop.service.UserService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.bind.annotation.RestController;

import org.springframework.web.servlet.ModelAndView;

@Controller

@RequestMapping("/bookshop")

public class UserController {

// 注入UserService

@Autowired

private UserService userService;

@RequestMapping("/login")

public String Login() {

return "login";

}

@RequestMapping("/registerform")

public String registerForm() {

return "register";

}

@RequestMapping("/deleteform2")

public String deleteForm2() {

return "deleteuser";

}

@RequestMapping("/pay")

public ModelAndView Pay(HttpServletRequest req) {

String recipient = req.getParameter("recipient");

String phone = req.getParameter("phone");

String address = req.getParameter("address");

ModelAndView mv=new ModelAndView();

mv.setViewName("pay");

mv.addObject("recipient",recipient);

mv.addObject("phone",phone);

mv.addObject("address",address);

return mv;

}

@RequestMapping("/saveuser")

public @ResponseBody String save(HttpServletRequest req) {

User user = new User();

user.setUsername(req.getParameter("username"));

user.setPassword(req.getParameter("password"));

user.setMoney(1000);

user = userService.save(user);

return "添加用户成功！";

}

@RequestMapping("/deleteuser")

public @ResponseBody String delete(HttpServletRequest req) {

userService.delete(Integer.parseInt(req.getParameter("userid")));

return "指定id用户注销成功！";

}

@RequestMapping("/getAlluser")

public ModelAndView getAllUser() {

Iterable<User> user= userService.getAll();

List<User> listUser = new ArrayList<User>();

user.forEach(single->listUser.add(single));

ModelAndView mv=new ModelAndView();

mv.setViewName("displayuser");

mv.addObject("users", listUser);

return mv;

}

}

1. BookController.java：

package org.fkit.bookshop.controller;

import java.util.ArrayList;

import java.util.List;

import java.util.Iterator;

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.fkit.bookshop.bean.\*;

import org.fkit.bookshop.service.BookService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.bind.annotation.RestController;

import org.springframework.web.servlet.ModelAndView;

@Controller

@RequestMapping("/bookshop")

public class BookController {

// 注入BookService

@Autowired

private BookService bookService;

@RequestMapping("/addform")

public String addForm() {

return "addbook";

}

@RequestMapping("/deleteform1")

public String deleteForm1() {

return "deletebook";

}

@RequestMapping("/loginform")

public String loginForm(HttpServletRequest req) {

String username = req.getParameter("username");

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

if((username.equals("admin"))&&(password.equals("admin")))

return "manage";

else if((req.getParameter("username").equals("admin"))&&(!req.getParameter("password").equals("admin")))

return "warning";

else {

return "list";

}

}

@RequestMapping("/lists")

public ModelAndView Lists() {

Iterable<Book> book=bookService.getAll();

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

book.forEach(single->listBook.add(single));

ModelAndView mv=new ModelAndView();

mv.setViewName("bookstore");

mv.addObject("books", listBook);

return mv;

}

@RequestMapping("/save")

public @ResponseBody String save(HttpServletRequest req) {

Book book = new Book();

book.setBookname(req.getParameter("bookname"));

book.setPrice(Float.parseFloat(req.getParameter("price")));

book.setQuantity(Integer.parseInt(req.getParameter("quantity")));

book = bookService.save(book);

System.out.println("添加数据成功，返回的结果：" + book);

return "添加书籍成功！";

}

@RequestMapping("/deletebook")

public @ResponseBody String delete(HttpServletRequest req) {

bookService.delete(Integer.parseInt(req.getParameter("bookid")));

return "指定id书籍下架成功！";

}

@RequestMapping("/getAllbook")

public ModelAndView getAllBook() {

Iterable<Book> book=bookService.getAll();

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

book.forEach(single->listBook.add(single));

ModelAndView mv=new ModelAndView();

mv.setViewName("displaybook");

mv.addObject("books", listBook);

return mv;

}

@RequestMapping(value = "/cart", method = RequestMethod.GET)

public ModelAndView Cart(HttpServletRequest request,HttpServletResponse response) {

HttpSession session = request.getSession();

ShoppingCart cart = (ShoppingCart) request.getAttribute("shoppingcart");

if (cart == null) {

cart = new ShoppingCart();

session.setAttribute("shoppingcart", cart);

