

**实习实训报告**

**课程名称： 基于springboot+vue的图书信息管理系统**

**学生姓名： 小帅**

**所在学院： 数学与计算机科学学院**

**专 业： 计算机科学与技术专业**

**学 号： 6666666666666**

**教 师： 大帅**

**2024年7月**

**云南民族大学数学与计算机科学学院**

目录

[一、 项目需求 1](#_Toc51752752)

[二、 开发环境 1](#_Toc51752753)

[三、 WBS工作任务分解（截图） 1](#_Toc51752754)

[四、 系统设计 1](#_Toc51752755)

[五、 主要功能和代码 1](#_Toc51752758)

[(一) 功能截图 1](#_Toc51752759)

[(二) 核心代码 1](#_Toc51752760)

[六、 总结 1](#_Toc51752761)

## 项目需求

该系统是基于springboot+vue的图书信息管理系统，使用jsp动态网页技术开发。该系统有管理员和读者两种用户，管理员在该系统中可以查看和管理读者的基本信息，管理员能够添加新书籍的基本信息，能够修改图书信息或者删除不再馆藏的书籍。支持根据书名、作者等关键字进行快速搜索和检索。新增读者个人信息，记录读者借阅的书籍信息，读者可以通过系统查询图书并借阅，系统自动更新库存信息和借阅记录。读者归还图书时，系统更新库存信息，并生成借阅归还记录。用户需要输入用户名和密码完成系统登录，并且有相关提示，密码要加密处理。

## 开发环境

开发平台：Windows10

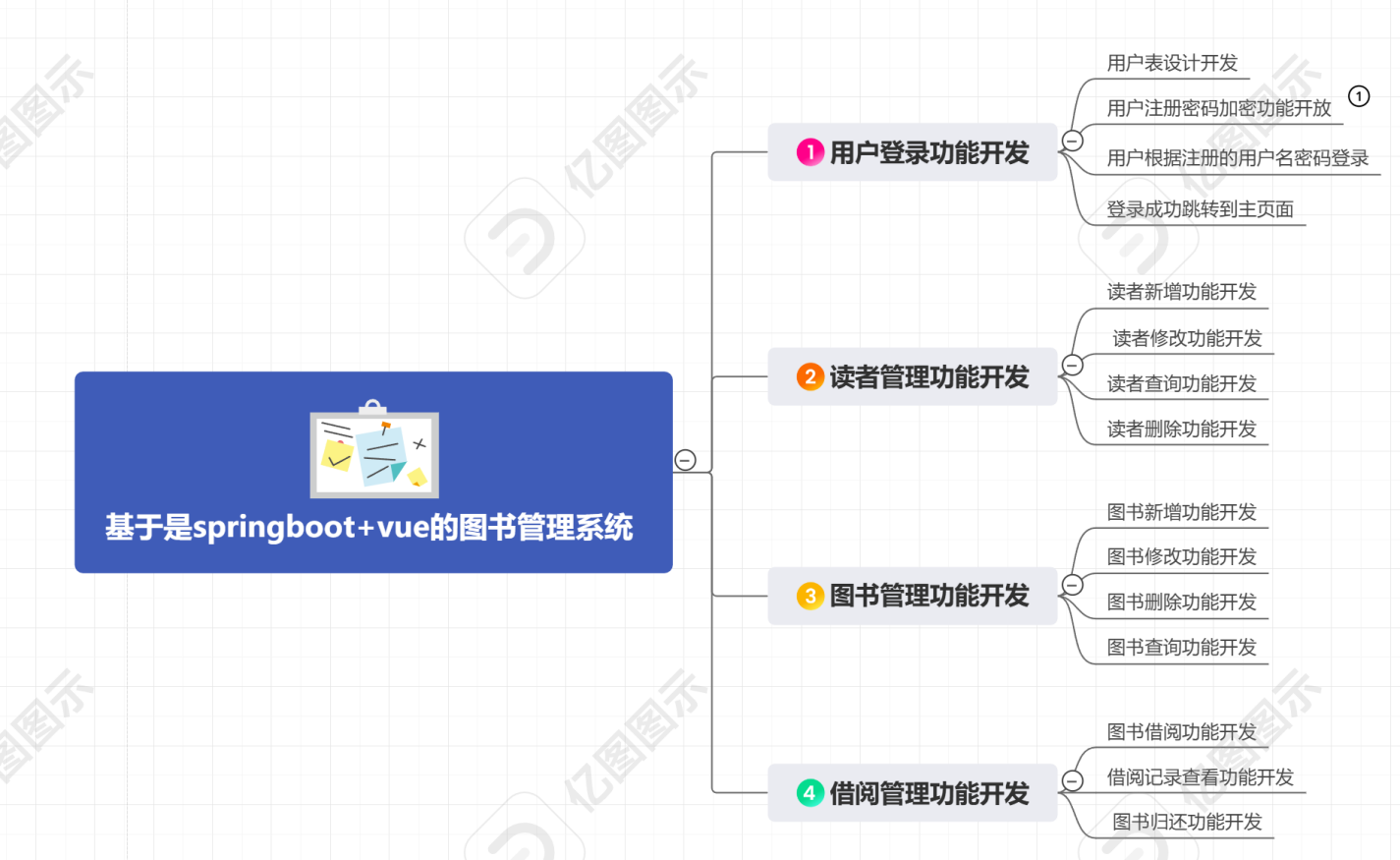
开发语言：Java

开发环境：JDK 8.0

开发工具：IDEA2019

数 据 库：MySQL 8.0

## WBS工作任务分解（截图）



## 系统设计

## 数据库设计

//图书信息表

create table book\_info  
(  
 book\_id int auto\_increment comment '图书ID，主键，自增长'  
 primary key,  
 title varchar(255) null comment '图书标题',  
 author varchar(100) null comment '作者',  
 publisher varchar(100) null comment '出版社',  
 publication\_year varchar(50) null,  
 genre varchar(50) null comment '图书类别',  
 total\_copies int null comment '总藏书量',  
 available\_copies int null comment '可借阅数量',  
 description text null comment '图书描述',  
 delete\_tag smallint default 0 null comment '0 代表未删除 1 代表删除',  
 password varchar(100) null,  
 salt varchar(100) null,  
 constraint title  
 unique (title)  
);

