Java应用开发课程设计

**基于MyBatis的超市管理系统**

**学 院： 数学与计算机学院**

**班 级： 物联网**

**姓 名： 余云飞**

**学 号： 1804240623**

**指导老师： 聂刚**

**成 绩：**

**完成日期： 2020年6月20日**

目录

[1.问题. 1](#_Toc533333826)

[1.1要解决的问题 1](#_Toc533333827)

[1.2要求 1](#_Toc533333828)

[2.可行性研究 1](#_Toc533333829)

[2.1 解决办法 1](#_Toc533333830)

[2.2 系统流程图 2](#_Toc533333831)

[2.3数据流图 2](#_Toc533333832)

[2.4数据字典精确定义数据流图中每个元素。 3](#_Toc533333833)

[3.需求分析 3](#_Toc533333834)

[3.1必须完成的功能 3](#_Toc533333835)

[3.2 数据库设计 4](#_Toc533333836)

[3.2.1 实体联系图（ER图） 5](#_Toc533333837)

[3.2.2关系模式 5](#_Toc533333838)

[3.2.3 数据库表 5](#_Toc533333839)

[3.3 状态转换图，描述作为外部事件结果的软件行为 6](#_Toc533333840)

[3.4 软件需求规格说明书 6](#_Toc533333841)

[4.总体设计 7](#_Toc533333842)

[4.1. 层次结构 7](#_Toc533333843)

[4.2 结构设计 8](#_Toc533333844)

[5.详细设计 8](#_Toc533333845)

[5.1 主程序流程图 8](#_Toc533333846)

[5.2 借阅图书程序流程图 9](#_Toc533333847)

[5.3 图书信息管理程序流程图 1](#_Toc533333848)0

[5.4程序描述 1](#_Toc533333849)0

[6. 系统测试 3](#_Toc533333850)6

[6.1.图书信息增加测试 3](#_Toc533333851)7

[6.2.图片信息删除测试 37](#_Toc533333852)

[6.3.图书信息修改测试 3](#_Toc533333853)8

[6.4.图书信息归还测试 39](#_Toc533333854)

