《WEB开发技术》

图书管理系统

日 期： 2022年6月

1. 可行性研究

采用现代化统一的计算机信息网站系统，能够有效优化图书馆管理系统，使其高效的发挥最大作用，能够迅捷的为读者提供相应的服务。开发本系统的可行性研究如下：

* 1. 技术可行性

技术上的可行性分析主要分析技术条件能否顺利完成开发工作，软、硬件能否满足需要。本系统前端采用Vue开发界面，便于用户理解、操作。后端采用SpringBoot开发，更快捷。数据库管理系统采用MySQL，它能够处理大量数据，同时保持数据的完整性、安全性。因此本系统的开发平台已成熟可行。硬件方面，在科技飞速发展的今天，硬件更新速度越来越快，容量越来越大，可靠性越来越高，价格越来越便宜，因此硬件平台也能够满足本系统所需[2]。

* 1. 经济可行性

鉴于计算机技术发展异常迅猛,在硬件软件配置以及开发技术均以可行的情况下开发这样一个管理系统成本不会很高，但其可以大大提高图书馆的工作效率,也是图书馆管理发展的必然趋势,其必将有比较宽阔的市场,因此改性统在经济可行性上时可行的[2]。

1. 系统需求分析
   1. 系统目标

该系统主要建立一个基于SpringBoot和Vue的图书馆管理系统,面对当起很多小型图书管理仍是人工管理带来的检索速度慢,效率低,借阅归还图书量大,图书统计工作量大,藏书不能完成及时更新的问题,该系统可以对跟系统的三个用户类型的使用实现:

1.对于读者在本系统的应用下可实现按照各种方式(如:书名,图书编号,作者)查询图书馆的藏书情况,或直接浏览图书列表，或按照图书分类浏览图书，方便的借阅图书,续借图书,归还图书,能够查询自己的借阅图书情况。

2.对于管理员(图书馆工作人员)能够实现方便的对图书情况进行查看(如已借图书,已还图书,预约信息,借阅排行),能进行图书审核(如节约审核,御姐审核,归还审核),同时能便捷的对图书信息进行添加、修改、删除,分类管理等操作。

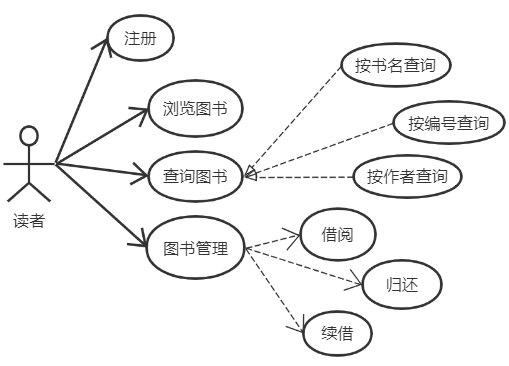
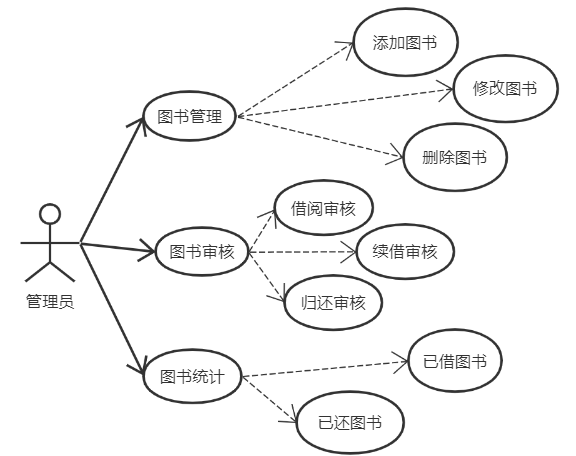
3.对于超级管理员可以添加或删除借书证，可以查看,修改,删除,或添加管理员,可以修改图书类别或添加图书类别，可以对图书情况进行查看(如已借图书,已还图书,预约信息,借阅排行)。

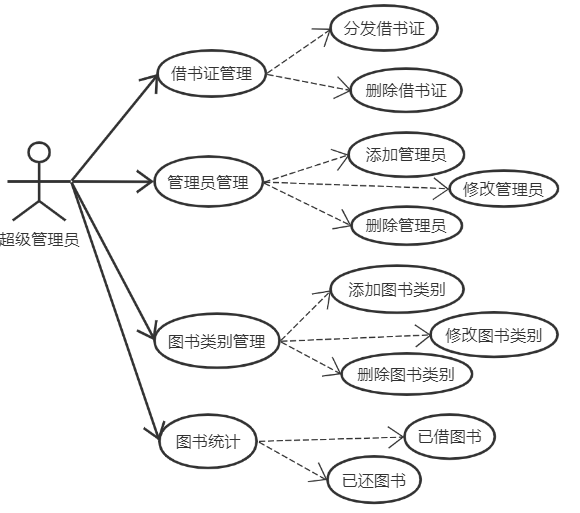
* 1. 用户类和用户特征

图书管理系统是一个基于SpringBoot和Vue的对图书馆进行高效率管理的应用系统,它的用户主要是读者和管理员,学生通过该系统进行图书查询进而对自己需要的图书进行借阅及自己的借阅情况进行查询,管理员则通过本系统实现对图书的高效管理和对读者借阅图书的高效审核和查看,除此之外,还需要一个超级管理员添加不同用户和添加修改图书类型等操作[2]。

三类用户的具体描述如下：

|  |  |
| --- | --- |
| 用户类 | 描述 |
| 读者 | 读者是该系统的重要的使用角色,他们通过该系统查询自己需要的图书信息,并向管理员提出借阅或归还图书的申请进而借阅或归还图书,同时可以进行个人信息的修改。读者账号可通过注册得到。 |
| 图书管理员 | 管理员是该系统的另一个重要使用者,管理员通过该系统进行图书的增加,修改,删除等操作,实现对读者借阅归还续借图书的便捷审核,实现对系统中图书借阅归还情况的查询,信息更改维护等操作。 |
| 超级管理员 | 超级管理员主要是图书管理系统中借书证和管理员的管理,借书证的分发删除和管理员的添加修改和删除，还包括图书类别的管理(添加,修改,删除图书类别),浏览已借图书,已还图书等详情。 |



* 1. 图书管理系统需求模型
     1. 功能描述

功能描图书借阅管理系统的主要任务是实现读者迅速检索查询,方便借阅归还图书,管理员高效的完成系统的各项基本操作,超级管理员是管理借书证和管理员等操作[3],从图2.1可以看出图书管理系统要完成一下功能:

1. 登录。

读者、管理员,超级管理员进入该系统必须登录，身份验证正确了才可以进入该系统，以不同身份进入该系统所对应的系统使用权限是不同的。

2.注册

读者可以通过注册获得账号。

3.系统设置功能：

超级管理员可以设置图书相关的参数信息。

4.用户管理功能：

超级管理员对用户进行借书证分发删除，对管理员进行添加修改删除等操作。

5.查询功能：

对图书的信息进行查询，查看读者已借已还图书，。

6.审核

管理员对读者的图书借阅、归还、续借申请进行审核。

7.图书管理

管理员进行添加修改删除图书的操作。

5.其他功能

* + 1. 读者功能描述

1.注册

读者通过注册获得账号。

2.个人信息管理

读者通过个人信息管理完善修改个人信息。

3.图书浏览

读者可以直接浏览或按分类浏览图书，也可以根据图书编号、名字、作者名字进行图书查询进信息浏览。

4.借阅、归还、续借图书

读者可以向管理员发出借阅、归还、续借图书申请。

* + 1. 管理员功能描述

1.图书管理

管理员可以添加修改删除图书。

2.图书审核

管理员可以对读者发出的借阅、续借、归还图书的请求进行审核。

3.图书统计

管理员可以查看已借、已还图书信息，可以查看预约信息和借阅排行榜。

4.个人信息管理

管理员通过个人信息管理完善修改个人信息。

* + 1. 超级管理员功能描述

1.图书类别管理

超级管理员可以添加修改删除图书类别。

2.借书证管理

超级管理员可以对分发及删除借书证。

3.管理员管理

超级管理员可以添加修改删除管理员。

4.图书统计

超级管理员可以查看已借、已还图书信息，可以查看预约信息和借阅排行榜。

5.个人信息管理

管理员通过个人信息管理完善修改个人信息。

* + 1. 主要用例的用例描述

|  |  |
| --- | --- |
| 用例名称 | 图书借阅 |
| 标识符 | UC-1 |
| 参与者 | 读者，管理员 |
| 描述 | 读书可以通过查询等方式获得自己想借阅的图书的名称,编号,等其他可唯一识别的信息,向图书管理员提出借阅请求,管理员在系统中审核相应信息,将图书交给读者,借阅成功。 |
| 前置条件 | 1. 登录；2. 读者有借书证 |
| 后置条件 | 1． 更新图书借阅列表； |
| 主干过程 | 1. 借阅图书 2. 读者请求借阅新的图书 3. 管理员审核并通过读者借阅请求 4. 系统更新借阅列表 |
| 异常 | 1.0.E.1 读者无借书证（第2步）  1．读者没有借书证  2. 系统将错误信息显示在借阅页面  3. 系统重新启动该用例 |
| 被包含用例 | 无 |
| 被扩展用例 | 无 |
| 优先级 | 高 |

|  |  |
| --- | --- |
| 用例名称 | 图书归还 |
| 标识符 | UC-2 |
| 参与者 | 读者，图书管理员 |
| 描述 | 读者将自己借阅的图书归还图书馆。 |
| 前置条件 | 1. 登录；2. 进入图书归还页面 |
| 后置条件 | 1． 更新图书归还列表； |
| 主干过程 | 1. 图书归还 2. 读者向管理员发出归还图书请求 3. 管理员审核并通过读者归还图书申请 4. 系统更图书归还列表 |
| 被包含用例 | 无 |
| 被扩展用例 | 无 |
| 优先级 | 高 |

1. 系统总体设计
   1. 数据库设计
      1. 数据库设计概述

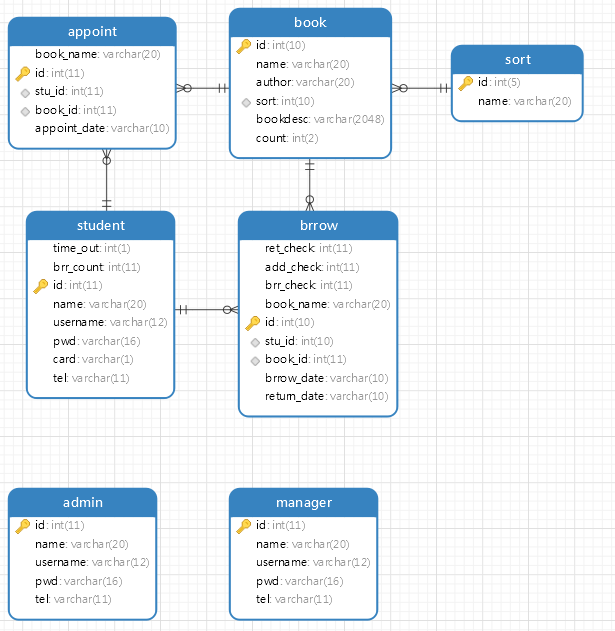
数据库是整个系统的基石,数据库的设计优劣直接影响到整个系统的设计成败,本节对数据库的设计进行专门阐述。

