**目录**

**[小组成员与分工 1](#_Toc16864)**

**[1 概述 3](#_Toc21202)**

[1.1 开发背景 4](#_Toc9436)

[1.2 开发意义 4](#_Toc27599)

**[2 开发介绍 4](#_Toc15537)**

[2.1 Java Web介绍 4](#_Toc528)

[2.2 开发工具介绍 5](#_Toc20046)

[2.3 SSM框架 6](#_Toc1838)

[2.4 系统的运行环境 6](#_Toc27570)

**[3 系统需求分析 6](#_Toc6377)**

[3.1系统需求 6](#_Toc10348)

[3.2 系统开发步骤 7](#_Toc32286)

[3.3 系统功能模块的划分 7](#_Toc21318)

[3.4 定义系统流程 7](#_Toc26698)

[3.5 分析系统流程 9](#_Toc26169)

**[4 系统概要设计 9](#_Toc1574)**

[4.1 系统框架 9](#_Toc10016)

[4.2系统总体功能 10](#_Toc6179)

[4.3系统流程概述 11](#_Toc1719)

**[5 系统详细设计 12](#_Toc32210)**

[5.1 数据库设计 12](#_Toc30959)

[5.2 定义系统静态结构 16](#_Toc11128)

[5.3 分析系统静态结构 17](#_Toc6313)

**[6 系统实现 23](#_Toc12643)**

[6.1登录界面 23](#_Toc14913)

[6.2查询和借阅图书 25](#_Toc2742)

[6.3 图书入库 27](#_Toc27549)

[6.4 修改密码 28](#_Toc22469)

[6.5 增加读者 30](#_Toc5537)

[6.6 数据库连接的实现 31](#_Toc2115)

**[7 测试 31](#_Toc21034)**

[7.1登录 32](#_Toc23489)

[7.2 查询图书 33](#_Toc25347)

[7.3 修改密码 33](#_Toc4952)

[7.4入库图书 34](#_Toc22903)

[7.5注册读者 34](#_Toc17780)

**[8 总结 35](#_Toc15203)**

**[参考文献 36](#_Toc3347)**

# 1 概述

Java是前SUN公司（已被甲骨文收购）在九十年代中期推出的一门面向对象的编程语言，目前广泛应用于开发基于WEB的大中型应用系统。本设计综合应用了Java web技术的特点，同时结合先进的IDEA开发工具和MySQL数据库技术，实现基于局域网的图书馆管理系统。

## 1.1 开发背景

图书馆管理系统的特点是从图书管理的角度出发，用集中的数据库将几乎所有与图书相关的数据统一管理起来，形成了集成的信息源。有好的用户界面，信息的共享使得物品管理人员得以摆脱繁重的日常工作，集中精力从战略的角度来考虑企业物品规划和政策。

## 1.2 开发意义

图书管理是高校都必须切实面对的工作，但一直以来人们使用传统的人工方式管理图书资料。这种方式存在着许多缺点，如效率低、保密性差且较为繁琐。另外，随着图书资料数量的增加，其工作量也将大大增加，这必将增加图书资料管理者的工作量和劳动强度，这将给图书资料信息的查找、更新和维护都带来了很多困难。

经过调查，目前我国各类高等学校中有相当一部分单位图书资料管理还停留在人工管理的基础上。这样的管理机制已经不能适应时代的发展，其管理方法将浪费许多人力和物力。随着科学技术的不断提高，这种传统的手工管理方法必然被以计算机为基础的信息管理方法所取代。

图书管理作为计算机应用的一个分支，有着手工管理无法比拟的优点，如检索迅速、查找方便、可靠性高、存储量大、保密性好、寿命长、成本低等。这些优点能够极大地提高图书管理的效率。因此，开发一套能够为用户提供充足的信息和快捷的查询手段的图书管理系统，将是非常必要的，也是十分及时的。

# 2 开发介绍

## 2.1 Java Web介绍

Java Web，是用Java技术来解决相关web互联网领域的技术栈。web包括：web服务端和web客户端两部分。Java在客户端的应用有Java Applet，不过使用得很少，Java在服务器端的应用非常的丰富，比如Servlet，JSP、第三方框架等等。Java技术对Web领域的发展注入了强大的动力。

Internet上供外界访问的Web资源分为：静态web资源（如html 页面）：指web页面中供人们浏览的数据始终是不变。动态web资源：指web页面中供人们浏览的数据是由程序产生的，不同时间点访问web页面看到的内容各不相同。静态web资源开发技术：HTML、CSS。动态web资源开发技术：JavaScript、JSP/Servlet、ASP、PHP、SSM框架等。在Java中，动态web资源开发技术统称为Java Web。

## 2.2 开发工具介绍

1. **IntelliJ IDEA概述**

IDEA 全称 IntelliJ IDEA，是java编程语言开发的集成环境。IntelliJ在业界被公认为最好的java开发工具，尤其在智能代码助手、代码自动提示、重构、JavaEE支持、各类版本工具(git、svn等)、JUnit、CVS整合、代码分析、 创新的GUI设计等方面的功能可以说是超常的。IDEA是JetBrains公司的产品，这家公司总部位于捷克共和国的首都布拉格，开发人员以严谨著称的东欧程序员为主。它的旗舰版本还支持HTML，CSS，PHP，MySQL，Python等。免费版只支持Java，Kotlin等少数语言。

