滨江学院

**详细设计报告**

( 2019 -- 2020 年度第 二 学期)

课程名称： 软件工程

题 目： 在线书店

专 业： 软件工程

班 级： 2班

学 号： 20182344050

姓 名： 毛济洲

日期：2020 年 6月 4日

**1. 引言**

**编写目的**

本说明书确定系统的详细功能模块和数据结构，为下阶段开发工作提供依据。

背景

软件系统的名称： 在线书店

本项目的任务开发者：毛济洲

软件系统的用户：书店老板及用户

**2. 界面设计要求**

用户界面采用Windows风格，便于用户进行操作。其主要界面，如下图所示。



用户界面设计规则

1.字体

宋体，正常体，黑色

2.控件

a).尺寸

在合理的布局下尽可能多的显示控件内的内容。

b).布局

按照操作流程或浏览顺序自左至右、由上而下的排放各种控件，使界面整体协调、美观大方。

c).自适应父对象的尺寸改变

控件应具有自适应父对象的尺寸改变的能力，当父对象的尺寸发生变化时，控件应能自动改变自己的尺寸并使界面保持整体协调，尽量减少因父对象的尺寸改变而带来的操作或浏览上的不便。

**3.数据库表设计**

1、用户表：

用户id，用户名，用户密码

2、管理员表：

管理员id，管理员名，管理员密码

3、图书信息表：

图书id，图书名，图书出版社，图书作者，图书状态

4、借阅记录表：

借阅记录id，用户id，用户名，图书id，图书名，是否归还

1. 用户表(user)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 是否主键 | 是否为空 | 长度 | 备注 |
| userId | varchar | 是 | 否 | 20 | 用户id |
| username | varchar | 否 | 否 | 20 | 用户名 |
| password | varchar | 否 | 否 | 20 | 用户密码 |

2、管理员表(admin)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 是否主键 | 是否为空 | 长度 | 备注 |
| adminId | varchar | 是 | 否 | 20 | 管理员id |
| adminname | varchar | 否 | 否 | 20 | 管理员名 |
| password | varchar | 否 | 否 | 20 | 管理员密码 |

3、图书表(book)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 是否主键 | 是否为空 | 长度 | 备注 |
| bookId | varchar | 是 | 否 | 20 | 图书id |
| bookName | varchar | 否 | 否 | 20 | 图书名 |
| press | varchar | 否 | 否 | 20 | 图书出版社 |
| author | varchar | 否 | 否 | 20 | 图书作者 |
| lend | int | 否 | 否 | 5 | 图书是否被加购  是为0  默认为1 |

4、加购记录表(lendrecord)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 是否主键 | 是否为空 | 长度 | 备注 |
| recordId | int | 是 | 否 | 20 | 加购记录id |
| userId | varchar | 否 | 否 | 20 | 用户id |
| userName | varchar | 否 | 否 | 20 | 用户名 |
| bookId | varchar | 否 | 否 | 20 | 图书id |
| bookName | varchar | 否 | 否 | 20 | 图书名 |
| back | int | 否 | 否 | 1 | 是否删除加购  是为0  默认为1 |

**4.** **重要代码分析**

a、Dao类（以BookDao为例）  
**Dao类中写的是对数据库的一些基本操作**

package com.book.dao;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import com.book.model.Book;

import com.book.model.User;

import com.book.util.StringUtil;

public class BookDao {

/\*\*

\* 图书添加

\* @param con

\* @param book

\* @return

\* @throws Exception

\*/

public int add(Connection con, Book book) throws Exception {

//插入图书前 检查bookId是否已经存在，若存在，返回0

String sqlid = "select \* from book where bookId = ?";

PreparedStatement pstmtid = (PreparedStatement) con.prepareStatement(sqlid);

pstmtid.setString(1, book.getBookId());

ResultSet rs = pstmtid.executeQuery();

if(rs.next()) {

return 0;

}

String sql = "insert into book values(?,?,?,?,?)";

PreparedStatement pstmt = (PreparedStatement) con.prepareStatement(sql);

pstmt.setString(1, book.getBookId());

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

pstmt.setString(3, book.getPress());

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

//pstmt.setString(5, book.getTypeId());

pstmt.setInt(5, book.getLend());

return pstmt.executeUpdate();

}

/\*\*

\* 图书查询

\* @param con

\* @param bookMessage

\* @return

\* @throws Exception

\*/

public ResultSet list(Connection con, Book bookMessage) throws Exception {

//将book表和bookType表连接

//StringBuffer sb = new StringBuffer("select \* from book,booktype where book.typeId=booktype.typeId");

StringBuffer sb = new StringBuffer("select \* from book where bookId = bookId ");

//向sb中添加，并进行模糊查询

if(StringUtil.isNotEmpty(bookMessage.getBookId())) {

sb.append(" and bookId like '%" + bookMessage.getBookId() + "%'");

}

//按书名模糊查询

if(StringUtil.isNotEmpty(bookMessage.getBookName())) {

sb.append(" and book.bookName like '%" + bookMessage.getBookName() + "%'");

