**JAVA + MySQL综合实训**

**报 告**

**题 目：** 图书管理系统

**院 （系）：** 信息工程系

**专 业：** 计算机科学与技术

**学生姓名：** 陈彦志

**学 号：** 1651200111

**指导教师：** 龙丹

**时 间：** 2018年12月23日

摘 要

本文介绍了图书管理系统开发的整个过程，该系统通过使用MySQL数据库作为后台数据库，通过使用MVC三层架构组织各个模块。该图书管理系统主要用于对个人图书日常工作的管理，主要任务是用计算机对用户管理员、图书等各种信息进行日常管理，如查询、修改、增加、删除、图书借阅和归还的管理等。本系统实现了图书的信息化管理，完成了图书管理系统的基本功能。该系统主要包括用户登录子系统、用户管理子系统、图书管理子系统、图书借阅信息与归还子系统三个部分，各部分主要功能如下： 用户登录子系统：用户登录。 用户管理子系统：用于用户信息的管理，包括用户信息的修改。 图书管理子系统：用于图书信息的管理，如图书信息显示、图书信息修改、图书信息删除、图书信息增加。 图书借阅与归还子系统：包括图书借阅信息显示以及归还图书。

关键字：Java, MySQL, 图书管理系统, MVC三层架构

Abstract

This paper introduces the whole process of library management system development. By using MySQL database as the back-end database, the system organizes each module by using MVC three-tier architecture. The book management system is mainly used for the management of the daily work of individual books. The main task is to use the computer to manage various information such as user administrators and books, such as query, modification, addition, deletion, book borrowing and return management. . The system realizes the information management of books and completes the basic functions of the book management system. The system mainly includes three parts: user login subsystem, user management subsystem, library management subsystem, book borrowing information and return subsystem. The main functions of each part are as follows: User login subsystem: user login. User Management Subsystem: Used for management of user information, including modification of user information. Book management subsystem: used for the management of book information, such as book information display, book information modification, book information deletion, book information increase. Book lending and returning subsystem: including book lending information display and returning books.

Keywords: Java, MySQL, library management system, MVC three-tier architecture

目录

[摘 要 2](#_Toc533333824)

[引言 5](#_Toc533333825)

[1.问题定义 5](#_Toc533333826)

[1.1要解决的问题 5](#_Toc533333827)

[1.2要求 5](#_Toc533333828)

[2.可行性研究 5](#_Toc533333829)

[2.1 行得通的解决办法 5](#_Toc533333830)

[2.2 系统流程图，描绘物理系统，表达数据在系统各部件之间流动的情况 6](#_Toc533333831)

[2.3数据流图，描绘信息流和数据从输入移动到输出的过程中所经受的变换 6](#_Toc533333832)

[2.4数据字典，通常数据字典和数据流图共同构成系统的逻辑模型，没有数据字典精确定义数据流图中每个元素，数据流图就不够严密；然而没有数据流图，数据字典也很难发挥作用。 7](#_Toc533333833)

[3.需求分析 7](#_Toc533333834)

[3.1必须完成的功能 7](#_Toc533333835)

[3.2 数据库设计 8](#_Toc533333836)

[3.2.1 实体联系图（ER图） 8](#_Toc533333837)

[3.2.2关系模式 8](#_Toc533333838)

[3.2.3 数据库表 8](#_Toc533333839)

[3.3 状态转换图，描述作为外部事件结果的软件行为 9](#_Toc533333840)

[3.4 软件需求规格说明书 9](#_Toc533333841)

[4.总体设计 11](#_Toc533333842)

[4.1.层次图，描绘软件层次结构 11](#_Toc533333843)

[4.2 结构图，软件结构设计 11](#_Toc533333844)

[5.详细设计 12](#_Toc533333845)

[5.1 主程序流程图 12](#_Toc533333846)

[5.2 借阅图书程序流程图 13](#_Toc533333847)

[5.3 图书信息管理程序流程图 14](#_Toc533333848)

[5.4程序描述 14](#_Toc533333849)

[6. 编码和单元测试 38](#_Toc533333850)

[6.1.图书信息增加测试 38](#_Toc533333851)

[6.2.图片信息删除测试 39](#_Toc533333852)

[6.3.图书信息修改测试 40](#_Toc533333853)

[6.4.图书信息归还测试 41](#_Toc533333854)

[6.5用户更改密码测试 42](#_Toc533333855)

[实训总结： 43](#_Toc533333856)

[参考文献 45](#_Toc533333857)

# 引言

该图书管理系统是采用Eclipse开发软件, 利用java语言开发的一个数据库管理系统。开发本系统的基本步骤为:问题定义、可行性研究、需求分析、总体设计、详细设计、软件测试。在系统分析中先后用到系统流程图、数据流图、数据字典、状态转换图、规格说明书、层次图、结构图和程序流程图。在系统的设计过程中, 采用了MVC三层架构模式, 比较详细、条理清晰地组织了各个模块，将多个类需要用到的模块封装到公共类，减少代码的冗余。在数据库库的数据表的设计当中, 力求设计简单实用。在系统具体实行阶段中，尽量使得窗机简洁美观。

# 1.问题定义

## 1.1要解决的问题

本软件要实现管理员的登录、图书信息的增加、删除、修改，图书分类以图书馆分类方法分类和图书信息的显示，以及显示借阅书籍和借阅书籍的归还。同时对于管理员来说，还要实现更改密码功能。

## 1.2要求

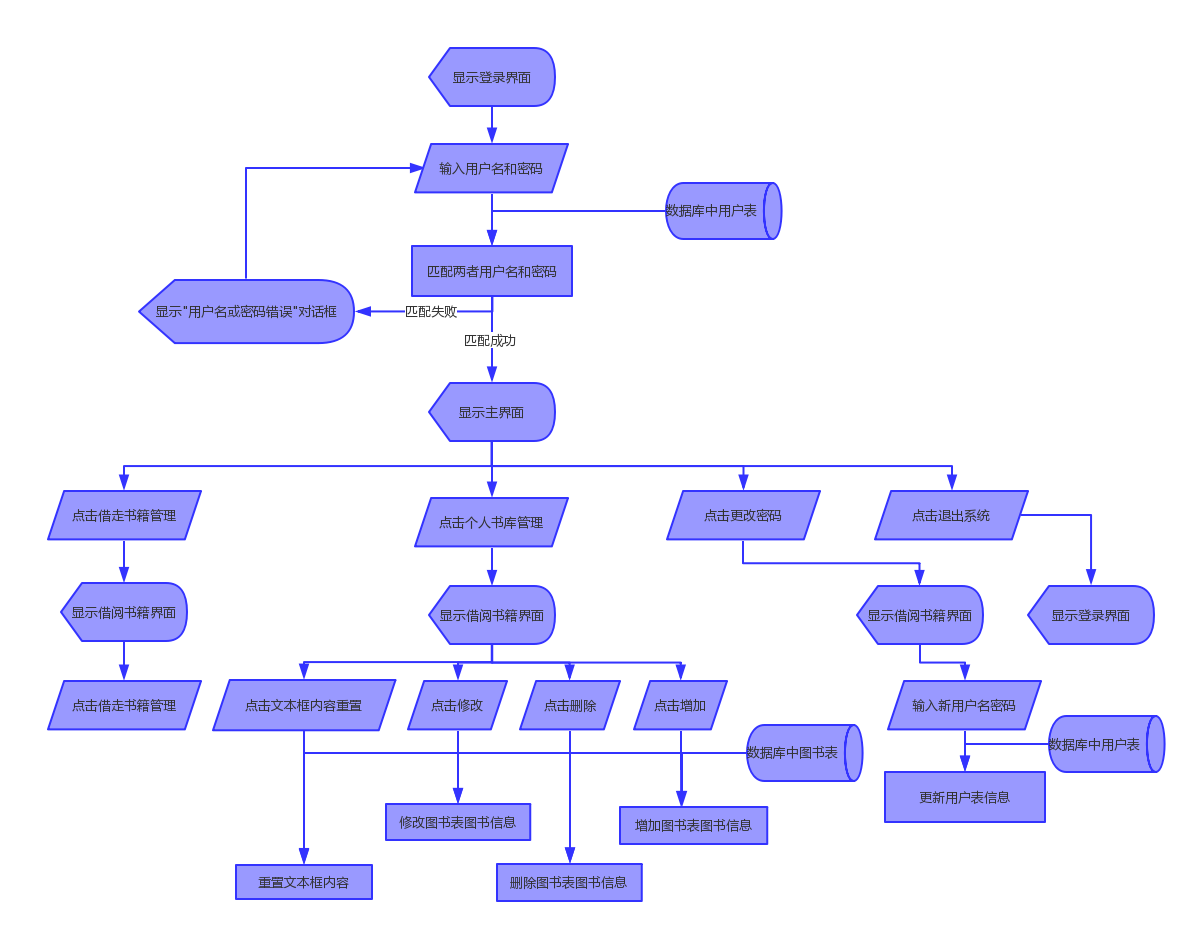
* 界面简洁美观
* 使用MySQL数据库
* 使用Java语言
* 软件支持完整的业务流程
* 提供一定的统计分析功能

