

**课 程 设 计 报 告**

|  |  |
| --- | --- |
| 院 系： | 计算机学院 |
| 专业名称： | 软件工程 |
| 课程名称： | 软件设计基础实践 |
| 学生姓名： | 04222119 张子恒 |
| 指导教师： | 崔瀚文 |
| 完成时间： | 2023年12月 |

第一题： C类题-订单管理系统

1、问题的需求分析

**题目要求：**订单管理的几种操作功能，包括订单的录入、修改、查询、读写文件、按产品交货日期排序等功能。以链表结构来表示订单库存的模型，链表的数据域包括订单编号、商品名称、订购数量、商品单价、卖家、买家、订单日期、总价。程序功能包括：建表、录入、查询以及文件读写等。

**问题分析：**订单管理系统的实现，主要包括以下功能

1. 用户登录
2. 订单管理
3. 供应商管理
4. 用户管理
5. 密码修改
6. 退出系统

2、设计思路

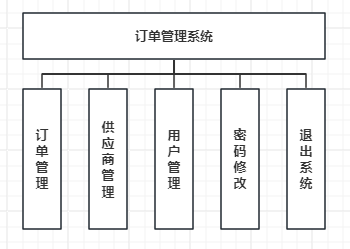
部分功能实现思路：

1. 创建一个Controller类，如LoginController，并定义一个处理登录请求的方法，如login()。在这个方法中，使用Spring的依赖注入功能将UserService注入到控制器中，然后调用UserService的login()方法来验证用户的用户名和密码。如果验证成功，可以将用户信息存储在Session中，并重定向到其他页面；如果验证失败，可以返回错误信息给用户。
2. 创建一个OrderController类，并定义处理订单相关请求的方法，如listOrders()、createOrder()、updateOrder()和deleteOrder()。在这些方法中，您可以使用Spring的依赖注入功能将OrderService注入到控制器中，然后调用OrderService的相关方法来执行相应的操作。
3. 创建一个SupplierController类，并定义处理供应商相关请求的方法，如listSuppliers()、createSupplier()、updateSupplier()和deleteSupplier()。在这些方法中，您可以使用Spring的依赖注入功能将SupplierService注入到控制器中，然后调用SupplierService的相关方法来执行相应的操作。
4. 创建一个UserController类，并定义处理用户相关请求的方法，如listUsers()、createUser()、updateUser()和deleteUser()。在这些方法中，您可以使用Spring的依赖注入功能将UserService注入到控制器中，然后调用UserService的相关方法来执行相应的操作。密码修改：changePassword()。在这个方法中，您可以使用Spring的依赖注入功能将UserService注入到控制器中，然后调用UserService的changePassword()方法来更改用户的密码。
5. 创建一个LogoutController类，并定义一个处理退出请求的方法，如logout()。在这个方法中，您可以清除Session中的用户信息，并重定向到登录页面。

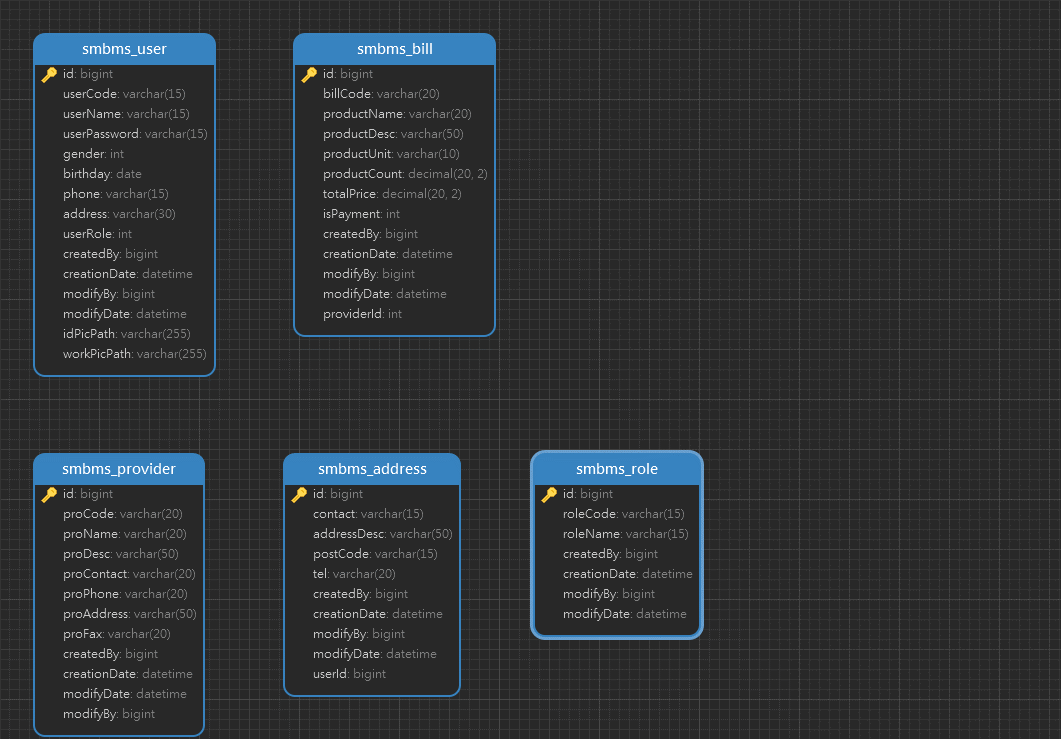
功能分析：

* 用户登录：允许用户输入他们的用户名和密码以访问系统。这是他们首次使用系统时必须执行的操作。
* 订单管理：允许用户查看、创建、修改和删除他们的订单。这包括查看订单状态、添加新的商品到订单中、更改订单的总金额等。
* 供应商管理：允许用户查看、创建、修改和删除他们的供应商信息。这可能包括查看供应商的联系信息、添加新的供应商、更改供应商的付款条款等。
* 用户管理：允许管理员（如系统管理员或经理）查看、创建、修改和删除其他用户的账户。这可能包括查看用户的个人信息、更改用户的权限级别、删除不再需要的用户账户等。
* 密码修改：允许用户更改他们的密码。这是为了保护他们的账户安全，防止未经授权的访问。
* 退出系统：允许用户结束当前的会话并退出系统。

订单管理HIPO图



数据库ER图



3、源码（详细设计）

pom.xml

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>    <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>com.kuang</groupId>   <artifactId>smbss</artifactId>   <version>1.0-SNAPSHOT</version>   <packaging>war</packaging>     <name>smbss Maven Webapp</name>   <!-- FIXME change it to the project's website -->   <url>http://www.example.com</url>     <properties>   <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>   <maven.compiler.source>1.7</maven.compiler.source>   <maven.compiler.target>1.7</maven.compiler.target>   </properties>     <dependencies>   <dependency>   <groupId>junit</groupId>   <artifactId>junit</artifactId>   <version>4.11</version>   </dependency>     <dependency>   <groupId>javax.servlet</groupId>   <artifactId>servlet-api</artifactId>   <version>2.5</version>   </dependency>     <dependency>   <groupId>javax.servlet.jsp</groupId>   <artifactId>javax.servlet.jsp-api</artifactId>   <version>2.3.3</version>   </dependency>     <dependency>   <groupId>javax.servlet.jsp.jstl</groupId>   <artifactId>jstl-api</artifactId>   <version>1.2</version>   </dependency>     <dependency>   <groupId>taglibs</groupId>   <artifactId>standard</artifactId>   <version>1.1.2</version>   </dependency>     <dependency>   <groupId>mysql</groupId>   <artifactId>mysql-connector-java</artifactId>   <version>5.1.49</version>   </dependency>     <dependency>   <groupId>com.alibaba</groupId>   <artifactId>fastjson</artifactId>   <version>1.2.68</version>   </dependency>   </dependencies>      </project> |

**以账单业务为例**

Dao/bill/BillDao.java

|  |
| --- |
| package com.zh.dao.bill;    import com.zh.pojo.Bill;    import java.sql.Connection;  import java.util.List;    public interface BillDao {   /\*\*   \* 增加订单   \*   \* @param connection   \* @param bill   \* @return   \* @throws Exception   \*/   public int add(Connection connection, Bill bill) throws Exception;     /\*\*   \* 通过查询条件获取供应商列表-模糊查询-getBillList   \*   \* @param connection   \* @param bill   \* @return   \* @throws Exception   \*/   public List<Bill> getBillList(Connection connection, Bill bill) throws Exception;     /\*\*   \* 通过delId删除Bill   \*   \* @param connection   \* @param delId   \* @return   \* @throws Exception   \*/   public int deleteBillById(Connection connection, String delId) throws Exception;       /\*\*   \* 通过billId获取Bill   \*   \* @param connection   \* @param id   \* @return   \* @throws Exception   \*/   public Bill getBillById(Connection connection, String id) throws Exception;     /\*\*   \* 修改订单信息   \*   \* @param connection   \* @param bill   \* @return   \* @throws Exception   \*/   public int modify(Connection connection, Bill bill) throws Exception;     /\*\*   \* 根据供应商ID查询订单数量   \*   \* @param connection   \* @param providerId   \* @return   \* @throws Exception   \*/   public int getBillCountByProviderId(Connection connection, String providerId) throws Exception;    } |

Filter/SysFilter.java

|  |
| --- |
| package com.zh.filter;    import com.zh.pojo.User;  import com.zh.util.Constants;    import javax.servlet.\*;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import java.io.IOException;    public class SysFilter implements Filter {     @Override   public void init(FilterConfig filterConfig) throws ServletException {     }     @Override   public void doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain filterChain) throws IOException, ServletException {   HttpServletRequest request = (HttpServletRequest)servletRequest;   HttpServletResponse response = (HttpServletResponse)servletResponse;     //过滤器，从session中获取用户   User user = (User)request.getSession().getAttribute(Constants.USER\_SESSION);   if(user == null) {//已经被移除或者注销了，或者未登录   response.sendRedirect(request.getContextPath()+"/error.jsp");   }else {   filterChain.doFilter(servletRequest, servletResponse);   }   }     @Override   public void destroy() {     }  } |