//用户表  
create table user\_info  
(  
 user\_id int auto\_increment  
 primary key,  
 user\_name varchar(40) null,  
 age varchar(40) null,  
 sex varchar(11) null,  
 delete\_tag smallint default 0 null comment '0 代表未删除 1 代表删除',  
 password varchar(100) null,  
 salt varchar(100) null,  
 contacting varchar(20) null,  
 borrow\_records varchar(20) null,  
 breach\_records varchar(20) null,  
 borrow\_date varchar(20) null,  
 due\_date varchar(20) null,  
 constraint user\_name  
 unique (user\_name),  
 constraint user\_name\_2  
 unique (user\_name)  
)  
 charset = utf8mb3;

//借阅表  
create table borrow  
(  
 borrow\_id int auto\_increment comment '借阅记录ID'  
 primary key,  
 reader\_id int not null comment '借阅者ID',  
 book\_id int not null comment '图书ID',  
 borrow\_date varchar(255) not null comment '借书日期',  
 due\_date varchar(255) not null comment '应还日期',  
 delete\_tag smallint default 0 null comment '0 代表未删除 1 代表删除',  
 constraint borrow\_ibfk\_1  
 foreign key (reader\_id) references user\_info (user\_id),  
 constraint borrow\_ibfk\_2  
 foreign key (book\_id) references book\_info (book\_id)  
);  
  
create index book\_id  
 on borrow (book\_id);  
  
create index reader\_id  
 on borrow (reader\_id);