**（2）Tomcat9.0概述**

Tomcat9.0 服务器是一个免费的开放源代码的Web 应用服务器,Tomcat是Apache 软件基金会（Apache Software Foundation）的Jakarta 项目中的一个核心项目，由Apache、[Sun](http://baike.baidu.com/view/24856.htm" \t "_blank) 和其他一些公司及个人共同开发而成。由于有了Sun 的参与和支持，最新的[Servlet](http://baike.baidu.com/view/25169.htm" \t "_blank) 和JSP 规范总是能在Tomcat 中得到体现，Tomcat 9.0支持最新的Servlet 3.0 和JSP 2.2 规范。因为Tomcat 技术先进、性能稳定，而且免费，因而深受Java 爱好者的喜爱并得到了部分软件开发商的认可，成为目前比较流行的Web 应用服务器。

**（3）Mysql概述**

MySQL是一个[关系型数据库管理系统](http://baike.baidu.com/view/1450387.htm" \t "_blank)，目前属于[Oracle](http://baike.baidu.com/view/15020.htm" \t "_blank)公司。MySQL是一种关联[数据库管理系统](http://baike.baidu.com/view/68446.htm" \t "_blank)，关联数据库将数据保存在不同的表中，而不是将所有数据放在一个大仓库内，这样就增加了速度并提高了[灵活](http://baike.baidu.com/view/474022.htm" \t "_blank)性。MySQL所使用的SQL语言是用于访问[数据库](http://baike.baidu.com/view/1088.htm" \t "_blank)的最常用标准化语言。软件采用了双授权政策，它分为社区版和商业版，由于其体积小、速度快、总体拥有成本低，尤其是[开放源码](http://baike.baidu.com/view/394804.htm" \t "_blank)这一特点，一般中小型系统的开发都选择MySQL作为[数据库](http://baike.baidu.com/view/1088.htm" \t "_blank)。

## 2.3 SSM框架

SSM框架是spring MVC ，spring和Mybatis框架的整合，是标准的MVC模式，将整个系统划分为表现层，controller层，service层，DAO层四层。

Spring框架是一个[开放源代码](https://baike.baidu.com/item/%E5%BC%80%E6%94%BE%E6%BA%90%E4%BB%A3%E7%A0%81/114160" \t "https://baike.baidu.com/item/spring/_blank)的[J2EE](https://baike.baidu.com/item/J2EE/110838" \t "https://baike.baidu.com/item/spring/_blank)应用程序框架，由[Rod Johnson](https://baike.baidu.com/item/Rod Johnson/1423612" \t "https://baike.baidu.com/item/spring/_blank)发起，是针对bean的生命周期进行管理的轻量级容器（lightweight container）。 Spring解决了开发者在J2EE开发中遇到的许多常见的问题，提供了功能强大IOC、[AOP](https://baike.baidu.com/item/AOP/1332219" \t "https://baike.baidu.com/item/spring/_blank)及Web MVC等功能。

  Spring MVC是一种基于Java的实现了Web MVC设计模式的请求驱动类型的轻量级Web框架，即使用了MVC架构模式的思想，将web层进行职责解耦，基于请求驱动指的就是使用请求-响应模型，框架的目的就是帮助我们简化开发，Spring Web MVC也是要简化我们日常Web开发的。

MyBatis 是一款优秀的数据持久层框架，它支持定制化 SQL、存储过程以及高级映射。MyBatis 避免了几乎所有的 JDBC 代码和手动设置参数以及获取结果集。MyBatis 可以使用简单的 XML 或注解来配置和映射原生信息，将接口和 Java 的 POJO(Plain Ordinary Java Object,普通的 Java对象)映射成数据库中的记录。

## 2.4 系统的运行环境

电脑配置：华硕 ，CPU：AMD，内存：16G，硬盘:300G。

操作系统：win 10

开发环境：IDEA

Web服务平台：Tomcat 9.0

数据库：MySql 8.0

JDK：1.8

# 3 系统需求分析

## 3.1系统需求

根据本校图书馆的工作流程与实际的需求和特色，本系统需满足以下几个方面的要求：

1. 对新书进行登记入库、下架管理。
2. 对借书读者信息提供维护功能
3. 提供图书借书、还书功能
4. 提供图书查阅功能，包括按照书名、作者、书号、简介和语言查询。

## 3.2 系统开发步骤

图书馆管理系统的建立与应用可以划分成开发工具选择、系统需求设计、系统概要设计、系统详细设计和测试五个阶段，其中系统详细设计阶段还可进一步分为数据库设计、定义系统静态结构和分析系统静态结构等工作环节。上述各个阶段排列成一个严格的线性开发序列。

上述开发方式的主要优点是便于开发工作的组织和管理，并且可大大降低在线学习系统开发的复杂性。国内外许多系统开发的实例都证明这是一种行之有效的开发方式。

## 3.3 系统功能模块的划分

软件所实现的功能强弱是衡量一个软件的最根本的标准。经过对系统的需求分析和实际应用需求，确定了本子系统的功能模块如图3.1所示**：**

 图3.1 系统功能模块图

## 3.4 定义系统流程

与本系统相关的角色包括：

图书馆管理员：拥有操作图书入库和下架，修改借阅者信息，查看借阅日志，修改管理员密码的权限。

图书借阅者：查看图书，图书借阅与归还，修改密码的权限。

用例图如下图3.2所示:



图3.2 系统用例图

表3.2 系统用例

|  |  |
| --- | --- |
| 系统用例名称 | 简述 |
| 信息查询 | 读者查询借阅信息 |
| 借阅与归还图书 | 借阅者借阅或归还图书 |
| 书籍查询 | 管理员或用户查询相关图书信息 |
| 登录 | 管理员或用户登录系统 |
| 书籍信息修改 | 管理员对书籍进行入库或者修改操作 |
| 增删用户 | 管理员增加或者删除用户 |