[6.5用户更改密码测试 4](#_Toc533333855)0

7[系统总结： 4](#_Toc533333856)1

# 1.问题

## 1.1要解决的问题

采用MyBatis框架实现基于DAO设计模式的图书馆系统：

## 1.2要求

用户角色划分：读者、图书馆人员

各角色基本功能如下：

读者角色：登陆，查询图书，借阅图书，归还图书，查询借阅记录，修改密码，

图书馆人员角色：登陆，图书增加，图书修改，图书删除，读者增加，读者修改，读者删除，借阅查询，借阅统计等。

完成需求分析绘制UML的用户用例图（Use Case）

设计对应的数据库，绘制数据库的ER图

完成每个类的设计，绘制系统的类图

对于每一个业务逻辑，绘制UML时序图

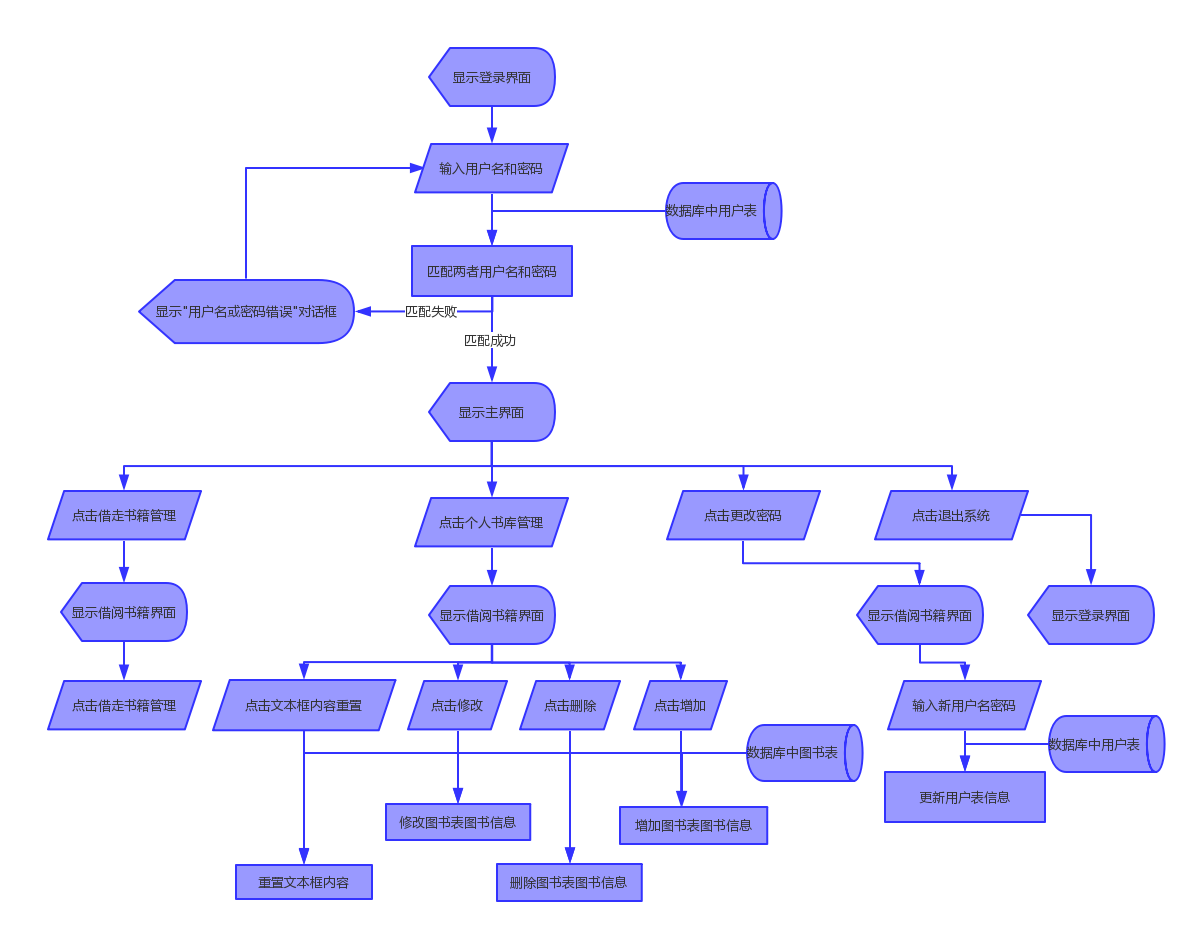
对于每一个方法实现，绘制相应的流程图

# 2.可行性研究

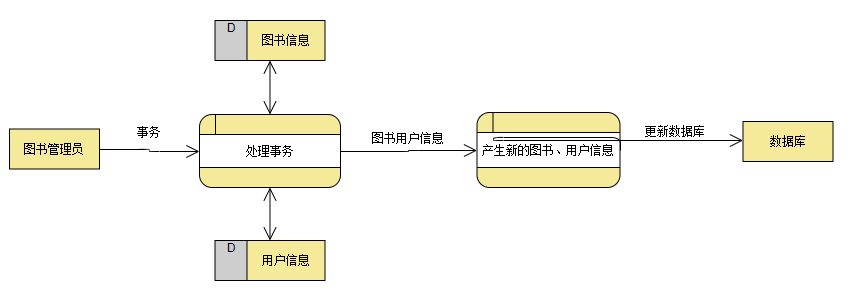
## 2.1 行得通的解决办法

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



## 2.3数据流图（描述信息流和数据从输入移动到输出过程之中的变换）



## 2.4数据字典精确定义数据流图中每个元素

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

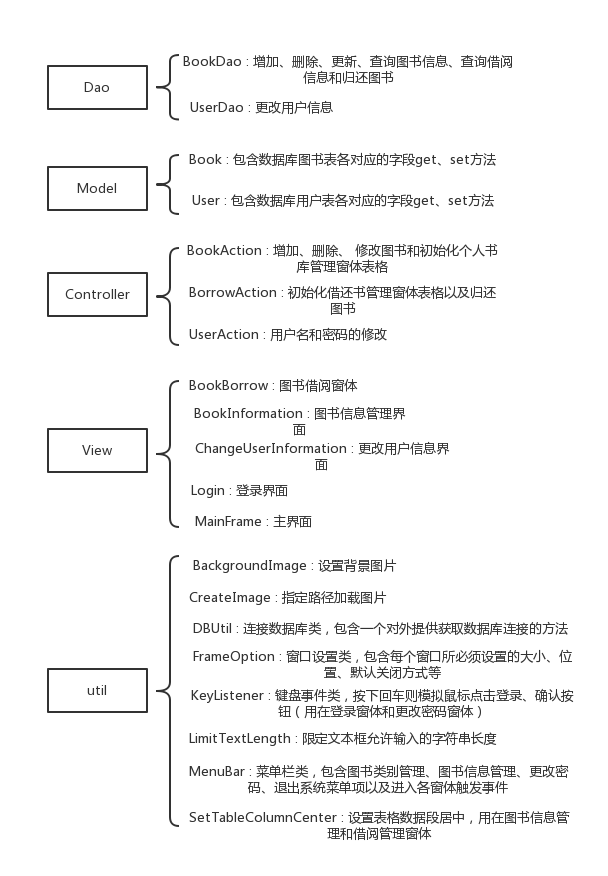
## 3.1功能

1管理员登录

2图书信息的增、删、改、查

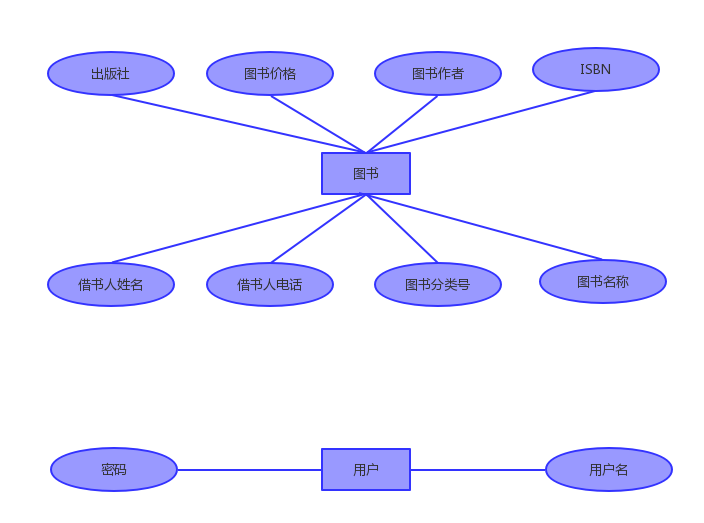
3图书借阅信息的显示及图书归还

4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