Pojo/Bill.java

|  |
| --- |
| package com.zh.pojo;    import java.math.BigDecimal;  import java.util.Date;    public class Bill {   private Integer id; //id   private String billCode; //账单编码   private String productName; //商品名称   private String productDesc; //商品描述   private String productUnit; //商品单位   private BigDecimal productCount; //商品数量   private BigDecimal totalPrice; //总金额   private Integer isPayment; //是否支付   private Integer providerId; //供应商ID   private Integer createdBy; //创建者   private Date creationDate; //创建时间   private Integer modifyBy; //更新者   private Date modifyDate;//更新时间     private String providerName;//供应商名称       public String getProviderName() {   return providerName;   }     public void setProviderName(String providerName) {   this.providerName = providerName;   }     public Integer getId() {   return id;   }     public void setId(Integer id) {   this.id = id;   }     public String getBillCode() {   return billCode;   }     public void setBillCode(String billCode) {   this.billCode = billCode;   }     public String getProductName() {   return productName;   }     public void setProductName(String productName) {   this.productName = productName;   }     public String getProductDesc() {   return productDesc;   }     public void setProductDesc(String productDesc) {   this.productDesc = productDesc;   }     public String getProductUnit() {   return productUnit;   }     public void setProductUnit(String productUnit) {   this.productUnit = productUnit;   }     public BigDecimal getProductCount() {   return productCount;   }     public void setProductCount(BigDecimal productCount) {   this.productCount = productCount;   }     public BigDecimal getTotalPrice() {   return totalPrice;   }     public void setTotalPrice(BigDecimal totalPrice) {   this.totalPrice = totalPrice;   }     public Integer getIsPayment() {   return isPayment;   }     public void setIsPayment(Integer isPayment) {   this.isPayment = isPayment;   }     public Integer getProviderId() {   return providerId;   }     public void setProviderId(Integer providerId) {   this.providerId = providerId;   }     public Integer getCreatedBy() {   return createdBy;   }     public void setCreatedBy(Integer createdBy) {   this.createdBy = createdBy;   }     public Date getCreationDate() {   return creationDate;   }     public void setCreationDate(Date creationDate) {   this.creationDate = creationDate;   }     public Integer getModifyBy() {   return modifyBy;   }     public void setModifyBy(Integer modifyBy) {   this.modifyBy = modifyBy;   }     public Date getModifyDate() {   return modifyDate;   }     public void setModifyDate(Date modifyDate) {   this.modifyDate = modifyDate;   }      } |

service/bill/BillService.java

|  |
| --- |
| package com.zh.service.bill;    import com.zh.pojo.Bill;    import java.util.List;    public interface BillService {   /\*\*   \* 增加订单   \*   \* @param bill   \* @return   \*/   public boolean add(Bill bill);     /\*\*   \* 通过条件获取订单列表-模糊查询-billList   \*   \* @param bill   \* @return   \*/   public List<Bill> getBillList(Bill bill);     /\*\*   \* 通过billId删除Bill   \*   \* @param delId   \* @return   \*/   public boolean deleteBillById(String delId);       /\*\*   \* 通过billId获取Bill   \*   \* @param id   \* @return   \*/   public Bill getBillById(String id);     /\*\*   \* 修改订单信息   \*   \* @param bill   \* @return   \*/   public boolean modify(Bill bill);    } |

servlet/bill/BillServlet.java

|  |
| --- |
| package com.zh.servlet.bill;    import com.alibaba.fastjson.JSONArray;  import com.mysql.jdbc.StringUtils;  import com.zh.pojo.Bill;  import com.zh.pojo.Provider;  import com.zh.pojo.User;  import com.zh.service.Provider.ProviderService;  import com.zh.service.Provider.ProviderServiceImpl;  import com.zh.service.bill.BillService;  import com.zh.service.bill.BillServiceImpl;  import com.zh.util.Constants;    import javax.servlet.ServletException;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import java.io.IOException;  import java.io.PrintWriter;  import java.math.BigDecimal;  import java.util.ArrayList;  import java.util.Date;  import java.util.HashMap;  import java.util.List;    public class BillServlet extends HttpServlet {   @Override   protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {   String method = req.getParameter("method");   if(method != null && method.equals("query")){   this.query(req,resp);   }else if(method != null && method.equals("add")){   this.add(req,resp);   }else if(method != null && method.equals("view")){   this.getBillById(req,resp,"billview.jsp");   }else if(method != null && method.equals("modify")){   this.getBillById(req,resp,"billmodify.jsp");   }else if(method != null && method.equals("modifysave")){   this.modify(req,resp);   }else if(method != null && method.equals("delbill")){   this.delBill(req,resp);   }else if(method != null && method.equals("getproviderlist")){   this.getProviderlist(req,resp);   }   }       @Override   protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {   doGet(req, resp);   }     @Override   public void destroy() {   super.destroy();   }     private void query(HttpServletRequest request, HttpServletResponse response)   throws ServletException, IOException {     List<Provider> providerList = new ArrayList<Provider>();   ProviderService providerService = new ProviderServiceImpl();   providerList = providerService.getProviderList("","");   request.setAttribute("providerList", providerList);     String queryProductName = request.getParameter("queryProductName");   String queryProviderId = request.getParameter("queryProviderId");   String queryIsPayment = request.getParameter("queryIsPayment");     if(StringUtils.isNullOrEmpty(queryProductName)){   queryProductName = "";   }     List<Bill> billList = new ArrayList<Bill>();   BillService billService = new BillServiceImpl();   Bill bill = new Bill();   if(StringUtils.isNullOrEmpty(queryIsPayment)){   bill.setIsPayment(0);   }else{   bill.setIsPayment(Integer.parseInt(queryIsPayment));   }     if(StringUtils.isNullOrEmpty(queryProviderId)){   bill.setProviderId(0);   }else{   bill.setProviderId(Integer.parseInt(queryProviderId));   }   bill.setProductName(queryProductName);   billList = billService.getBillList(bill);   request.setAttribute("billList", billList);   request.setAttribute("queryProductName", queryProductName);   request.setAttribute("queryProviderId", queryProviderId);   request.setAttribute("queryIsPayment", queryIsPayment);   request.getRequestDispatcher("billlist.jsp").forward(request, response);     }     private void add(HttpServletRequest request, HttpServletResponse response)   throws ServletException, IOException {   String billCode = request.getParameter("billCode");   String productName = request.getParameter("productName");   String productDesc = request.getParameter("productDesc");   String productUnit = request.getParameter("productUnit");     String productCount = request.getParameter("productCount");   String totalPrice = request.getParameter("totalPrice");   String providerId = request.getParameter("providerId");   String isPayment = request.getParameter("isPayment");     Bill bill = new Bill();   bill.setBillCode(billCode);   bill.setProductName(productName);   bill.setProductDesc(productDesc);   bill.setProductUnit(productUnit);   bill.setProductCount(new BigDecimal(productCount).setScale(2,BigDecimal.ROUND\_DOWN));   bill.setIsPayment(Integer.parseInt(isPayment));   bill.setTotalPrice(new BigDecimal(totalPrice).setScale(2,BigDecimal.ROUND\_DOWN));   bill.setProviderId(Integer.parseInt(providerId));   bill.setCreatedBy(((User)request.getSession().getAttribute(Constants.USER\_SESSION)).getId());   bill.setCreationDate(new Date());   boolean flag = false;   BillService billService = new BillServiceImpl();   flag = billService.add(bill);   System.out.println("add flag -- > " + flag);   if(flag){   response.sendRedirect(request.getContextPath()+"/jsp/bill.do?method=query");   }else{   request.getRequestDispatcher("billadd.jsp").forward(request, response);   }   }     private void getBillById(HttpServletRequest request, HttpServletResponse response,String url)   throws ServletException, IOException {   String id = request.getParameter("billid");   if(!StringUtils.isNullOrEmpty(id)){   BillService billService = new BillServiceImpl();   Bill bill = null;   bill = billService.getBillById(id);   request.setAttribute("bill", bill);   request.getRequestDispatcher(url).forward(request, response);   }   }     private void modify(HttpServletRequest request, HttpServletResponse response)   throws ServletException, IOException {   System.out.println("modify===============");   String id = request.getParameter("id");   String productName = request.getParameter("productName");   String productDesc = request.getParameter("productDesc");   String productUnit = request.getParameter("productUnit");   String productCount = request.getParameter("productCount");   String totalPrice = request.getParameter("totalPrice");   String providerId = request.getParameter("providerId");   String isPayment = request.getParameter("isPayment");     Bill bill = new Bill();   bill.setId(Integer.valueOf(id));   bill.setProductName(productName);   bill.setProductDesc(productDesc);   bill.setProductUnit(productUnit);   bill.setProductCount(new BigDecimal(productCount).setScale(2,BigDecimal.ROUND\_DOWN));   bill.setIsPayment(Integer.parseInt(isPayment));   bill.setTotalPrice(new BigDecimal(totalPrice).setScale(2,BigDecimal.ROUND\_DOWN));   bill.setProviderId(Integer.parseInt(providerId));     bill.setModifyBy(((User)request.getSession().getAttribute(Constants.USER\_SESSION)).getId());   bill.setModifyDate(new Date());   boolean flag = false;   BillService billService = new BillServiceImpl();   flag = billService.modify(bill);   if(flag){   response.sendRedirect(request.getContextPath()+"/jsp/bill.do?method=query");   }else{   request.getRequestDispatcher("billmodify.jsp").forward(request, response);   }   }     private void delBill(HttpServletRequest request, HttpServletResponse response)   throws ServletException, IOException {   String id = request.getParameter("billid");   HashMap<String, String> resultMap = new HashMap<String, String>();   if(!StringUtils.isNullOrEmpty(id)){   BillService billService = new BillServiceImpl();   boolean flag = billService.deleteBillById(id);   if(flag){//删除成功   resultMap.put("delResult", "true");   }else{//删除失败   resultMap.put("delResult", "false");   }   }else{   resultMap.put("delResult", "notexit");   }   //把resultMap转换成json对象输出   response.setContentType("application/json");   PrintWriter outPrintWriter = response.getWriter();   outPrintWriter.write(JSONArray.toJSONString(resultMap));   outPrintWriter.flush();   outPrintWriter.close();   }     private void getProviderlist(HttpServletRequest request, HttpServletResponse response)   throws ServletException, IOException {     System.out.println("getproviderlist ========================= ");     List<Provider> providerList = new ArrayList<Provider>();   ProviderService providerService = new ProviderServiceImpl();   providerList = providerService.getProviderList("","");   //把providerList转换成json对象输出   response.setContentType("application/json");   PrintWriter outPrintWriter = response.getWriter();   outPrintWriter.write(JSONArray.toJSONString(providerList));   outPrintWriter.flush();   outPrintWriter.close();   }  } |