# 2.可行性研究

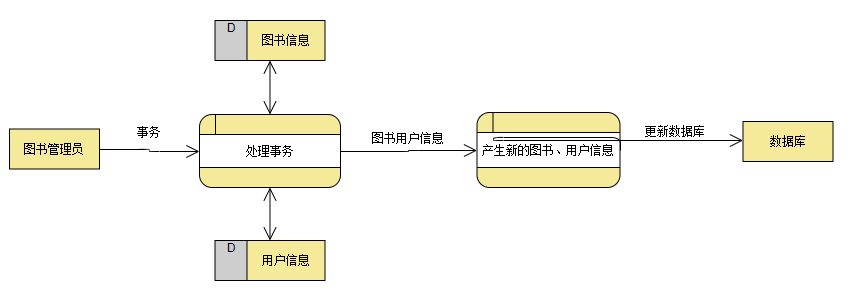
## 2.1 行得通的解决办法

使用Java + MySQL设计，利用Java Swing设计窗体，将主窗体底层面板插入背景图片标签后，通过设置窗体转换为容器后再将容器转换为的面板为透明来实现设置背景图片。通过表格组件与数据库的交互来显示当前图书全部信息。通过文本框和按钮与表格和数据库的交互来显示增加、删除、修改数据库数据后图书的信息。

## 2.2 系统流程图，描绘物理系统，表达数据在系统各部件之间流动的情况



## 2.3数据流图，描绘信息流和数据从输入移动到输出的过程中所经受的变换



## 2.4数据字典，通常数据字典和数据流图共同构成系统的逻辑模型，没有数据字典精确定义数据流图中每个元素，数据流图就不够严密；然而没有数据流图，数据字典也很难发挥作用。

名字：图书信息

别名：

描述：数据库中图书表各字段的集合，包含图书名称，图书作者、图书价格、ISBN、出版社、图书分类号、借书人姓名、借书人电话号码

定义：图书信息 = 图书名称 + 图书作者 + 图书价格 + ISBN + 出版社 + 图书分类号 + 借书人姓名 + 借书人电话号码

位置：数据库、事务

名字：用户信息

别名：

描述：数据库中用户表各字段的集合，包含用户名、密码

定义：用户信息 = 用户名 + 密码

位置：数据库 + 事务

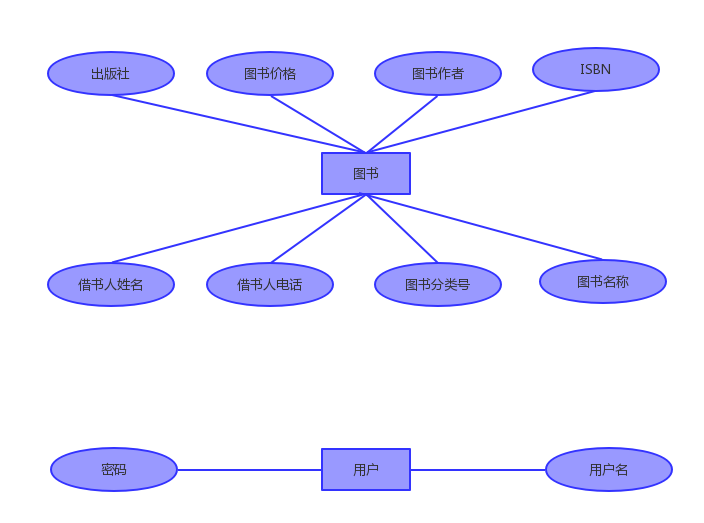
# 3.需求分析

## 3.1必须完成的功能

* 管理员的登录
* 图书信息的增删改查
* 图书借阅信息的显示以及图书归还
* 管理员更改密码

## 3.2 数据库设计

### 3.2.1 实体联系图（ER图）



### 3.2.2关系模式

管理员（管理员编号，用户名，密码）

图书（图书编号，图书名称，图书价格，图书作者，ISBN，出版社，图书分类号，借书人姓名，借书人电话）

### 3.2.3 数据库表

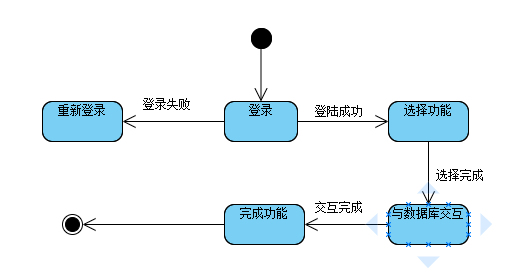
表1.1 用户表（tb\_user）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 是否主键 |
| id | int | 255 | 不为空 | 主键 |
| User | Varchar | 255 |  |  |
| Passwd | Varchar | 255 |  |  |

表1.2 图书表（tb\_books）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 主键否 | 是否为空 | 描述 |
| ID | int | 255 | 主键 | 不为空 | 图书编号 |
| Book\_name | Varchar | 255 |  | 不为空 | 图书名称 |
| ISBN | Varchar | 255 |  | 不为空 | ISBN |
| Book\_price | Decimal | 10 |  |  | 图书价格 |
| Book\_author | Varchar | 255 |  |  | 图书作者 |
| Published\_house | Varchar | 255 |  |  |  |
| Borrower\_name | Varchar | 255 |  |  |  |
| Borrower\_phone | Varchar | 255 |  |  |  |

## 3.3 状态转换图，描述作为外部事件结果的软件行为



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 功能名称 | 图书借阅管理 | 功能编号 |  | 设计者 | 陈彦志 |
| 功能需求提出者： |  | 陈彦志 | | 完成时间 | 2018.12.18 |
| 功能修改提出者： |  | 陈彦志 | | 修改时间 | 2018.12.19 |
| 功能修改批准者 | 陈彦志 | 功能修改者 | 陈彦志 | 修改次数 | 1 |
| FDD功能框图 | | | | | |
| 说明 | 显示借阅书籍的名称，借书人姓名和借书人电话号码，包含归还图书功能。 | | | | |

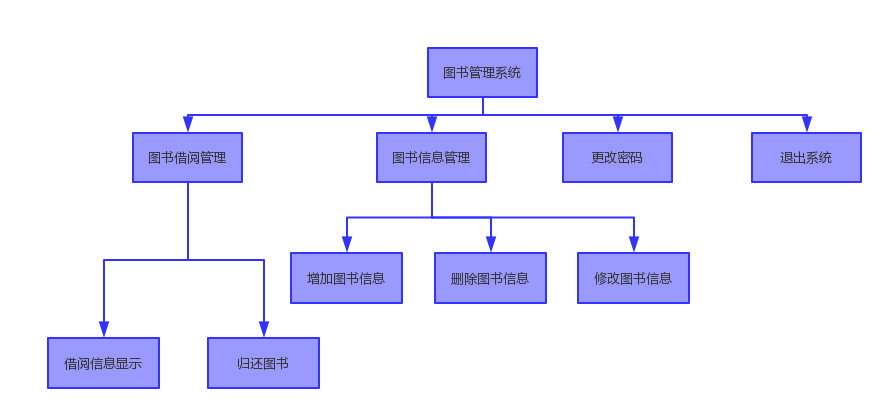
## 3.4 软件需求规格说明书

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 功能名称 | 图书信息管理 | 功能编号 |  | 设计者 | 陈彦志 |
| 功能需求提出者： | | 陈彦志 | | 完成时间 | 2018.12.18 |
| 功能修改提出者： | | 陈彦志 | | 修改时间 | 2018.12.19 |
| 功能修改批准者 | 陈彦志 | 功能修改者 | 陈彦志 | 修改次数 | 1 |
| FDD功能框图 | | | | | |
| 说明 | 显示图书的名称、作者、价格、ISBN、分类号等信息和增加图书信息、删除图书信息、修改图书信息等功能。 | | | | |

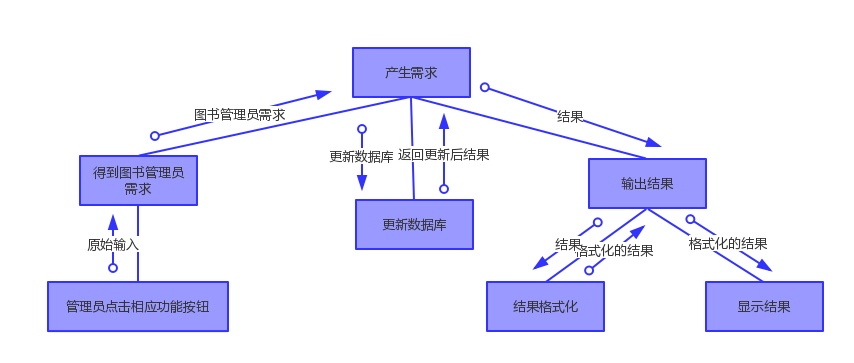
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 功能名称 | 更改密码 | 功能编号 |  | 设计者 | 陈彦志 |
| 功能需求提出者： | | 陈彦志 | | 完成时间 | 2018.12.18 |
| 功能修改提出者： | | 陈彦志 | | 修改时间 | 2018.12.19 |
| 功能修改批准者 | 陈彦志 | 功能修改者 | 陈彦志 | 修改次数 | 1 |
| FDD功能框图 | | | | | |
| 说明 | 图书管理员密码更改 | | | | |