## 原型设计

**用户登录：**

用户输入用户名和密码进行登录，密码会经过哈希函数进行加密处理，提高用户账号的安全性。

**首页界面：**

展示各个功能模块的菜单栏，包括系统首页、读者管理、图书管理、借阅管理等。

显示员工数量、薪酬总额、罚款总额、迟到次数等信息。

展示员工年龄段分布和本月迟到最多员工的信息。

读者管理：

用户新增：在网页上实现对用户信息的新增，包括姓名、年龄、性别、联系方式、密码等。

用户修改：对已有的用户信息进行修改，可修改姓名、年龄、性别、联系方式等。

用户查询：通过输入用户名、性别等信息来查询用户信息，并展示查询结果。

用户删除：通过删除按钮进行用户信息的删除，删除前会弹出确认提示。

**图书管理：**

图书新增：在网页上实现对图书信息的新增，包括标题、作者、出版社、出版年份、图书类别等。

图书修改：对已有的图书信息进行修改，可修改标题、作者、出版社、出版年份、图书类别等。

图书删除：通过删除按钮进行图书信息的删除，删除前会弹出确认提示。

图书查询：输入图书的标题、出版社、类别进行查询，支持模糊查询和精确查询。

**借阅管理：**

图书借阅：实现对图书的借阅，借阅后，图书库存量减一。

借阅记录查看：点击用户界面的查看按钮，可以查看用户的借阅信息，包括借书日期、归还日期、借阅ID、图书ID等。

图书归还：读者归还图书时，系统更新库存信息，并生成借阅归还记录。

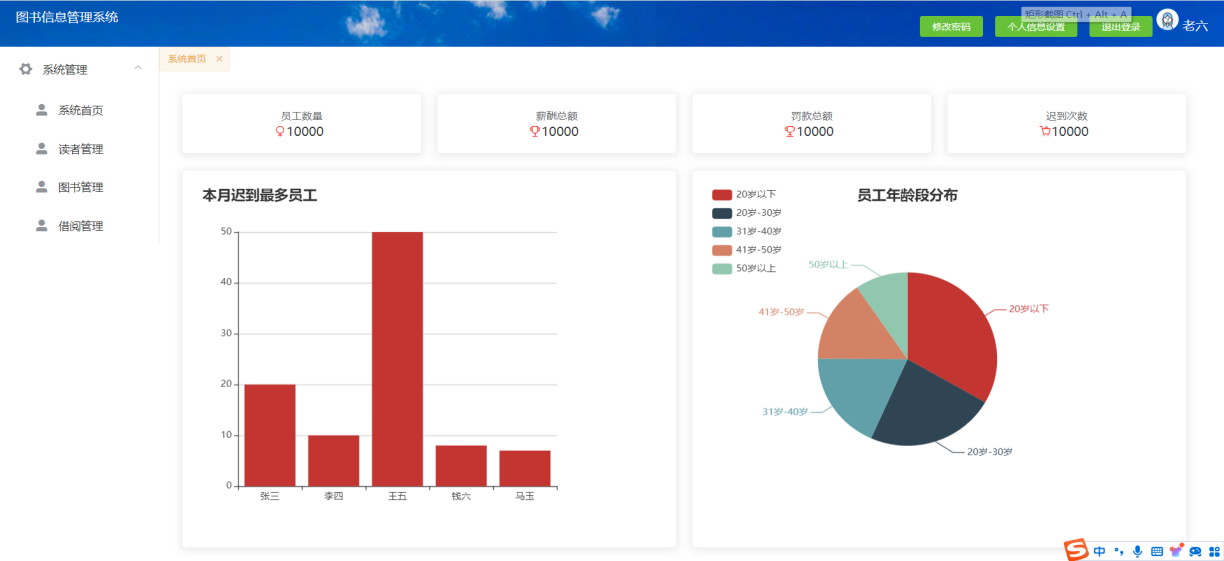
## 主要功能和代码

## 功能截图

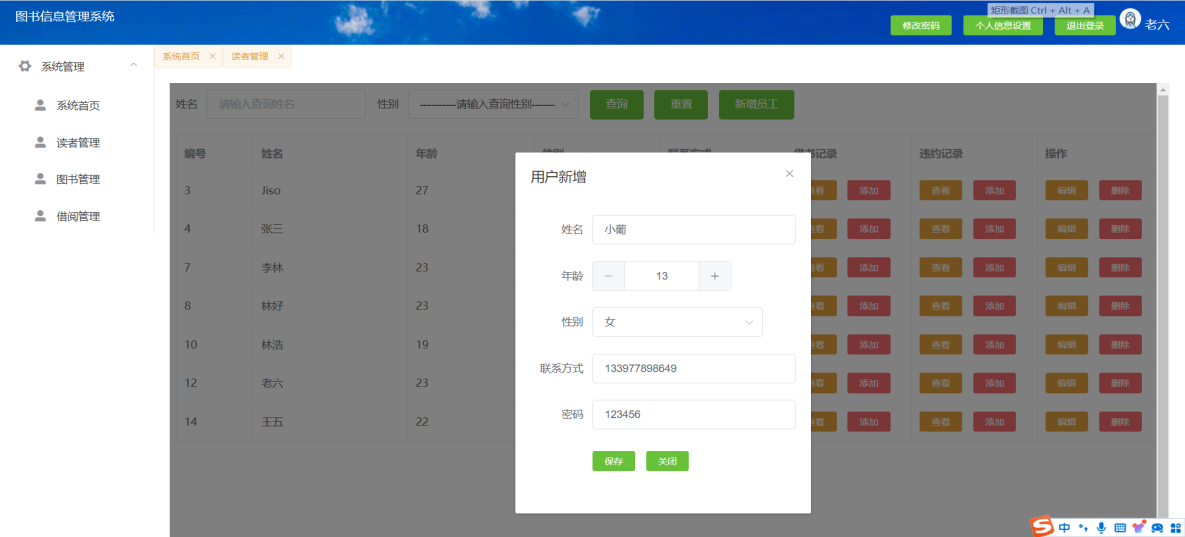
用户登录时，当用户密码时，用户输入的密码会经过哈希函数进行加密处理，让用户账号更加安全。