数据库是数据管理的最新技术。十多年来数据库管理系统已从专用的应用程序发展成为通用的系统软件。由于数据库具有数据结构化，最低冗余度，较高的程序与数据独立性，易于扩充，易于编制应用程序等优点，较大的信息系统都是建立在数据库设计之上的。因此不仅大型计算机及中小型计算机，甚至微型机都配有数据库管理系统。

数据库系统的出现使信息系统从以加工数据的程序为中心转向围绕共享的数据库为中心的新阶段。这样既便于数据的集中管理，又有利于应用程序的研制和维护，提高了数据的利用性和相容性，提高了决策的可靠性。目前，数据库已经成为现代信息系统不可分割的重要组成部分。数据库技术也是计算机领域中发展最快的技术之一。

数据库设计是把现实世界的实体模型与需求转换成数据库的模型的过程，它是建立数据库应用系统的核心问题。数据库及其应用的性能都建立在良好的数据库设计的基础之上，数据库的数据是一切操作的基础，如果数据库设计不好，那么其它一切用于提高数据库性能的方法收效都是有限的。数据库设计的关键是如何使设计的数据库能合理地存储用户的数据，方便用户进行数据处理。

在这里使用E-R图描述了图书借阅管理系统的数据模型。图3.1图书借阅管理系统E-R图描述了该系统所涉及到的实体以及他们之间的关系。具体结构如下图所示:



* + 1. 数据库设计信息表

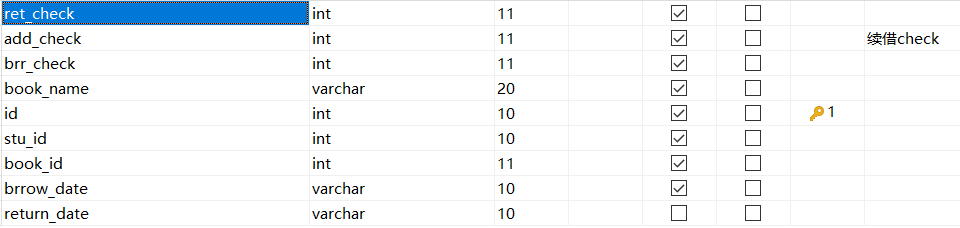
admin表



appoint表 

book表 

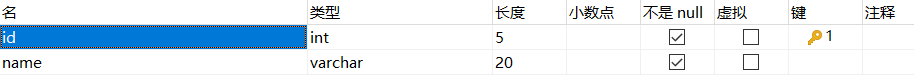
borrow表



manager表



sort表

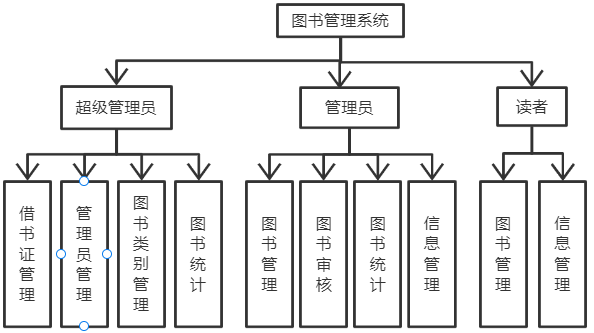


student表



* 1. 系统总体结构设计

图书管理系统主要分为三个系统角色，分别是超级管理员、管理员和学生，其中超级管理员主要对学生借书证、管理员信息、图书类别和图书数据统计进行管理；管理员主要对图书信息、图书借阅、图书信息统计进行管理；学生主要是查询图书、借阅图书和归还图书等功能。各个功能模块下还具有对应的子功能模块，子功能模块为画出。



1. 系统详细设计
   1. 后台设计

后台采用SpringBoot技术，IDEA工具，MySQL8.0数据库。

* + 1. 跨域配置
       1. CrosConfig

package com.dong.library.controller;  
  
import com.dong.library.Service.AdminService;  
import com.dong.library.entity.Admin;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.\*;  
import java.util.List;  
  
@RestController//等于@ResponseBody+@Controller，  
 // 代表这是一个Controller类，并且将返回的数据对象转换为json格式  
@RequestMapping("/admins") //代表对指定的请求路径进行响应,一级路径  
public class AdminController {  
  
 @Autowired //对成员变量、方法和构造函数进行标注，来完成自动装配的工作  
 // 根据类型进行自动装配的。  
 private AdminService adminService;  
  
 */\*\*  
 \* 获取所有管理员  
 \** ***@return*** *\*/* @GetMapping("/findAll") //get请求使用GetMapping  
 public List<Admin> list(){  
 List<Admin> adminList=adminService.findAll();  
 return adminList;  
 }  
  
 */\*\*  
 \* 管理员登录  
 \** ***@param*** *admin  
 \** ***@return*** *\*/* @PostMapping("/findById") //@RequestBody:用于读取http请求的内容(字符串)  
 // 通过springmvc提供的HttpMessageConverter接口将读到的内容（json数据）转换为java对象并绑定到Controller方法的参数上。  
 public Admin findByAdmin(@RequestBody Admin admin){  
 Admin admin1 = adminService.findAdmin(admin);  
 return admin1;  
 }  
  
 */\*\*  
 \* 通过名字查找  
 \** ***@param*** *name  
 \** ***@return*** *\*/* @GetMapping("/findByName/{name}")  
 public Admin findByName(@PathVariable("name") String name ){  
 Admin admin1 = adminService.findbyname(name);  
  
 return admin1;  
 }  
  
 */\*\*  
 \* 通过id查找  
 \** ***@param*** *\** ***@return*** *\*/* @GetMapping("/findById/{id}")  
 public Admin findById(@PathVariable("id") int id ){  
 Admin admin1 = adminService.findbyId(id);  
 return admin1;  
 }  
  
 */\*\*  
 \* 更新员工信息  
 \** ***@param*** *admin  
 \** ***@return*** *\*/* @PutMapping("/update")  
 public String update(@RequestBody Admin admin){  
 String status = adminService.updateAdmin(admin);  
 return status;  
 }  
}

* + - 1. MpConfig

package com.dong.library.config;  
  
import com.baomidou.mybatisplus.extension.plugins.PaginationInterceptor;  
import org.mybatis.spring.annotation.MapperScan;  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.Configuration;  
  
//作用于类上表示这是一个配置类，可理解为用spring的时候xml里面的< beans>标签  
@Configuration  
//com.dong.library.mapper下面的接口类，在编译之后都会生成相应的实现类  
//开启mapper的扫描配置，扫描com.dong.library.mapper目录下的所有Mapper文件作为Mapper接口类  
@MapperScan("com.dong.library.mapper")  
public class MpConfig {  
  
 */\*\*  
 \* 分页插件  
 \** ***@return*** *\*/* @Bean  
 public PaginationInterceptor paginationInterceptor(){  
 return new PaginationInterceptor();  
 }  
}

* + 1. 控制层
       1. AdminController

package com.dong.library.controller;  
  
import com.dong.library.Service.AdminService;  
import com.dong.library.entity.Admin;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.\*;  
import java.util.List;  
  
@RestController//等于@ResponseBody+@Controller，  
 // 代表这是一个Controller类，并且将返回的数据对象转换为json格式  
@RequestMapping("/admins") //代表对指定的请求路径进行响应,一级路径  
public class AdminController {  
  
 @Autowired //对成员变量、方法和构造函数进行标注，来完成自动装配的工作  
 // 根据类型进行自动装配的。  
 private AdminService adminService;  
  
 */\*\*  
 \* 获取所有管理员  
 \** ***@return*** *\*/* @GetMapping("/findAll") //get请求使用GetMapping  
 public List<Admin> list(){  
 List<Admin> adminList=adminService.findAll();  
 return adminList;  
 }  
  
 */\*\*  
 \* 管理员登录  
 \** ***@param*** *admin  
 \** ***@return*** *\*/* @PostMapping("/findById") //@RequestBody:用于读取http请求的内容(字符串)  
 // 通过springmvc提供的HttpMessageConverter接口将读到的内容（json数据）转换为java对象并绑定到Controller方法的参数上。  
 public Admin findByAdmin(@RequestBody Admin admin){  
 Admin admin1 = adminService.findAdmin(admin);  
 return admin1;  
 }  
  
 */\*\*  
 \* 通过名字查找  
 \** ***@param*** *name  
 \** ***@return*** *\*/* @GetMapping("/findByName/{name}")  
 public Admin findByName(@PathVariable("name") String name ){  
 Admin admin1 = adminService.findbyname(name);  
  
 return admin1;  
 }  
  
 */\*\*  
 \* 通过id查找  
 \** ***@param*** *\** ***@return*** *\*/* @GetMapping("/findById/{id}")  
 public Admin findById(@PathVariable("id") int id ){  
 Admin admin1 = adminService.findbyId(id);  
 return admin1;  
 }  
  
 */\*\*  
 \* 更新员工信息  
 \** ***@param*** *admin  
 \** ***@return*** *\*/* @PutMapping("/update")  
 public String update(@RequestBody Admin admin){  
 String status = adminService.updateAdmin(admin);  
 return status;  
 }  
}

* + - 1. AppointController

package com.dong.library.controller;  
  
import com.dong.library.Service.AppointService;  
import com.dong.library.entity.Appoint;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.\*;  
import java.util.List;  
  
@RestController  
@RequestMapping("/appoints")  
public class AppointContrller {  
  
 @Autowired  
 private AppointService appointService;  
  
 */\*\*  
 \* 找到所有预约的书籍  
 \** ***@return*** *\*/* @GetMapping("/findAll")  
 public List<Appoint> list(){  
 List<Appoint> list = appointService.findAll();  
 return list;  
 }  
  
 */\*\*  
 \* 添加appoint数据  
 \** ***@param*** *appoint  
 \** ***@return*** *\*/* @PostMapping("/add")  
 public boolean add(@RequestBody Appoint appoint){  
 Boolean flage =appointService.save(appoint);  
 return flage;  
 }  
}

* + - 1. BookController

package com.dong.library.controller;  
  
import com.baomidou.mybatisplus.extension.plugins.pagination.Page;  
import com.dong.library.Service.BookService;  
import com.dong.library.entity.Book;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.ResponseEntity;  
import org.springframework.web.bind.annotation.\*;  
import java.util.List;  
  
*/\*\*  
 \* controller控制层  
 \*/*@RestController  
@RequestMapping("/books")  
public class BookController {  
  
 @Autowired  
 private BookService bookService;  
  