# 4.总体设计

## 4.1.层次图，描绘软件层次结构

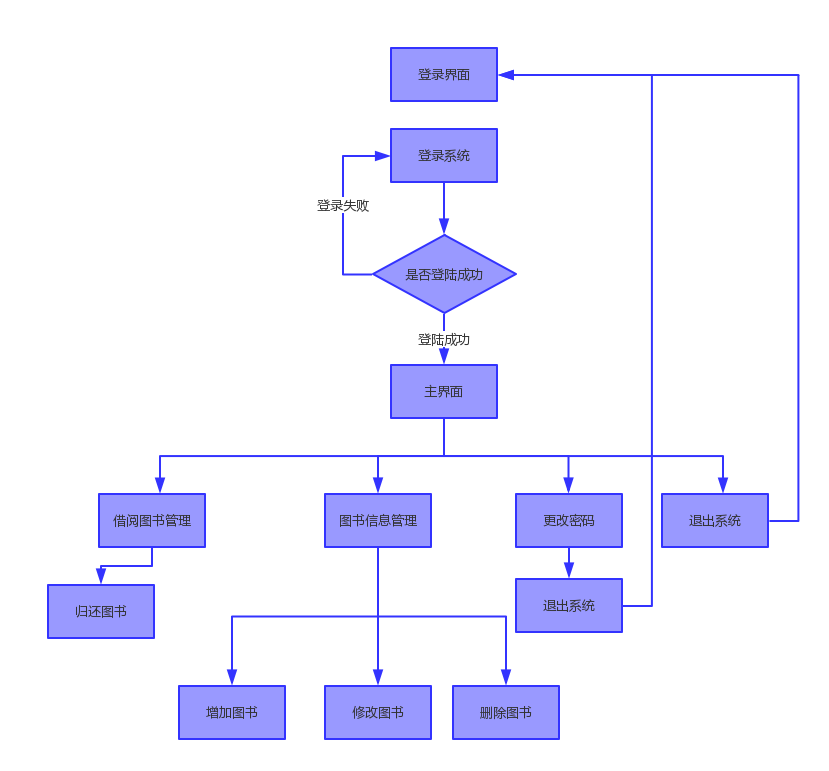


## 4.2 结构图，软件结构设计

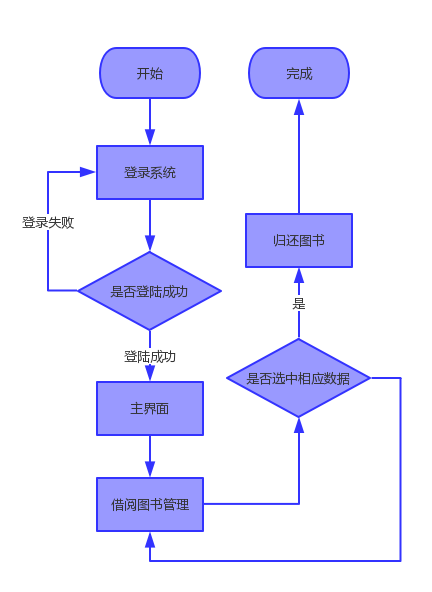


# 5.详细设计

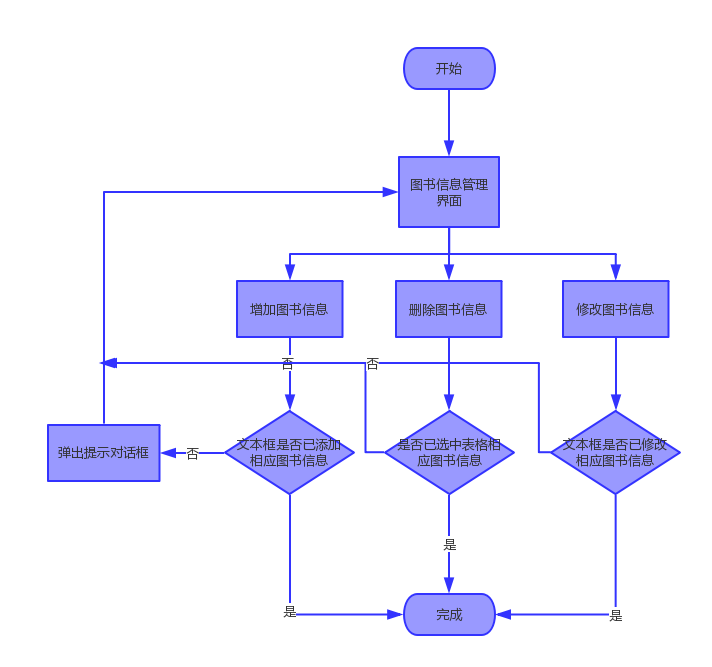
## 5.1 主程序流程图



## 5.2 借阅图书程序流程图



## 5.3 图书信息管理程序流程图



## 5.4程序描述

1. 主菜单模块：

选择相应的功能，需要在当前页面继续显示。

功能描述

功能类型：其他

功能概述：总体归纳图书管理功能

前提业务：登陆模块

后续业务：图书借阅管理、图书信息管理、更改密码

功能约束：权限约束

约束描述：

操作权限：图书管理员

主菜单，展示所有功能，进入功能各项模块所必需.

**public** **class** MainFrame {

JFrame frame = **new** JFrame("个人书屋");

Container container = frame.getContentPane();

**public** MainFrame() {

// 设置背景图片

**new** BackgroundImage(frame,container,"mainFrame.jpg");

// 添加工具栏以及各组件和监听事件

**new** MenuBar(frame);

// 设置窗口大小、位置、可视、默认关闭方式等

**new** FrameOption(frame);

}

}

工具栏类

**public** **class** MenuBar {

JMenuBar menuBar;

JMenuItem menuItemBookInformation;

JMenuItem menuItemBorrowManage;

JMenuItem menuItemChangUser;

JMenuItem menuItemExit;

**public** MenuBar(JFrame frame) {

menuBar = **new** JMenuBar();

// 图书信息管理菜单项

menuItemBookInformation = **new** JMenuItem();

setMenuItemBookInformationn(frame);

// 图书借阅管理菜单项

menuItemBorrowManage = **new** JMenuItem();

setMenuItemBorrowManage(frame);

// 用户信息更改菜单项

menuItemChangUser = **new** JMenuItem();

setMenuItemChangeUser(frame);

// 退出系统菜单项

menuItemExit = **new** JMenuItem();

setMenuItemExit(frame);

menuBar.add(menuItemBorrowManage);

menuBar.add(menuItemBookInformation);

menuBar.add(menuItemChangUser);

menuBar.add(menuItemExit);

frame.setJMenuBar(menuBar);

}

/\*\*

\* 设置退出系统菜单项

\*/

**private** **void** setMenuItemExit(JFrame frame) {

menuItemExit.setIcon(**new** ImageIcon("res/menuItemExit.jpg"));

menuItemExit.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent e) {

// 使父窗体不可见

frame.setVisible(**false**);

**new** Login();

}

});

}

**private** **void** setMenuItemChangeUser(JFrame frame) {

menuItemChangUser.setIcon(**new** ImageIcon("res/menuItemChangePassword.jpg"));

menuItemChangUser.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent e) {

// 使父窗体不可见

frame.setVisible(**false**);

**new** ChangeUserInformation();

}

});

}

**private** **void** setMenuItemBorrowManage(JFrame frame) {

menuItemBorrowManage.setIcon(**new** ImageIcon("res/menuBookCategoryManage.jpg"));

menuItemBorrowManage.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent e) {

// 使父窗体不可见

frame.setVisible(**false**);

**new** BookBorrow();

}

});

}

**private** **void** setMenuItemBookInformationn(JFrame frame) {

menuItemBookInformation.setIcon(**new** ImageIcon("res/menuBookInformationManage.jpg"));

menuItemBookInformation.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent e) {

// 使父窗体不可见

frame.setVisible(**false**);

**new** BookInformation();

}

});

}

}

设置窗口相关设置类

**public** **class** FrameOption {

**public** FrameOption(JFrame frame) {

frame.setVisible(**true**);

// 窗口不可调整大小

frame.setResizable(**false**);

frame.setSize(800, 508);

frame.setLocation(200,100);

frame.setDefaultCloseOperation(WindowConstants.***EXIT\_ON\_CLOSE***);

}

}

（2）图书借阅管理模块

需要说明的问题

操作权限：图书管理员

功能类型：数据库处理

功能概述：显示图书借阅信息、归还图书

前提业务：登录系统

后继业务：

功能约束：权限约束

约束描述：

操作权限：面向图书管理员

**public** **class** BookBorrow {

JFrame frame = **new** JFrame("个人书屋");

Container container = frame.getContentPane();

// 表格

JTable table;