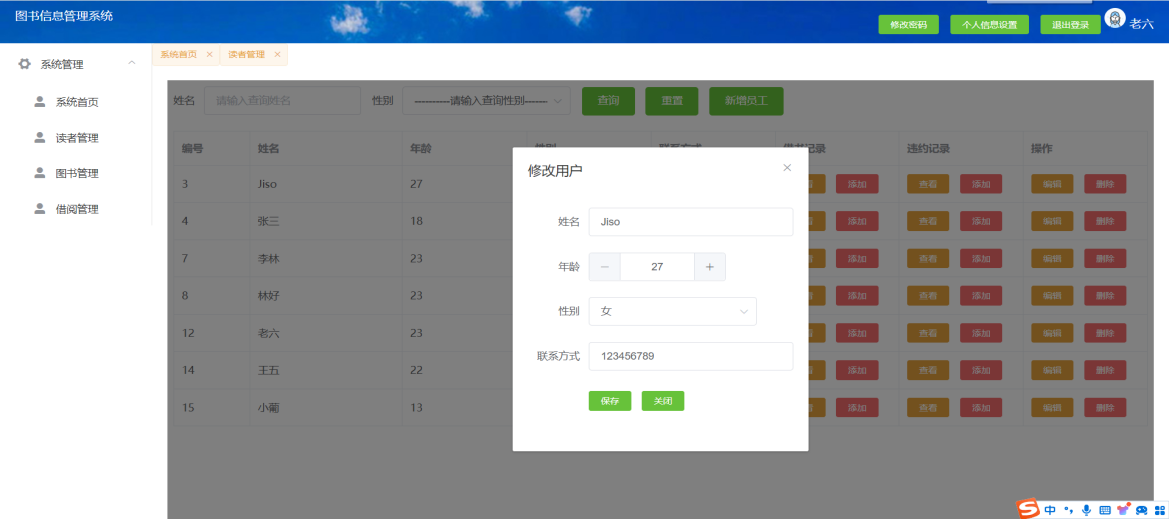
首页界面有各个功能模块的的菜单栏



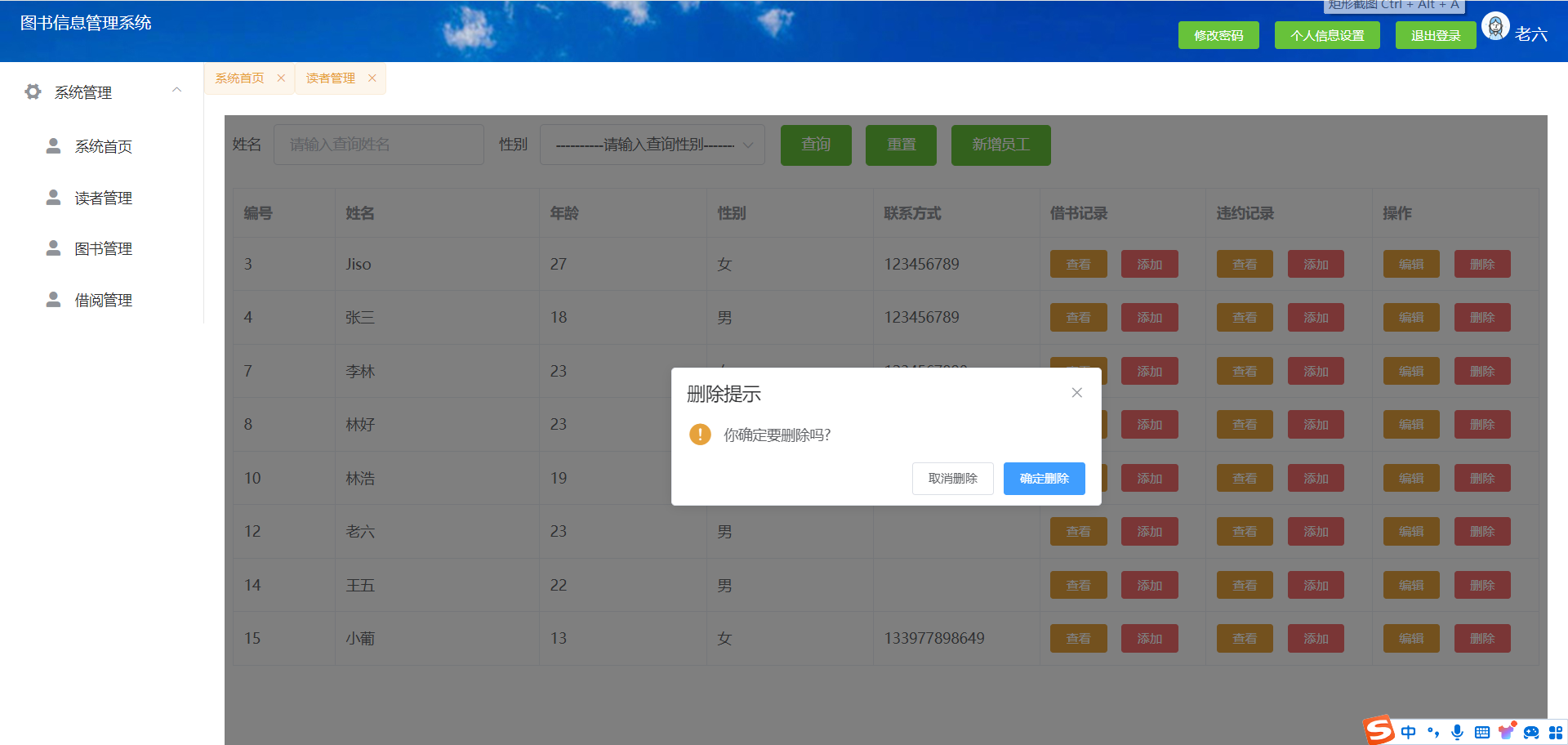
在网页上实现对用户信息的新增



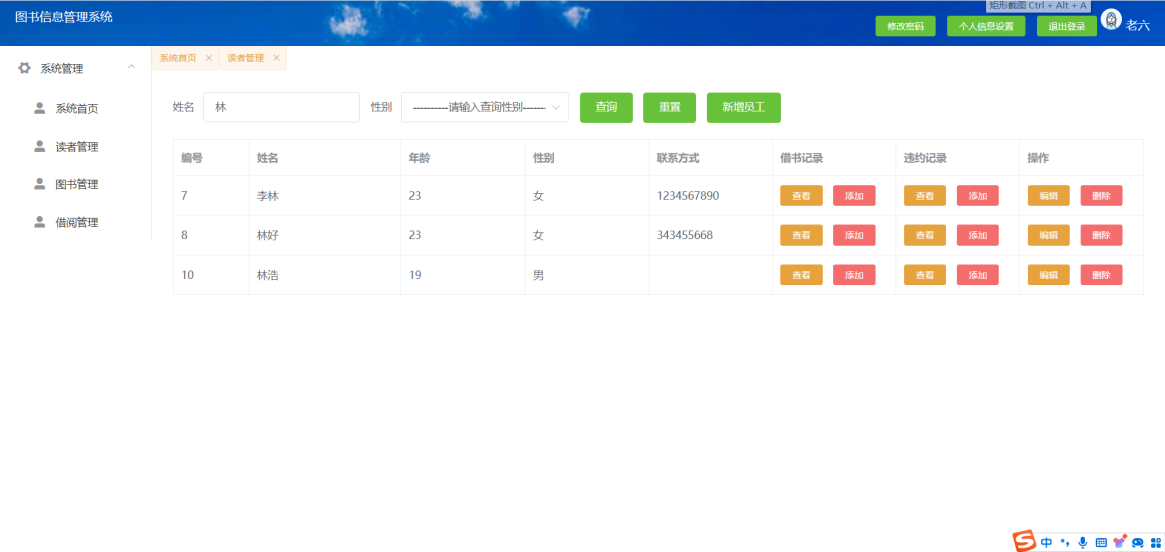
对已有的用户信息进行修改



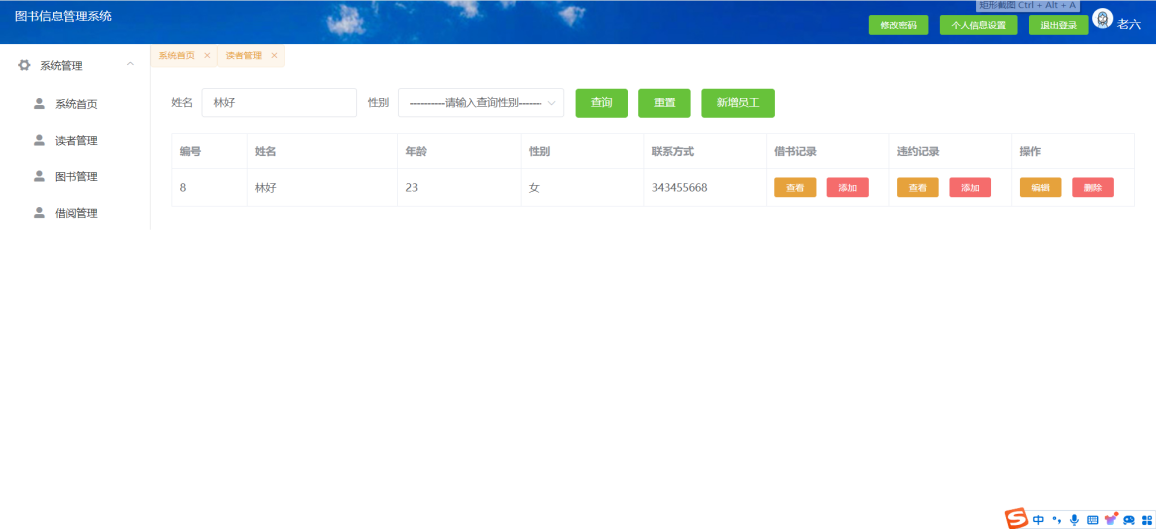
通过删除按钮进行图书信息的删除



通过输入用户名，性别来查询用户信息



通过输入用户名，性别来查询用户信息

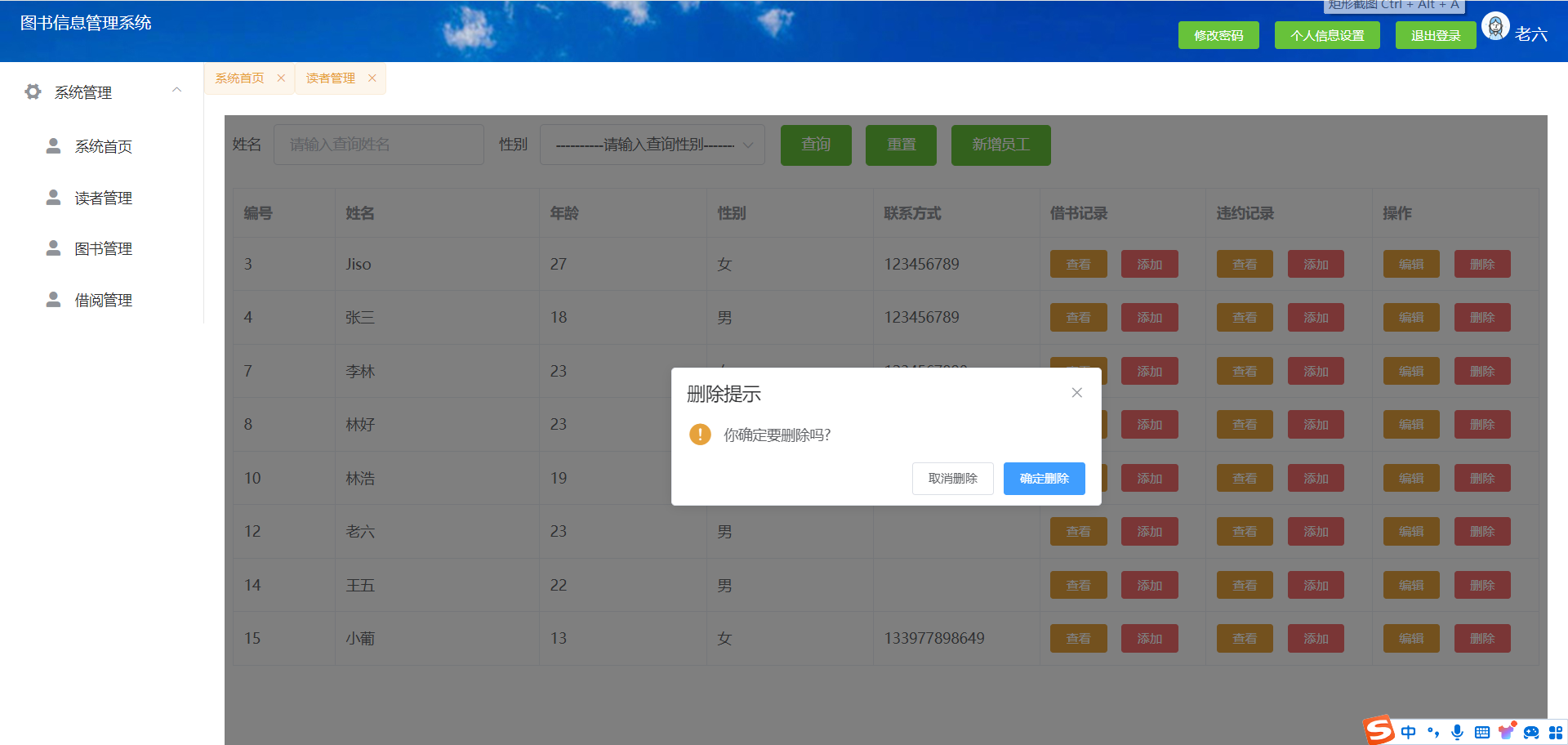


在网页上实现对图书信息的新增



对已有的图书信息进行修改和通过删除按钮删除图书信息



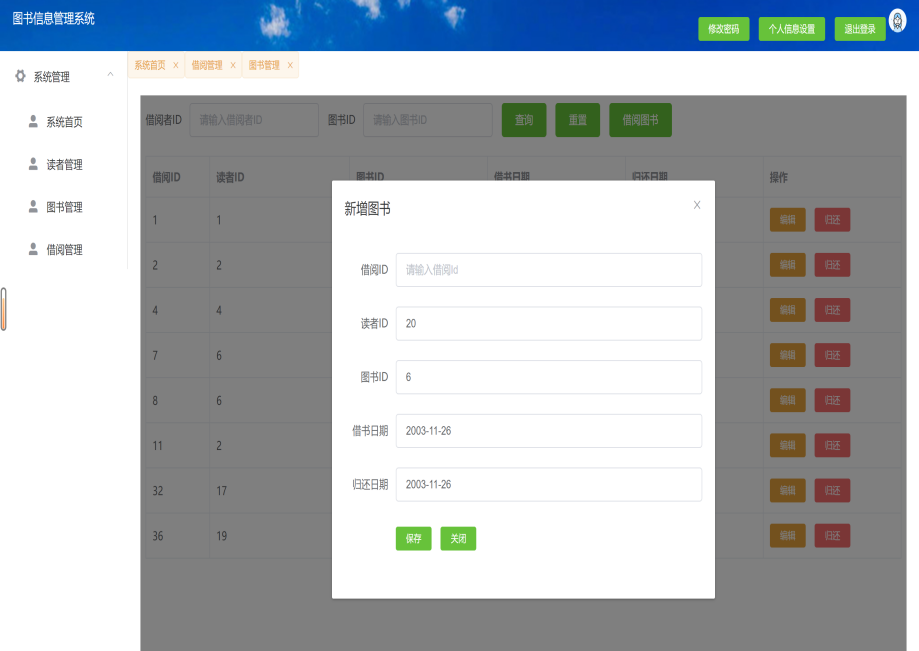


输入图书的标题，出版社，类别进行查询模糊查询和精确查询

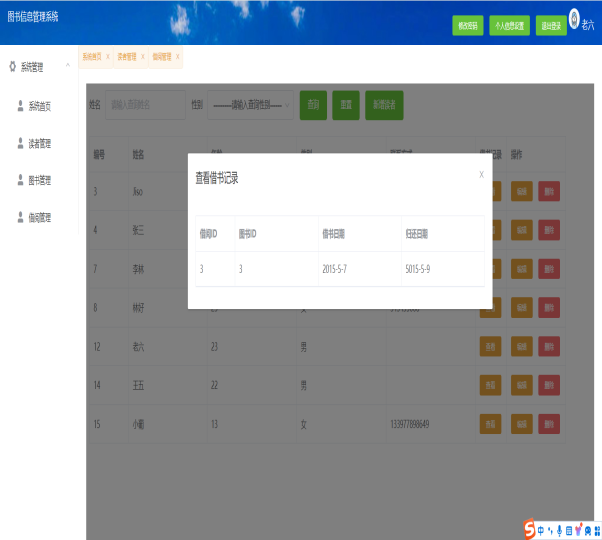
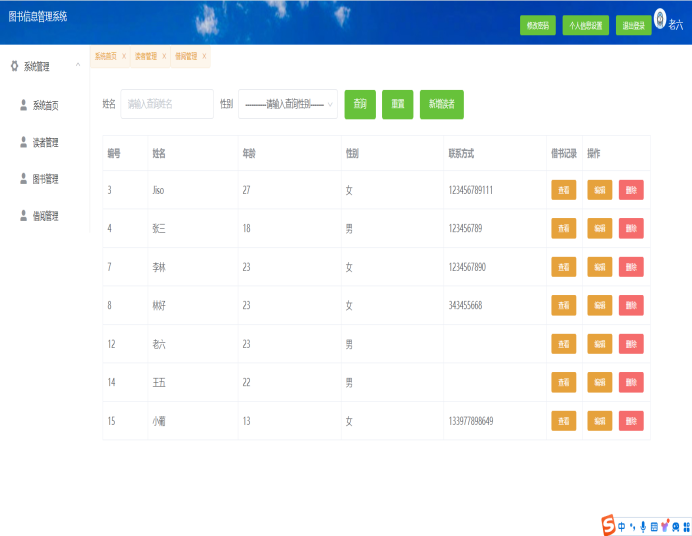




实现对图书的借阅，借阅后，图书库存量减一，归还后，图书库存加一



点击用户界面的查看按钮，就可以查看用户的借阅信息



## 核心代码

package com.zcs.service;