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 |  |  |

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

登陆失败 登陆成功

登陆

选择功能

重新登陆

选择完成

交互完成

完成功能

与数据库交互

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。

图书借阅管理

图书归还

图书借阅信息显示

## 3.4 软件需求规格说明

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。

图书信息管理

图书信息修改

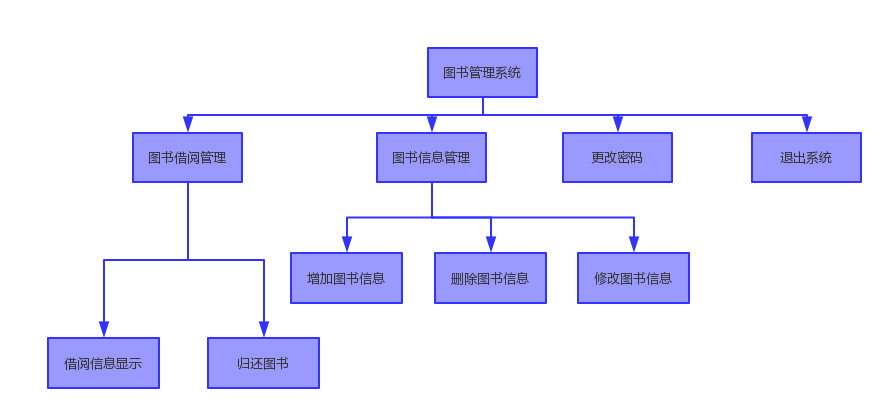
图书显示

图书信息删除

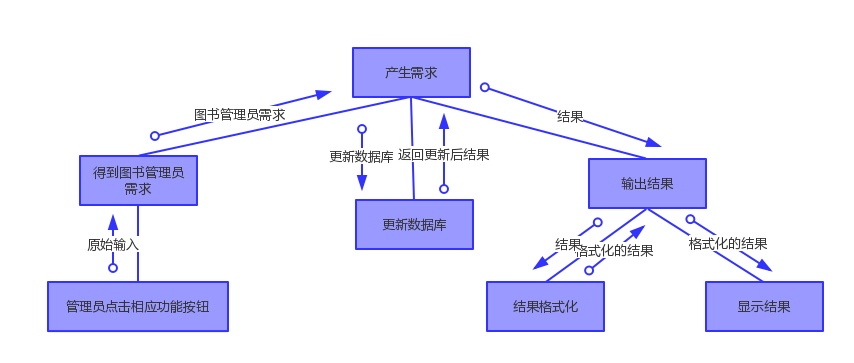
图书信息增加

# 4.总体设计

## 4.1.层次结构

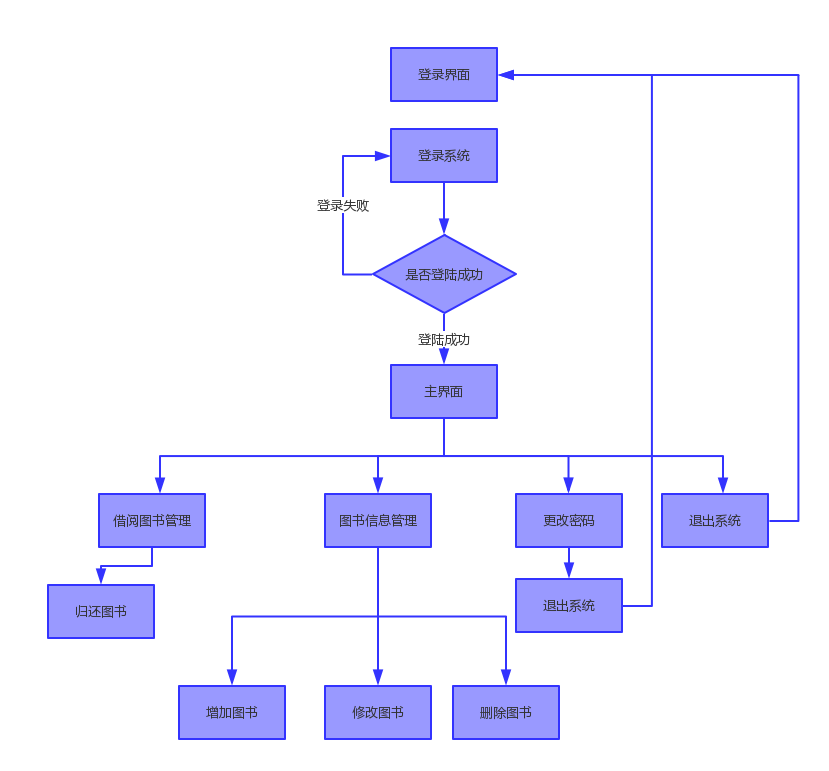


## 4.2 结构设计

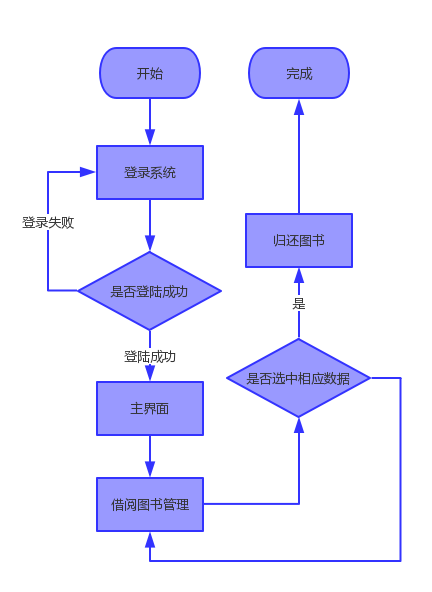


# 5.详细设计

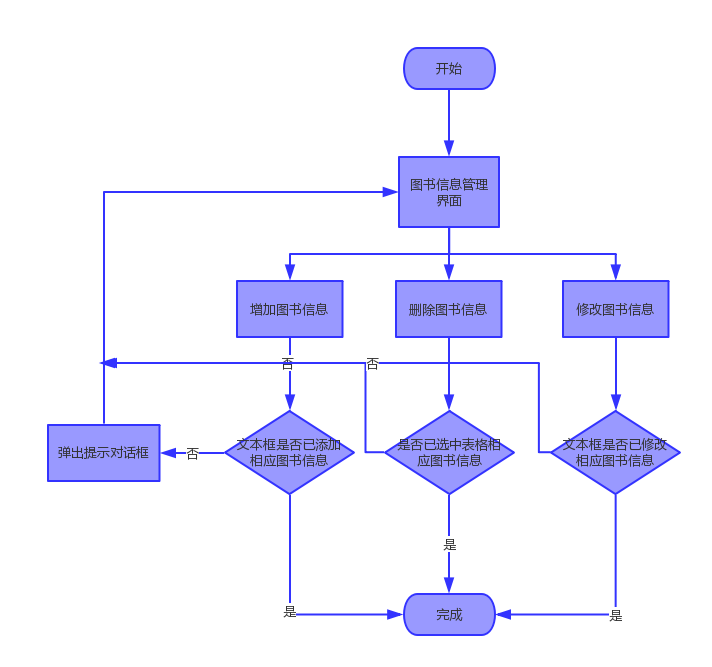
## 5.1 主程序流程图



## 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.

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() {

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() {

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() {