 */\*\*  
 \* 找到所有书籍  
 \** ***@return*** *\*/* @GetMapping("/findAll")  
 public List<Book> list(){  
 List<Book> bookList = bookService.findAll();  
  
 return bookList;  
 }  
  
 */\*\*  
 \* 分页查询数据  
 \** ***@param*** *page  
 \** ***@param*** *size  
 \** ***@return*** *\*/* @GetMapping("/findAll/{page}/{size}")  
 public Page<Book> listByPage(@PathVariable("page") int page,@PathVariable("size") int size){  
 Page<Book> all = bookService.findByPage(page,size);  
  
 return all;  
 }  
  
 */\*\*  
 \* 根据id获取book  
 \** ***@param*** *id  
 \** ***@return*** *\*/* @GetMapping("/findById/{id}")  
 public Book findById(@PathVariable("id") int id){  
 Book book = bookService.findById(id);  
 return book;  
 }  
  
 */\*\*  
 \* 修改图书信息  
 \** ***@param*** *book  
 \** ***@return*** *\*/* @PutMapping("/update")  
 public String update(@RequestBody Book book){  
 String status = bookService.updateBook(book);  
 return status;  
 }  
  
 */\*\*  
 \* 根据id删除图书  
 \** ***@param*** *id  
 \*/* @DeleteMapping("/delete/{id}") //PathVariable:获取路径参数,即url/{id}这种形式  
 public void delete(@PathVariable("id") int id){  
 bookService.deleteById(id);  
 }  
  
 */\*\*  
 \* 添加图书  
 \** ***@param*** *book  
 \** ***@return*** *\*/* @PostMapping("/add")  
 public boolean add(@RequestBody Book book){  
 Boolean falg = bookService.save(book);  
 return falg;  
 }  
  
 */\*\*  
 \* 通过关键字来查询数据  
 \** ***@param*** *keyword  
 \** ***@return*** *\*/* @GetMapping("/search") //RequestParam:获取查询参数  
 public ResponseEntity<Object> search(@RequestParam("keyword") String keyword){  
 ResponseEntity<Object> responseEntity = bookService.searchByKey(keyword);  
 return responseEntity;  
 }  
  
 */\*\*  
 \* 根据sort查询符合书籍  
 \** ***@param*** *keyword  
 \** ***@return*** *\*/* @GetMapping("/searchBySort")  
 public ResponseEntity<Object> searchBySort(@RequestParam("keyword") String keyword){  
 ResponseEntity<Object> responseEntity = bookService.searchBysort(keyword);  
 return responseEntity;  
 }  
}

* + - 1. BorrowController

package com.dong.library.controller;  
  
import com.dong.library.Service.BrrowService;  
import com.dong.library.entity.Brrow;  
import com.dong.library.entity.Rank;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.ResponseEntity;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.\*;  
  
@RestController  
@RequestMapping("/brrows")  
public class BrrowController {  
  
 @Autowired  
 private BrrowService brrowService;  
  
 */\*\*  
 \* 找到所有借阅图书  
 \** ***@return*** *\*/* @GetMapping("/findAll")  
 public List<Brrow> list(){  
 List<Brrow> brrowList = brrowService.findAll();  
 return brrowList;  
 }  
  
 */\*\*  
 \* 处理图书借阅排行  
 \** ***@return*** *\*/* @GetMapping("/findRank")  
 public List<Rank> brrList(){  
 List<Rank> rankList = brrowService.bookRank();  
 return rankList;  
 }  
  
 */\*\*  
 \* 添加brrow记录  
 \** ***@param*** *brrow  
 \** ***@return*** *\*/* @PostMapping("/add")  
 public boolean add(@RequestBody Brrow brrow){  
 boolean flage = brrowService.save(brrow);  
 return flage;  
 }  
  
 */\*\*  
 \* 通过学生id查看图书  
 \** ***@param*** *keyword  
 \** ***@return*** *\*/* @GetMapping("/searchByStuId")  
 public ResponseEntity<Object> searchByStuId(@RequestParam("keyword") String keyword){  
 ResponseEntity<Object> responseEntity= brrowService.findByStuIdContaining(keyword);  
 return responseEntity;  
 }  
  
 */\*\*  
 \* 通过图书id查找图书  
 \** ***@param*** *keyword  
 \** ***@return*** *\*/* @GetMapping("/searchByBookId")  
 public ResponseEntity<Object> searchByBookId(@RequestParam("keyword") String keyword){  
 ResponseEntity<Object> responseEntity = brrowService.findByBookId(keyword);  
 return responseEntity;  
 }  
  
  
  
 */\*\*  
 \* 借书数据更新  
 \** ***@param*** *brrow  
 \** ***@return*** *\*/* @PutMapping("/update")  
 public String update(@RequestBody Brrow brrow){  
 String status = brrowService.updateBrrow(brrow);  
  
 return status;  
 }  
}

* + - 1. ManagerController

package com.dong.library.controller;  
  
import com.dong.library.Service.ManagerService;  
import com.dong.library.entity.Manager;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.\*;  
import java.util.List;  
  
@RestController  
@RequestMapping("/managers")  
public class ManagerController {  
  
 @Autowired  
 private ManagerService managerService;  
  
 */\*\*  
 \* 找到所有员工  
 \** ***@return*** *\*/* @GetMapping("findAll")  
 public List<Manager> list(){  
 List<Manager> managerList = managerService.findAll();  
 return managerList;  
 }  
  
 */\*\*  
 \* 员工登录  
 \** ***@param*** *manager  
 \** ***@return*** *\*/* @PostMapping("/findById") //RequestBody:  
 public Manager findByManager(@RequestBody Manager manager){  
 Manager manager1 = managerService.findManager(manager);  
 return manager1;  
 }  
  
 //通过id查找员工  
 @GetMapping("findById/{id}")  
 public Manager findManagerById(@PathVariable("id") int id){  
 Manager manager = managerService.FindById(id);  
 return manager;  
 }  
  
 //更新员工信息  
 @PutMapping("/update")  
 public String update(@RequestBody Manager manager){  
 String status= managerService.updateManager(manager);  
 return status;  
 }  
  
 //添加员工  
 @PostMapping("/add")  
 public boolean add(@RequestBody Manager manager){  
 boolean flage = managerService.addManager(manager);  
 return flage;  
 }  
  
 */\*\*  
 \* 删除员工  
 \** ***@param*** *id  
 \*/* @DeleteMapping("/delete/{id}")  
 public void delete(@PathVariable("id") int id){  
 managerService.deleteManager(id);  
 }  
  
}

* + - 1. SortController

package com.dong.library.controller;  
  
import com.dong.library.Service.SortService;  
import com.dong.library.entity.Sort;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.\*;  
import java.util.List;  
  
@RestController  
@RequestMapping("/sorts")  
public class SortController {  
  
 @Autowired  
 private SortService sortService;  
  
 */\*\*  
 \* 找到所有图书分类  
 \** ***@return*** *\*/* @GetMapping("/findAll")  
 public List<Sort> list(){  
 List<Sort> sortList = sortService.findAll();  
 return sortList;  
 }  
  
 */\*\*  
 \* 通过id查找图书分类  
 \** ***@param*** *id  
 \** ***@return*** *\*/* @GetMapping("/findById/{id}")  
 public Sort findById(@PathVariable("id") int id){  
 Sort sort = sortService.findById(id);  
 return sort;  
 }  
  
 */\*\*  
 \* 更新图书分类  
 \** ***@param*** *sort  
 \** ***@return*** *\*/* @PutMapping("/update")  
 public String update(@RequestBody Sort sort){  
 String status = sortService.updateSort(sort);  
 return status;  
 }  
  
 */\*\*  
 \* 通过id删除分类  
 \** ***@param*** *id  
 \*/* @DeleteMapping("delete/{id}")  
 public void delete(@PathVariable("id") int id){  
 sortService.deleteById(id);  
 }  
  
 */\*\*  
 \* 添加分类  
 \** ***@param*** *sort  
 \** ***@return*** *\*/* @PostMapping("/add")  
 public boolean add(@RequestBody Sort sort){  
 boolean flage = sortService.insertSort(sort);  
 return flage;  
 }  
}

* + - 1. StudentController

package com.dong.library.controller;  
  
import com.dong.library.Service.StudentService;  
import com.dong.library.entity.Student;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.ResponseEntity;  
import org.springframework.web.bind.annotation.\*;  
import java.util.List;  
  
@RestController  
@RequestMapping("/students")  
public class StudentController {  
  
 @Autowired  
 private StudentService studentService;  
  
 */\*\*  
 \* 查询所有Student  
 \** ***@return*** *\*/* @GetMapping("/findAll")  
 public List<Student>list(){  
 List<Student> studentList=studentService.findAll();  
 return studentList;  
 }  
  
 */\*\*  
 \* 通过名字去查寻  
 \** ***@param*** *\** ***@return*** *\*/* @PostMapping("/findById")  
 public Student findByName(@RequestBody Student student){  
 Student student1 = studentService.findUser(student);  
 return student1;  
 }  
  
 */\*\*  
 \* 通过id去查找学生信息  
 \** ***@param*** *id  
 \** ***@return*** *\*/* @GetMapping("/findById/{id}")  
 public Student findById(@PathVariable("id") int id){  
 Student student = studentService.findById(id);  
 return student;  
 }  
  
 */\*\*  
 \* 添加学生  
 \** ***@param*** *student  
 \** ***@return*** *\*/* @PostMapping("/add")  
 public String add(@RequestBody Student student){  
 Student save = studentService.savestudent(student);  
 return save.toString();  
 }  
  
 */\*\*  
 \* 修改学生信息  
 \** ***@param*** *student  
 \** ***@return*** *\*/* @PutMapping("/update")  
 public String update(@RequestBody Student student){  
 String status = studentService.updateCard(student); //返回修改成功与否  
 return status;  
 }  
  
 //查询学生借书证的情况  
 @GetMapping("/searchByCard")  
 public ResponseEntity<Object> searchByCard(@RequestParam("keyword") String keyword){  
 ResponseEntity<Object> responseEntity = studentService.findByCardContaining(keyword);  
 return responseEntity;  
 }  
}

* + 1. 实体层
       1. Admin

package com.dong.library.entity;  
  
import com.baomidou.mybatisplus.annotation.IdType;  
import com.baomidou.mybatisplus.annotation.TableId;  
import lombok.Data;  
import javax.persistence.Entity;  
import javax.persistence.Id;  
  
/\*标注Admin是一个实体类\*/  
@Entity  
@Data  
public class Admin {  
 @Id  
 @TableId(type= IdType.*AUTO*)  
 private Integer id;  
 private String name;  
 private String username;  
 private String pwd;  
 private String tel;  
  
 public Integer getId() {  
 return id;  
 }  
  
 public void setId(Integer id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getUsername() {  
 return username;  
 }  
  
 public void setUsername(String username) {  
 this.username = username;  
 }  
  
 public String getPwd() {  
 return pwd;  
 }  
  
 public void setPwd(String pwd) {  
 this.pwd = pwd;  
 }  
  
 public String getTel() {  
 return tel;  
 }  
  
 public void setTel(String tel) {  
 this.tel = tel;  
 }  
  
 @Override  
 public String toString() {  
 return "Admin{" +  
 "id=" + id +  
 ", name='" + name + '\'' +  
 ", username='" + username + '\'' +  
 ", pwd='" + pwd + '\'' +  
 ", tel='" + tel + '\'' +  
 '}';  
 }  
}