4、运行、测试与分析

登录



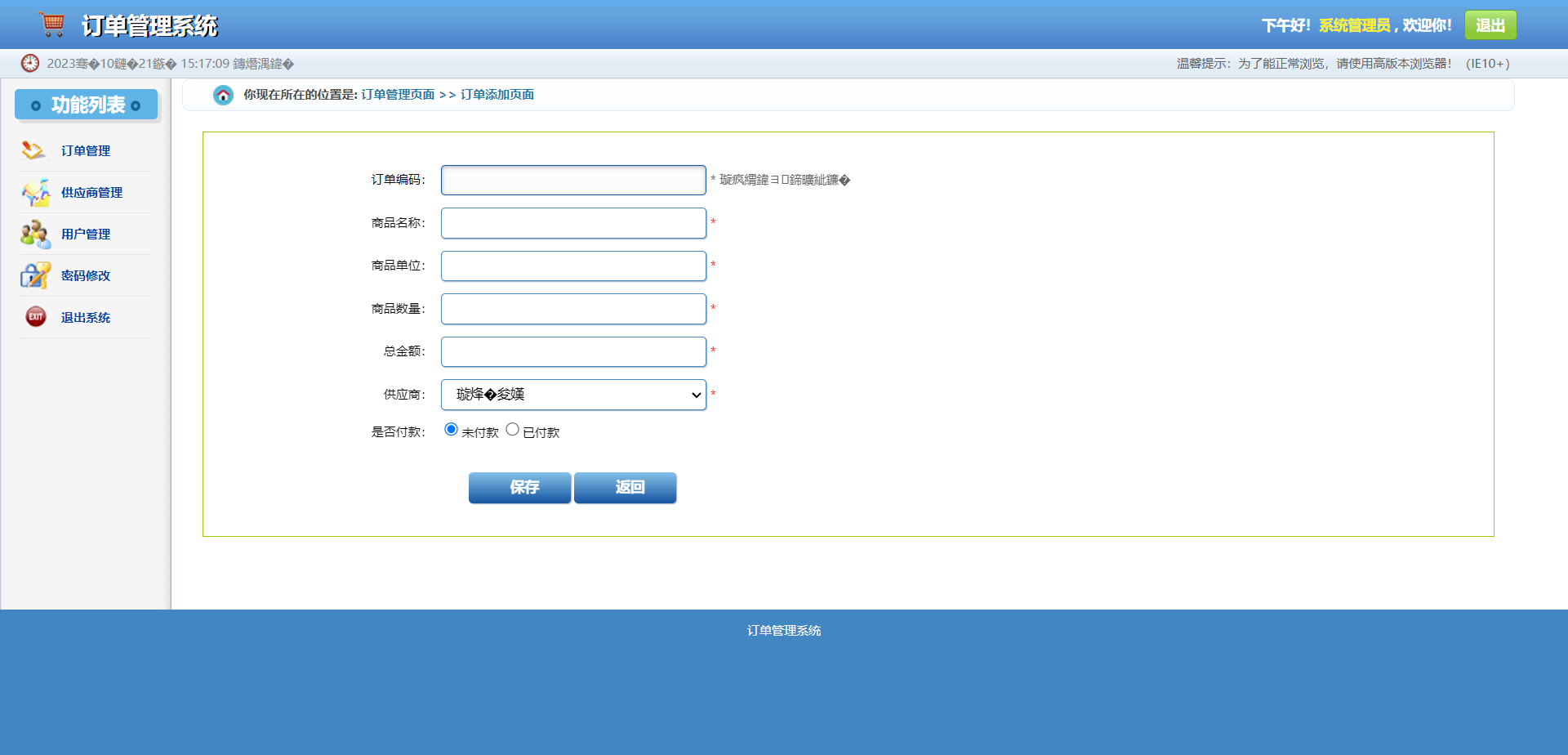
首页



订单管理



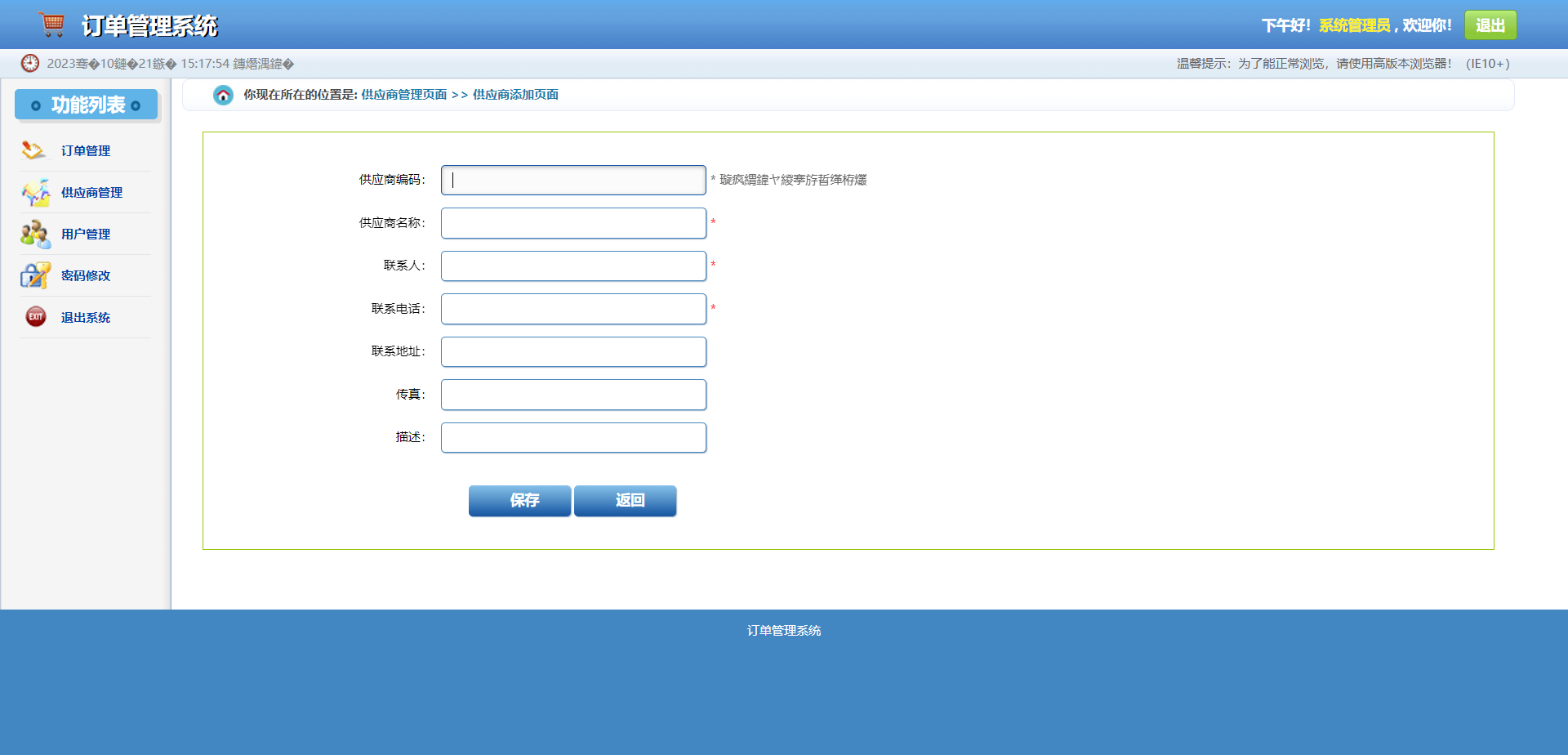
添加订单



供应商管理



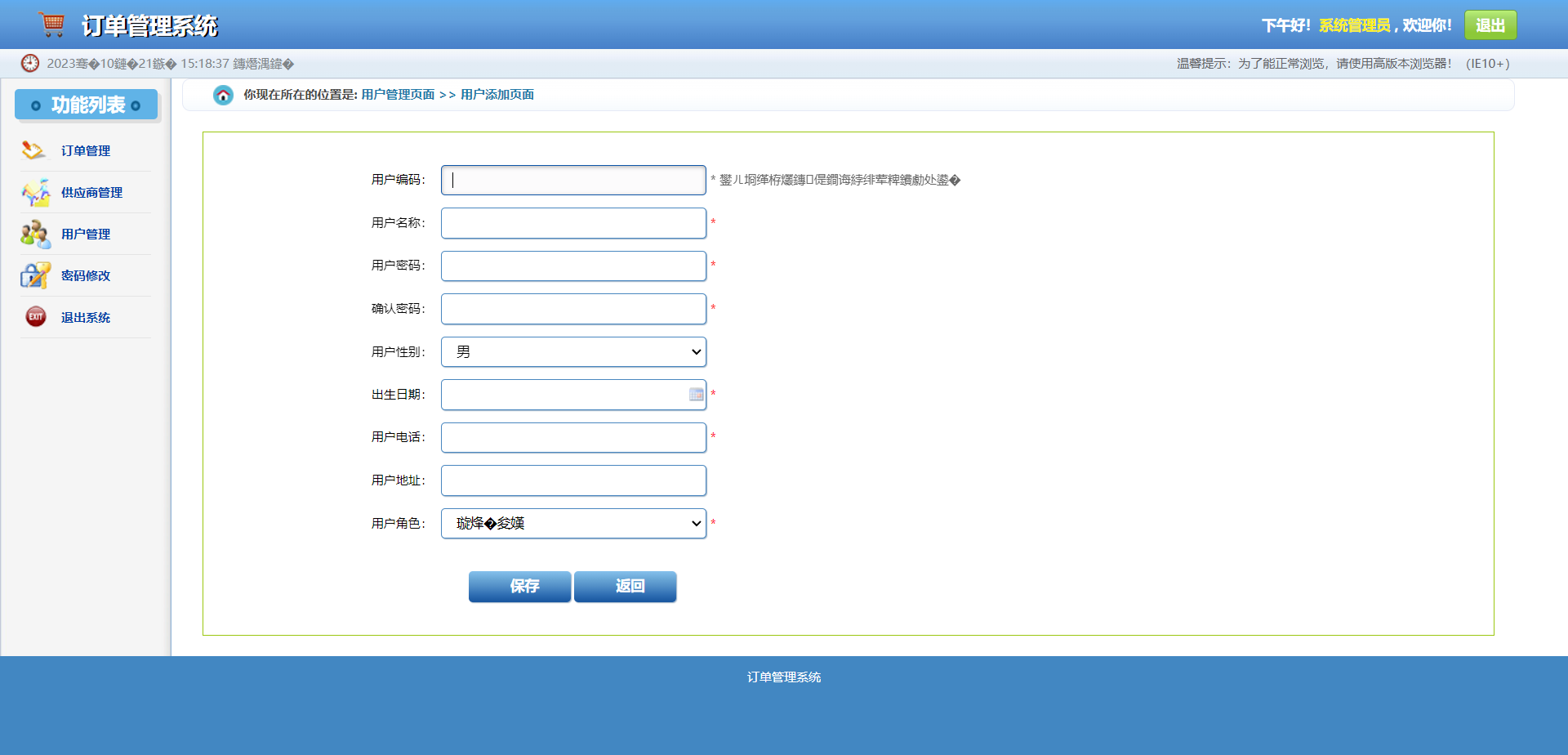
添加供应商



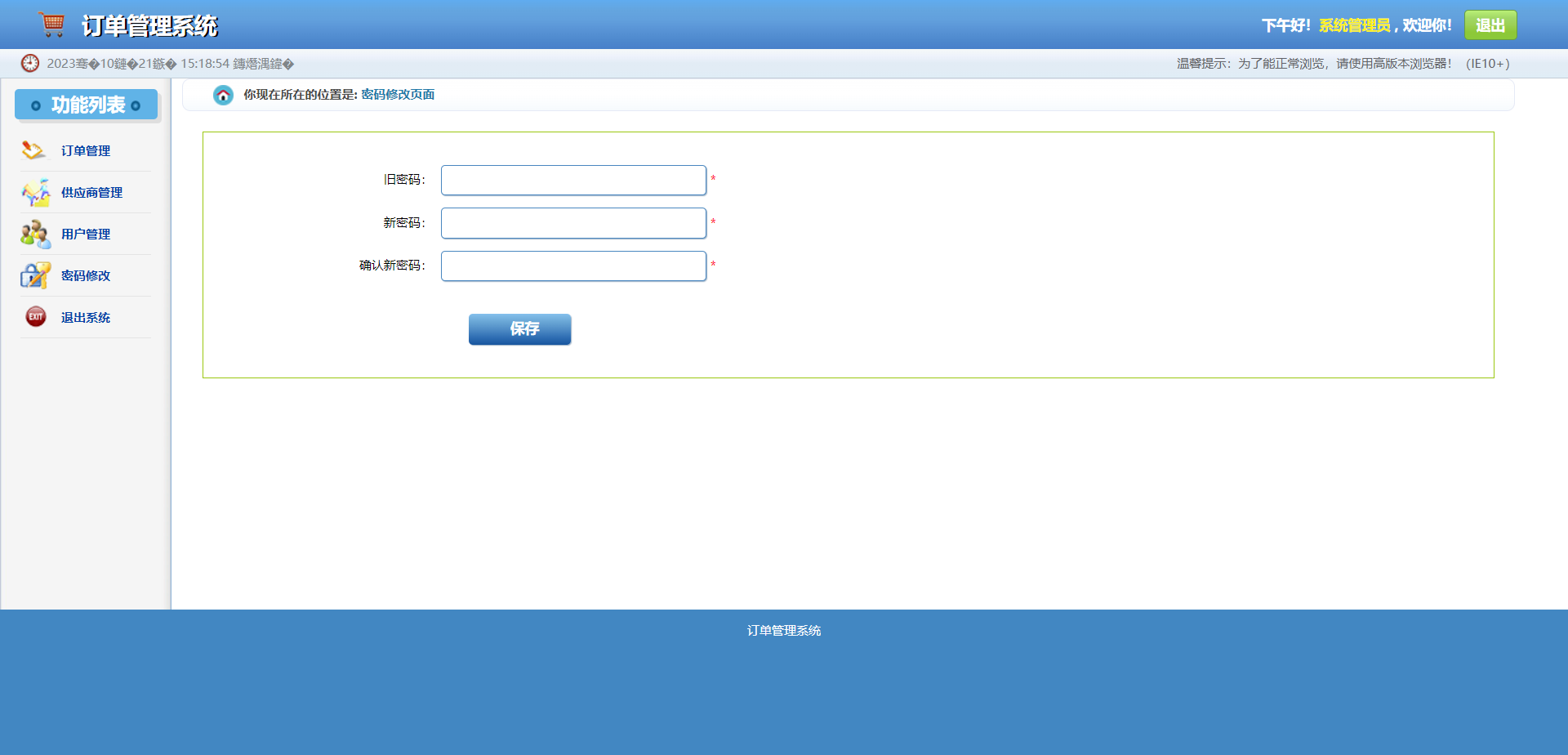
用户管理



添加用户



密码修改



第二题： C类题-图书信息管理系统

1. 问题的需求分析

**题目要求：**

1、 用数据库建立 1 或 2 个图书信息表。（不限使用哪种数据库）

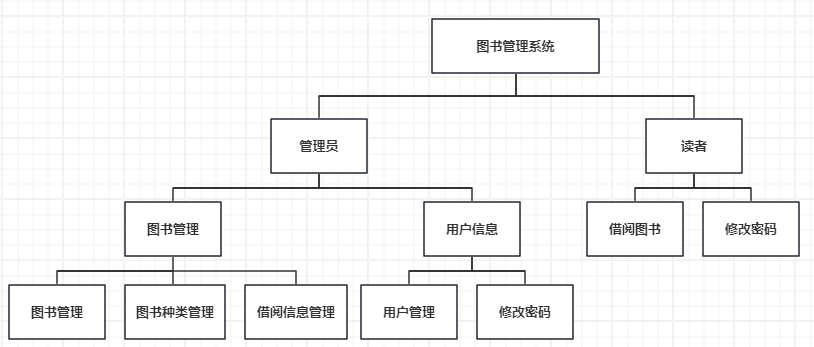
2、 能连接数据库并实现对图书的查询、增、删、改等功能。

**问题分析**：图书信息管理系统是一个用于管理图书馆内图书信息的系统。它的主要功能包括添加、删除、修改和查询图书信息，以及管理读者信息等。

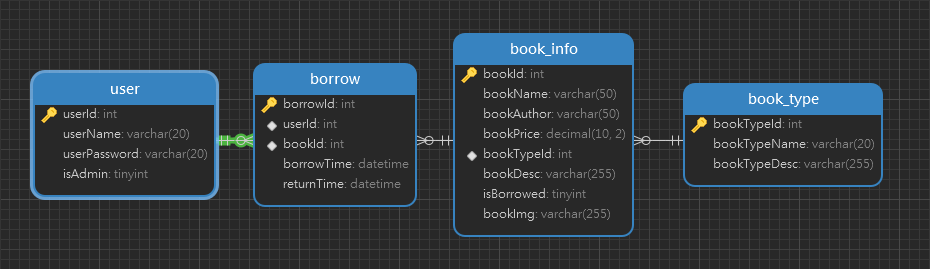
2、设计思路

1. 数据库设计：设计一个适合存储图书和读者信息的数据库，包括表的设计和关系的建立。可以使用关系型数据库如MySQL进行设计。
2. 用户界面设计：使用户能够方便地进行图书和读者信息的管理。使用前端框架技术Vue并结合图形界面库Element UI进行设计。
3. 功能模块划分：将系统的功能划分为不同的模块，如添加图书模块、删除图书模块、修改图书模块、查询图书模块和借阅管理模块等。每个模块负责处理特定的功能。
4. 数据库操作：根据用户的操作请求，执行相应的数据库操作，如插入、删除、更新和查询等。可以使用SQL语句。

图书管理系统HPIO图



数据库ER图



3、源码（详细设计）

application.properties

# 端口号  
server.port=8092  
# 上下文根  
server.servlet.context-path=/BookManager  
  
# 配置数据库  
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver  
spring.datasource.url=jdbc:mysql://localhost:3306/book\_manager?useAffectedRows=true  
spring.datasource.username=root  
spring.datasource.password=root  
  