import com.zcs.dto.BookInfoDto;

import com.zcs.pojo.BookInfo;

import com.zcs.util.ResultVo;

public interface BookInfoService {

//新增图书

ResultVo addBookInfo(BookInfo bookInfo);

//修改图书

ResultVo updateBookInfo(BookInfo bookInfo);

//删除图书

ResultVo delBookInfo(Integer bookId);

//查询图书

ResultVo query(BookInfoDto dto);

}

package com.zcs.service;

import com.zcs.dto.BorrowInfoDto;

import com.zcs.pojo.BorrowInfo;

import com.zcs.util.ResultVo;

/\*\*

\* @version 1.0

\* @Author 张成双

\* @Date 2024/7/18 上午10:21

\* {@code @注释:定义具体业务实现}

\*\*/

public interface BorrowInfoService {

//新增用户功能

ResultVo addBorrowInfo(BorrowInfo borrowInfo);

//修改用户

ResultVo updateBorrowInfo(BorrowInfo borrowInfo);

//删除用户

ResultVo deleteBorrowInfo(Integer id);

//查询用户

ResultVo query(BorrowInfoDto dto);

}

package com.zcs.service;  
  
import com.zcs.dto.UserInfoDto;  
import com.zcs.pojo.UserInfo;  
import com.zcs.util.ResultVo;  
  
import javax.servlet.http.HttpSession;  
  
public interface UserInfoService {  
  
    //新增用户  
    ResultVo addUserInfo(UserInfo userInfo);  
  
    //修改用户  
    ResultVo updateUserInfo(UserInfo userInfo);  
  
    //删除用户  
    ResultVo delUserInfo(Integer id);  
  
    //查询用户  
    ResultVo query(UserInfoDto dto);  
  