* + - 1. Appoint

package com.dong.library.entity;  
  
import com.baomidou.mybatisplus.annotation.IdType;  
import com.baomidou.mybatisplus.annotation.TableId;  
import lombok.Data;  
import javax.persistence.Entity;  
import javax.persistence.Id;  
  
//预约图书实体  
@Entity //定义Appoint为实体类  
@Data // 自动生成各种各样的getter和setter方法  
public class Appoint {  
 @Id//绑定Appoint表里面的id  
 @TableId(type = IdType.*AUTO*)  
 private Integer id;  
 private String stuId;  
 private String bookId;  
 private String appointDate;  
 private String bookName;  
  
 public Integer getId() {  
 return id;  
 }  
  
 public void setId(Integer id) {  
 this.id = id;  
 }  
  
 public String getStuId() {  
 return stuId;  
 }  
  
 public void setStuId(String stuId) {  
 this.stuId = stuId;  
 }  
  
 public String getBookId() {  
 return bookId;  
 }  
  
 public void setBookId(String bookId) {  
 this.bookId = bookId;  
 }  
  
 public String getAppointDate() {  
 return appointDate;  
 }  
  
 public void setAppointDate(String appointDate) {  
 this.appointDate = appointDate;  
 }  
  
 public String getBookName() {  
 return bookName;  
 }  
  
 public void setBookName(String bookName) {  
 this.bookName = bookName;  
 }  
  
 @Override  
 public String toString() {  
 return "Appoint{" +  
 "id=" + id +  
 ", stuId='" + stuId + '\'' +  
 ", bookId='" + bookId + '\'' +  
 ", appointDate='" + appointDate + '\'' +  
 ", bookName='" + bookName + '\'' +  
 '}';  
 }  
}

* + - 1. Book

package com.dong.library.entity;  
  
import com.baomidou.mybatisplus.annotation.IdType;  
import com.baomidou.mybatisplus.annotation.TableField;  
import com.baomidou.mybatisplus.annotation.TableId;  
import lombok.Data;  
import javax.persistence.\*;  
  
@Entity  
@Data  
public class Book {  
 @Id  
 @TableId(type = IdType.*AUTO*)  
 private Integer id;  
 private String name;  
 private String author;  
 private String sort;  
 private String bookdesc;  
 private Integer count;  
  
 //非数据库字段  
 @TableField(exist = false)  
 private String typename;  
  
 public Integer getId() {  
 return id;  
 }  
  
 public void setId(Integer id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getAuthor() {  
 return author;  
 }  
  
 public void setAuthor(String author) {  
 this.author = author;  
 }  
  
 public String getSort() {  
 return sort;  
 }  
  
 public void setSort(String sort) {  
 this.sort = sort;  
 }  
  
 public String getBookdesc() {  
 return bookdesc;  
 }  
  
 public void setBookdesc(String bookdesc) {  
 this.bookdesc = bookdesc;  
 }  
  
 public Integer getCount() {  
 return count;  
 }  
  
 public void setCount(Integer count) {  
 this.count = count;  
 }  
  
 public String getTypename() {  
 return typename;  
 }  
  
 public void setTypename(String typename) {  
 this.typename = typename;  
 }  
  
 @Override  
 public String toString() {  
 return "Book{" +  
 "id=" + id +  
 ", name='" + name + '\'' +  
 ", author='" + author + '\'' +  
 ", sort='" + sort + '\'' +  
 ", bookdesc='" + bookdesc + '\'' +  
 ", count=" + count +  
 ", typename='" + typename + '\'' +  
 '}';  
 }  
}

* + - 1. Borrow

package com.dong.library.entity;  
  
import com.baomidou.mybatisplus.annotation.IdType;  
import com.baomidou.mybatisplus.annotation.TableId;  
import lombok.Data;  
import javax.persistence.Entity;  
import javax.persistence.Id;  
  
@Entity  
@Data  
public class Brrow {  
 @Id  
 @TableId(type= IdType.AUTO)  
 private Integer id;  
 private String stuId;  
 private String bookId;  
 private String brrowDate;  
 private String returnDate;  
 private String bookName;  
// 借阅，还书，续借审核 0 默认 1 提交申请 2 同意申请  
 private Integer brrCheck;  
 private Integer retCheck;  
 private Integer addCheck;  
  
 public Integer getId() {  
 return id;  
 }  
  
 public void setId(Integer id) {  
 this.id = id;  
 }  
  
 public String getStuId() {  
 return stuId;  
 }  
  
 public void setStuId(String stuId) {  
 this.stuId = stuId;  
 }  
  
 public String getBookId() {  
 return bookId;  
 }  
  
 public void setBookId(String bookId) {  
 this.bookId = bookId;  
 }  
  
 public String getBrrowDate() {  
 return brrowDate;  
 }  
  
 public void setBrrowDate(String brrowDate) {  
 this.brrowDate = brrowDate;  
 }  
  
 public String getReturnDate() {  
 return returnDate;  
 }  
  
 public void setReturnDate(String returnDate) {  
 this.returnDate = returnDate;  
 }  
  
 public String getBookName() {  
 return bookName;  
 }  
  
 public void setBookName(String bookName) {  
 this.bookName = bookName;  
 }  
  
 public Integer getBrrCheck() {  
 return brrCheck;  
 }  
  
 public void setBrrCheck(Integer brrCheck) {  
 this.brrCheck = brrCheck;  
 }  
  
 public Integer getRetCheck() {  
 return retCheck;  
 }  
  
 public void setRetCheck(Integer retCheck) {  
 this.retCheck = retCheck;  
 }  
  
 public Integer getAddCheck() {  
 return addCheck;  
 }  
  
 public void setAddCheck(Integer addCheck) {  
 this.addCheck = addCheck;  
 }  
  
 @Override  
 public String toString() {  
 return "Brrow{" +  
 "id=" + id +  
 ", stuId='" + stuId + '\'' +  
 ", bookId='" + bookId + '\'' +  
 ", brrowDate='" + brrowDate + '\'' +  
 ", returnDate='" + returnDate + '\'' +  
 ", bookName='" + bookName + '\'' +  
 ", brrCheck=" + brrCheck +  
 ", retCheck=" + retCheck +  
 ", addCheck=" + addCheck +  
 '}';  
 }  
}

* + - 1. Manager

package com.dong.library.entity;  
  
import com.baomidou.mybatisplus.annotation.IdType;  
import com.baomidou.mybatisplus.annotation.TableId;  
import lombok.Data;  
import javax.persistence.Entity;  
import javax.persistence.Id;  
  
@Entity  
@Data  
public class Manager {  
 @Id  
 @TableId(type = IdType.AUTO)  
 private Integer id;  
 private String name;  
 private String username;  
 private String pwd;  
 private String tel;  
  
 public Integer getId() {  
 return id;  
 }  
  
 public void setId(Integer id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getUsername() {  
 return username;  
 }  
  
 public void setUsername(String username) {  
 this.username = username;  
 }  
  
 public String getPwd() {  
 return pwd;  
 }  
  
 public void setPwd(String pwd) {  
 this.pwd = pwd;  
 }  
  
 public String getTel() {  
 return tel;  
 }  
  
 public void setTel(String tel) {  
 this.tel = tel;  
 }  
  
 @Override  
 public String toString() {  
 return "Manager{" +  
 "id=" + id +  
 ", name='" + name + '\'' +  
 ", username='" + username + '\'' +  
 ", pwd='" + pwd + '\'' +  
 ", tel='" + tel + '\'' +  
 '}';  
 }  
}

* + - 1. Rank

package com.dong.library.entity;  
  
import lombok.Data;  
  
*/\*\*  
 \* 用来统计借阅排行榜  
 \*/*@Data  
public class Rank {  
  
 private String id;  
 private String name;  
 private Integer brrCount;  
  
 public String getId() {  
 return id;  
 }  
  
 public void setId(String id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public Integer getBrrCount() {  
 return brrCount;  
 }  
  
 public void setBrrCount(Integer brrCount) {  
 this.brrCount = brrCount;  
 }  
  
 @Override  
 public String toString() {  
 return "Rank{" +  
 "id='" + id + '\'' +  
 ", name='" + name + '\'' +  
 ", brrCount=" + brrCount +  
 '}';  
 }  
}

* + - 1. Sort

package com.dong.library.entity;  
  
import com.baomidou.mybatisplus.annotation.IdType;  
import com.baomidou.mybatisplus.annotation.TableId;  
import lombok.Data;  
import javax.persistence.Entity;  
import javax.persistence.Id;  
  
@Entity  
@Data  
public class Sort {  
 @Id  
 @TableId(type = IdType.AUTO)  
 private Integer id;  
 private String name;  
  
 public Integer getId() {  
 return id;  
 }  
  
 public void setId(Integer id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 @Override  
 public String toString() {  
 return "Sort{" +  
 "id=" + id +  
 ", name='" + name + '\'' +  
 '}';  
 }  
}

* + - 1. Student

package com.dong.library.entity;  
  
import com.baomidou.mybatisplus.annotation.IdType;  
import com.baomidou.mybatisplus.annotation.TableId;  
import lombok.Data;  
import javax.persistence.Entity;  
import javax.persistence.Id;  
  
@Entity  
@Data  
public class Student {  
 @Id  
 @TableId(type = IdType.AUTO)  
 private Integer id;  
 private String name;  
 private String username;  
 private String pwd;  
 private String card;  
 private String tel;  
 private Integer brrCount;  
 private Integer timeOut;  
  
 public Integer getId() {  
 return id;  
 }  
  
 public void setId(Integer id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getUsername() {  
 return username;  
 }  
  
 public void setUsername(String username) {  
 this.username = username;  
 }  
  
 public String getPwd() {  
 return pwd;  
 }  
  
 public void setPwd(String pwd) {  
 this.pwd = pwd;  
 }  
  
 public String getCard() {  
 return card;  
 }  
  
 public void setCard(String card) {  
 this.card = card;  
 }  
  
 public String getTel() {  
 return tel;  
 }  
  
 public void setTel(String tel) {  
 this.tel = tel;  
 }  
  
 public Integer getBrrCount() {  
 return brrCount;  
 }  
  
 public void setBrrCount(Integer brrCount) {  
 this.brrCount = brrCount;  
 }  
  
 public Integer getTimeOut() {  
 return timeOut;  
 }  
  
 public void setTimeOut(Integer timeOut) {  
 this.timeOut = timeOut;  
 }  
  
 @Override  
 public String toString() {  
 return id.toString()+" , "+username;  
 }  
}

* + 1. 持久层
       1. Adminmapper

package com.dong.library.mapper;

import com.baomidou.mybatisplus.core.mapper.BaseMapper;

import com.dong.library.entity.Admin;

import org.springframework.stereotype.Repository;