## 3.5 分析系统流程

具体活动图如下图3.3所示，管理员与用户首先需要登录，若账户和密码正确，则登录成功，反之则登录失败。登录成功后，系统根据登录者的身份提供相应的功能，读者用户可以进行书籍信息查找，借阅或者归还图书，修改自己的密码，管理员可以对书籍信息，用户信息，借阅日志进行修改或删除操作。



图3.3活动图

# 4 系统概要设计

## 4.1 系统框架

在B/S的系统中，用户可以通过浏览器向分布在网络上的许多服务器发出请求。B/S结构极大的简化了客户机的工作，客户机上只需安装、配置少量的客户端软件即可，服务器将担负更多的工作，对数据库的访问和应用程序的执行将在服务器上完成。

在B/S三层体系结构下，表示层(view)、功能层(controller)、数据层(model)被分割成三个相对独立的单元：

（1）、第一层--表示层：Web浏览器

在表示层中包含系统的显示逻辑，位于客户端。它的任务是由Web浏览器向网络上的某一Web服务器提出服务请求，Web服务器对用户身份进行验证后用HTTP协议把所需的主页传送给客户端，客户机接受传来的主页文件，并把它显示在Web浏览器上。

（2）、第二层--功能层：具有应用程序扩展功能的Web服务器

在功能层中包含系统的事务处理逻辑，位于Web服务器端。它的任务是接受用户的请求，首先需要执行相应的扩展应用程序与数据库进行连接，通过SQL等方式向数据库服务器提出数据处理申请，后台数据库服务器将数据处理的结果提交给Web服务器，再由Web服务器传送回客户端。

（3）、第三层--数据层：数据库服务器

在数据层中包含系统的数据处理逻辑，位于数据库服务器端。它的任务是接受Web服务器对数据库操纵的请求，实现对数据库查询、修改、更新、删除功能，把运行结果提交给Web服务器。

从上面的分析中可以看出，B/S体系结构是把C/S结构的事务处理逻辑模块从客户机的任务中分离出来，由单独组成的一层来负担其任务，这样客户机的压力大大减轻了，把负荷均衡地分配给了Web服务器，于是由原来的C/S结构转变成B/S结构。

## 4.2系统总体功能

（1）、用户管理

用户管理包括管理员为用户分配账号，修改或删除用户信息功能。此模块只能由图书管理员统一配置，用户不能自行注册，管理员可以修改自己的密码，但是不可以修改用户密码。

（2）、 图书管理

管理员可以对图书进行入库或者修改图书信息的操作，并能查看或者删除已经归还的图书借阅日志。

（3）、图书借阅与归还

用户可以借阅图书馆中的图书，若图书数量不足，则不可以借阅，同时，用户可以归还图书，并查看自己的借阅记录。

## 4.3系统流程概述

1. 、系统用户首先需要通过输入账号和密码的方式来登录图书管理系统，若账号不存在或者密码错误，则登录失败，反之则登录成功。系统流程图如下图4.1所示。



图4.1 系统流程图

1. 、系统确定账号和密码无误后，确定系统用户的身份，若用户身份为图书管理员，则进入图书管理界面，若用户为普通读者，则进入图书借阅界面。
2. 、管理员进入管理界面后，可以查看全部的图书信息、根据图书名、作者、出版社、ISBN或者语言查找图书信息。身为管理员，可以入库新的图书，也可以注册新的用户，修改用户的信息（除密码以外），查看或者删除已经归还的借阅信息。同时，管理员也可以修改自己的管理员账户密码。
3. 、读者进入图书馆后，可以查阅图书馆所有存在的图书信息，同时可以根据图书名称进行模糊查询。身为读者，可以借阅图书馆中存在的图书，也可归还自己借阅的图书；在归还图书后，读者可以查看归还日志；用户可以修改自己的信息（包括密码）。

# 5 系统详细设计

## 5.1 数据库设计

数据库的设计在一个系统中的作用十分重要，数据库建立的是否完善直接影响到你个系统的实现。数据库的设计既要满足用户的需求又要尽最大可能的降低数据的冗余，尽可能降低数据间的依赖，将他们分离。在各种信息的提供，保存，更新和查询，这就要求数据库结构能充分满足各种信息的输出和输入，收集基本数据，数据结构，以及数据处理的流程，组成一份详尽的数据字典，为后面的具体设计打下基础．

5.1.1 概念设计

用户分为两类：读者、图书馆管理员。读者仅可以修改个人信息、借阅或归还书籍和查看自己的借还日志。如下图5.1所示。

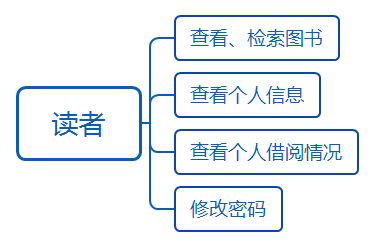


图5.1 读者权限

图书馆管理员可以修改读者信息，修改书目信息，查看所有借还日志等。如下图5.2所示。

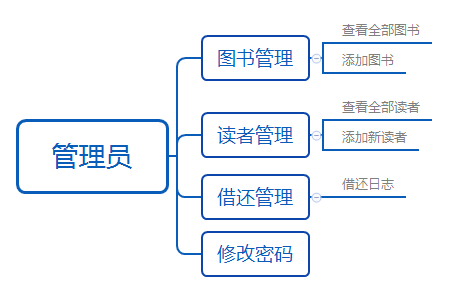


图5.2 管理员权限

### 5.1.2 ER图设计

E-R图也称实体-联系图(Entity Relationship Diagram)，提供了表示实体类型、属性和联系的方法，它是描述现实世界关系概念模型的有效方法，是表示概念关系模型的一种方式；图书管理系统的ER图如下图5.3所示。