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() {

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

业务：登录系统

功能约束：权限约束

操作权限：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() {

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

\* 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

\* 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

\*/

@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() {

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() {

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() {

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() {

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() {

public void mouseReleased(MouseEvent e) {

// TODO Auto-generated method stub

}

public void mousePressed(MouseEvent e) {

// TODO Auto-generated method stub

}

public void mouseExited(MouseEvent e) {

// TODO Auto-generated method stub

}

public void mouseEntered(MouseEvent e) {

// TODO Auto-generated method stub

}

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

\*/

public class BookAction {

/\*\*

\* 初始化窗体表格

\* 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。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

\* 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

业务：登录系统

功能约束：权限约束

/\*\*

\* 更改用户信息界面类

\*/

@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() {

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() {

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

\*/

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

\*/

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);

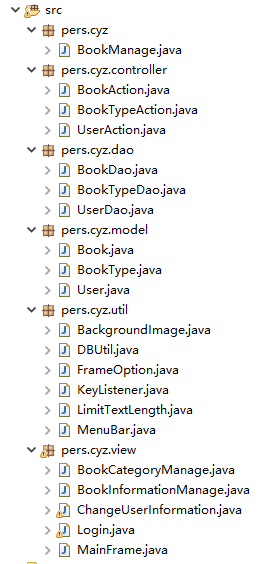
preparedStatement.setString(2, passwd);

//执行SQL语句

preparedStatement.execute();