// 显示表格的滚动面板

JScrollPane scrollPane;

// 归还图书按钮

JButton buttonReturn;

BorrowAction borrowAction;

**public** BookBorrow() {

frame.setLayout(**null**);

// 设置背景图片

**new** BackgroundImage(frame,container,"BookBorrow.jpg");

// 添加工具栏以及各组件和监听事件

**new** MenuBar(frame);

// 设置表格

setTable();

// 归还图书按钮

buttonReturn = **new** JButton();

setButtonReturn();

container.add(buttonReturn);

container.add(scrollPane);

// 设置窗口大小、位置、可视、默认关闭方式等

**new** FrameOption(frame);

}

/\*\*

\* 设置归还图书按钮

\*/

**private** **void** setButtonReturn() {

buttonReturn.setBounds(580,390,100,25);

buttonReturn.setIcon(**new** ImageIcon("res/button\_return.jpg"));

buttonReturn.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent e) {

**try** {

borrowAction = **new** BorrowAction();

borrowAction.BorrowBook(table);

frame.setVisible(**false**);

**new** BookBorrow();

} **catch**(Exception e1) {

JOptionPane.*showMessageDialog*(**null**,"请先选中要归还的表格项","错误"

, JOptionPane.***PLAIN\_MESSAGE***);

}

}

});

}

/\*\*

\* 设置表格

\*/

**private** **void** setTable() {

String[] columnNames = {"ID", "图书名称", "借书人姓名", "借书人电话"};

**try** {

BorrowAction borrowAction = **new** BorrowAction();

Object[][] results = borrowAction.initializTable(columnNames);

table = **new** JTable(results,columnNames);

**new** SetTableColumnCenter(table);

scrollPane = **new** JScrollPane(table);

scrollPane.setViewportView(table);

scrollPane.setBounds(20,80,760,190);

} **catch**(Exception e) {

e.printStackTrace();

}

}

}

借还书相应行为类

**public** **class** BorrowAction {

BookDao borrowerDao;

/\*\*

\* 初始化借还书管理窗体表格

\* **@return**

\* results

\*/

@SuppressWarnings("rawtypes")

**public** Object[][] initializTable(String[] columnNames) **throws** Exception{

borrowerDao = **new** BookDao();

List list = borrowerDao.borrowQuery();

Object[][] results = **new** Object[list.size()][columnNames.length];

**for**(**int** i = 0; i < list.size(); i++) {

Book book = (Book)list.get(i);

results[i][0] = book.getID();

results[i][1] = book.getBookName();

results[i][2] = book.getBorrowerName();

results[i][3] = book.getBorrowerPhone();

}

**return** results;

}

/\*\*

\* 归还图书

\*/

**public** **void** BorrowBook (JTable table) **throws** Exception{

borrowerDao=**new** BookDao();

Book book=**new** Book();

**int** selRow = table.getSelectedRow();

**int** ID = Integer.*parseInt*(table.getValueAt(selRow, 0).toString());

book.setID(ID);

book.setBorrowerName(**null**);

book.setBorrowerPhone(**null**);

// 归还图书

borrowerDao.returnBook(book);

}

}

借还书数据库操作类

/\*\*

\* 查询借阅信息

\*

\* **@return**

\* bookList

\*/

**public** List<Book> borrowQuery() **throws** Exception{

Connection con = DBUtil.*getConnection*();

Statement stmt = con.createStatement();

ResultSet rs = stmt.executeQuery(""

// ID、书名、借书人姓名、借书人电话

+ "SELECT ID, book\_name, borrower\_name, borrower\_phone "

+ "FROM tb\_books "

+ "WHERE borrower\_name IS NOT NULL"

);

List<Book> bookList = **new** ArrayList<Book>();

Book book = **null**;

**while** (rs.next()){//如果对象中有数据，就会循环打印出来

book = **new** Book();

book.setID(rs.getInt("ID"));

book.setBookName(rs.getString("book\_name"));

book.setBorrowerName(rs.getString("borrower\_name"));

book.setBorrowerPhone(rs.getString("borrower\_phone"));

bookList.add(book);

}

**return** bookList;

}

/\*\*

\* 更新图书信息，归还图书

\*/

**public** **void** returnBook(Book book) **throws** SQLException{

Connection con=DBUtil.*getConnection*();//首先拿到数据库的连接

String sql="UPDATE tb\_books "

// ISBN、图书名称、作者、价格

+ "SET "

// 借书人姓名、借书人电话

+ "borrower\_name = ?, borrower\_phone = ? "

// 参数用?表示，相当于占位符

+ "WHERE ID = ?";

// 预编译sql语句

PreparedStatement psmt = con.prepareStatement(sql);

// 先对应SQL语句，给SQL语句传递参数

psmt.setString(1, book.getBorrowerName());

psmt.setString(2, book.getBorrowerPhone());

psmt.setInt(3, book.getID());

// 执行SQL语句

psmt.execute();

}

（3）图书信息管理模块

需要说明的问题

操作权限：图书管理员

功能类型：数据库处理

功能概述：显示图书信息、增加图书、删除图书、修改图书

前提业务：登录系统

后继业务：

功能约束：权限约束

约束描述：

操作权限：面向图书管理员

/\*\*

\* 图书信息管理界面类

\*

\* **@author** 1651200111 陈彦志

\*/

@SuppressWarnings("serial")

**public** **class** BookInformation **extends** JFrame {

JFrame frame = **new** JFrame("个人书屋");

Container container = frame.getContentPane();

// 增加、删除、修改按钮

JButton buttonAdd, buttonDel, buttonChange,buttonReset;

// ISBN、图书名称、图书价格、图书作者文本框

JTextField textFieldISBN, textFieldBookName, textFieldPrice, textFieldAuthor;

// 出版社、图书分类号、借书人姓名文本框

JTextField textFieldPublishedHouse, textFieldBookCategory;

// 借书人姓名、借书人电话

JTextField textFieldBorrowName, textFieldBorrowPhone;

// 表格

JTable table;

// 显示表格的滚动面板

JScrollPane scrollPane;

BookAction bookAction;

**public** BookInformation() {

frame.setLayout(**null**);

// 设置背景图片

**new** BackgroundImage(frame,container,"BookInformation.jpg");

// 添加工具栏以及各组件和监听事件

**new** MenuBar(frame);

bookAction = **new** BookAction();

// ISBN文本框

textFieldISBN = **new** JTextField();

setTextFieldISBN();

// 图书名称文本框

textFieldBookName = **new** JTextField();

setTextFieldBookName();

// 图书价格文本框

textFieldPrice = **new** JTextField();

setTextFieldBookPrice();

// 图书作者文本框

textFieldAuthor = **new** JTextField();

setTextFieldAuthor();

// 出版社文本框

textFieldPublishedHouse = **new** JTextField();

setTextFieldPublishedHouse();

// 图书分类号文本框

textFieldBookCategory = **new** JTextField();

setTextFieldBookCategory();

// 借书人姓名文本框

textFieldBorrowName = **new** JTextField();

setTextFieldBorrowName();

// 借书人电话文本框

textFieldBorrowPhone = **new** JTextField();

setTextFieldBorrowPhone();

// 设置窗体表格

setTable();

// 增加按钮

buttonAdd = **new** JButton();

setButtonAdd();

// 删除按钮

buttonDel = **new** JButton();

setButtonDel();

// 修改按钮

buttonChange = **new** JButton();

setButtonChange();

// 重置按钮

buttonReset = **new** JButton();

setButtonReset();

container.add(scrollPane);

container.add(buttonAdd);

container.add(buttonDel);

container.add(buttonReset);

container.add(buttonChange);

container.add(textFieldISBN);

container.add(textFieldBookName);

container.add(textFieldAuthor);

container.add(textFieldPrice);

container.add(textFieldBookCategory);

container.add(textFieldPublishedHouse);

container.add(textFieldBorrowName);

container.add(textFieldBorrowPhone);

// 设置窗口大小、位置、可视、默认关闭方式等

**new** FrameOption(frame);

}

/\*\*

\* 设置借书人电话文本框

\*/

**private** **void** setTextFieldBorrowPhone() {

textFieldBorrowPhone.setBounds(490,312,232,23);

}

/\*\*

\* 设置借书人姓名文本框

\*/

**private** **void** setTextFieldBorrowName() {

textFieldBorrowName.setBounds(150,312,200,23);

}

/\*\*

\* 设置修改按钮

\*/

**private** **void** setButtonChange() {

buttonChange.setBounds(470,390,60,25);

buttonChange.setIcon(**new** ImageIcon("res/button\_change.jpg"));

buttonChange.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent e) {

**try** {

bookAction.changeBookInformation(textFieldISBN, textFieldBookName

,textFieldPrice, textFieldAuthor, textFieldPublishedHouse

, textFieldBookCategory, textFieldBorrowName, textFieldBorrowPhone, table);

frame.setVisible(**false**);

**new** BookInformation();

}**catch**(Exception e1) {

JOptionPane.*showMessageDialog*(**null**,"表中没有该数据","错误"