    //登录  
    ResultVo login(UserInfo userInfo, HttpSession session); /session会话  
}  
  
  package com.zcs.service.impl;

import com.baomidou.mybatisplus.core.conditions.query.QueryWrapper;

import com.zcs.dao.BookInfoDao;

import com.zcs.dto.BookInfoDto;

import com.zcs.pojo.BookInfo;

import com.zcs.service.BookInfoService;

import com.zcs.util.ResultVo;

import org.springframework.stereotype.Service;

import javax.annotation.Resource;

import java.util.List;

@Service

public class BookInfoServiceImpl implements BookInfoService {

//获取BookInfo的实现类对象

@Resource

BookInfoDao dao;

@Override

public ResultVo addBookInfo(BookInfo bookInfo) {

try{

//新增

int n =dao.insert(bookInfo);

if (n>0){

return ResultVo.success();

}else {

return ResultVo.error();

}

}catch(Exception e){

//e.printStackTrace();

System.out.println(e.getMessage());//打印出错误

return ResultVo.error("图书重名",null);

}

}

@Override

public ResultVo updateBookInfo(BookInfo bookInfo) {

try{

//修改

int n =dao.updateById(bookInfo);

if (n>0){

return ResultVo.success();

}else {

return ResultVo.error();

}

}catch(Exception e){

//e.printStackTrace();

System.out.println(e.getMessage());//打印出错误

return ResultVo.error("图书重名",null);

}

}

@Override

public ResultVo delBookInfo(Integer bookId) {

//删除

int n =dao.deleteById(bookId);

if (n>0){

return ResultVo.success();

}else {

return ResultVo.error();

}

}

@Override

public ResultVo query(BookInfoDto dto) {

QueryWrapper<BookInfo> queryWrapper = new QueryWrapper<>();

//定义图书名的查询条件

boolean isBookName = null!=dto.getTitle()&&!dto.getTitle().equals("");

queryWrapper.like(isBookName,"title",dto.getTitle());

//定义出版社的查询条件

boolean isPublisher = null!=dto.getPublisher()&&!dto.getPublisher().equals("");

queryWrapper.like(isPublisher,"publisher",dto.getPublisher());

//定义类别查询条件

boolean isGenre = null!= dto.getGenre() &&!dto.getGenre().equals("");

queryWrapper.eq(isGenre,"genre",dto.getGenre());

List<BookInfo> list =dao.selectList(queryWrapper);

if (list.size()>0){

return ResultVo.success("查询到了数据",list);

}

return ResultVo.error("没有查询数据",null);

}

}package com.zcs.service.impl;

import com.baomidou.mybatisplus.core.conditions.query.QueryWrapper;

import com.zcs.dao.BorrowInfoDao;

import com.zcs.dao.BookInfoDao;

import com.zcs.dto.BorrowInfoDto;

import com.zcs.pojo.BookInfo;

import com.zcs.pojo.BorrowInfo;

import com.zcs.service.BorrowInfoService;

import com.zcs.util.ResultVo;

import org.springframework.stereotype.Service;

import javax.annotation.Resource;

import java.util.List;

/\*\*

\* @version 1.0

\* @Author 张成双

\* @Date 2024/7/25 下午7:24

\* @注释:

\*\*/

@Service

public class BorrowInfoServiceImpl implements BorrowInfoService {

//获取UserInfoDao的实现类对象

@Resource//资源注入，通过dao来调用UserInfoDao中定义的方法从而实现于数据库的交互

BorrowInfoDao dao;//定义的接口类对象

BookInfoDao daoTwo;//定义的接口类对象

// 添加构造函数，注入 daoTwo

public BorrowInfoServiceImpl(BookInfoDao daoTwo) {

this.daoTwo = daoTwo;

}

@Override//重写父类或接口中的同名方法

public ResultVo addBorrowInfo(BorrowInfo borrowInfo) {

try {

//新增之前验证新增的用户名是否重名，用代码控制

// QueryWrapper<UserInfo> queryWrapper = new QueryWrapper<>();//UserInfo对应的数据库

// queryWrapper.eq("user\_name", userInfo.getUserName());//eq等于什么

// UserInfo u = dao.selectOne(queryWrapper);//返回单个数据，登录注册用,u不能于传入的参数一样

// if (u != null) {

// return ResultVo.error("用户名重名，请重新输入", "null");

// }

//新增

int n = dao.insert(borrowInfo);

if (n > 0) {

// 更新bookInfo表的available\_copies字段

BookInfo bookInfo = daoTwo.selectById(borrowInfo.getBookId());

bookInfo.setAvailableCopies(bookInfo.getAvailableCopies() - 1);

daoTwo.updateById(bookInfo);

return ResultVo.success();

} else {

return ResultVo.error();

}

}catch (Exception e){

//e.printStackTrace(); 打印堆栈

System.out.println(e.getMessage());//看起来比较友好，也是打印异常

return ResultVo.error("用户名重名","null");

}

}

@Override//修改业务

public ResultVo updateBorrowInfo(BorrowInfo borrowInfo) {

try {

int n = dao.updateById(borrowInfo);//updateById根据id进行修改

if (n > 0) {

//System.out.println("修改成功");

return ResultVo.success();

} else {

//System.out.println("修改失败");

return ResultVo.error();

}

}catch (Exception e){

System.out.println(e.getMessage());

return ResultVo.error("用户名重名","null");

}

}

@Override

public ResultVo deleteBorrowInfo(Integer id) {

BorrowInfo borrowInfo = dao.selectById(id);

int n = dao.deleteById(id);

if (n > 0) {

// 更新bookInfo表的available\_copies字段

BookInfo bookInfo = daoTwo.selectById(borrowInfo.getBookId());

bookInfo.setAvailableCopies(bookInfo.getAvailableCopies() + 1);

daoTwo.updateById(bookInfo);

return ResultVo.success();

} else {

return ResultVo.error();