图 5.3 图书管理系统ER图

图书管理系统的实体类型包括：读者（reader）、书籍（book\_info）和管理员（admin）。

实体类读者的属性：读者号（reader\_id）、姓名（name）、性别（sex）、生日（birth）、地址（address）和电话号码（phone）。

实体类管理员的属性：管理员编号（id）、管理员名（username）和密码（password）。

实体类书籍的属性：数量（number）、简介（introduction）、出版时间 （pub\_time）、书名（name）、作者（author）、出版社（publish）、标准书号（ISBN）、语言（language）、价格（price）和分类（class\_id）。

实体类书籍和实体类读者通过借阅（lendlist）联系起来，读者可以对书籍进行借阅操作，其中借阅包含属性借阅日期（lend data）和归还日期（back data）。

实体类管理员和书籍通过管理分类（class\_info）联系起来，管理员可以对图书进行分类操作。

5.1.2 逻辑设计

逻辑结构设计的原则如下：

（1）、尽可能的减少数据冗余和重复。

（2）、结构设计与操作设计相结合。

（3）、数据结构具有相对的稳定性。

（4）、遵循数据库设计三范式。

基于以上设计原则，本系统设计的数据库：全部都使用逻辑删除，为了便于开发，所以不添加任何表关联，该数据库的表设计如下:

**管理员表admin：**

| **字段名** | **类型** | **长度** | **小数点** | **NULL** | **用途** | **键** |
| --- | --- | --- | --- | --- | --- | --- |
| admin\_id | bigint | 20 | 0 | 否 | 账号 | ✔ |
| password | varchar | 15 | 0 | 否 | 密码 |  |
| username | varchar | 15 | 0 | 是 | 用户名 |  |

**表5.1.2.1管理员表**

**书目表book\_info**

| **名** | **类型** | **长度** | **小数点** | **NULL** | **用途** | **键** |
| --- | --- | --- | --- | --- | --- | --- |
| book\_id | bigint | 20 | 0 | 否 | 图书号 | ✔ |
| name | varchar | 200 | 0 | 否 | 书名 |  |
| author | varchar | 150 | 0 | 否 | 作者 |  |
| publish | varchar | 200 | 0 | 否 | 出版社 |  |
| ISBN | varchar | 15 | 0 | 否 | 标准书号 |  |
| introduction | text | 0 | 0 | 是 | 简介 |  |
| language | varchar | 4 | 0 | 否 | 语言 |  |
| price | decimal | 10 | 2 | 否 | 价格 |  |
| pub\_date | date | 0 | 0 | 否 | 出版时间 |  |
| class\_id | int | 11 | 0 | 是 | 分类号 |  |
| number | int | 11 | 0 | 是 | 剩余数量 |  |

**表5.1.2.2书目表**

**图书分类表class\_info**

| **名** | **类型** | **长度** | **小数点** | **NULL** | **用途** | **键** |
| --- | --- | --- | --- | --- | --- | --- |
| class\_id | int | 11 | 0 | 否 | 类别号 | ✔ |
| class\_name | varchar | 15 | 0 | 否 | 类别名 |  |

**表5.1.2.3图书目表**

**借阅信息表lend\_list**

| **字段名** | **类型** | **长度** | **小数点** | **NULL** | **用途** | **键** |
| --- | --- | --- | --- | --- | --- | --- |
| ser\_num | bigint | 20 | 0 | 否 | 流水号 | ✔ |
| book\_id | bigint | 20 | 0 | 否 | 图书号 |  |
| reader\_id | bigint | 20 | 0 | 否 | 读者证号 |  |
| lend\_date | timestamp | 0 | 0 | 是 | 借出日期 |  |
| back\_date | timestamp | 0 | 0 | 是 | 归还日期 |  |

**借阅卡信息表reader\_card**

| **字段名** | **类型** | **长度** | **小数点** | **NULL** | **用途** | **键** |
| --- | --- | --- | --- | --- | --- | --- |
| reader\_id | bigint | 20 | 0 | 否 | 读者证号 | ✔ |
| password | varchar | 15 | 0 | 否 | 密码 |  |
| username | varchar | 15 | 0 | 是 | 用户名 |  |

**读者信息表reader\_info**

| **字段名** | **类型** | **长度** | **小数点** | **NULL** | **用途** | **键** |
| --- | --- | --- | --- | --- | --- | --- |
| reader\_id | bigint | 20 | 0 | 否 | 读者证号 | ✔ |
| name | varchar | 10 | 0 | 否 | 姓名 |  |
| sex | varchar | 2 | 0 | 否 | 性别 |  |
| birth | date | 0 | 0 | 否 | 生日 |  |
| address | varchar | 50 | 0 | 否 | 地址 |  |
| phone | varchar | 15 | 0 | 否 | 电话 |  |

## 5.2 定义系统静态结构

类图用于描述系统中所包含的类以及它们之间的相互关系，帮助用户简化对系统的理解，它是系统分析和设计阶段的重要产物，也是系统编码和测试的重要模型依据，用类图表达图书管理系统内部的静态结构。系统只有具备稳定且具弹性的静态结构，才能够顺应需求变更，迅速支撑多样化的系统用例。该类图包括图书管理员，读者，借阅图书，读者卡，书籍；各部分联合起来的整体类图如图5.4所示。