/\*\*

\*

\* @repository则需要在Spring中配置扫描包地址，然后生成dao层的bean，

\* 之后被注入到ServiceImpl中。

\*/

@Repository //用在持久层的接口上，这个注解是将接口的一个实现类交给spring管理。

public interface Adminmapper extends BaseMapper<Admin> {

}

* + - 1. AppointMapper

package com.dong.library.mapper;

import com.baomidou.mybatisplus.core.mapper.BaseMapper;

import com.dong.library.entity.Appoint;

import org.springframework.stereotype.Repository;

@Repository//用在持久层的接口上，这个注解是将接口的一个实现类交给spring管理。

public interface AppointMapper extends BaseMapper<Appoint> {

}

* + - 1. BookMapper

package com.dong.library.mapper;  
  
import com.baomidou.mybatisplus.core.mapper.BaseMapper;  
import com.dong.library.entity.Book;  
import org.springframework.stereotype.Repository;  
  
@Repository//用在持久层的接口上，这个注解是将接口的一个实现类交给spring管理。  
public interface BookMapper extends BaseMapper<Book> {  
}

* + - 1. BorrowMapper

package com.dong.library.mapper;  
import com.baomidou.mybatisplus.core.mapper.BaseMapper;  
import com.dong.library.entity.Brrow;  
import org.springframework.stereotype.Repository;  
  
@Repository//用在持久层的接口上，这个注解是将接口的一个实现类交给spring管理。  
public interface Brrowmapper extends BaseMapper<Brrow> {  
}

* + - 1. ManagerMapper

package com.dong.library.mapper;  
  
import com.baomidou.mybatisplus.core.mapper.BaseMapper;  
import com.dong.library.entity.Manager;  
import org.springframework.stereotype.Repository;  
  
@Repository//用在持久层的接口上，这个注解是将接口的一个实现类交给spring管理。  
public interface Managermapper extends BaseMapper<Manager> {  
}

* + - 1. SortMapper

package com.dong.library.mapper;  
  
import com.baomidou.mybatisplus.core.mapper.BaseMapper;  
import com.dong.library.entity.Sort;  
import org.springframework.stereotype.Repository;  
  
@Repository//用在持久层的接口上，这个注解是将接口的一个实现类交给spring管理。  
public interface Sortmapper extends BaseMapper<Sort> {  
}

* + - 1. StudentMapper

package com.dong.library.mapper;  
  
import com.baomidou.mybatisplus.core.mapper.BaseMapper;  
import com.dong.library.entity.Student;  
import org.springframework.stereotype.Repository;  
  
@Repository//用在持久层的接口上，这个注解是将接口的一个实现类交给spring管理。  
public interface Studentmapper extends BaseMapper<Student> {  
}

* + 1. 服务层
       1. AdminService

package com.dong.library.Service;  
  
import com.baomidou.mybatisplus.core.conditions.query.QueryWrapper;  
import com.dong.library.entity.Admin;  
import com.dong.library.mapper.Adminmapper;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
import org.springframework.transaction.annotation.Transactional;  
import java.util.List;  
  
@Service  
public class AdminService {  
  
 @Autowired//对成员变量、方法和构造函数进行标注，来完成自动装配的工作  
 // 根据类型进行自动装配的。  
 private Adminmapper adminmapper;  
  
 public Admin findAdmin(Admin admin) {  
 Admin admin1 = adminmapper.selectOne(new QueryWrapper<Admin>()  
 .eq("username",admin.getUsername())  
 .eq("pwd",admin.getPwd()));  
 if(admin1 !=null){  
 return admin1;  
 }  
 return new Admin();  
 }  
  
 */\*\*  
 \* 通过名字去查找  
 \** ***@param*** *name  
 \** ***@return*** *\* QueryWrapper：条件构造器  
 \*/* public Admin findbyname(String name) {  
 Admin name1 = adminmapper.selectOne(new QueryWrapper<Admin>().eq("name", name));  
 return name1;  
 }  
  
 */\*\*  
 \* 更新系统管理员信息  
 \** ***@param*** *admin  
 \** ***@return*** *\*/* @Transactional  
 public String updateAdmin(Admin admin) {  
 int i = adminmapper.update(admin,new QueryWrapper<Admin>().eq("id",admin.getId()));  
 if(i != 0){  
 return "success";  
 }  
 return "error";  
 }  
  
 */\*\*  
 \* 通过id去查找  
 \** ***@param*** *id  
 \** ***@return*** *\*/* public Admin findbyId(int id) {  
 Admin admin = adminmapper.selectById(id);  
 return admin;  
 }  
  
 */\*\*  
 \* 获取所有管理员信息  
 \** ***@return*** *\*/* public List<Admin> findAll() {  
 List<Admin> admins = adminmapper.selectList(null);  
 return admins;  
 }  
}

* + - 1. AppointService

package com.dong.library.Service;  
  
  
import com.dong.library.entity.Appoint;  
import com.dong.library.mapper.AppointMapper;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
import java.util.List;  
  
@Service  
public class AppointService {  
 @Autowired  
 public AppointMapper appointMapper;  
  
 */\*\*  
 \* 找到所有预约的书籍  
 \** ***@return*** *\*/* public List<Appoint> findAll() {  
 List<Appoint> list = appointMapper.selectList(null);  
 return list;  
 }  
  
 */\*\*  
 \* 添加appoint数据  
 \** ***@param*** *appoint  
 \** ***@return*** *\*/* public Boolean save(Appoint appoint) {  
 int insert = appointMapper.insert(appoint);  
 if(insert != -1){  
 return true;  
 }  
 return false;  
 }  
}

* + - 1. BookService

package com.dong.library.Service;  
  
import com.baomidou.mybatisplus.core.conditions.query.QueryWrapper;  
import com.baomidou.mybatisplus.extension.plugins.pagination.Page;  
import com.dong.library.entity.Book;  
import com.dong.library.mapper.BookMapper;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.HttpStatus;  
import org.springframework.http.ResponseEntity;  
import org.springframework.stereotype.Service;  
import org.springframework.transaction.annotation.Transactional;  
  
import java.util.List;  
  
@Service  
public class BookService {  
  
 @Autowired  
 private BookMapper bookMapper;  
  
 */\*\*  
 \* 通过分页查找图书  
 \** ***@param*** *page  
 \** ***@param*** *size  
 \** ***@return*** *\*/* public Page<Book> findByPage(int page, int size) {  
 Page<Book> page1 = new Page<>(page, size);  
 Page<Book> bookPage = bookMapper.selectPage(page1, null);  
 return bookPage;  
 }  
  
 */\*\*  
 \* 通过关键字来查询数据  
 \** ***@param*** *keyword  
 \** ***@return*** *\* 逻辑判断使用Wrapper来处理（Map只能做等值判断）  
 \*/* public ResponseEntity<Object> searchByKey(String keyword) {  
 List<Book> books = bookMapper.selectList(new QueryWrapper<Book>()  
 .like("id", keyword).or()  
 .like("name", keyword).or()  
 .like("author", keyword));  
 if (books.isEmpty()) {  
 return new ResponseEntity<>(null, HttpStatus.*NO\_CONTENT*);  
 }  
 return new ResponseEntity<>(books, HttpStatus.OK);  
 }  
  
 */\*\*  
 \* 根据sort查询符合书籍  
 \** ***@param*** *keyword  
 \** ***@return*** *\*/* public ResponseEntity<Object> searchBysort(String keyword) {  
 List<Book> sort = bookMapper.selectList(new QueryWrapper<Book>().eq("sort", keyword));  
 if (sort.isEmpty()) {  
 return new ResponseEntity<>(null, HttpStatus.NO\_CONTENT);  
 }  
 return new ResponseEntity<>(sort, HttpStatus.OK);  
 }  
  
 */\*\*  
 \* 找到所有图书  
 \** ***@return*** *\*/* public List<Book> findAll() {  
 return bookMapper.selectList(null);  
 }  
  
 */\*\*  
 \* 根据id获取book  
 \** ***@param*** *id  
 \** ***@return*** *\*/* public Book findById(int id) {  
 Book book = bookMapper.selectById(id);  
 if (book != null) {  
 return book;  
 }  
 return new Book();  
 }  
  
 */\*\*  
 \* 员工修改图书信息  
 \** ***@param*** *book  
 \** ***@return*** *\*/* @Transactional  
 public String updateBook(Book book) {  
 int i = bookMapper.updateById(book);  
 if (i != 0) {  
 return "success";  
 }  
 return "error";  
 }  
  
 */\*\*  
 \* 根据id删除图书  
 \** ***@param*** *id  
 \*/* @Transactional  
 public void deleteById(int id) {  
 bookMapper.deleteById(id);  
 }  
  
 */\*\*  
 \* 添加图书  
 \** ***@param*** *book  
 \** ***@return*** *\*/* @Transactional  
 public Boolean save(Book book) {  
 int insert = bookMapper.insert(book);  
 if(insert != -1){  
 return true;  
 }  
 return false;  
 }  
}

* + - 1. BorrowService

package com.dong.library.Service;  
  
import com.baomidou.mybatisplus.core.conditions.query.QueryWrapper;  
import com.dong.library.entity.Brrow;  
import com.dong.library.entity.Rank;  
import com.dong.library.mapper.Brrowmapper;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.HttpStatus;  
import org.springframework.http.ResponseEntity;  
import org.springframework.stereotype.Service;  
import org.springframework.transaction.annotation.Transactional;  
import java.util.ArrayList;  
import java.util.Comparator;  
import java.util.List;  
  
@Service  
public class BrrowService {  
  
 @Autowired  
 private Brrowmapper brrowmapper;  
  
 */\*\*  
 \* 找到所有的图书  
 \** ***@return*** *\*/* public List<Brrow> findAll() {  
 List<Brrow> brrowList = brrowmapper.selectList(null);  
 return brrowList;  
 }  
  
 */\*\*  
 \* 处理图书借阅排行  
 \** ***@return*** *\*/* public List<Rank> bookRank() {  
 List<Brrow> brrowList = findAll();  
 List<Rank> rankList = new ArrayList<>();  
 int length = brrowList.size();  
 int t=length-1;  
 int i,j;  
 for ( i = 0; i < t-1; i++) {  
 int count = 1;  
 Rank rank = new Rank();  
 for (j = t; j > i; j--) {  
 if (brrowList.get(i).getBookId().equals(brrowList.get(j).getBookId()) ) {  
 count++;  
 brrowList.remove(j);  
 t--;  
 }  
 }  
 rank.setBrrCount(count);  
 rank.setId(brrowList.get(i).getBookId());  
 rank.setName(brrowList.get(i).getBookName());  
 rankList.add(rank);  
 }  
 //List.sort()  
 //进行排序  
 rankList.sort(Comparator.*comparing*(Rank::getBrrCount).reversed());  
 return rankList;  
 }  
  