, JOptionPane.***PLAIN\_MESSAGE***);

}

}

});

}

/\*\*

\* 设置图书分类号文本框

\*/

**private** **void** setTextFieldBookCategory() {

textFieldBookCategory.setBounds(582,340,140,23);

}

/\*\*

\* 设置出版社文本框

\*/

**private** **void** setTextFieldPublishedHouse() {

textFieldPublishedHouse.setBounds(348,340,140,23);

}

/\*\*

\* 设置图书作者文本框

\*/

**private** **void** setTextFieldAuthor() {

textFieldAuthor.setBounds(586,280,140,23);

}

/\*\*

\* 设置图书价格文本框

\*/

**private** **void** setTextFieldBookPrice() {

textFieldPrice.setBounds(120,340,140,23);

}

/\*\*

\* 设置图书名称文本框

\*/

**private** **void** setTextFieldBookName() {

textFieldBookName.setBounds(348,280,140,23);

}

/\*\*

\* 设置ISBN文本框

\*/

**private** **void** setTextFieldISBN() {

// 限制文本框长度为17

textFieldISBN.setDocument(**new** LimitTextLength(17));

textFieldISBN.setBounds(120,280,140,23);

}

/\*\*

\* 设置删除按钮

\*/

**private** **void** setButtonDel() {

buttonDel.setBounds(580,390,60,25);

buttonDel.setIcon(**new** ImageIcon("res/button\_del.jpg"));

buttonDel.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent e) {

// **TODO** Auto-generated method stub

**try** {

bookAction.delBookInformation(table);

frame.setVisible(**false**);

**new** BookInformation();

} **catch**(Exception e1) {

e1.printStackTrace();

}

}

});

}

/\*\*

\* 设置文本框重置按钮

\*/

**private** **void** setButtonReset() {

buttonReset.setBounds(270,390,150,25);

buttonReset.setIcon(**new** ImageIcon("res/button\_textReset.jpg"));

buttonReset.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent e) {

// **TODO** Auto-generated method stub

textFieldBookName.setText("");

textFieldAuthor.setText("");

textFieldISBN.setText("");

textFieldPrice.setText("");

textFieldBookCategory.setText("");

textFieldPublishedHouse.setText("");

}

});

}

/\*\*

\* 设置添加按钮

\*/

**private** **void** setButtonAdd() {

buttonAdd.setBounds(700,390,60,25);

buttonAdd.setIcon(**new** ImageIcon("res/button\_add.jpg"));

buttonAdd.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent e) {

// **TODO** Auto-generated method stub

**if**(textFieldISBN.getText().length() == 0) {

JOptionPane.*showMessageDialog*(**null**,"ISBN号不能为空","错误"

, JOptionPane.***PLAIN\_MESSAGE***);

}

**else** **if**(textFieldISBN.getText().length() != 17) {

JOptionPane.*showMessageDialog*(**null**,"ISBN号位数必须是13位","错误"

, JOptionPane.***PLAIN\_MESSAGE***);

}

**else** **if**(textFieldBookName.getText().length() == 0) {

JOptionPane.*showMessageDialog*(**null**, "图书名称不能为空", "错误"

, JOptionPane.***PLAIN\_MESSAGE***);

}

**else** **if**(textFieldAuthor.getText().length() == 0) {

JOptionPane.*showMessageDialog*(**null**, "图书作者不能为空", "错误"

, JOptionPane.***PLAIN\_MESSAGE***);

}

**else** **if**(textFieldPrice.getText().length() == 0) {

JOptionPane.*showMessageDialog*(**null**, "图书价格不能为空", "错误"

, JOptionPane.***PLAIN\_MESSAGE***);

}

**else** **if**(textFieldPublishedHouse.getText().length() == 0) {

JOptionPane.*showMessageDialog*(**null**, "出版社不能为空", "错误"

, JOptionPane.***PLAIN\_MESSAGE***);

}

**else** **if**(textFieldBookCategory.getText().length() == 0) {

JOptionPane.*showMessageDialog*(**null**, "图书分类号不能为空", "错误"

, JOptionPane.***PLAIN\_MESSAGE***);

}

**else** **if**(textFieldPrice.getText().length() > 4) {

JOptionPane.*showMessageDialog*(**null**, "图书价格不能超过4位数", "错误"

, JOptionPane.***PLAIN\_MESSAGE***);

}

**else** {

**try** {

bookAction.addBookInformation(textFieldISBN, textFieldBookName, textFieldPrice,textFieldAuthor

,textFieldPublishedHouse,textFieldBookCategory,textFieldBorrowName,textFieldBorrowPhone);

frame.setVisible(**false**);

**new** BookInformation();

}**catch**(Exception e1) {

e1.printStackTrace();

}

}

}

});

}

/\*\*

\* 设置窗体表格

\*/

**private** **void** setTable() {

String[] columnNames = {"ID", "图书名称","图书作者","图书价格(元)","ISBN"

,"出版社","分类号", "借书人姓名", "借书人电话"

};

**try** {

BookAction bookAction = **new** BookAction();

Object[][] results = bookAction.initializTable(columnNames);

table = **new** JTable(results,columnNames);

// 设置表格字段居中

**new** SetTableColumnCenter(table);

scrollPane = **new** JScrollPane(table);

scrollPane.setViewportView(table);

scrollPane.setBounds(20,80,760,190);

table.addMouseListener(**new** MouseListener() {

@Override

**public** **void** mouseReleased(MouseEvent e) {

// **TODO** Auto-generated method stub

}

@Override

**public** **void** mousePressed(MouseEvent e) {

// **TODO** Auto-generated method stub

}

@Override

**public** **void** mouseExited(MouseEvent e) {

// **TODO** Auto-generated method stub

}

@Override

**public** **void** mouseEntered(MouseEvent e) {

// **TODO** Auto-generated method stub

}

@Override

**public** **void** mouseClicked(MouseEvent e) {

// **TODO** Auto-generated method stub

String ISBN, bookName, price, author;

String publishedHouse, category, borrowName, borrowPhone;

**int** selRow = table.getSelectedRow();

bookName = table.getValueAt(selRow, 1).toString();

author = table.getValueAt(selRow, 2).toString();

price = table.getValueAt(selRow, 3).toString();

ISBN = table.getValueAt(selRow, 4).toString();

publishedHouse = table.getValueAt(selRow, 5).toString();

category = table.getValueAt(selRow, 6).toString();

**if** (table.getValueAt(selRow, 7).toString() == **null**) {

borrowName = "";

}

**else** {

borrowName = table.getValueAt(selRow, 7).toString();

}

**if** (table.getValueAt(selRow, 8).toString() == **null**) {

borrowPhone = "";

}

**else** {

borrowPhone = table.getValueAt(selRow, 8).toString();

}

textFieldBookName.setText(bookName);

textFieldAuthor.setText(author);

textFieldPrice.setText(price);

textFieldISBN.setText(ISBN);

textFieldPublishedHouse.setText(publishedHouse);

textFieldBookCategory.setText(category);

textFieldBorrowName.setText(borrowName);

textFieldBorrowPhone.setText(borrowPhone);

}

});

} **catch**(Exception e) {

e.printStackTrace();

}

}

}

图书信息增加删除修改行为类

/\*\*

\* 图书信息行为控制类，包含增加图书、删除图书

\* 、 修改图书、和初始化个人书库管理窗体表格

\*

\* **@author** 1651200111 陈彦志

\*/

**public** **class** BookAction {

/\*\*

\* 初始化窗体表格

\* **@return**

\* results

\*/

@SuppressWarnings("rawtypes")

**public** Object[][] initializTable(String[] columnNames) **throws** Exception{

BookDao bookDao = **new** BookDao();

List list = bookDao.query();

Object[][] results = **new** Object[list.size()][columnNames.length];

**for**(**int** i = 0; i < list.size(); i++) {

Book book = (Book)list.get(i);

results[i][0] = book.getID();

results[i][1] = book.getBookName();

results[i][2] = book.getAuthor();

results[i][3] = book.getPrice();

results[i][4] = book.getISBN();

results[i][5] = book.getPublishHouse();

results[i][6] = book.getBookCategory();

String borrowerName = book.getBorrowerName();

**if** (borrowerName == **null**) {

borrowerName = "";

results[i][7] = borrowerName;

}

**else** {

results[i][7] = borrowerName;

}

String borrowerPhone = book.getBorrowerPhone();

**if** (borrowerPhone == **null**) {

borrowerPhone = "";

results[i][8] = borrowerPhone;

}

**else** {

results[i][8] = borrowerPhone;

}

}

**return** results;

}

/\*\*

\* 添加图书信息

\*/

**public** **void** addBookInformation (JTextField textFieldISBN, JTextField textFieldName

,JTextField textFieldPrice, JTextField textFieldAuthor, JTextField textFieldPublishedHouse

, JTextField textFieldBookCategory, JTextField textFieldBorrowName

, JTextField textFieldBorrowPhone) **throws** Exception {

BookDao bookDao=**new** BookDao();