}

}

@Override

public ResultVo query(BorrowInfoDto dto) {

QueryWrapper<BorrowInfo> queryWrapper = new QueryWrapper<>();

//定义读者Id

boolean isreaderID = null!=dto.getReaderId()&&!dto.getReaderId().equals("");

//根据isUserName的值，对user\_name字段进行模糊查询。如果isUserName为true，则使用dto.getUserName()作为查询条件

queryWrapper.eq(isreaderID,"reader\_id",dto.getReaderId());

//定义图书ID查询

boolean isbookID = null!=dto.getBookId()&&!dto.getBookId().equals("");

queryWrapper.eq(isbookID,"book\_id",dto.getBookId());

List<BorrowInfo> list = dao.selectList(queryWrapper);

if (list.size() > 0) {

return ResultVo.success("查询到了数据", list);

}

return ResultVo.error("没有查询数据", null);

}

}

package com.zcs.service.impl;

import com.baomidou.mybatisplus.core.conditions.query.QueryWrapper;

import com.zcs.dao.UserInfoDao;

import com.zcs.dto.UserInfoDto;

import com.zcs.pojo.UserInfo;

import com.zcs.service.UserInfoService;

import com.zcs.util.ResultVo;

import com.zcs.util.SHA3Util;

import org.springframework.stereotype.Service;

import javax.annotation.Resource;

import javax.servlet.http.HttpSession;

import java.util.List;

/\*\*

\* 项目:yunnanmd

\* 描述:

\* 时间:2024/7/18 10:34

\* 作者:admin

\* 版本:1.0

\*\*/

@Service

public class UserInfoServiceImpl implements UserInfoService {

//获取UserInfoDao的实现类对象

@Resource

UserInfoDao dao;

@Override

public ResultVo addUserInfo(UserInfo userInfo) {

try {

//生成盐值

String salt = SHA3Util.generateRandomSalt(16);

//加密密码

String pwd = SHA3Util.hashWithSalt(userInfo.getPassword(),salt);

userInfo.setSalt(salt);

userInfo.setPassword(pwd);

//新增

int n = dao.insert(userInfo);

if (n > 0) {

return ResultVo.success();

} else {

return ResultVo.error();

}

} catch (Exception e) {

//e.printStackTrace();

System.out.println(e.getMessage());

return ResultVo.error("用户名重名", null);

}

}

@Override

public ResultVo updateUserInfo(UserInfo userInfo) {

try {

//生成盐值

String salt = SHA3Util.generateRandomSalt(16);

//加密密码

String pwd = SHA3Util.hashWithSalt(userInfo.getPassword(),salt);

userInfo.setSalt(salt);

userInfo.setPassword(pwd);

//修改

int n = dao.updateById(userInfo);

if (n > 0) {

return ResultVo.success();

} else {

return ResultVo.error();

}

} catch (Exception e) {

System.out.println(e.getMessage());

return ResultVo.error("用户名重名", null);

}

}

@Override

public ResultVo delUserInfo(Integer id) {

//删除

int n = dao.deleteById(id);

if (n > 0) {

return ResultVo.success();

} else {

return ResultVo.error();

}

}

@Override

public ResultVo query(UserInfoDto dto) {

QueryWrapper<UserInfo> queryWrapper = new QueryWrapper<>();

//定义用户名的查询条件

boolean isUserName = null != dto.getUserName() && !dto.getUserName().equals("");

queryWrapper.like(isUserName, "user\_name", dto.getUserName());

//定义性别的查询条件

boolean isSex = null != dto.getSex() && !dto.getSex().equals("");

queryWrapper.eq(isSex, "sex", dto.getSex());

List<UserInfo> list = dao.selectList(queryWrapper);

if (list.size() > 0) {

return ResultVo.success("查询到了数据", list);

}

return ResultVo.error("没有查询数据", null);

}

@Override

public ResultVo login(UserInfo userInfo, HttpSession session) {

QueryWrapper<UserInfo> queryWrapper = new QueryWrapper<>();

queryWrapper.eq("user\_name",userInfo.getUserName());

UserInfo u = dao.selectOne(queryWrapper);

if(u==null){

return ResultVo.error("用户名不正确",null);

}

//加密密码

String pwd = SHA3Util.hashWithSalt(userInfo.getPassword(),u.getSalt());

queryWrapper.eq("password",pwd);

UserInfo uu = dao.selectOne(queryWrapper);

if(uu==null){

return ResultVo.error("密码不正确",null);

}

//记录登录状态

session.setAttribute("name",u.getUserName());

return ResultVo.success();

}

}

## 总结

通过本次基于springboot + vue的图书信息管理系统的实习实训，我不仅成功地实现了系统的各项功能，还在这个过程中获得了许多宝贵的经验和成长。

在系统开发过程中，我深入学习了springboot和vue的相关知识，掌握了它们的基本原理和应用技巧。同时，我提高了自己的网页开发能力。

在数据库设计方面，我学会了如何合理地设计表结构，建立关联关系，以确保数据的完整性和一致性。通过对图书信息表、用户表和借阅表的设计和管理，我对数据库的操作有了更深入的理解。

在功能实现上，我学会了如何将系统的需求转化为具体的代码实现，解决了遇到的各种问题和挑战。例如，用户登录时的密码加密处理，图书的增删改查功能，以及借阅管理中的库存更新和借阅记录生成等。

此外，在团队协作和沟通方面，我也取得了进步。与团队成员共同讨论、解决问题，提高了我的团队合作能力和沟通能力。

通过这次实习实训，我不仅学到了专业知识和技能，还培养了自己的耐心、细心和解决问题的能力。我意识到在实际开发中，需要不断学习和探索，才能更好地应对各种挑战。

在未来的学习和工作中，我将继续努力提升自己的技术水平，不断完善和优化系统，为实际应用提供更好的解决方案。