 */\*\*  
 \* 通过学生id查询图书  
 \** ***@param*** *keyword  
 \** ***@return*** *\*/* public ResponseEntity<Object> findByStuIdContaining(String keyword) {  
 List<Brrow> brrowList = brrowmapper.selectList(new QueryWrapper<Brrow>().eq("stu\_id", keyword));  
 if(brrowList.isEmpty()){  
 return new ResponseEntity<>(null, HttpStatus.*NO\_CONTENT*);  
 }  
 return new ResponseEntity<>(brrowList, HttpStatus.*OK*);  
 }  
  
 */\*\*  
 \* 通过图书id查找图书  
 \** ***@param*** *keyword  
 \** ***@return*** *\*/* public ResponseEntity<Object> findByBookId(String keyword) {  
 List<Brrow> brrowList = brrowmapper.selectList(new QueryWrapper<Brrow>().eq("book\_id", keyword));  
 if(brrowList.isEmpty()){  
 return new ResponseEntity<>(null, HttpStatus.*NO\_CONTENT*);  
 }  
 return new ResponseEntity<>(brrowList, HttpStatus.*OK*);  
 }  
  
 */\*\*  
 \* 保存借书记录  
 \** ***@param*** *brrow  
 \** ***@return*** *\*/* @Transactional  
 public boolean save(Brrow brrow) {  
 int insert = brrowmapper.insert(brrow);  
 if(insert !=-1){  
 return true;  
 }  
 return false;  
 }  
  
 */\*\*  
 \* 借书数据更新  
 \** ***@param*** *brrow  
 \** ***@return*** *\*/* @Transactional  
 public String updateBrrow(Brrow brrow) {  
 int i = brrowmapper.updateById(brrow);  
 if(i != 0){  
 return "success";  
 }  
 return "error";  
 }  
}

* + - 1. ManagerService

package com.dong.library.Service;  
  
  
import com.baomidou.mybatisplus.core.conditions.query.QueryWrapper;  
import com.dong.library.entity.Manager;  
import com.dong.library.mapper.Managermapper;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
import java.util.List;  
  
@Service  
public class ManagerService {  
  
 @Autowired  
 //导入 Mapper 下的 Managermapper 数据库接口  
 private Managermapper managermapper;  
  
 */\*\*  
 \* 员工登录  
 \** ***@param*** *manager  
 \** ***@return*** *\*/* public Manager findManager(Manager manager) {  
 Manager manager1 = managermapper.selectOne(new QueryWrapper<Manager>()  
 .eq("username",manager.getUsername())  
 .eq("pwd",manager.getPwd()));  
  
 if(manager1 != null){  
 return manager1;  
 }  
 return new Manager();  
 }  
  
 */\*\*  
 \* 根据id删除员工  
 \** ***@param*** *id  
 \*/* public void deleteManager(int id) {  
 managermapper.deleteById(id);  
 }  
  
 */\*\*  
 \* 通过id查找员工  
 \** ***@param*** *id  
 \** ***@return*** *\*/* public Manager FindById(int id) {  
 Manager manager = managermapper.selectById(id);  
 return manager;  
 }  
  
 */\*\*  
 \* 更新员工信息  
 \** ***@param*** *manager  
 \** ***@return*** *\*/* public String updateManager(Manager manager) {  
 int i = managermapper.update(manager, new QueryWrapper<Manager>().eq("id", manager.getId()));  
 if(i != 0){  
 return "success";  
 }  
 return "error";  
 }  
  
 */\*\*  
 \* 添加员工操作  
 \** ***@param*** *manager  
 \** ***@return*** *\*/* public boolean addManager(Manager manager) {  
 int insert = managermapper.insert(manager); //调用API接口实现插入数据操作  
  
 if(insert !=-1){  
 return true;  
 }  
 return false;  
 }  
  
 */\*\*  
 \* 找到所有员工  
 \** ***@return*** *\*/* public List<Manager> findAll() {  
 return managermapper.selectList(null);  
 }  
}

* + - 1. SortService

package com.dong.library.Service;  
  
import com.baomidou.mybatisplus.core.conditions.query.QueryWrapper;  
import com.dong.library.entity.Sort;  
import com.dong.library.mapper.Sortmapper;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
import org.springframework.transaction.annotation.Transactional;  
import java.util.List;  
  
@Service  
public class SortService {  
 @Autowired  
 private Sortmapper sortmapper;  
  
 */\*\*  
 \* 找到所有图书分类  
 \** ***@return*** *\*/* public List<Sort> findAll() {  
 return sortmapper.selectList(null);  
 }  
  
 */\*\*  
 \* 通过id删除图书  
 \** ***@param*** *id  
 \*/* @Transactional  
 public void deleteById(int id) {  
 sortmapper.deleteById(id);  
 }  
  
 */\*\*  
 \* 通过id查找图书分类  
 \** ***@param*** *id  
 \** ***@return*** *\*/* public Sort findById(int id) {  
 Sort sort = sortmapper.selectById(id);  
 if(sort !=null){  
 return sort;  
 }  
 return new Sort();  
 }  
  
 */\*\*  
 \* 更新图书分类  
 \** ***@return*** *\*/* @Transactional  
 public String updateSort(Sort sort) {  
 int i = sortmapper.update(sort, new QueryWrapper<Sort>().eq("id", sort.getId()));  
 if(i != 0){  
 return "success";  
 }  
 return "error";  
 }  
  
 */\*\*  
 \* 新增图书分类  
 \** ***@param*** *sort  
 \** ***@return*** *\*/* public boolean insertSort(Sort sort) {  
 int insert = sortmapper.insert(sort);  
 if(insert != -1){  
 return true;  
 }  
 return false;  
 }  
}

* + - 1. StudentService

package com.dong.library.Service;  
  
import com.baomidou.mybatisplus.core.conditions.query.QueryWrapper;  
import com.dong.library.entity.Student;  
import com.dong.library.mapper.Studentmapper;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.HttpStatus;  
import org.springframework.http.ResponseEntity;  
import org.springframework.stereotype.Service;  
import org.springframework.transaction.annotation.Transactional;  
import java.util.List;  
  
@Service  
public class StudentService {  
  
 @Autowired  
 private Studentmapper studentmapper;  
  
 //登录时查看用户是否在表中存在,eq等值判断  
 public Student findUser(Student student) {  
 Student student1 = studentmapper.selectOne(new QueryWrapper<Student>()  
 .eq("username",student.getUsername())  
 .eq("pwd",student.getPwd()));  
 if(student1 !=null){  
 return student1;  
 }  
 return new Student();  
 }  
  
 //保存用户注册信息  
 @Transactional  
 public Student savestudent(Student student) {  
 int insert = studentmapper.insert(student);  
 if(insert != -1){  
 return student;  
 }  
 return new Student();  
 }  
  
 //查看借书卡的学生的情况  
 public ResponseEntity<Object> findByCardContaining(String keyword) {  
  
 List<Student> studentList = studentmapper.selectList(new QueryWrapper<Student>().eq("card",keyword));  
 if(studentList.isEmpty()){  
 return new ResponseEntity<>(null, HttpStatus.*NO\_CONTENT*);  
 }  
 return new ResponseEntity<>(studentList, HttpStatus.*OK*);  
  
 }  
  
 //更新学生的借书卡  
 @Transactional  
 public String updateCard(Student student) {  
 int i = studentmapper.updateById(student);  
 if(i != 0){  
 return "success";  
 }  
 return "error";  
 }  
  
 */\*\*  
 \* 通过id去查找学生信息  
 \** ***@return*** *\*/* public Student findById(int id) {  
 Student student = studentmapper.selectById(id);  
 return student;  
 }  
  
 */\*\*  
 \* 查找所有学生  
 \** ***@return*** *\* queryWrapper:条件查询  
 \*/* public List<Student> findAll() {  
 return studentmapper.selectList(null);  
 }  
}

* 1. 前台设计

前台采用Vue技术，IDEA工具。(具体页面设计省略)

* + 1. global.js

export default{  
 stuCount:"",  
 stuTimeOut:"",  
 stuName:"",  
 stuCard:"",  
 stuBrrCount:0,  
 manCount:"",  
 manName:"",  
 admCount:"",  
 admName:"",  
   
 setStuBrrCount(stuBrrCount){  
 this.stuBrrCount=stuBrrCount;  
 },  
 setStuTimeOut(stuTimeOut){  
 this.stuTimeOut=stuTimeOut;  
 },  
 setStuCard(stuCard){  
 this.stuCard=stuCard;  
 },  
 setStuCount(stuCount){  
 this.stuCount=stuCount;  
 },setManCount(manCount){  
 this.manCount=manCount;  
 },  
 setAdmCount(admCount){  
 this.admCount=admCount;  
 },  
   
 setStuName(stuName){  
 this.stuName=stuName;  
 },  
 setManName(manName){  
 this.manName=manName;  
  
 },setAdmName(admName){  
 this.admName=admName;  
 }  
}

* + 1. index.js

import Vue from 'vue'  
  
import Router from 'vue-router'  
import stuLog from "../views/stuLog.vue"  
import stuRegistry from "../views/stuRegistry.vue"  
import student from "../views/student.vue"  
import stuBookLook from "../views/stuBookLook.vue"  
import stuBookBorrow from "../views/stuBookBorrow.vue"  
import stuBookQuery from "../views/stuBookQuery.vue"  
import stuBookSort from "../views/stuBookSort.vue"  
import stuInfo from "../views/stuInfo.vue"  
import stuBrrHistory from "../views/stuBrrHistory.vue"  
  
import manager from "../views/manager.vue"  
  
import manBookLook from "../views/manBookLook.vue"  
import manAddBook from "../views/manAddBook.vue"  
import manUpdateBook from "../views/manUpdateBook.vue"  
import manBrrCheck from "../views/manBrrCheck.vue"  
import manAddCheck from "../views/manAddCheck.vue"  
import manRetCheck from "../views/manRetCheck.vue"  
import manBrrInfo from "../views/manBrrInfo.vue"  
import manRetInfo from "../views/manRetInfo.vue"  
import manBrrTotalInfo from "../views/manBrrTotalInfo.vue"  
import manAppointInfo from "../views/manAppointInfo.vue"  
import manInfo from "../views/manInfo.vue"  
  
import admin from "../views/admin.vue"  
import admAddCard from "../views/admAddCard.vue"  
import admDeleteCard from "../views/admDeleteCard.vue"  
import admManLook from "../views/admManLook.vue"  
import admAddMan from "../views/admAddMan.vue"  
import admUpdateMan from "../views/admUpdateMan.vue"  
import admSortLook from "../views/admSortLook.vue"  
import admAddSort from "../views/admAddSort.vue"  
import admUpdateSort from "../views/admUpdateSort.vue"  
import admBrrInfo from "../views/admBrrInfo.vue"  
import admRetInfo from "../views/admRetInfo.vue"  
import admTotalBrrInfo from "../views/admTotalBrrInfo.vue"  
import admInfo from "../views/admInfo.vue"  
  