 **图5.4 系统类图**

## 5.3 分析系统静态结构

顺序图以读者和管理员为对象，可显示不同的功能如何交互，纵向是时间轴，时间沿竖线向下延伸。横向轴代表了在协作中图书管理系统的各个对象。对象用生命线表示。当对象存在时，角色用一条虚线表示，当对象的过程处于激活状态时，生命线是一个双道线。

1. **、登录时序图**

读者和管理员作为一个对象登录图书管理系统，在系统登录界面中的文本框中分别输入用户名和密码，点击登录按钮，系统将向查询结果发送一条信息，

查询结果对象在数据库中进行查找操作，若匹配成功则跳转至相应的界面，若匹配失败登录界面将弹出“密码错误”的提示信息，登录时序图如下图5.5所示。



**图5.5 登录时序图**

**（2）、查找图书时序图**

读者和管理员作为一个对象使用系统的查询图书的功能，在系统界面中的文本框中输入关键字查找相关的图书信息，点击查找按钮，系统将向查询结果角色发送一条信息，将输入的书名进行匹配，若匹配成功则在操作界面输出相关图书信息，若匹配失败则返回不存在信息，操作界面将弹出“你要查找的图书不存在”的信息。查询图书时序图如下图5.6所示。



**图5.6 查询图书时序图**

1. **、图书借阅与归还时序图**

读者作为一个对象使用系统的图书借阅的功能，在图书显示界面中点击借阅按钮，系统将向查询结果角色发送一条信息，将借阅的书名进行匹配，操作界面的“借阅”按钮将变成“已借”按钮。图书借阅时序图如下图5.7所示。

  **图5.7 图书借阅与归还时序图**

1. **、修改密码时序图**

读者和管理员作为一个对象使用系统的修改密码的功能，在修改密码界面中的文本框中输入旧密码和新密码，点击“修改”按钮，系统首先向查询结果发送一条消息判断旧密码是否输入正确，若输入正确，则将新密码写入数据库，并在界面显示密码修改成功的信息；反之，提示旧密码不匹配或者两次新密码输入不一致。修改密码时序图如下图5.8所示。

 图5.8 修改密码时序图

1. **、修改信息时序图**

****

**图5.9修改信息时序图**

1. **、管理员时序图**

管理员作为系统的使用对象，有着对图书的信息修改和对图书入库与修改的操作，每当管理员执行这些操作时，将在图书相关界面输入图书的信息进行修改或入库操作，点击确认按钮后，将会给查询结果发送请求，此时，查询结果接收到请求，将会向数据库发送修改数据的操作，届时，操作界面反馈给管理员修改成功的信息。

**图5.10 管理员时序图**

# 6 系统实现

Bootstrap，来自 Twitter，是目前最受欢迎的前端框架之一。Bootstrap 是基于 HTML、CSS、JAVASCRIPT 的，它简洁灵活，使得 Web 开发更加快捷。

本图书管理系统采用bootstrap框架编写前端界面，使得代码变得更加的简洁，界面变得更加的美观，便于用户能够快速的上手本系统。

## 6.1登录界面

在浏览器中打开系统后，映入眼帘的是登录界面，如下图6.1所示。在这个个白色的界面中分别有两个文本框、一个可选的记住密码复选框以及登录按钮，用户依次输入属于自己的账号和密码，点击登录，若账号和密码正确，即可登录到相应的系统主界面。若密码输入不正确，则在界面中提示账号和密码不正确的信息。



图6.1 系统登录界面

**前端登录界面主要代码如下（index.jsp）:**

<div class="panel panel-default" id="login">  
 <div class="panel-heading" style="background-color: #fff">  
 <h3 class="panel-title">请登录</h3>  
 </div>  
 <div class="panel-body">  
 <div class="form-group">  
 <label for="id">账号</label>  
 <input type="text" class="form-control" id="id" placeholder="请输入账号">  
 </div>  
 <div class="form-group">  
 <label for="passwd">密码</label>  
 <input type="password" class="form-control" id="passwd" placeholder="请输入密码">  
 </div>  
 <div class="checkbox text-left">  
 <label>  
 <input type="checkbox" id="remember">记住密码  
 </label>  
 </div>

**后端判断密码是否正确的主要代码如下（LoginController.java）:**

@RequestMapping(value = "/api/loginCheck", method = RequestMethod.POST)

public @ResponseBody

Object loginCheck(HttpServletRequest request) {

long id = Long.parseLong(request.getParameter("id"));//得到用户的账号

String passwd = request.getParameter("passwd");//得到用户的密码

//与数据库中查询到的相匹配，若正确，返回true,反之返回false

boolean isReader = loginService.hasMatchReader(id, passwd);

boolean isAdmin = loginService.hasMatchAdmin(id, passwd);

HashMap<String, String> res = new HashMap<>();

if (isAdmin) {

Admin admin = new Admin();

admin.setAdminId(id);

String username = loginService.getAdminUsername(id);

admin.setUsername(username);

//设置session，只需登录一次，即可进行多界面操作。

request.getSession().setAttribute("admin", admin);

//设置登录状态，管理员为1，读者为2

res.put("stateCode", "1");

res.put("msg", "管理员登陆成功！");

} else if (isReader) {

ReaderCard readerCard = loginService.findReaderCardByReaderId(id);

request.getSession().setAttribute("readercard", readerCard);

res.put("stateCode", "2");

res.put("msg", "读者登陆成功！");

} else {

res.put("stateCode", "0");

res.put("msg", "账号或密码错误！");

}

return res;