}

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

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

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

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

id = StringUtil.filterHtml(id);

name = StringUtil.filterHtml(name);

price = StringUtil.filterHtml(price);

quantity = StringUtil.filterHtml(quantity);

if (StringUtil.validateNull(quantity))

cart.addCartItem(new CartItem(Integer.parseInt(id), name, Float.parseFloat(price),1));

else

cart.addCartItem(new CartItem(Integer.parseInt(id), name, Float.parseFloat(price), Integer.parseInt(quantity)));

ArrayList<CartItem> ct = new ArrayList<CartItem>();

Iterator<CartItem> it = ct.iterator();

List<CartItem> listCart = new ArrayList<CartItem>();

ct.forEach(single->listCart.add(single));

ModelAndView mv=new ModelAndView();

mv.setViewName("getcart");

mv.addObject("cart", listCart);

return mv;

}

}

6、UserRepository.java：

package org.fkit.bookshop.repository;

import org.fkit.bookshop.bean.User;

import org.springframework.data.repository.CrudRepository;

public interface UserRepository extends CrudRepository<User, Integer>{

}

1. BookRepository.java：略
2. UserService.java：

package org.fkit.bookshop.service;

import java.util.Optional;

import javax.annotation.Resource;

import javax.transaction.Transactional;

import org.fkit.bookshop.bean.Book;

import org.fkit.bookshop.bean.User;

import org.fkit.bookshop.repository.UserRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.boot.web.servlet.server.Session;

import org.springframework.stereotype.Service;

@Service

public class UserService {

@Autowired

private UserRepository userRepository;

@Transactional

public User save(User User) {

return userRepository.save(User);

}

@Transactional

public void delete(int id) {

userRepository.deleteById(id);

}

public Iterable<User> getAll() {

return userRepository.findAll();

}

public User getById(Integer id) {

Optional<User> op = userRepository.findById(id);

return op.get();

}

}

1. BookService.java：略

**5.3 src/main/webapp/WEB-INF/jsp**

1、login.jsp：

<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*

pageEncoding=*"UTF-8"*%>

<html>

<head>

<title>欢迎登陆</title>

</head>

<h3><div align=*"center"*>欢迎登陆：</div></h3><br>

<body>

<form method=*"POST"* action=*"/bookshop/loginform"*>

<div align=*"center"*>账号：<input type=*"text"* name=*"username"* size=*"20"*><br></div>

<div align=*"center"*>密码：<input type=*"password"* name=*"password"* size=*"20"*><br></div>

<div align=*"center"*><p>

<input type=*"submit"* value=*"提交"* name=*"B1"*>

<input type=*"reset"* value=*"重置"* name=*"B2"*>

</p></div>

<br>

</form>

<div align=*"center"*>没有账号？<a href=*"/bookshop/registerform"*>立即注册</a></div>

</body>

</html>

1. manage.jsp：

<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*

pageEncoding=*"UTF-8"*%>

<html>

<head>

<title>管理员操作菜单</title>

</head>

<h3><div align=*"center"*>欢迎使用管理员权限：</div></h3>

<body>

<div align=*"center"*><font color=*"#FF0000"*>对书籍操作：</font></div><br>

<div align=*"center"*><a href=*"/bookshop/addform"*><input type=*"button"* value=*"添加书籍"*></a></div><br>

<div align=*"center"*><a href=*"/bookshop/deleteform1"*><input type=*"button"* value=*"移除书籍"*></a></div><br>

<div align=*"center"*><a href=*"/bookshop/getAllbook"*><input type=*"button"* value=*"查看库存"*></a></div><br>

<br>

<div align=*"center"*><font color=*"#FF0000"*>对用户操作：</font></div><br>

<div align=*"center"*><a href=*"/bookshop/deleteform2"*><input type=*"button"* value=*"注销用户"*></a></div><br>

<div align=*"center"*><a href=*"/bookshop/getAlluser"*><input type=*"button"* value=*"查看全部用户"*></a></div><br>

</body>

</html>

3、addbook.jsp:

<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*

pageEncoding=*"UTF-8"*%>

<html>

<head>

<title>添加书籍</title>

</head>

<h3><div align=*"center"*>添加新书到库存：</div></h3><br>

<body>

<form method=*"POST"* action=*"/bookshop/save"*>

<div align=*"center"*>书名：<input type=*"text"* name=*"bookname"* size=*"20"*></div><br>