}

**系统实现**



BookManage:主程序类，调用登录界面类

BookAction、BookTypeAction、UserAction：控制层，负责协调各自视图层和Dao层。

BookDao、BookTypeDao、UserDao：Dao层，包含需要操作数据库三张表各字段的增删改查等。

Book、BookType、User：模型层，按照数据库表对应的字段编写的各自类。

BackgroundImage：设置背景图片类

DBUtil：连接数据库类

FrameOption：窗口设置类，包含每个窗口所必须设置的大小、位置、默认关闭方式等

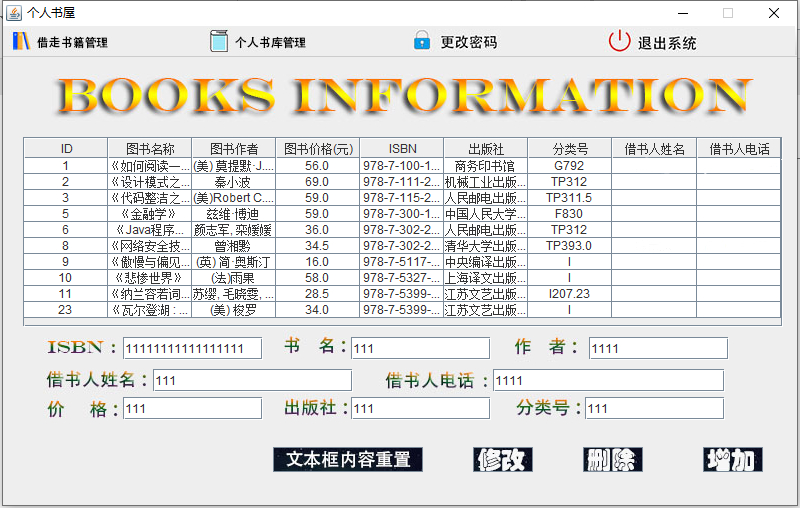
KeyListener：键盘事件类，如果在文本框或密码框按下回车则模拟鼠标点击相应按钮。