}

**涉及到的sql语句如下:**

**读者登录（readerCard.xml）：**

<select id="getIdMatchCount" resultType="int">

select count(\*) from reader\_card where

reader\_id = #{reader\_id} and password = #{password}

</select>

**管理员登录（admin.xml）：**

<select id="getMatchCount" resultType="int">

select count(\*) from admin

where admin\_id = #{admin\_id}

and password = #{password}

</select>

## 6.2查询和借阅图书

当读者或者管理员登录成功后，即可使用查询和借阅图书的功能，如下图6.2所示。读者输入图书的关键字（包括图书名、作者、出版社、图书编号以及简介）进行查找相关的信息，若图书数量充足，即可借阅。



图6.2 查询借阅图书

**前端界面代码如下（readerbooks.jsp）:**

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

<tr>

<td><c:out value="${book.name}"></c:out></td>

<td><c:out value="${book.author}"></c:out></td>

<td><c:out value="${book.publish}"></c:out></td>

<td><c:out value="${book.isbn}"></c:out></td>

<td><c:out value="${book.price}"></c:out></td>

<td><c:out value="${book.number}"></c:out></td>

<c:set var="flag" value="false"/>

<c:forEach var="lend" items="${myLendList}">

<c:if test="${lend eq book.bookId}">

<c:set var="flag" value="true"/>

</c:if>

</c:forEach>

**后端查询图书主要代码如下（BookController.jsp）：**

@RequestMapping("/reader\_querybook\_do.html")

public ModelAndView readerQueryBookDo(String searchWord) {

//返回Boolean类型的值，查找到了就放入list集合中

if (bookService.matchBook(searchWord)) {

ArrayList<Book> books = bookService.queryBook(searchWord);

ModelAndView modelAndView = new ModelAndView("reader\_books"); //使用ModelAndView类用来存储处理完后的结果数据，以及显示该数据的视图。

modelAndView.addObject("books", books);

return modelAndView;

} else {

return new ModelAndView("reader\_books", "error", "没有匹配的图书");

}

}

**涉及到的sql语句如下：**

**管理员查询图书（book.xml）：**

<select id="queryBook"resultType="com.library.bean.Book"parameterType="String">

select \* from book\_info where name like #{search}

or author like #{search} or introduction like #{search}

or publish like #{search} or language like #{search}

or ISBN like #{search}

</select>

<select id="matchBook" resultType="int" parameterType="String">

select count(\*) from book\_info where name like #{search}

or author like #{search} or introduction like #{search}

or publish like #{search} or language like #{search}

or ISBN like #{search}

</select>

**读者借阅图书（lend.xml）：**

<insert id="lendBookOne">

insert into lend\_list values(null , #{book\_id} , #{reader\_id} , sysdate() , null)

</insert>

<update id="lendBookTwo" parameterType="long">

update book\_info set number = number - 1 where book\_id = #{book\_id}

</update>

## 6.3 图书入库

作为图书管理员，拥有图书入库以及修改图书信息的权限，界面如下图6.3所示，当管理员输入完成图书信息后，点击“添加”按钮，系统将提示添加成功的信息，并跳转至显示所有图书信息的界面。



图6.3 添加图书

**前端主要代码如下所示（admin\_book\_add.jsp）:**

<form action="book\_add\_do.html" method="post" id="addbook">

<div class="form-group">

<label for="name">图书名</label>

<input type="text" class="form-control" name="name" id="name"

..........................................

<div class="form-group">

<label for="number">数量</label>

<input type="text" class="form-control" name="number" id="number" placeholder="请输入图书数量">

</div>

<input type="submit" value="添加" class="btn btn-success btn-sm" class="text-left">

**后端主要代码如下所示（BookController.java）:**

@RequestMapping("/book\_add\_do.html")

public String addBookDo(@RequestParam(value = "pubstr") String pubstr, Book book, RedirectAttributes redirectAttributes) {

//格式化日期

book.setPubdate(getDate(pubstr));

//添加图书，若添加成功，为true

if (bookService.addBook(book)) {

//达到重新向带参,其原理就是放到session中，session在跳到页面后销毁

redirectAttributes.addFlashAttribute("succ", "图书添加成功！");

} else {

redirectAttributes.addFlashAttribute("succ", "图书添加失败！");

}

return "redirect:/admin\_books.html";

}

**涉及到sql语句如下（book.xml）:**

<insert id="addBook" parameterType="com.library.bean.Book">

insert into book\_info values

(null, #{name}, #{author}, #{publish}, #{ISBN},

#{introduction}, #{language}, #{price}, #{pub\_date},

#{class\_id}, #{number})

</insert>

## 6.4 修改密码

读者或者图书管理员登录系统后，能修改自己的密码，如下图6.4所示，点击修改密码按钮，跳转至修改密码界面，用户分别输入自己的旧密码、新密码和确认新密码后，点击提交按钮，系统将提示密码修改成功信息，若旧密码输入错误，将提示密码错误，若两次密码输入不一致，则返回不一致的信息。

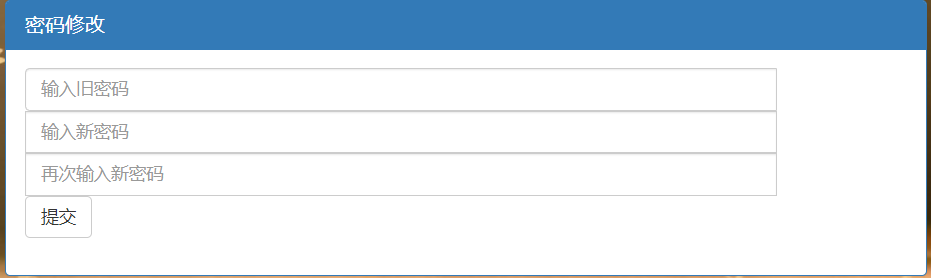


图6.4 修改密码界面

**前端代码如下所示（LoginController.java）:**

<div class="panel-body">

<form method="post" action="admin\_repasswd\_do" class="form-inline" id="repasswd">