Vue.use(Router)  
  
export default new Router({  
 mode: 'hash',  
 routes: [  
 {  
 path:'',  
 redirect:'/stuLog'  
 },  
 //登录路由  
 {  
 path: '/stuLog',  
 name: 'stuLog',  
 component: stuLog  
 },  
 //注册路由  
 {  
 path: '/stuRegistry',  
 name: 'stuRegistry',  
 component: stuRegistry  
 },  
  
 //学生路由  
 {  
 path: '/student',  
 name: 'student',  
 redirect: '/stuBookLook',  
 component: student,  
 children: [  
 {  
 path: '/stuBookLook',  
 name: 'stuBookLook',  
 show: true,  
 component: stuBookLook  
 },  
 {  
 path: '/stuInfo',  
 name: 'stuInfo',  
 component: stuInfo  
 },  
  
 {  
 path: '/stuBookBorrow',  
 name: 'stuBookBorrow',  
 component: stuBookBorrow  
 },  
 {  
 path: '/stuBrrHistory',  
 name: 'stuBrrHistory',  
 component: stuBrrHistory  
 },  
 {  
 path: '/stuBookSort',  
 name: 'stuBookSort',  
 component: stuBookSort  
 }  
 ,  
 {  
 path: '/stuBookQuery',  
 name: 'stuBookQuery',  
 component: stuBookQuery  
 }  
 ]  
  
 },  
 //员工路由  
 {  
 path: '/manager',  
 name: 'manager',  
 component: manager,  
 redirect: '/manBookLook',  
 children: [  
  
 {  
 path: '/manBookLook',  
 name: 'manBookLook',  
 component: manBookLook  
 },  
 {  
 path: '/manAddBook',  
 name: 'manAddBook',  
 component: manAddBook  
 },  
 {  
 path: '/manUpdateBook',  
 name: 'manUpdateBook',  
 component: manUpdateBook  
 },  
 {  
 path: '/manBrrCheck',  
 name: 'manBrrCheck',  
 component: manBrrCheck  
 },  
 {  
 path: '/manAddCheck',  
 name: 'manAddCheck',  
 component: manAddCheck  
 },  
 {  
 path: '/manRetCheck',  
 name: 'manRetCheck',  
 component: manRetCheck  
 },  
 {  
 path: '/manBrrInfo',  
 name: 'manBrrInfo',  
 component: manBrrInfo  
 },  
 {  
 path: '/manAppointInfo',  
 name: 'manAppointInfo',  
 component: manAppointInfo  
 },  
 {  
 path: '/manRetInfo',  
 name: 'manRetInfo',  
 component: manRetInfo  
 },  
 {  
 path: '/manBrrTotalInfo',  
 name: 'manBrrTotalInfo',  
 component: manBrrTotalInfo  
 },  
  
 {  
 path: '/manInfo',  
 name: 'manInfo',  
 component: manInfo  
 }  
 ]  
 },  
 //管理员路由  
 {  
 path: '/admin',  
 name: 'admin',  
 component: admin,  
 redirect: '/admAddCard',  
 children:[  
 {  
 path:'/admAddCard',  
 name:'admAddCard',  
 component:admAddCard  
 },  
 {  
 path:'/admDeleteCard',  
 name:'admDeleteCard',  
 component:admDeleteCard  
 },  
 {  
 path:'/admManLook',  
 name:'admManLook',  
 component:admManLook  
 },  
 {  
 path:'/admAddMan',  
 name:'admAddMan',  
 component:admAddMan  
 },  
 {  
 path:'/admUpdateMan',  
 name:'admUpdateMan',  
 component:admUpdateMan  
 },  
 {  
 path:'/admSortLook',  
 name:'admSortLook',  
 component:admSortLook  
 },  
 {  
 path:'/admAddSort',  
 name:'admAddSort',  
 component:admAddSort  
 },  
 {  
 path:'/admUpdateSort',  
 name:'admUpdateSort',  
 component:admUpdateSort  
 },  
 {  
 path:'/admBrrInfo',  
 name:'admBrrInfo',  
 component:admBrrInfo  
 },  
 {  
 path:'/admRetInfo',  
 name:'admRetInfo',  
 component:admRetInfo  
 },  
 {  
 path:'/admTotalBrrInfo',  
 name:'admTotalBrrInfo',  
 component:admTotalBrrInfo  
 },  
 {  
 path:'/admInfo',  
 name:'admInfo',  
 component:admInfo  
 }  
 ]  
 }  
  
  
 ],  
  
  
})

* + 1. main.js

// The Vue build version to load with the `import` command  
// (runtime-only or standalone) has been set in webpack.base.conf with an alias.  
import Vue from 'vue'  
import App from './App'  
import router from './router'  
import ElementUI from 'element-ui';  
import 'element-ui/lib/theme-chalk/index.css'  
// 记得引入axios  
import ***axios*** from 'axios'  
import global from "./components/global.js"  
Vue.prototype.$axios = ***axios***;  
Vue.prototype.global = global;  
  
  
Vue.config.productionTip = false  
  
Vue.use(ElementUI)  
/\* eslint-disable no-new \*/  
new Vue({  
 el: '#app',  
 router,  
 components: { App },  
 template: '<App/>'  
})

* + 1. App.vue

<template>  
 <div id="app">  
 <router-view v-if="isRouterAlive"></router-view>  
 </div>  
</template>  
  
<style>  
   
</style>  
  
<script>  
export default {  
 provide(){  
 return {  
 reload:this.reload  
 }  
 },  
 data(){  
 return{  
 isRouterAlive:true   
 }  
 },  
 methods:{  
 reload(){  
 this.isRouterAlive=false;  
 this.$nextTick(function(){  
 this.isRouterAlive=true  
 })  
 }  
 }  
}  
</script>