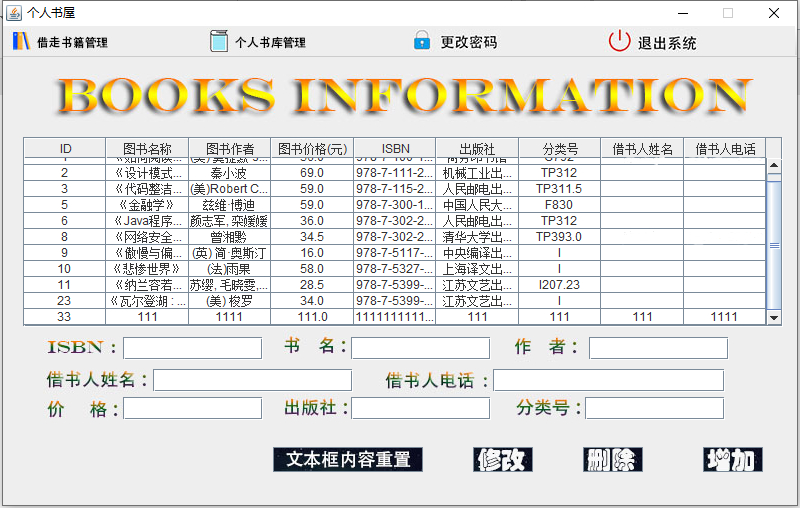
LimitTextLength：限定文本框允许输入的字符串长度类。

MenuBar：菜单类，因为除了登录界面，都需要用到菜单栏所以讲菜单栏封装为一个类。

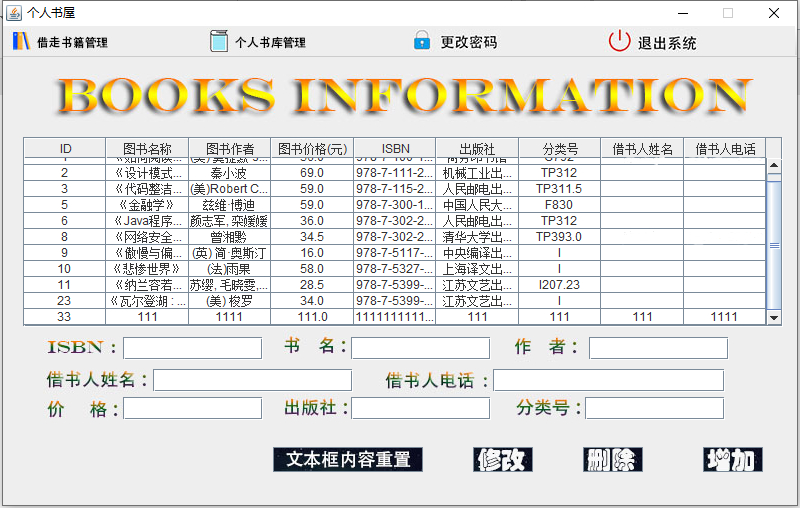
# 6. 编码和单元测试

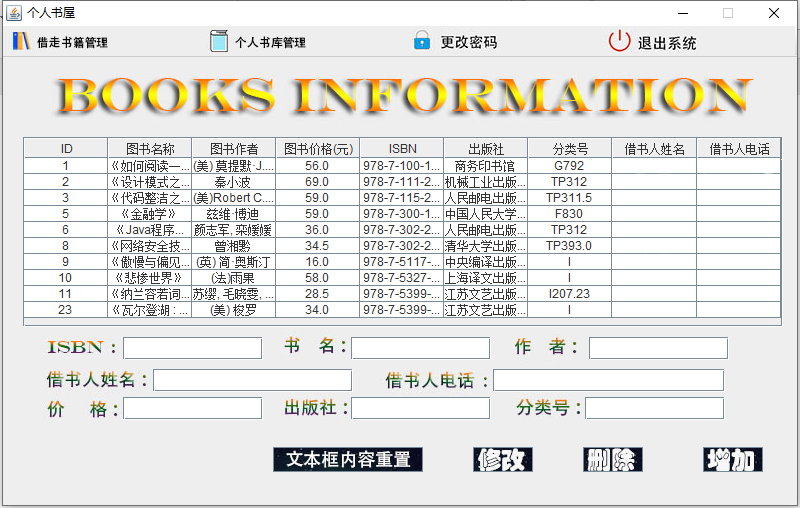
## 6.1.图书信息增加测试



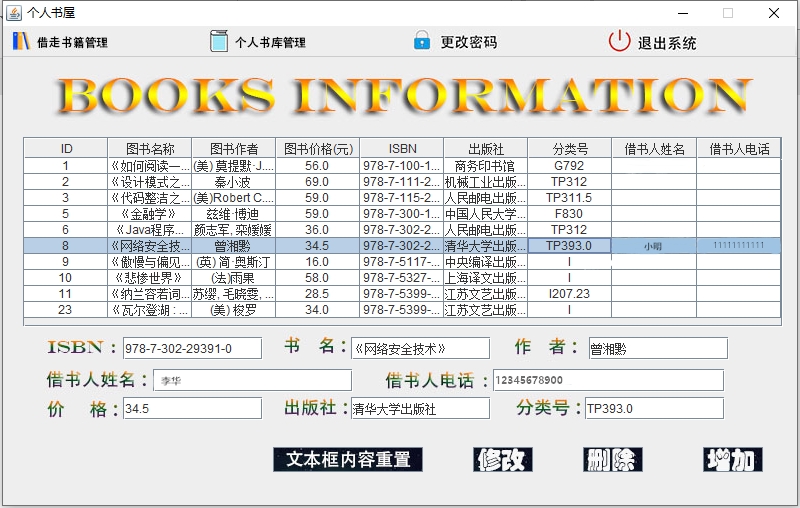


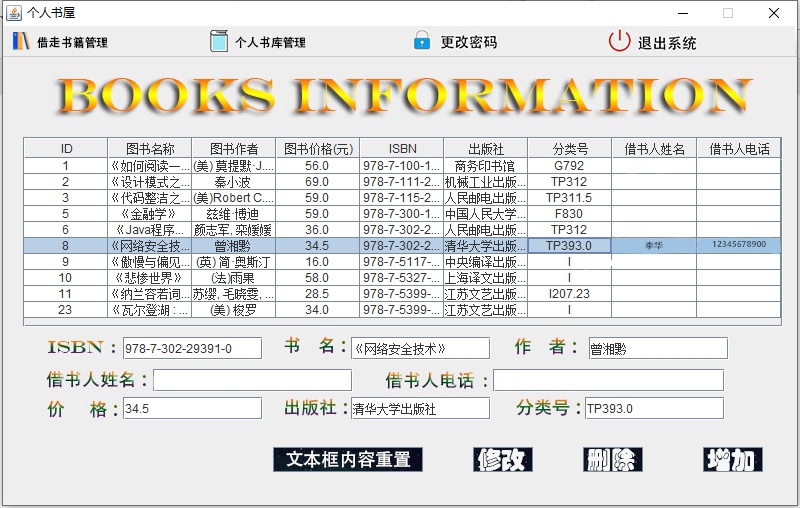