# 配置Redis  
spring.redis.host=192.168.88.130  
spring.redis.port=6379  
spring.redis.password=123456  
  
# 上传文件大小限制  
spring.servlet.multipart.max-file-size=10MB  
spring.servlet.multipart.max-request-size=100MB  
  
# 关闭multipart，该添加会将springboot的默认的multipart取消，避免上传文件时报错  
spring.servlet.multipart.enabled=false

**以图书信息业务为例**

**package com.zh.model;**

public class BookInfo {  
  
 private Integer bookid;  
 //书名  
 private String bookname;  
 //作者  
 private String bookauthor;  
 //价格  
 private BigDecimal bookprice;  
 //类型id  
 private Integer booktypeid;  
 //类型名称  
 private String booktypename;  
 //简介  
 private String bookdesc;  
 //借书情况  
 private Byte isborrowed;  
 //图书照片  
 private String bookimg;  
  
 public Integer getBookid() {  
 return bookid;  
 }  
  
 public void setBookid(Integer bookid) {  
 this.bookid = bookid;  
 }  
  
 public String getBookname() {  
 return bookname;  
 }  
  
 public void setBookname(String bookname) {  
 this.bookname = bookname;  
 }  
  
 public String getBookauthor() {  
 return bookauthor;  
 }  
  
 public void setBookauthor(String bookauthor) {  
 this.bookauthor = bookauthor;  
 }  
  
 public BigDecimal getBookprice() {  
 return bookprice;  
 }  
  
 public void setBookprice(BigDecimal bookprice) {  
 this.bookprice = bookprice;  
 }  
  
 public Integer getBooktypeid() {  
 return booktypeid;  
 }  
  
 public void setBooktypeid(Integer booktypeid) {  
 this.booktypeid = booktypeid;  
 }  
  
 public String getBooktypename() {  
 return booktypename;  
 }  
  
 public void setBooktypename(String booktypename) {  
 this.booktypename = booktypename;  
 }  
  
 public String getBookdesc() {  
 return bookdesc;  
 }  
  
 public void setBookdesc(String bookdesc) {  
 this.bookdesc = bookdesc;  
 }  
  
 public Byte getIsborrowed() {  
 return isborrowed;  
 }  
  
 public void setIsborrowed(Byte isborrowed) {  
 this.isborrowed = isborrowed;  
 }  
  
 public String getBookimg() {  
 return bookimg;  
 }  
  
 public void setBookimg(String bookimg) {  
 this.bookimg = bookimg;  
 }  
}

**package com.zh.controller;**

@RestController  
@RequestMapping(value = "/bookInfo")  
public class BookInfoController {  
  
 @Autowired  
 BookInfoService bookInfoService;  
  
 */\*\*  
 \* 获取图书数量  
 \** ***@return*** *\*/* @GetMapping(value = "/getCount")  
 public Integer getCount(){  
 return bookInfoService.getCount();  
 }  
  
 */\*\*  
 \* 查询所有图书信息  
 \** ***@return*** *\*/* @GetMapping(value = "/queryBookInfos")  
 public List<BookInfo> queryBookInfos(){  
 return bookInfoService.queryBookInfos();  
 }  
  
 */\*\*  
 \* 分页搜索查询图书信息  
 \* params: {page, limit, bookname, bookauthor, booktypeid}  
 \** ***@param*** *params  
 \** ***@return*** *\*/* @GetMapping(value = "/queryBookInfosByPage")  
 public Map<String, Object> queryBookInfosByPage(@RequestParam Map<String, Object> params){  
 MyUtils.*parsePageParams*(params);  
 int count = bookInfoService.getSearchCount(params); // 获得总数  
 List<BookInfo> bookInfos = bookInfoService.searchBookInfosByPage(params); // 分页查询  
 return MyResult.*getListResultMap*(0, "success", count, bookInfos);  
 }  
  
 */\*\*  
 \* 添加图书信息  
 \** ***@param*** *bookInfo  
 \** ***@return*** *\*/* @PostMapping(value = "/addBookInfo")  
 public Integer addBookInfo(@RequestBody BookInfo bookInfo){  
 return bookInfoService.addBookInfo(bookInfo);  
 }  
  
 */\*\*  
 \* 删除图书信息  
 \** ***@param*** *bookInfo  
 \** ***@return*** *\*/* @DeleteMapping(value = "/deleteBookInfo")  
 public Integer deleteBookInfo(@RequestBody BookInfo bookInfo){  
 return bookInfoService.deleteBookInfo(bookInfo);  
 }  
  
 */\*\*  
 \* 删除一些图书信息  
 \** ***@param*** *bookInfos  
 \** ***@return*** *\*/* @DeleteMapping(value = "/deleteBookInfos")  
 public Integer deleteBookInfos(@RequestBody List<BookInfo> bookInfos){  
 return bookInfoService.deleteBookInfos(bookInfos);  
 }  
  
 */\*\*  
 \* 更新一些图书信息  
 \** ***@param*** *bookInfo  
 \** ***@return*** *\*/* @PutMapping(value = "/updateBookInfo")  
 public Integer updateBookInfo(@RequestBody BookInfo bookInfo){  
 return bookInfoService.updateBookInfo(bookInfo);  
 }  
}

**package com.zh.service;**

public interface BookInfoService {  
  
 */\*\*  
 \* 获取图书数量  
 \*  
 \** ***@return*** *\*/* Integer getCount();  
  
 */\*\*  
 \* 查询所有图书信息  
 \*  
 \** ***@return*** *\*/* List<BookInfo> queryBookInfos();  
  
 */\*\*  
 \* 根据id查询图书信息  
 \** ***@param*** *bookid  
 \** ***@return*** *\*/* BookInfo queryBookInfoById(Integer bookid);  
  
 */\*\*  
 \* 统计搜索总数  
 \** ***@param*** *params  
 \** ***@return*** *\*/* Integer getSearchCount(Map<String, Object> params);  
  
 */\*\*  
 \* 分页搜索查询图书信息  
 \*  
 \** ***@param*** *params  
 \** ***@return*** *\*/* List<BookInfo> searchBookInfosByPage(Map<String, Object> params);  
  
 */\*\*  
 \* 添加图书信息  
 \*  
 \** ***@param*** *bookInfo  
 \** ***@return*** *\*/* Integer addBookInfo(BookInfo bookInfo);  
  
 */\*\*  
 \* 删除图书信息  
 \*  
 \** ***@param*** *bookInfo  
 \** ***@return*** *\*/* Integer deleteBookInfo(BookInfo bookInfo);  
  
 */\*\*  
 \* 删除一些图书信息  
 \*  
 \** ***@param*** *bookInfos  
 \** ***@return*** *\*/* Integer deleteBookInfos(List<BookInfo> bookInfos);  
  
 */\*\*  
 \* 更新一些图书信息  
 \*  
 \** ***@param*** *bookInfo  
 \** ***@return*** *\*/* Integer updateBookInfo(BookInfo bookInfo);  
}

**package com.zh.service.impl;**

@Service  
public class BookInfoServiceImpl implements BookInfoService {  
  
 @Resource  
 private BookInfoMapper bookInfoMapper;  
  
 */\*\*  
 \* 获取图书数量  
 \*  
 \** ***@return*** *\*/* @Override  
 public Integer getCount() {  
 return bookInfoMapper.selectCount();  
 }  
  
 */\*\*  
 \* 查询所有图书信息  
 \*  
 \** ***@return*** *\*/* @Override  
 public List<BookInfo> queryBookInfos() {  
 return bookInfoMapper.selectAll();  
 }  
  
 */\*\*  
 \* 根据id查询图书信息  
 \*  
 \** ***@param*** *bookid  
 \** ***@return*** *\*/* @Override  
 public BookInfo queryBookInfoById(Integer bookid) {  
 return bookInfoMapper.selectByPrimaryKey(bookid);  
 }  
  
 */\*\*  
 \* 统计搜索总数  
 \*  
 \** ***@param*** *params  
 \** ***@return*** *\*/* @Override  
 public Integer getSearchCount(Map<String, Object> params) {  
 return bookInfoMapper.selectCountBySearch(params);  
 }  
  
 */\*\*  
 \* 分页搜索查询图书信息  
 \*  
 \** ***@param*** *params  
 \** ***@return*** *\*/* @Override  
 public List<BookInfo> searchBookInfosByPage(Map<String, Object> params) {  
 return bookInfoMapper.selectBySearch(params);  
 }  
  
 */\*\*  
 \* 添加图书信息  
 \*  
 \** ***@param*** *bookInfo  
 \** ***@return*** *\*/* @Override  
 public Integer addBookInfo(BookInfo bookInfo) {  
 return bookInfoMapper.insertSelective(bookInfo);  
 }  
  
 */\*\*  
 \* 删除图书信息  
 \*  
 \** ***@param*** *bookInfo  
 \** ***@return*** *\*/* @Override  
 public Integer deleteBookInfo(BookInfo bookInfo) {  
 int count = 0;  
 try {  
 count = bookInfoMapper.deleteByPrimaryKey(bookInfo.getBookid());  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 return count;  
 }  
  
 */\*\*  
 \* 删除一些图书信息  
 \*  
 \** ***@param*** *bookInfos  
 \** ***@return*** *\*/* @Override  
 public Integer deleteBookInfos(List<BookInfo> bookInfos) {  
 int count = 0;  
 for (BookInfo bookInfo : bookInfos) {  
 count += deleteBookInfo(bookInfo);  
 }  
 return count;  
 }  
  
 */\*\*  
 \* 更新一些图书信息  
 \*  
 \** ***@param*** *bookInfo  
 \** ***@return*** *\*/* @Override  
 public Integer updateBookInfo(BookInfo bookInfo) {  
 return bookInfoMapper.updateByPrimaryKeySelective(bookInfo);  
 }  
  