1. 测试
   1. 各个页面及功能测试
      1. 登陆页面



* + 1. 测试数据

所用测试数据：

有借书证学生：yinxiao 密码：123456

无借书证学生：11111 密码：123456

管理员：0801 密码：123456

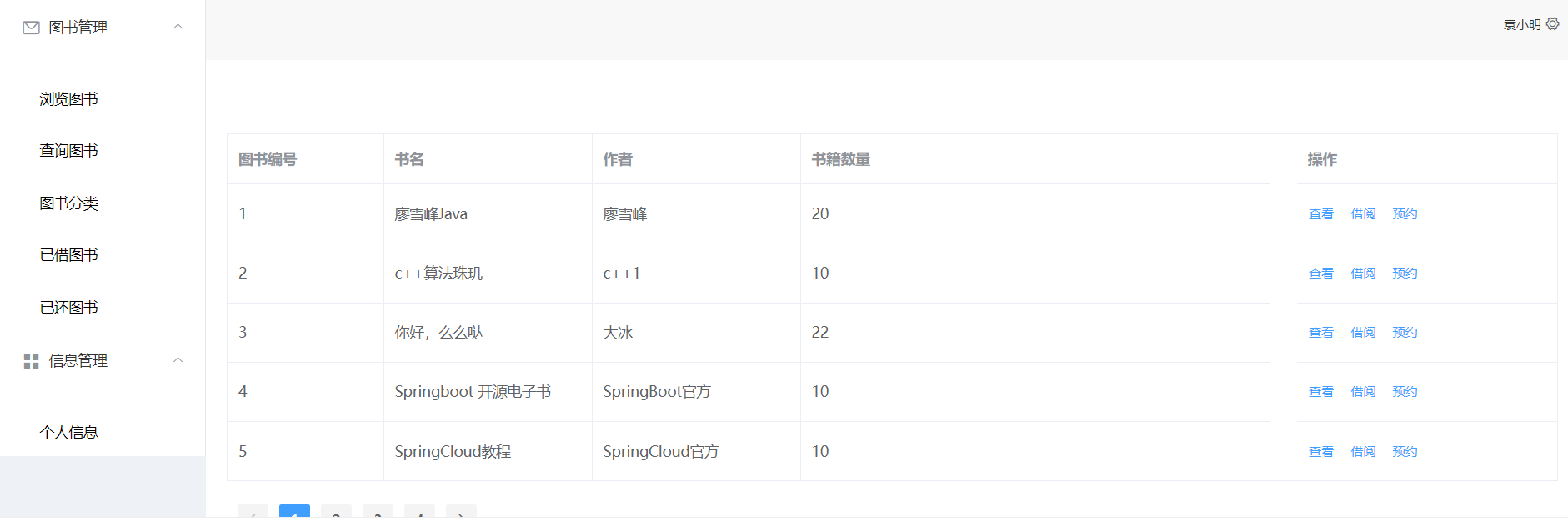
超级管理员：root 密码：123456

* + 1. 注册页面（读者可通过注册获得账号）



* + 1. 读者页面

浏览图书



查询图书：

图书分类



借阅图书：

（有借书证借阅申请提交成功）



（无借书证借阅时，提示没有借书证）



已借图书：



续借：



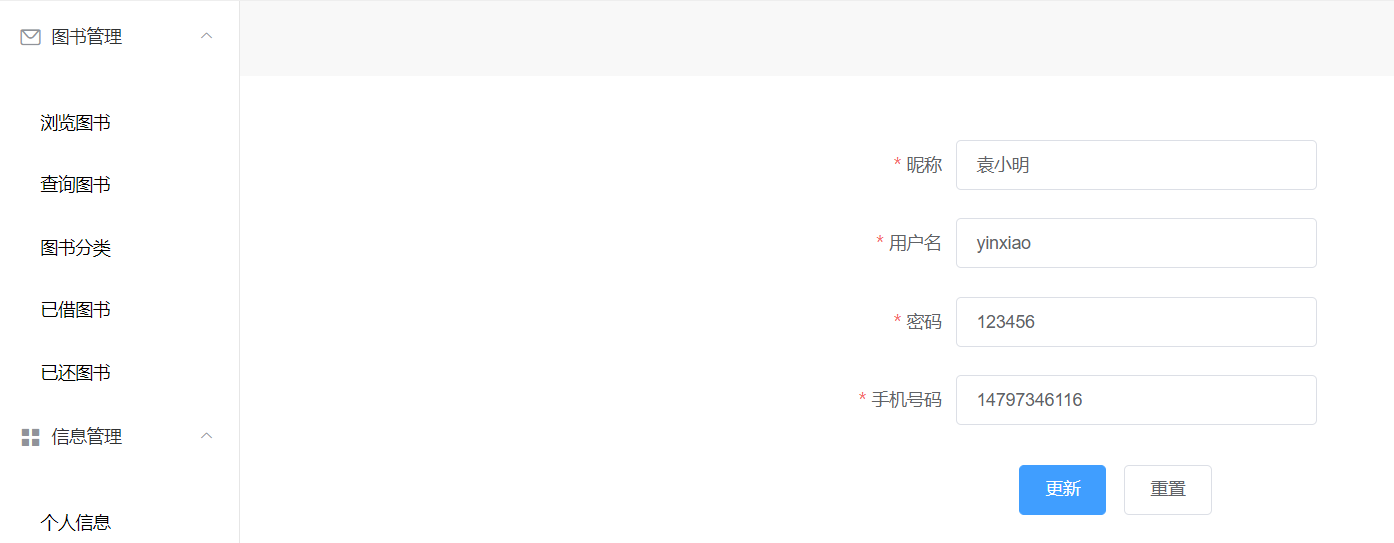
归还：



已还图书：



个人信息：



* + 1. 管理员页面

查看图书：



修改图书：



增加图书：



删除图书：



借阅审核：



续借审核：



归还审核：



已借图书：



已还图书：



借阅排行：

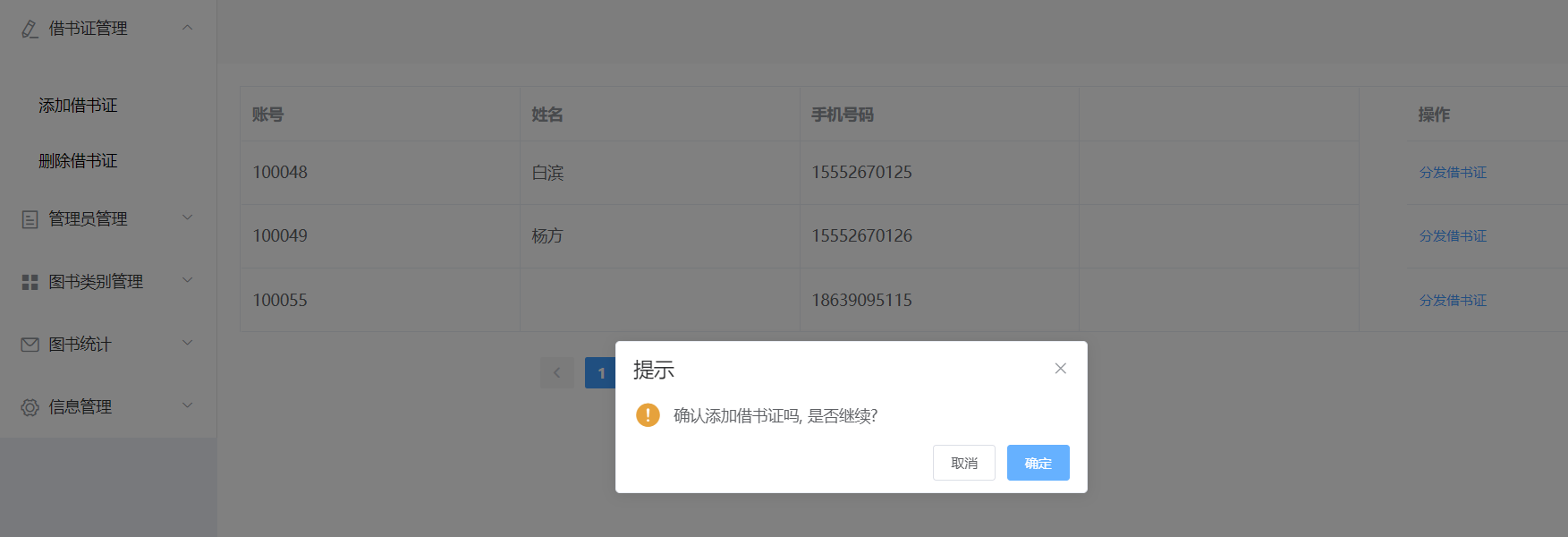


个人信息管理：



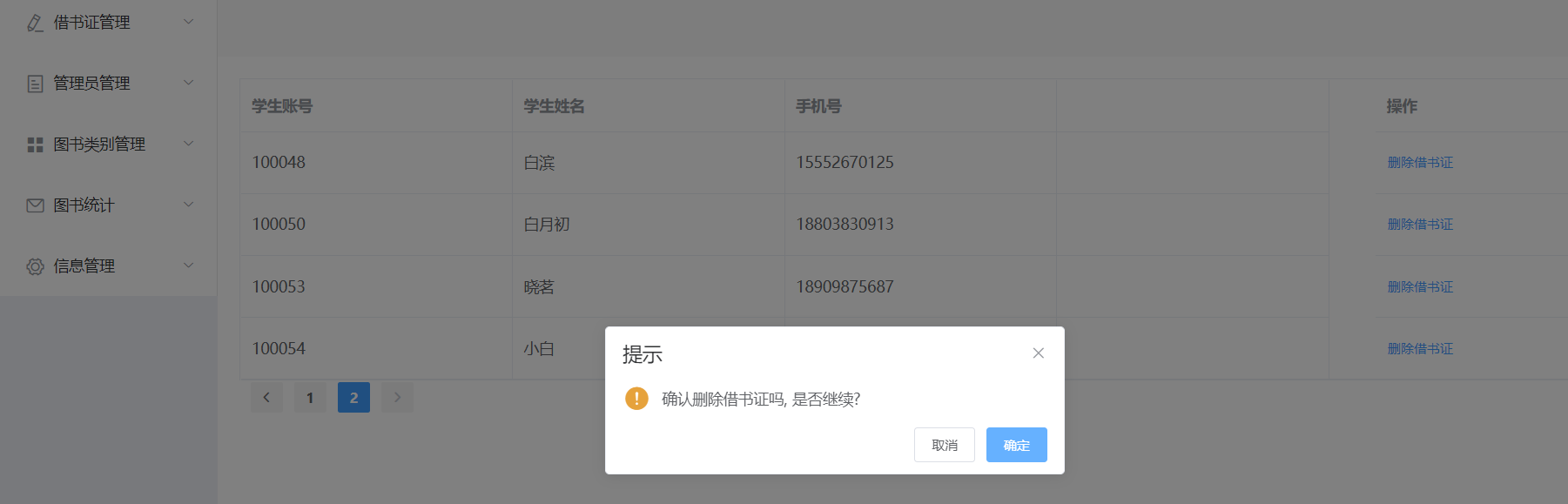
* + 1. 超级管理员页面

添加借书证





删除借书证：





查看管理员



修改管理员：



删除管理员：



添加管理员：

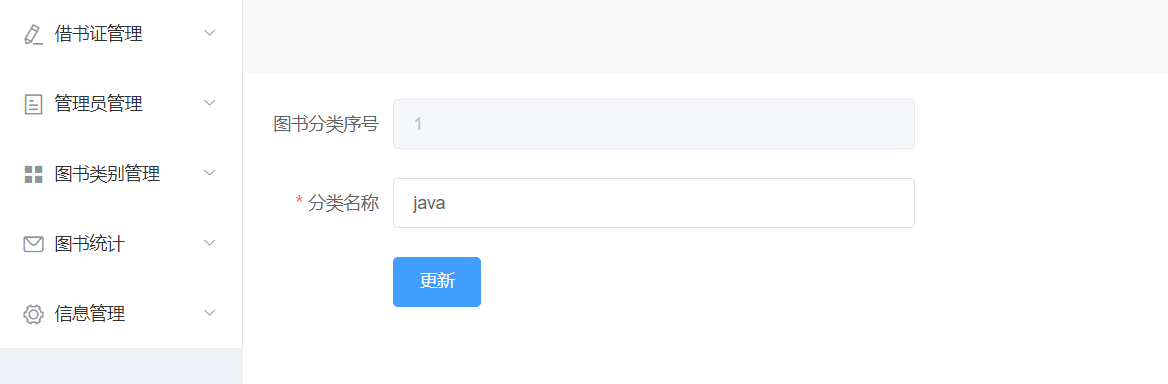




查看图书类别：



修改图书类别：



删除图书类别：



添加图书类别：



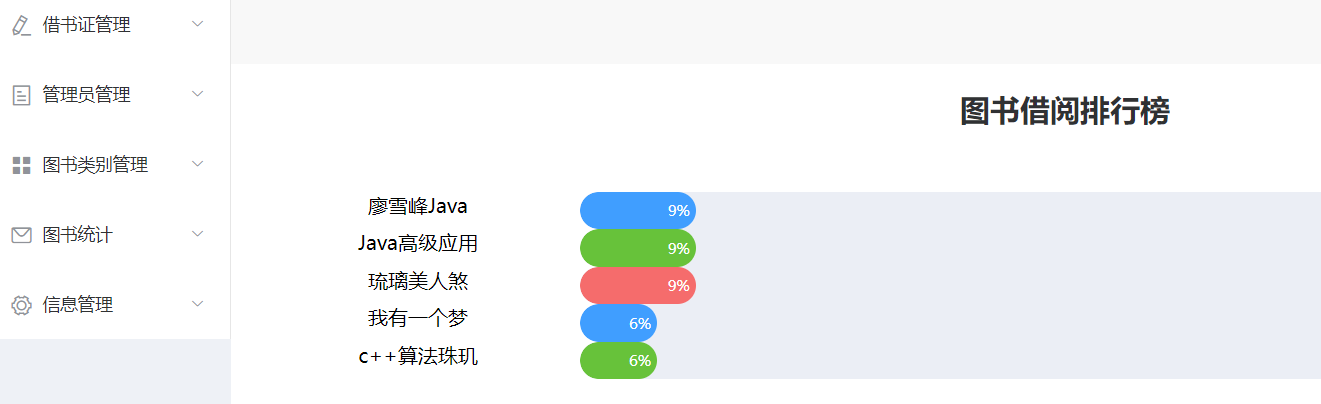
已借图书：



已还图书：



借阅排行榜：



个人信息管理：



* 1. 项目部署方法

第一步：将后端项目：library\_back 以 open 的方式导入 IDEA 中（第一个 idea 窗口）

**第二步：**将前端项目：library\_font 以 open 的方式导入 IDEA 中（第一个 idea 窗口）

**第三步：**将 book.sql 文件导入到 Navicate premium 中

**第四步**：在后端项目：library\_back 中修改 application.yml 文件，

application.yml 文件中的数据库以及登录数据库的账号和密码应该与所需要连接的数据库相匹配

**第五步：**运行后端项目：运行 Java 包下的 LibraryApplication.java 类或者创建对应的Spring Boot项目启动装置

**第六步：**运行前端项目：

1、下载 Vue.js 插件：File-->settings-->plugins-->在Marketplace下输入（Vue.js），

2、安装 node.js 插件

3、在 IDEA 前端项目控制台输入 ：npm install ，安装 Vue 项目运行所需要的环境

4、安装好所需要的环境之后，在 IDEA 前端项目控制台输入 ：npm run dev，启动项目（注意：后端项目必需要运行起来）

5、前后端项目成功运行之后，前端项目会出现一个链接，点击该链接，会出现在系统的注册和登录页面，说明项目成功运行。

**第七步**：同时运行前后端项目后，项目成功运行后，会在前端项目的 idea 控制台出现一个链接，点击该链接便可以进入到系统注册和登录页面。

1. 可行性研究(标题的字体和字号：黑体，三号)
2. (标题的字体和字号：黑体，小三)
3. (标题的字体和字号：黑体，四号)

(内容的字体和字号：宋体，小四)

1. 系统需求分析
2. 系统总体设计
3. 系统详细设计
4. 测试