## 6.2.图片信息删除测试



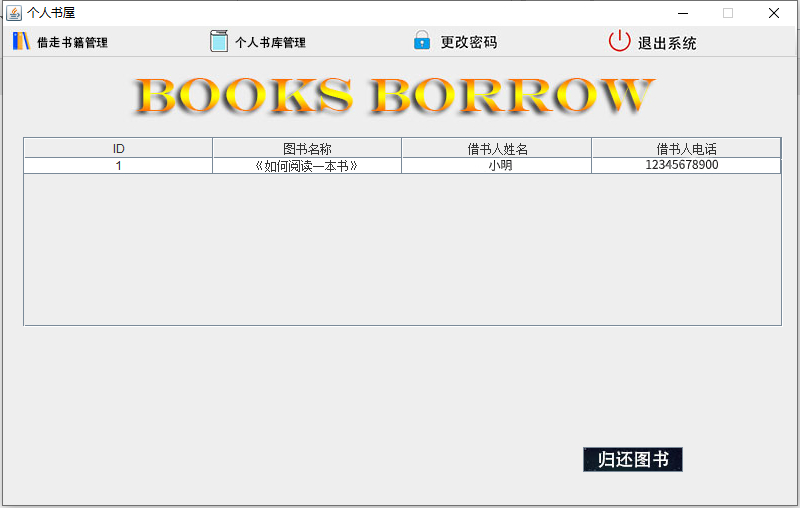


## 6.3.图书信息修改测试



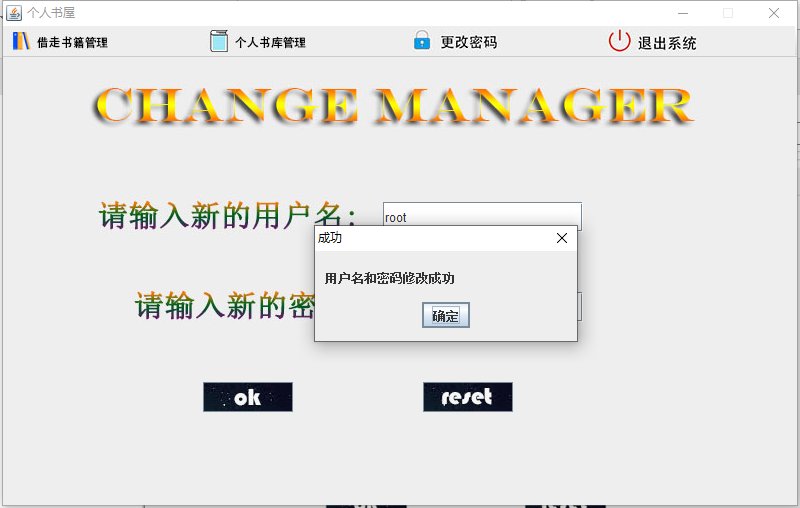


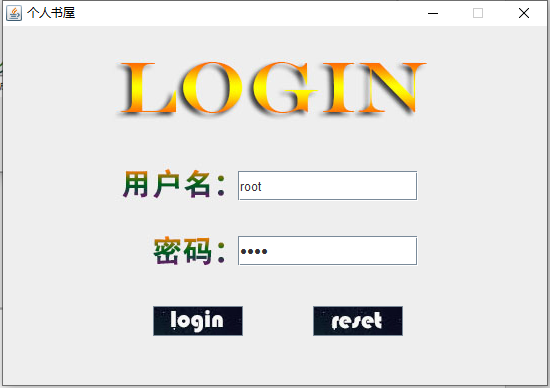
## 6.4.图书信息归还测试





## 6.5用户更改密码测试





# 7系统总结：

总体上收获还是挺大的，这次实训任务是图书管理系统，该系统主要实现了图书管理的功能，从功能分析，界面的搭建、美化，再到前后台的连接，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，也让我学会了很多东西，对java有了更深的认识。