}

**package com.zh.mapper;**

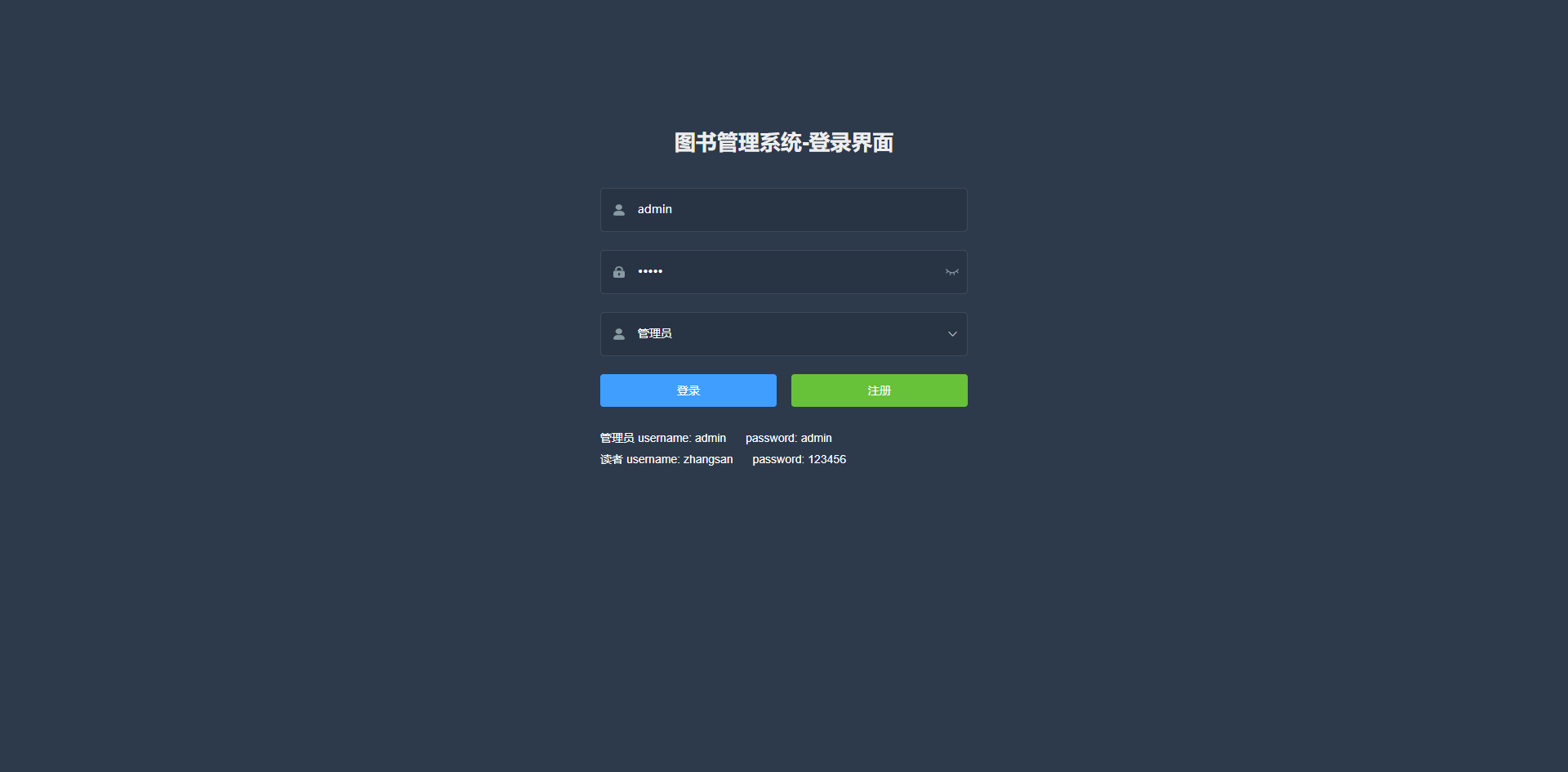
import com.zh.model.BookInfo;  
import org.apache.ibatis.annotations.Param;  
  
import java.util.List;  
import java.util.Map;  
  
public interface BookInfoMapper {  
  
 */\*\*  
 \* 获取图书数量  
 \** ***@return*** *\*/* Integer selectCount();  
  
 */\*\*  
 \* 查询所有图书信息  
 \** ***@return*** *\*/* List<BookInfo> selectAll();  
  
 */\*\*  
 \* 根据id查询图书  
 \** ***@param*** *bookid  
 \** ***@return*** *\*/* BookInfo selectByPrimaryKey(Integer bookid);  
  
 */\*\*  
 \* 统计搜索总数  
 \** ***@param*** *searchParam  
 \** ***@return*** *\*/* int selectCountBySearch(Map<String, Object> searchParam);  
  
 */\*\*  
 \*搜索查询图书信息  
 \** ***@param*** *searchParam  
 \** ***@return*** *\*/* List<BookInfo> selectBySearch(Map<String, Object> searchParam);  
  
 */\*\*  
 \* 根据id删除图书  
 \** ***@param*** *bookid  
 \** ***@return*** *\*/* int deleteByPrimaryKey(Integer bookid);  
  
 */\*\*  
 \* 添加图书  
 \** ***@param*** *record  
 \** ***@return*** *\*/* int insert(BookInfo record);  
  
 */\*\*  
 \* 添加图书-动态  
 \** ***@param*** *record  
 \** ***@return*** *\*/* int insertSelective(BookInfo record);  
  
 */\*\*  
 \* 更新图书信息-动态  
 \** ***@param*** *record  
 \** ***@return*** *\*/* int updateByPrimaryKeySelective(BookInfo record);  
  
 */\*\*  
 \* 更新图书信息  
 \** ***@param*** *record  
 \** ***@return*** *\*/* int updateByPrimaryKey(BookInfo record);  
  
 */\*\*  
 \* 分页查询图书  
 \** ***@param*** *begin  
 \** ***@param*** *size  
 \** ***@return*** *\*/* List<BookInfo> selectAllByLimit(@Param("begin") Integer begin, @Param("size") Integer size);  
  
 */\*\*  
 \* 根据种类统计图书数量  
 \** ***@param*** *map  
 \** ***@return*** *\*/* int selectCountByType(Map<String, Object> map);  
  
 */\*\*  
 \* 根据种类查询图书信息  
 \** ***@param*** *map  
 \** ***@return*** *\*/* List<BookInfo> selectByType(Map<String, Object> map);  
}

**BookInfoMapper.xml**

<?xml version="1.0" encoding="UTF-8"?>  
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd">  
<mapper namespace="com.zh.mapper.BookInfoMapper">  
 <resultMap id="BaseResultMap" type="com.zh.model.BookInfo">  
 <id column="bookId" jdbcType="INTEGER" property="bookid" />  
 <result column="bookName" jdbcType="VARCHAR" property="bookname" />  
 <result column="bookAuthor" jdbcType="VARCHAR" property="bookauthor" />  
 <result column="bookPrice" jdbcType="DECIMAL" property="bookprice" />  
 <result column="bookTypeId" jdbcType="INTEGER" property="booktypeid" />  
 <result column="bookTypeName" jdbcType="INTEGER" property="booktypename" />  
 <result column="bookDesc" jdbcType="VARCHAR" property="bookdesc" />  
 <result column="isBorrowed" jdbcType="TINYINT" property="isborrowed" />  
 <result column="bookImg" jdbcType="TINYINT" property="bookimg" />  
 </resultMap>  
 <sql id="Base\_Column\_List">  
 bookId, bookName, bookAuthor, bookPrice, bookTypeId, bookDesc, isBorrowed, bookImg  
 </sql>  
  
 <select id="selectCount" resultType="int">  
 select count(\*) from book\_manager.book\_info  
 </select>  
  
 <select id="selectAll" resultMap="BaseResultMap">  
 select bookId, bookName, bookAuthor,bookPrice, bookTypeId, bookDesc,isBorrowed, bookImg,  
 (select bookTypeName from book\_manager.book\_type where book\_manager.book\_type.bookTypeId = book\_manager.book\_info.bookTypeId) as bookTypeName  
 from book\_manager.book\_info  
 </select>  
  
 <select id="selectByPrimaryKey" parameterType="java.lang.Integer" resultMap="BaseResultMap">  
 select  
 <include refid="Base\_Column\_List" />  
 from book\_manager.book\_info  
 where bookId = #{bookid,jdbcType=INTEGER}  
 </select>  
  
 <select id="selectCountBySearch" resultType="int">  
 select count(\*) from book\_manager.book\_info  
 <where>  
 <if test="bookname != null and bookname != '' ">and bookName like concat('%',#{bookname},'%')</if>  
 <if test="bookauthor != null and bookauthor != '' ">and bookAuthor like concat('%',#{bookauthor},'%')</if>  
 <if test="booktypeid != null and booktypeid != '' ">and bookTypeId = #{booktypeid}</if>  
 </where>  
 </select>  
  
 <select id="selectBySearch" resultMap="BaseResultMap">  
 select bookId, bookName, bookAuthor,bookPrice, bookTypeId, bookDesc,isBorrowed, bookImg,  
 (select bookTypeName from book\_manager.book\_type where book\_manager.book\_type.bookTypeId = book\_manager.book\_info.bookTypeId) as bookTypeName  
 from book\_manager.book\_info  
 <where>  
 <if test="bookname != null and bookname != '' ">and bookName like concat('%',#{bookname},'%')</if>  
 <if test="bookauthor != null and bookauthor != '' ">and bookAuthor like concat('%',#{bookauthor},'%')</if>  
 <if test="booktypeid != null and booktypeid != '' ">and bookTypeId = #{booktypeid}</if>  
 </where>  
 limit #{begin}, #{size}  
 </select>  
  
 <delete id="deleteByPrimaryKey" parameterType="java.lang.Integer">  
 delete from book\_manager.book\_info  
 where bookId = #{bookid,jdbcType=INTEGER}  
 </delete>  
  
 <insert id="insert" parameterType="com.zh.model.BookInfo">  
 insert into book\_manager.book\_info (bookId, bookName, bookAuthor,  
 bookPrice, bookTypeId, bookDesc,   
 isBorrowed, bookImg)  
 values (#{bookid,jdbcType=INTEGER}, #{bookname,jdbcType=VARCHAR}, #{bookauthor,jdbcType=VARCHAR},   
 #{bookprice,jdbcType=DECIMAL}, #{booktypeid,jdbcType=INTEGER}, #{bookdesc,jdbcType=VARCHAR},   
 #{isborrowed,jdbcType=TINYINT}, #{bookimg,jdbcType=TINYINT})  
 </insert>  
  