<div class="input-group">

<input type="password" id="oldPasswd" name="oldPasswd" placeholder="输入旧密码" class="form-control">

<input type="password" id="newPasswd" name="newPasswd" placeholder="输入新密码" class="form-control">

<input type="password" id="reNewPasswd" name="reNewPasswd" placeholder="再次输入新密码">

<em id="tishi" style="color: red"></em>

<br/>

<span>

<input type="submit" value="提交" class="btn btn-default">

</span>

**后端主要代码如下（LoginController.java）:**

@RequestMapping("/admin\_repasswd\_do")

public String reAdminPasswdDo(HttpServletRequest request, String oldPasswd, String newPasswd, String reNewPasswd, RedirectAttributes redirectAttributes) {

//得到登录时设置的session，

Admin admin = (Admin) request.getSession().getAttribute("admin");

long id = admin.getAdminId();

//获取当前管理员的密码

String password = loginService.getAdminPassword(id);

if (password.equals(oldPasswd)) {

//若用户输入的旧密码与数据库中的相匹配，则修改成功

if (loginService.adminRePassword(id, newPasswd)) {

redirectAttributes.addFlashAttribute("succ", "密码修改成功！");

//重定向至修改密码界面

return "redirect:/admin\_repasswd.html";

} else {

redirectAttributes.addFlashAttribute("error", "密码修改失败！");

return "redirect:/admin\_repasswd.html";

}

} else {

redirectAttributes.addFlashAttribute("error", "旧密码错误！");

return "redirect:/admin\_repasswd.html";

}

}

**涉及到的sql语句如下：**

**管理员（admin.xml）:**

<update id="resetPassword">

update admin set password = #{password}

where admin\_id = #{admin\_id}

</update>

<select id="getPassword" resultType="String" parameterType="long">

select password from admin where admin\_id = #{admin\_id}

</select>

**读者（readerCard.xml）:**

<update id="resetPassword">

update reader\_card set password = #{password}

where reader\_id = #{reader\_id}

</update>

<select id="getPassword" resultType="String" parameterType="long">

select password from reader\_card where reader\_id = #{reader\_id}

</select>

## 6.5 增加读者

管理员登录图书管理系统，可以录入新的读者信息，用于读者登录图书馆借阅图书，管理员点击“增加读者”按钮，跳转至录入读者信息的界面，界面如下图6.5所示，输入读者相关信息后，点击添加按钮，系统将返回添加成功的信息。



图6.5 录入读者界面

**前端主要代码如下所示（admin\_reader\_add.jsp）:**

<form action="reader\_add\_do.html" method="post" id="readeredit" >

<div class="input-group" style="padding-top: 20px;">

<span class="input-group-addon">密码</span>

<input type="password" class="form-control" name="password" id="password" ">

</div>

..................

<div class="input-group" style="padding-top: 20px;">

<span class="input-group-addon">姓名</span>

<input type="text" class="form-control" name="name" id="name" >

</div>

<input style="align-items: center" type="submit" value="添加" class="btn btn-success btn-sm"class="text-left">

**后端主要代码如下所示（ReadController.java）:**

@RequestMapping("reader\_add\_do.html")

public String readerInfoAddDo(String name, String sex, String birth, String address, String phone, String password, RedirectAttributes redirectAttributes) {

ReaderInfo readerInfo = getReaderInfo(0, name, sex, birth, address, phone);

//录入到数据库中，返回读者的登录账号。

long readerId = readerInfoService.addReaderInfo(readerInfo);

readerInfo.setReaderId(readerId);

//添加读者卡的相关信息到读者卡表中

if (readerId > 0 && readerCardService.addReaderCard(readerInfo, password)) {

redirectAttributes.addFlashAttribute("succ", "添加读者信息成功！");

} else {

redirectAttributes.addFlashAttribute("succ", "添加读者信息失败！");

}

return "redirect:/allreaders.html";

}

**涉及到的sql语句如下（readerCard.xml）:**

<insert id="addReaderInfo" parameterType="com.library.bean.ReaderInfo">

insert into reader\_info values

(null, #{name},#{sex},#{birth},#{address},#{phone})

</insert>

## 6.6 数据库连接的实现

本系统在数据库连接方面，使用统一的连接数据库的方法，使用maven管理相关jar包后，调用其中的类。通过db.properties配置文件读取数据库信息和登陆数据库帐户的用户名和密码，实现数据库的连接。

**db.properties文件代码如下：**

jdbc.driver=com.mysql.jdbc.Driver

jdbc.url=jdbc:mysql://localhost:3306/library?useUnicode=true&characterEncoding=UTF-8&serverTimezone=Asia/Shanghai

jdbc.username=root//数据库账号

jdbc.password=wuhuai123//数据库密码

# 7 测试

在开发大型软件系统的过程中，需要面对错综复杂的问题，因此，在软件生存周期的每个阶段都不可避免地会产生错误。我们力求在每个阶段结束之前通过严格的技术审查，尽可能早的发现并纠正错误。测试的目的就是在软件投入生产性运行之前，尽可能多地发现软件中的错误。目前软件测试仍然是保证软件质量的关键步骤，它是对软件规格说明、设计和编码的最后复审。