Book book=**new** Book();

book.setISBN(textFieldISBN.getText());

book.setBookName(textFieldName.getText());

**float** price = Float.*parseFloat*(textFieldPrice.getText());

book.setPrice(price);

book.setAuthor(textFieldAuthor.getText());

book.setPublishHouse(textFieldPublishedHouse.getText());

book.setBookCategory(textFieldBookCategory.getText());

**if** (textFieldBorrowName.getText() == **null** ||textFieldBorrowName.getText() == "" ) {

book.setBorrowerName(**null**);

}

**else** {

book.setBorrowerName(textFieldBorrowName.getText());

}

**if** (textFieldBorrowPhone.getText() == **null** || textFieldBorrowPhone.getText() == "") {

book.setBorrowerPhone(**null**);

}

**else** {

book.setBorrowerPhone(textFieldBorrowPhone.getText());

}

//添加图书

bookDao.addBook(book);

}

/\*\*

\* 删除图书信息

\*/

**public** **void** delBookInformation (JTable table) **throws** Exception {

**int** selRow = table.getSelectedRow();

**int** ID = Integer.*parseInt*(table.getValueAt(selRow, 0).toString());

BookDao bookDao=**new** BookDao();

Book book=**new** Book();

book.setID(ID);

// 删除图书信息

bookDao.delBook(ID);

}

/\*\*

\* 修改图书信息

\*/

**public** **void** changeBookInformation (JTextField textFieldISBN, JTextField textFieldName

,JTextField textFieldPrice, JTextField textFieldAuthor, JTextField textFieldPublishedHouse

, JTextField textFieldBookCategory, JTextField textFieldBorrowerName

, JTextField textFieldBorrowerPhone, JTable table) **throws** Exception{

BookDao bookDao=**new** BookDao();

Book book=**new** Book();

**int** selRow = table.getSelectedRow();

**int** ID = Integer.*parseInt*(table.getValueAt(selRow, 0).toString());

book.setID(ID);

book.setISBN(textFieldISBN.getText());

book.setBookName(textFieldName.getText());

book.setAuthor(textFieldAuthor.getText());

**float** price = Float.*parseFloat*(textFieldPrice.getText());

book.setPrice(price);

book.setPublishHouse(textFieldPublishedHouse.getText());

book.setBookCategory(textFieldBookCategory.getText());

book.setBorrowerName(textFieldBorrowerName.getText());

book.setBorrowerPhone(textFieldBorrowerPhone.getText());

//修改图书

bookDao.changeBook(book);

}

}

图书信息增加删除修改数据库操作类

**public** **class** BookDao {

/\*\*

\* 增加图书信息

\*/

**public** **void** addBook(Book book) **throws** Exception{

// 首先拿到数据库的连接

Connection con = DBUtil.*getConnection*();

String sql="insert into tb\_books"

// ISBN、书名、图书价格、图书作者、出版社

+ "(ISBN, book\_name, book\_price, book\_author, published\_house,"

// 分类号、借书人姓名、借书人电话、借书日期，已借天数

+ "book\_category, borrower\_name, borrower\_phone) "

+ "values("

/\*

\* 参数用?表示，相当于占位符，然后在对参数进行赋值。当真正执行时，这些参数会加载在SQL语句中，把SQL语句拼接完整才去执行。这样就会减少对数据库的操作

\*/

+ "?,?,?,?,?,?,?,?)";

/\*

\* prepareStatement这个方法会将SQL语句加载到驱动程序conn集成程序中，

\* 但是并不直接执行,而是当它调用execute()方法的时候才真正执行；

\*/

PreparedStatement psmt = con.prepareStatement(sql);

// 先对应SQL语句，给SQL语句传递参数

psmt.setString(1, book.getISBN());

psmt.setString(2, book.getBookName());

psmt.setFloat(3, book.getPrice());

psmt.setString(4, book.getAuthor());

psmt.setString(5, book.getPublishHouse());

psmt.setString(6, book.getBookCategory());

**if** (book.getBorrowerName() == **null** || book.getBorrowerName() == "") {

psmt.setString(7, **null**);

}

**else** {

psmt.setString(7, book.getBorrowerName());

}

**if** (book.getBorrowerPhone() == **null** || book.getBorrowerPhone() == "") {

psmt.setString(8, **null**);

}

**else** {

psmt.setString(8, book.getBorrowerPhone());

}

//执行SQL语句

psmt.execute();

}

/\*\*

\* 删除图书信息

\*/

**public** **void** delBook(**int** ID) **throws** SQLException{

// 首先拿到数据库的连接

Connection con=DBUtil.*getConnection*();

String sql="" +

"DELETE FROM tb\_books "+

// 参数用?表示，相当于占位符

"WHERE ID = ?";

// 预编译sql语句

PreparedStatement psmt = con.prepareStatement(sql);

// 先对应SQL语句，给SQL语句传递参数

psmt.setInt(1, ID);

// 执行SQL语句

psmt.execute();

}

/\*\*

\* 更新图书信息

\*/

**public** **void** changeBook(Book book) **throws** SQLException{

// 首先拿到数据库的连接

Connection con=DBUtil.*getConnection*();

String sql="update tb\_books "

+ "set ISBN = ?, book\_name = ?, book\_price = ?, book\_author = ?"

+ ",published\_house = ?, book\_category = ?, borrower\_name = ?, borrower\_phone = ? "

// 参数用?表示，相当于占位符

+ "where ID = ?";

// 预编译sql语句

PreparedStatement psmt = con.prepareStatement(sql);

// 先对应SQL语句，给SQL语句传递参数

psmt.setString(1, book.getISBN());

psmt.setString(2, book.getBookName());

psmt.setFloat(3, book.getPrice());

psmt.setString(4, book.getAuthor());

psmt.setString(5, book.getPublishHouse());

psmt.setString(6, book.getBookCategory());

**if** (book.getBorrowerName().equals("")) {

psmt.setString(7, **null**);

}

**else** {

psmt.setString(7, book.getBorrowerName());

}

**if** (book.getBorrowerPhone().equals("")) {

psmt.setString(8, **null**);

}

**else** {

psmt.setString(8, book.getBorrowerPhone());

}

psmt.setInt(9, book.getID());

// 执行SQL语句

psmt.execute();

}

/\*\*

\* 查询书籍信息

\*/

**public** List<Book> query() **throws** Exception{

Connection con = DBUtil.*getConnection*();

Statement stmt = con.createStatement();

ResultSet rs = stmt.executeQuery("select "

// ISBN、书名、作者、图书价格、出版社

+ "ID, ISBN, book\_name, book\_author, book\_price, published\_house, "

// 分类号、借书人姓名、借书人电话

+ "book\_category, borrower\_name, borrower\_phone "

+ "from tb\_books");

List<Book> bookList = **new** ArrayList<Book>();

Book book = **null**;

// 如果对象中有数据，就会循环打印出来

**while** (rs.next()){

book = **new** Book();

book.setID(rs.getInt("ID"));

book.setISBN(rs.getString("ISBN"));

book.setBookName(rs.getString("book\_name"));

book.setAuthor(rs.getString("book\_author"));

book.setPrice(rs.getFloat("book\_price"));

book.setPublishHouse(rs.getString("published\_house"));

book.setBookCategory(rs.getString("book\_category"));

book.setBorrowerName(rs.getString("borrower\_name"));

book.setBorrowerPhone(rs.getString("borrower\_phone"));

bookList.add(book);

}

**return** bookList;

}

/\*\*

\* 查询借阅信息

\*

\* **@return**

\* bookList

\*/

**public** List<Book> borrowQuery() **throws** Exception{

Connection con = DBUtil.*getConnection*();

Statement stmt = con.createStatement();

ResultSet rs = stmt.executeQuery(""

// ID、书名、借书人姓名、借书人电话

+ "SELECT ID, book\_name, borrower\_name, borrower\_phone "

+ "FROM tb\_books "

+ "WHERE borrower\_name IS NOT NULL"

);

List<Book> bookList = **new** ArrayList<Book>();

Book book = **null**;

// 如果对象中有数据，就会循环打印出来

**while** (rs.next()){

book = **new** Book();

book.setID(rs.getInt("ID"));

book.setBookName(rs.getString("book\_name"));

book.setBorrowerName(rs.getString("borrower\_name"));

book.setBorrowerPhone(rs.getString("borrower\_phone"));

bookList.add(book);

}

**return** bookList;

}

/\*\*

\* 更新图书信息，归还图书

\*/

**public** **void** returnBook(Book book) **throws** SQLException{

// 首先拿到数据库的连接

Connection con=DBUtil.*getConnection*();

String sql="UPDATE tb\_books "

// ISBN、图书名称、作者、价格

+ "SET "

// 借书人姓名、借书人电话

+ "borrower\_name = ?, borrower\_phone = ? "

// 参数用?表示，相当于占位符

+ "WHERE ID = ?";

// 预编译sql语句

PreparedStatement psmt = con.prepareStatement(sql);

// 先对应SQL语句，给SQL语句传递参数

psmt.setString(1, book.getBorrowerName());

psmt.setString(2, book.getBorrowerPhone());

psmt.setInt(3, book.getID());