 <insert id="insertSelective" parameterType="com.zh.model.BookInfo">  
 insert into book\_manager.book\_info  
 <trim prefix="(" suffix=")" suffixOverrides=",">  
 <if test="bookid != null">bookId,</if>  
 <if test="bookname != null">bookName,</if>  
 <if test="bookauthor != null">bookAuthor,</if>  
 <if test="bookprice != null">bookPrice,</if>  
 <if test="booktypeid != null">bookTypeId,</if>  
 <if test="bookdesc != null">bookDesc,</if>  
 <if test="isborrowed != null">isBorrowed,</if>  
 <if test="bookimg != null">bookImg,</if>  
 </trim>  
 <trim prefix="values (" suffix=")" suffixOverrides=",">  
 <if test="bookid != null">#{bookid,jdbcType=INTEGER},</if>  
 <if test="bookname != null">#{bookname,jdbcType=VARCHAR},</if>  
 <if test="bookauthor != null">#{bookauthor,jdbcType=VARCHAR},</if>  
 <if test="bookprice != null">#{bookprice,jdbcType=DECIMAL},</if>  
 <if test="booktypeid != null">#{booktypeid,jdbcType=INTEGER},</if>  
 <if test="bookdesc != null">#{bookdesc,jdbcType=VARCHAR},</if>  
 <if test="isborrowed != null">#{isborrowed,jdbcType=TINYINT},</if>  
 <if test="bookimg != null">#{bookimg,jdbcType=TINYINT},</if>  
 </trim>  
 </insert>  
  
 <update id="updateByPrimaryKeySelective" parameterType="com.zh.model.BookInfo">  
 update book\_manager.book\_info  
 <set>  
 <if test="bookname != null">bookName = #{bookname,jdbcType=VARCHAR},</if>  
 <if test="bookauthor != null">bookAuthor = #{bookauthor,jdbcType=VARCHAR},</if>  
 <if test="bookprice != null">bookPrice = #{bookprice,jdbcType=DECIMAL},</if>  
 <if test="booktypeid != null">bookTypeId = #{booktypeid,jdbcType=INTEGER},</if>  
 <if test="bookdesc != null">bookDesc = #{bookdesc,jdbcType=VARCHAR},</if>  
 <if test="isborrowed != null">isBorrowed = #{isborrowed,jdbcType=TINYINT},</if>  
 <if test="bookimg != null">bookImg = #{bookimg,jdbcType=TINYINT},</if>  
 </set>  
 where bookId = #{bookid,jdbcType=INTEGER}  
 </update>  
  
 <update id="updateByPrimaryKey" parameterType="com.zh.model.BookInfo">  
 update book\_manager.book\_info  
 set bookName = #{bookname,jdbcType=VARCHAR},  
 bookAuthor = #{bookauthor,jdbcType=VARCHAR},  
 bookPrice = #{bookprice,jdbcType=DECIMAL},  
 bookTypeId = #{booktypeid,jdbcType=INTEGER},  
 bookDesc = #{bookdesc,jdbcType=VARCHAR},  
 isBorrowed = #{isborrowed,jdbcType=TINYINT},  
 bookImg = #{bookimg,jdbcType=TINYINT}  
 where bookId = #{bookid,jdbcType=INTEGER}  
 </update>  
  
 <select id="selectAllByLimit" resultMap="BaseResultMap">  
 select bookId, bookName, bookAuthor,bookPrice, bookTypeId, bookDesc,isBorrowed, bookImg,  
 (select bookTypeName from book\_manager.book\_type where book\_manager.book\_type.bookTypeId = book\_manager.book\_info.bookTypeId) as bookTypeName  
 from book\_manager.book\_info  
 limit #{begin}, #{size}  
 </select>  
  
  
  
  
 <select id="selectCountByType" resultType="int">  
 select count(\*) from book\_manager.book\_info  
 <where>  
 <if test="booktypeid != null and booktypeid != '' ">and bookTypeId = #{booktypeid}</if>  
 </where>  
 </select>  
  
 <select id="selectByType" resultMap="BaseResultMap">  
 select bookId, bookName, bookAuthor,bookPrice, bookTypeId, bookDesc,isBorrowed, bookImg,  
 (select bookTypeName from book\_manager.book\_type where book\_manager.book\_type.bookTypeId = book\_manager.book\_info.bookTypeId) as bookTypeName  
 from book\_manager.book\_info  
 <where>  
 <if test="booktypeid != null and booktypeid != '' ">and bookTypeId = #{booktypeid}</if>  
 </where>  
 limit #{begin}, #{size}  
 </select>  
  