<div align=*"center"*>价格：<input type=*"text"* name=*"price"* size=*"20"*></div><br>

<div align=*"center"*>数量：<input type=*"text"* name=*"quantity"* size=*"20"*></div><br>

<div align=*"center"*>

<p>

<input type=*"submit"* value=*"提交"* name=*"B1"*>

<input type=*"reset"* value=*"重置"* name=*"B2"*>

</p>

</div>

</form>

</body>

</html>

4、bookstore.jsp:

<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"* pageEncoding=*"UTF-8"*%>

<%@ taglib prefix=*"c"* uri=*"http://java.sun.com/jsp/jstl/core"* %>

<html>

<head>

<title>书店首页</title>

</head>

<body>

<div>

<div align=*"center"*><h3>以下为全部书籍信息：</h3></div>

</div>

<table align=*"center"*>

<thead>

<table border=*"1"* width=*"631"* align=*"center"*> <tr>

<td width=*"58"*><div align=*"center"*>id</div></td>

<td width=*"194"*><div align=*"center"*>书名</div></td>

<td width=*"47"*><div align=*"center"*>单价</div></td>

<td width=*"58"*><div align=*"center"*>库存</div></td>

<td width=*"58"*><div align=*"center"*>操作</div></td>

</tr>

</thead>

<tbody>

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

<tr>

<td width=*"174"*><div align=*"center"*>${book.id}</div></td>

<td width=*"174"*><div align=*"center"*>${book.bookname}</div></td>

<td width=*"174"*><div align=*"center"*>${book.price}</div></td>

<td width=*"174"*><div align=*"center"*>${book.quantity}</div></td>

<td width=*"174"*><div align=*"center"*><a href=*"./cart?id=*${book.id}*&bookname=*${book.bookname}*&price=*${book.price}*"* target=*"\_self"*>加购物车</a></td>

</tr>

</c:forEach>

</tbody>

</table>

</body>

</html>

5、pay.jsp：

<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"* pageEncoding=*"UTF-8"*%>

<html>

<head>

<title>支付</title>

</head>

<div align=*"center"*><h3>提示信息：</h3></div>

<body>

<div align=*"center"*>支付成功！您的当前账户余额为>${money}元。。。</div>

<div align=*"center"*>如有需要，可联系管理员（QQ：123456789）进行充值</div

<HR style="FILTER: *alpha(opacity*=*100,finishopacity*=*0,style*=*3)*" width=*"60%"* color=*#987cb9* SIZE=*3*><br>

<div align=*"center"*>您提交的个人信息为：</div>

<table border=*"1"* align=*"center"*>

<tr><td width=*"80"*><div align=*"center"*>收件人</div></td>

<td width=*"200"*><div align=*"center"*>${recipient}</div></td></tr>

<tr><td width=*"80"*><div align=*"center"*>手机号</div></td>

<td width=*"200"*><div align=*"center"*>${phone}</div></td></tr>

<tr><td width=*"80"*><div align=*"center"*>地址</div></td>

<td width=*"200"*><div align=*"center"*>${address}</div></td></tr>

</table>

</body>

</html>

1. **总结及心得体会**

本次作业设计开发的网上书店系统主要包括购书界面的设计和管理平台两个方面。在开发的工程中，采用了软件工程的设计思想，根据关系模式设计数据库表，用模块化设计方法将系统的各个功能模块层层分解，设计出系统的模块化结构，并根据系统的具体要求，详细地定义了系统各个模块的具体功能。本系统分为普通用户系统和管理员系统两大模块。根据功能要求，采用了开放式数据库连接技术对数据库进行访问。

通过对本次《JAVA EE》课程的学习，我掌握了web开发的基本技能，熟悉了servlet、JSP、spring等知识。其中前端包括对html、jsp的学习，后端包括servlet、javabean以及对数据库的连接、访问操作，除此之外，还熟悉了web项目的配置与搭建和部署到服务器上并运行的方法。《JAVA EE》课程极大地提高了我对web开发的认知和兴趣，在课程结束之后，我会深入学习关于spring等框架方面的知识，提升自己的开发能力！