// 执行SQL语句

psmt.execute();

}

}

（4）更改密码模块

需要说明的问题

操作权限：图书管理员

功能类型：数据库处理

功能概述：更改密码

前提业务：登录系统

后继业务：

功能约束：权限约束

约束描述：

操作权限：面向图书管理员

/\*\*

\* 更改用户信息界面类

\*

\* **@author** 1651200111 陈彦志

\*/

@SuppressWarnings("serial")

**public** **class** ChangeUserInformation **extends** JFrame {

JFrame frame = **new** JFrame("个人书屋");

Container container = frame.getContentPane();

**private** JTextField textFieldUserName = **new** JTextField();

**private** JTextField textFieldPasswd = **new** JTextField();

**private** JButton buttonYes = **new** JButton();

**private** JButton buttonReset = **new** JButton();

**public** ChangeUserInformation() {

frame.setLayout(**null**);

// 设置背景图片

**new** BackgroundImage(frame,container,"changeUser.jpg");

// 添加工具栏以及各组件和监听事件

**new** MenuBar(frame);

// 用户名框

setUserTextField();

// 密码框

setPasswordField();

// 确认按钮

setButtonYes();

// 重置按钮

setButtonReset();

container.add(textFieldPasswd);

container.add(textFieldUserName);

container.add(buttonReset);

container.add(buttonYes);

// 设置窗口大小、位置、可视、默认关闭方式等

**new** FrameOption(frame);

}

/\*\*

\* 设置重置按钮

\*/

**private** **void** setButtonReset() {

buttonReset.setIcon(**new** ImageIcon("res/button\_reset.jpg"));

buttonReset.setBounds(420,325,90,30);

buttonReset.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent e) {

// **TODO** Auto-generated method stub

textFieldUserName.setText("");

textFieldPasswd.setText("");

}

});

}

/\*\*

\* 设置确认按钮，添加操作数据库更改用户名密码事件

\*/

**public** **void** setButtonYes(){

buttonYes.setIcon(**new** ImageIcon("res/button\_ok.jpg"));

buttonYes.setBounds(200,325,90,30);

buttonYes.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent e) {

// **TODO** Auto-generated method stub

UserAction userAction = **new** UserAction();

**try** {

userAction.edit(textFieldUserName,textFieldPasswd);

JOptionPane.*showMessageDialog*(**null**,"用户名和密码修改成功","成功"

,JOptionPane.***PLAIN\_MESSAGE***);

frame.setVisible(**false**);

**new** Login();

}**catch**(Exception e1) {

e1.printStackTrace();

}

}

});

}

/\*\*

\* 设置密码框

\*/

**public** **void** setPasswordField() {

textFieldPasswd.setBounds(380,235,200,30);

**new** KeyListener(textFieldPasswd,buttonYes);

}

/\*\*

\* 用户框

\*/

**public** **void** setUserTextField() {

textFieldUserName.setBounds(380,145,200,30);

**new** KeyListener(textFieldUserName,buttonYes);

}

}

用户行为控制类

/\*\*

\* 用户行为控制类，包含用户名和密码的修改

\*

\* **@author** 1651200111 陈彦志

\*/

**public** **class** UserAction {

/\*\*

\* 修改用户名密码

\*/

**public** **void** edit(JTextField username,JTextField passwd) **throws** Exception{

User user = **new** User();

UserDao userdao=**new** UserDao();

userdao.updateUser(user,username.getText(),passwd.getText());

}

}

用户表信息访问数据对象类

/\*\*

\* 用户表信息数据访问对象类，包含更改用户信息

\*

\* **@author** 1651200111 陈彦志

\*/

**public** **class** UserDao {

/\*\*

\* 更改用户信息

\*/

**public** **void** updateUser (User user, String username, String passwd) **throws** SQLException{

// 首先拿到数据库的连接

Connection con = DBUtil.*getConnection*();

String sql="update tb\_user "

// 参数用?表示，相当于占位符

+"set user=?,passwd=? "

+ "where id=1";

//预编译sql语句

PreparedStatement preparedStatement = con.prepareStatement(sql);

//先对应SQL语句，给SQL语句传递参数

preparedStatement.setString(1, username);

preparedStatement.setString(2, passwd);

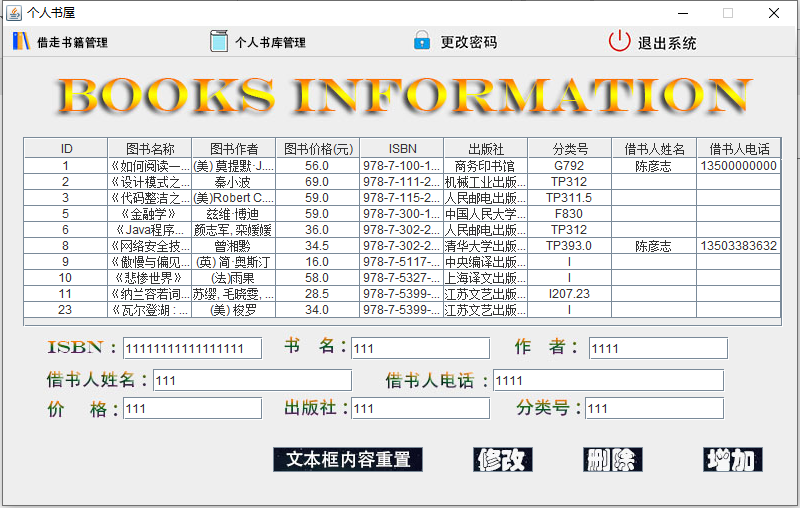
//执行SQL语句

preparedStatement.execute();