</mapper>

4、程序运行、测试与分析

登录



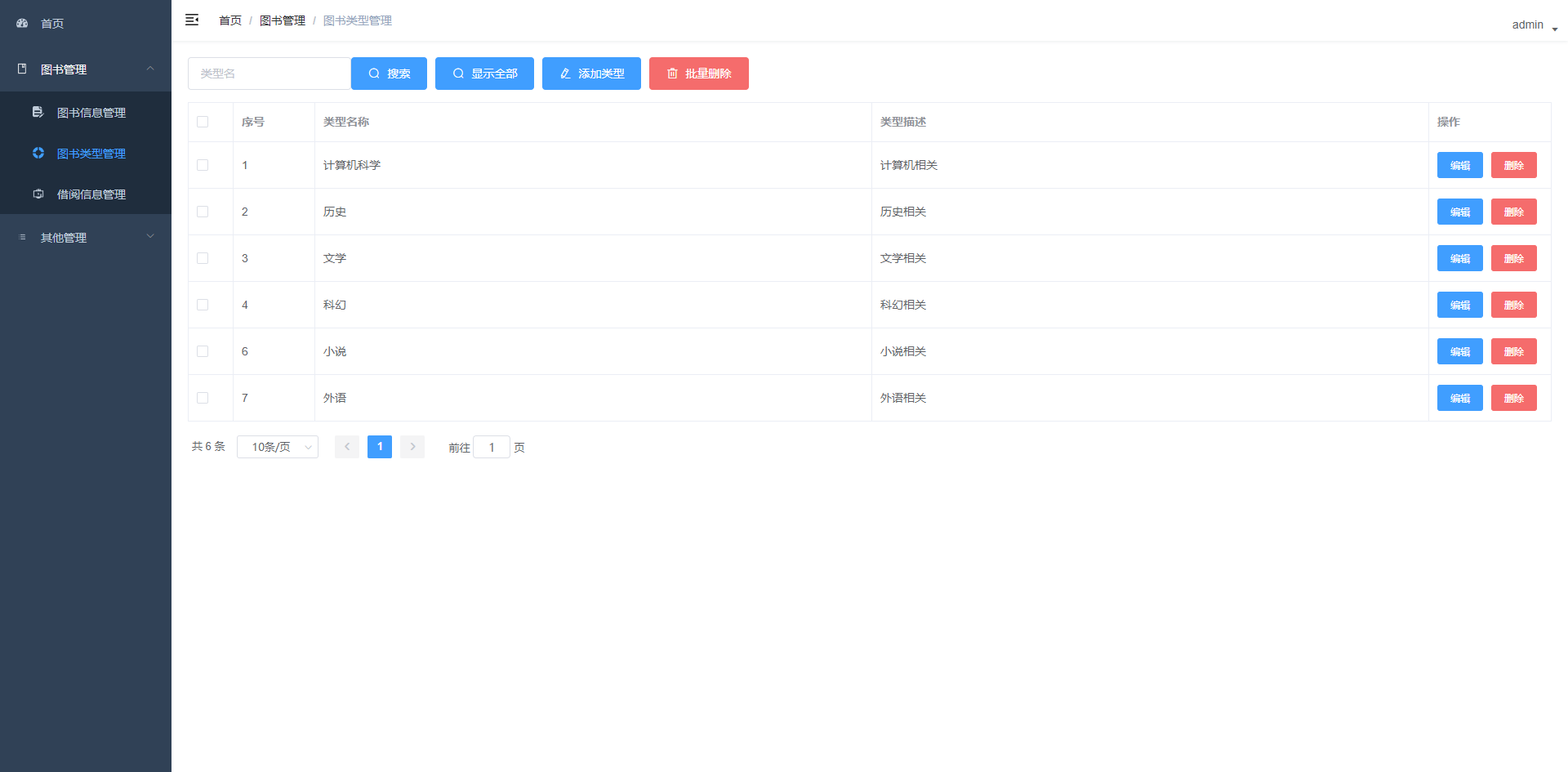
图书信息管理



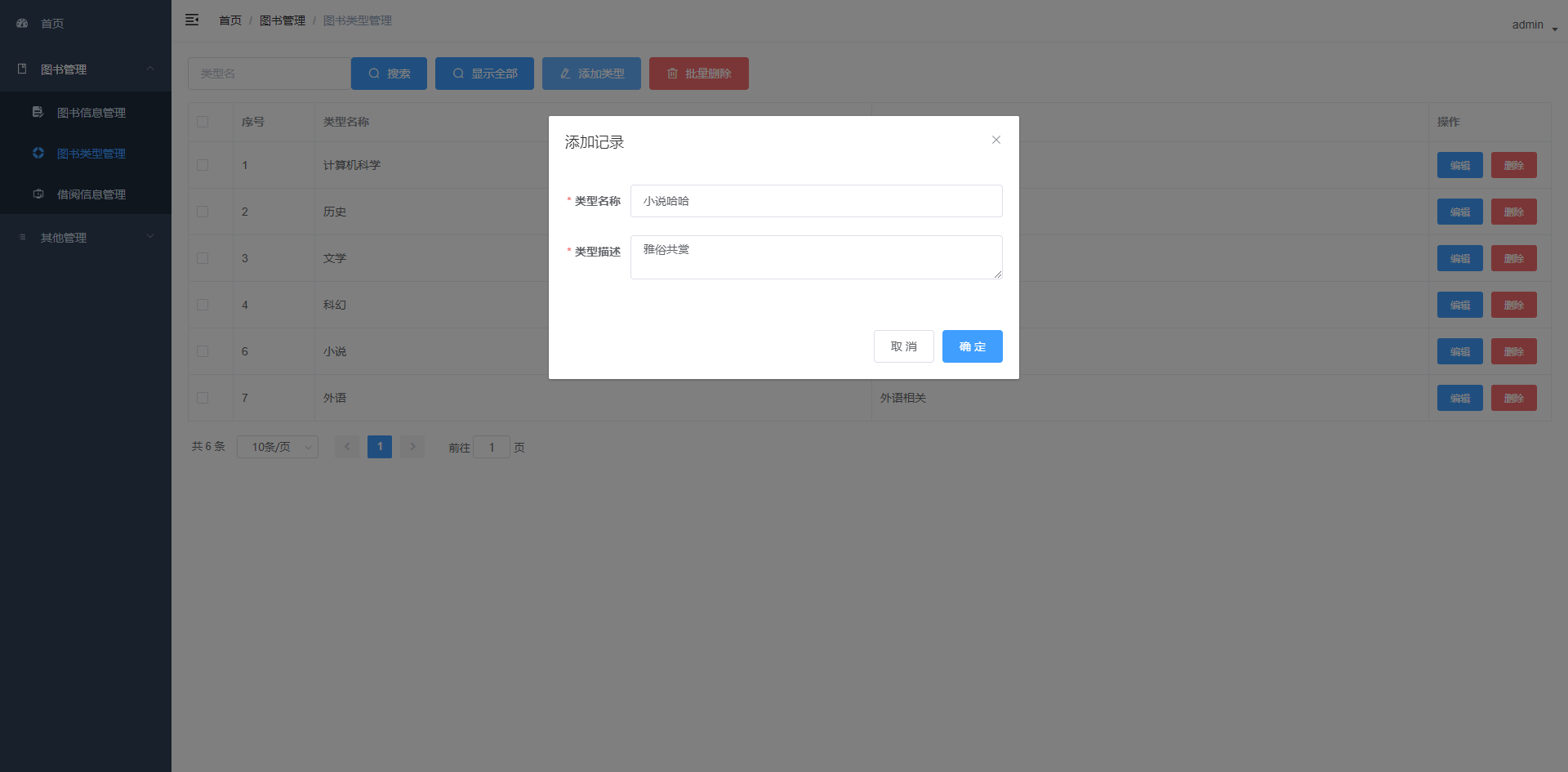
添加图书



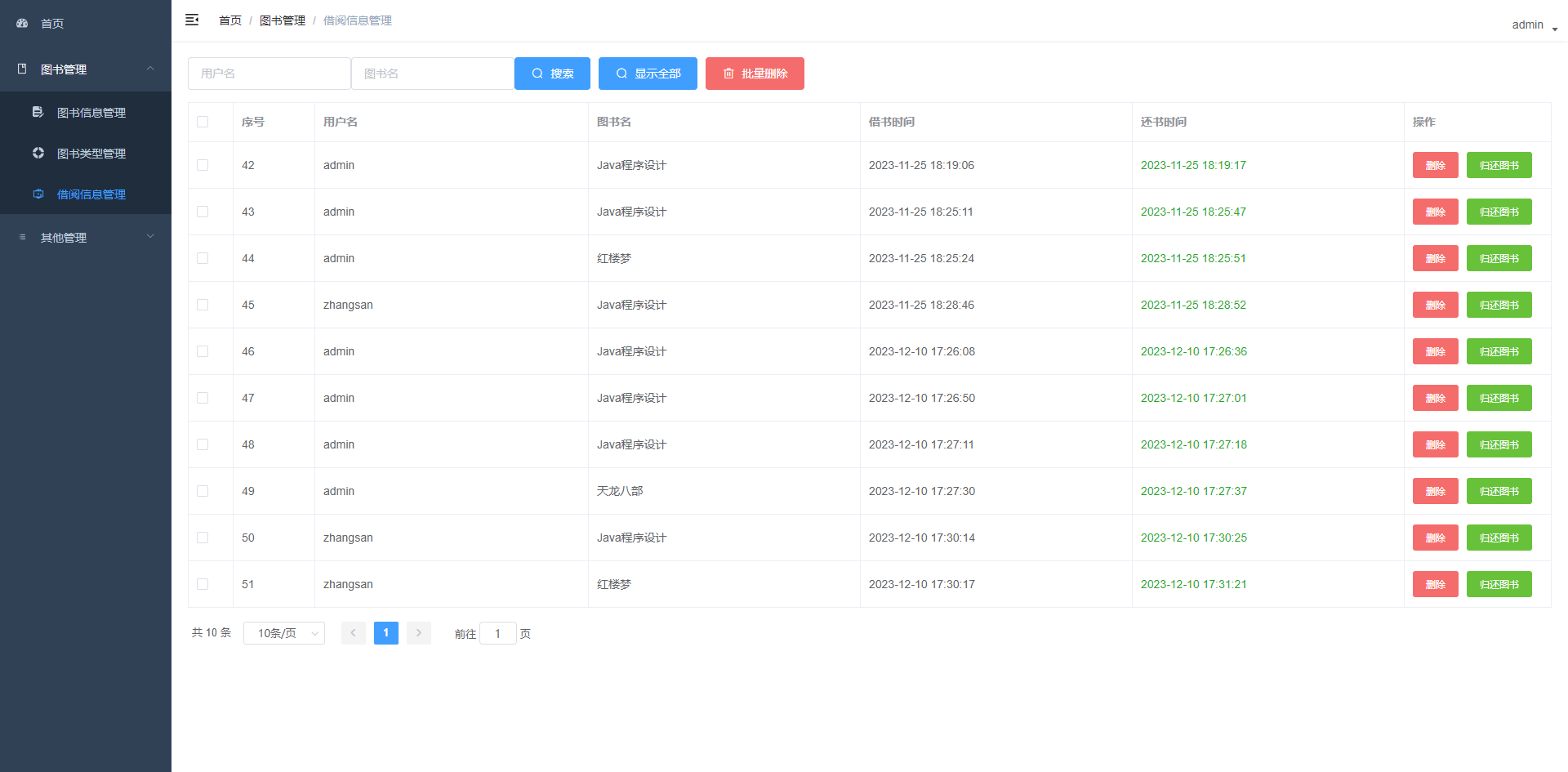
图书类型管理



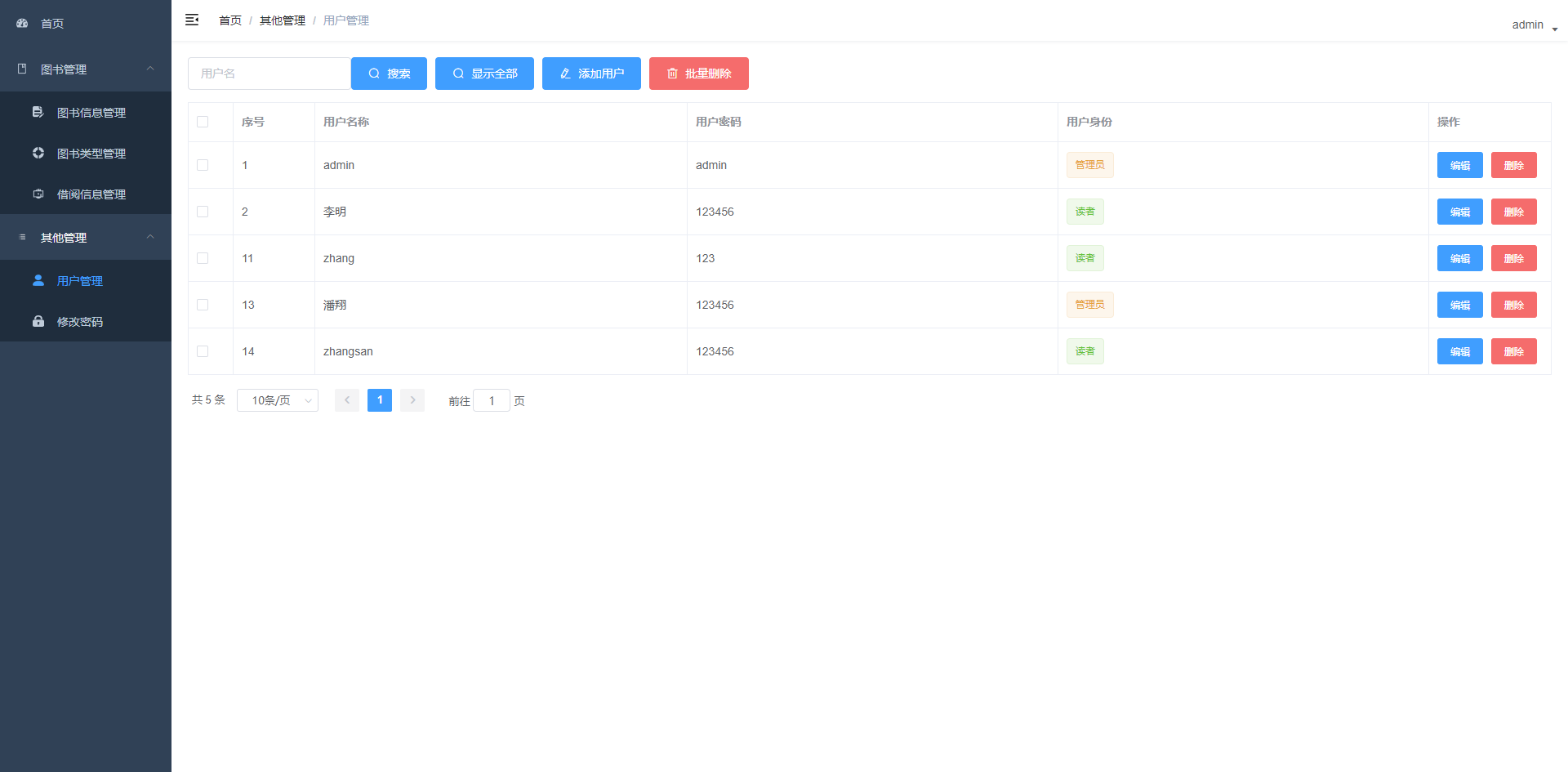
添加图书类型



借阅信息管理



用户管理



添加用户



修改密码