尽管面向对象技术的基本思想保证了软件应该有更高的质量，但无论采用什么样的编程技术，编程人员的错误都是不可避免的，而且由于面向对象技术开发的软件代码重用率高，更需要严格测试，避免错误的繁衍。因此，软件测试在面向对象编程里更具有它的重要性。

与开发过程类似，测试过程也必须分步骤进行，每个步骤在逻辑上是前一个步骤地继续。大型软件系统通常由若干个子系统组成，每个子系统又由许多模块组成。因此，大型软件系统的测试的基本层次如图7.1所示。



图7.1 测试的基本层次

(1) 单元测试：单元测试的用例从单元详细设计中导出。在单元测试中可以采用功能性测试和结构性测试两种。

(2) 集成测试：在这个测试步骤中所发现的往往是概要设计的错误。

(3) 系统测试：在这个测试步骤中所发现的往往是需求规格说明的错误。一般来说，系统测试是功能性测试，不是结构性测试。

## 7.1登录

用户在登录界面输入属于自己的用户名和账号，若密码正确，登录成功，跳转至系统的主界面，反之登陆失败，提示密码错误，如下图7.1所示。



图7.1 登录

## 7.2 查询图书

读者与图书管理员可以通过输入自己想要查询的图书关键字查询相关图书信息，这些关键字包括书名、作者、出版社、语言以及简介；若存在，显示相关图书的信息，如下图7.2所示；若不存在，返回图书不存在的提示框，如下图7.3所示。



图7.2 查询结果



图7.3 没有匹配的图书

## 7.3 修改密码

读者或者图书管理员登录系统后，能修改自己的密码，点击修改密码按钮，跳转至修改密码界面，用户分别输入自己的旧密码、新密码和确认新密码后，点击提交按钮，系统将提示密码修改成功信息，若旧密码输入错误，将提示密码错误，若两次密码输入不一致，则返回不一致的信息，如下图7.4所示。

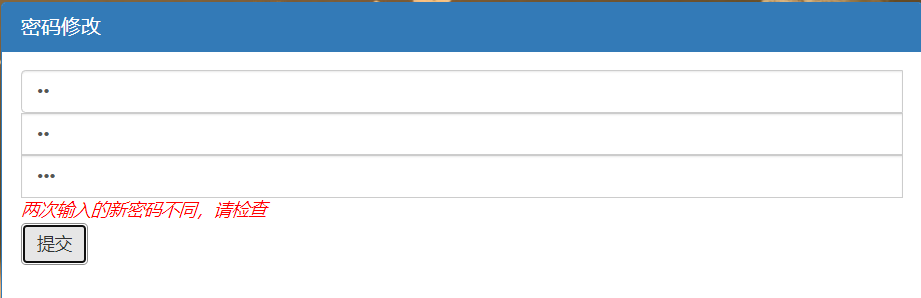


图7.4 修改密码

## 7.4入库图书

当图书馆有新书入库时，图书馆管理员可以在系统中选择增加图书功能，添加新书，当输入新书完整信息后，点击“添加”按钮，如下图7.5所示，即可完成添加操作。



图7.5 入库图书

## 7.5注册读者

身为图书馆管理员，可以注册新的读者，在增加读者的界面分别输入读者的相关信息，点击添加，如下图7.6所示，即可注册新的读者，读者的登录账号为读书号，为自增长段，如下图7.7所示。



图7.6注册读者



图7.7 注册成功

# 8 总结

通过这次的课程设计，可以巩固我们对数据库基本原理和基础理论的理解，掌握数据库应用系统设计开发的基本方法，进一步提高我们综合运用所学知识的能力。

　　当我们这组决定做图书馆管理系统时，我们并没有着手写程序。而是大家一起商量这个系统概述、系统目标、系统需求、业务流程分析、数据流程分析和数据词典。当这些都准备好了之后，我们进行模块的分工。每个人都有自己的模块设计，而且写出来的代码要求可以实现相应模块的功能，得到理想的效果。当每个人都把自己的分工做好了，最后会由一个人把这些全部组合搭建在一起。我们使用的是jsp和SSM框架技术相互嵌套使用，当一个系统做好了之后，我会好好地把程序都看一遍，理会其中的奥秘。

　　通过两周的系统开发，系统完成。在系统的开发过程中，以前学习感到很抽象的课程，如软件工程、数据库原理、面向对象程序设等变得清晰起来。感觉到这几门理论课程的重要性。

以前做一些应用的系统开发，随意性大，想到那里做到那里，没有统一的规划，一旦系统成型，出现错误返工修改工作量大，最终的系统可靠性不高。另外一直以来，进行程序设计均采用结构化开发方法，对系统的全貌难以在开发前看到，导致实现的系统不满足需求。

　　知识的获得是无止境的，只要你想学，只要你行动，没有什么会难倒我们的。回首这一个多星期的课程设计，我很欣慰。因为我有了动力，有了勇气。谢谢老师对我们的不懈帮助，谢谢学校给了我们这一次实践的机会，也谢谢组员们的关怀。这些美好的回忆美好的东西将永远伴随着我。

# 参考文献

[1]刘纪刚。高校图书馆自建特色数据库的着作权分析[J].图书馆研究与工作，2017,（01）：52-57.[2017-08-25].

[2]戴声，王波。基于图数据库的大规模配置管理数据库关联查询[J].计算机与现代化，2017,（01）：51-56.（2017-01-12）[2017-08-25]

[3] [美]Todd Cook. JSP从入门到精通[M].北京：电子工业出版社，2003. 1-385.

[4] Paul C.Jorgensen.Software Testing A Craftsman’s Approach(second Edition)[M].China Machine Press,2003,170-175.