}

# 6. 编码和单元测试

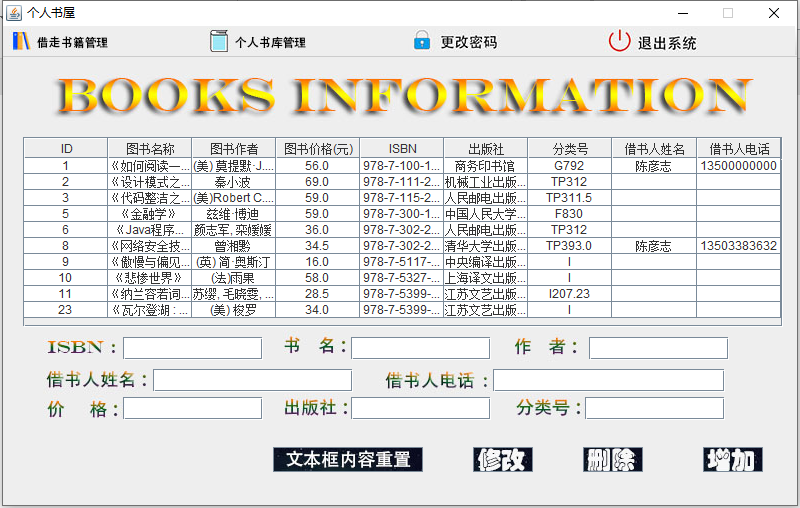
## 6.1.图书信息增加测试



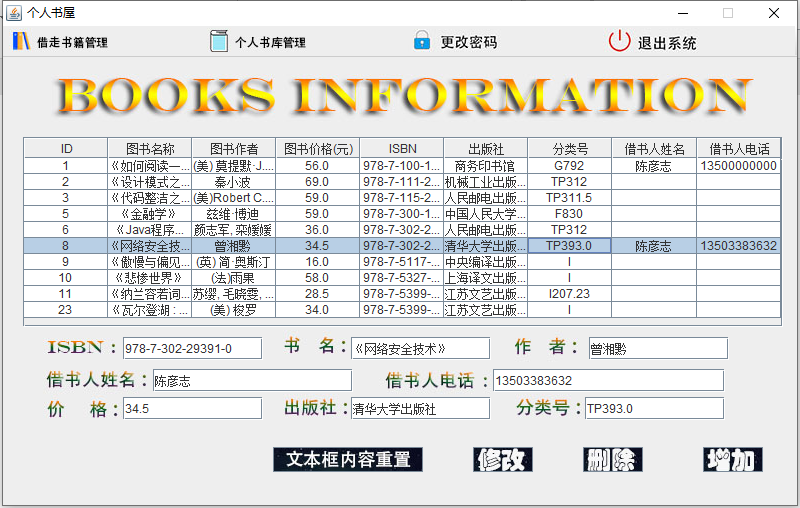


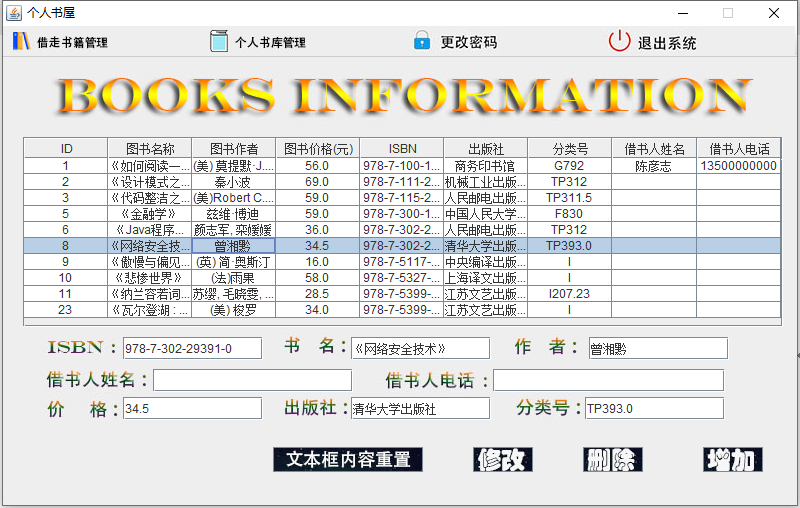
## 6.2.图片信息删除测试



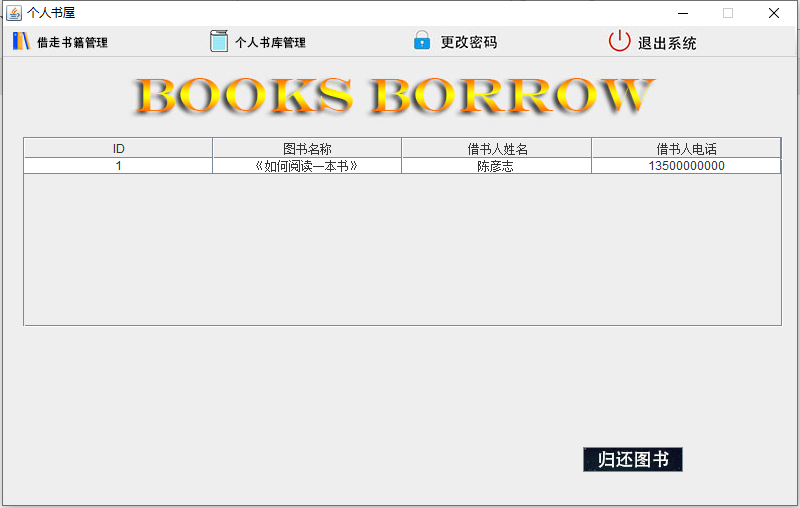


## 6.3.图书信息修改测试



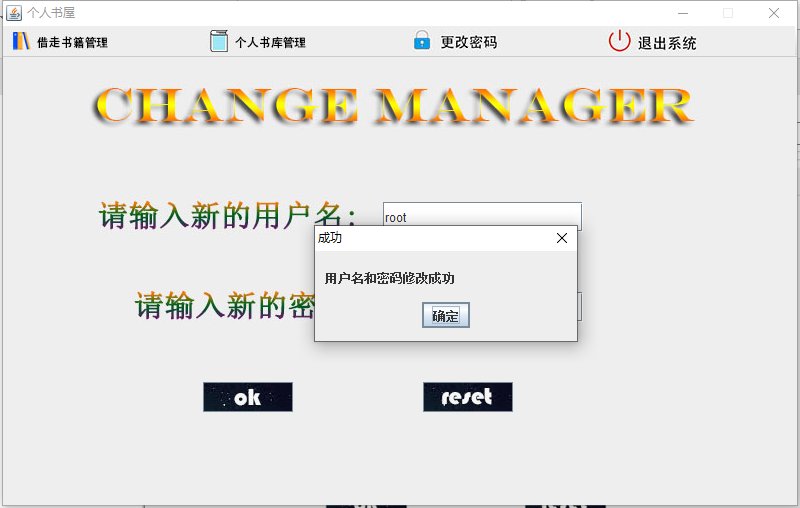


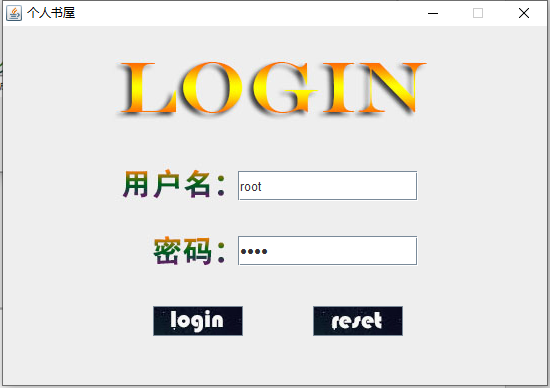
## 6.4.图书信息归还测试

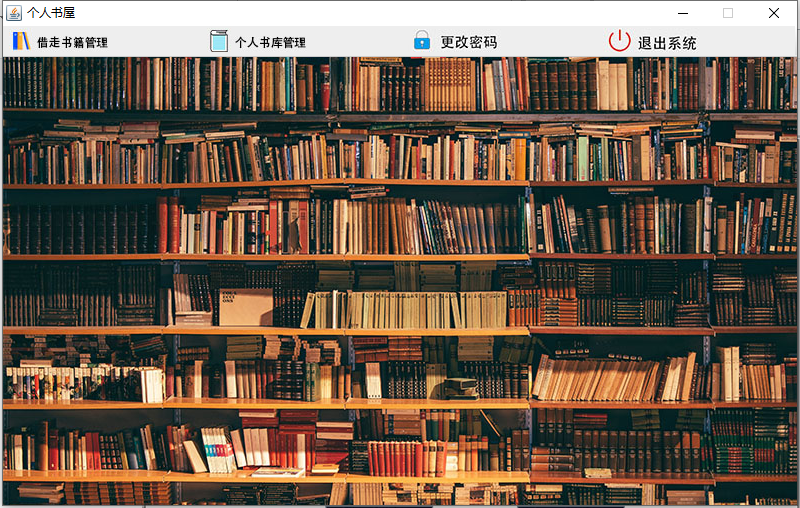




## 6.5用户更改密码测试







# 实训总结：

虽然本次实训只有短短的三个星期，但对于我来说，感觉自己总体上收获还是挺大的，这次实训我做的是图书管理系统，该系统主要实现了图书管理的功能，从需求分析，界面的搭建、美化，再到数据库的连接，表格，存储过程和存储过程等的建立。一系列的过程步骤让我对于Java + MySQL有了更深刻的理解，在实训过程中，也让我学会了很多东西并对面向对象有了更深的认识，首先，让我学会了在java中实现MySQL数据库的操作，学会了在图形化界面中设置背景图片以及PS里边特效字体的绘制。其次，让我更深刻的理解了面向对象设计（java）就是类与类之间的调用，还有MVC三层架构的组织模块方法。最后，通过书写实训报告，又加深了对于软件工程开发软件编写报告的整个过程。

其中用到的知识虽然大部分都已经学过，但真正实践起来才发现自己还是有很多不足，于是我又将课本中的知识重新回炉了一遍，加深了对课本知识的理解，知道了监听器的工作原理。通过对不同事件进行监听去做出相应的不同的操作。还理解了对图书管理系统的整体进行良好的布局，所以我在设计界面上花了很多功夫，为了让界面美观，我用了几天时间去对每个界面的背景图片去ps，虽然仍有不足之处，但因为时间短也就只能稍有遗憾，等到日后再做完善。然后开始往整体系统中去添加相应的功能，在添加功能上，除了一些必须的功能和龙丹老师中期检查时要求的功能外，本来还想再添加借书日期和已借天数功能，但因为刚要添加，在调试时又遇到一些bug，比如，在图书信息管理界面将借书人姓名、借书人电话置空后图书借阅管理界面还有该条信息等，于是又用了一两天的时间去改bug，等到改完就马上要验收了，这也是一个实训过程中的遗憾。

在这次的实训中，层次化，模块化，抽象化也是我学到的一个重要的经验，我在学校的图书馆借阅了相关的书籍，参考一些资料后发现模块化能使程序设计更加简单，设计代码时目标更加明确，更有条理，效率也更高，以前虽然也知道这些道理，但自己真正实施起来才真正发现这样做的好处，在本次实训中，我使用的是MVC三层架构方式组织包，将多个界面都需要用到的模块封装在公共类中，通过这样组织，使代码更加简洁，也更容易理解。

本次实训虽然开题早，但是选完课题后，又赶上了接二连三的考试，一边要抓紧时间复习考试，一边要在考完试准备下一个科目的空档中构思实训的设计，显然没有办法投入太多时间去完成课题的设计开发。但是即使在短暂的零碎时间里，通过自己的摸索、参考书籍、以及和同学的交流探讨，我还是学到了很多东西，当然也意识到了自己目前尚存的不足之处。相信通过本次的实训以及以后的更加努力的学习，在即将到来的毕业设计中，能有更大的突破，做出自己应有的水平。

# 参考文献

[1] 徐云彪编著.数据库原理与技术[M].浙江：浙江大学出版社，2002：23-59.

[2] 陆晶编著.Java程序设计[M].北京：清华大学出版社，2002：10-136.

[3] 孙元编著.Java语言 SQL接口[M].北京：清华大学出版社，1997：3-59.

[5] 苗春义著.Java项目开发全程实录[M].北京：清华大学出版社，2008：45-159.