}

//按作者模糊查询

if(StringUtil.isNotEmpty(bookMessage.getAuthor())) {

sb.append(" and book.author like '%" + bookMessage.getAuthor() + "%'");

}

// //按图书类型模糊查询

// if(StringUtil.isNotEmpty(bookMessage.getTypeId())) {

// sb.append(" and book.typeId like '%" + bookMessage.getTypeId() + "%'");

// }

PreparedStatement pstmt=(PreparedStatement) con.prepareStatement(sb.toString());

return pstmt.executeQuery();

}

//历史记录查询

public ResultSet listHistory(Connection con, User userMessage) throws Exception {

String sql = "select recordId,userName, bookName, (case back when 0 then '否' when 1 then '是' end) as back from lendrecord where userName = ?";

PreparedStatement pstmt = (PreparedStatement) con.prepareStatement(sql);

pstmt.setString(1, userMessage.getUsername());

return pstmt.executeQuery();

}

/\*\*

\* 图书删除

\* @param con

\* @param id

\* @return

\* @throws Exception

\*/

public int delete(Connection con, String id) throws Exception {

String sql = "delete from book where bookId = ?";

PreparedStatement pstmt = (PreparedStatement) con.prepareStatement(sql);

pstmt.setString(1, id);

return pstmt.executeUpdate();

}

/\*\*

\* 图书修改

\* @param con

\* @param bookMessage1

\* @return

\* @throws Exception

\*/

public int update(Connection con, Book bookMessage1) throws Exception {

String sql = "update book set bookName=?, press=?, author=? where bookId=?";

PreparedStatement pstmt = (PreparedStatement) con.prepareStatement(sql);

pstmt.setString(1, bookMessage1.getBookName());

pstmt.setString(2, bookMessage1.getPress());

pstmt.setString(3, bookMessage1.getAuthor());

pstmt.setString(4, bookMessage1.getBookId());

return pstmt.executeUpdate();

}

//检查该id的图书是否存在,若存在，返回0，否则返回1

public ResultSet bookCheck(Connection con, String id) throws Exception{

//插入图书前 检查bookId是否已经存在，若存在，返回0

String sqlid = "select \* from book where bookId = ?";

PreparedStatement pstmtid = (PreparedStatement) con.prepareStatement(sqlid);

pstmtid.setString(1, id);

ResultSet rs = pstmtid.executeQuery();

return rs;

}

//

/\*\*

\* 图书借阅

\* 修改book表中的lend字段和 加购记录表（lendrecord）中的back字段 lend=1,back=0

\* @param con

\* @param textid

\* @param bookName

\* @param userMessage

\* @return

\* @throws Exception

\*/

public int lend (Connection con, String textid, String bookName, User userMessage) throws Exception {

//先修改book表中图书的状态

String sql = "update book set lend=? where bookId=?";

PreparedStatement pstmt = (PreparedStatement) con.prepareStatement(sql);

pstmt.setString(1, "1");

pstmt.setString(2, textid);

pstmt.executeUpdate();

//插入加购记录表(lendrecord) 信息

String insql = "insert into lendrecord (userId,userName,bookId,bookName,back)values(?,?,?,?,?)";

PreparedStatement lendpstmt = (PreparedStatement) con.prepareStatement(insql);

lendpstmt.setString(1, userMessage.getUserId());

lendpstmt.setString(2, userMessage.getUsername());

lendpstmt.setString(3, textid);

lendpstmt.setString(4, bookName);

lendpstmt.setString(5, "0");

lendpstmt.executeUpdate();

return 0;//成功

}

//图书加购删除

/\*\*

\* 修改book表中的lend字段和 借阅记录表（lendrecord）中的back字段 lend=0,back=1

\* @param con

\* @param textid

\* @param userMessage

\* @return

\* @throws Exception

\*/

public int back (Connection con, String textid, User userMessage) throws Exception {

//先修改book表中图书的状态

String sql = "update book set lend=? where bookId=?";

PreparedStatement pstmt = (PreparedStatement) con.prepareStatement(sql);

pstmt.setString(1, "0");

pstmt.setString(2, textid);

pstmt.executeUpdate();

//修改加购记录表(lendrecord) 信息 ，back=1,表示已经归还

String insql = "update lendrecord set back=? where bookId=? and userName=? ";

PreparedStatement lendpstmt = (PreparedStatement) con.prepareStatement(insql);

lendpstmt.setString(1, "1");

lendpstmt.setString(2, textid);

lendpstmt.setString(3, userMessage.getUsername());

lendpstmt.executeUpdate();

return 0;//成功

}

}

**b、Model类（以Book为例）**  
**Medel类，主要是get,set方法  
get和set方法可以自动生成，快捷键：  
Shift+Alt+S –> Generate Getters and Setters –>选择你需要的get,set参数**

package com.book.model;

public class Book {

private String bookId;//图书id

private String bookName;//图书名

private String press;//图书出版社

private String author;//图书作者

//private String typeId;//图书类别id

private int lend;//图书是否被加入购物车, 是为1, 默认为0

public String getBookId() {

return bookId;

}

public void setBookId(String bookId) {

this.bookId = bookId;

}

public String getBookName() {

return bookName;

}

public void setBookName(String bookName) {

this.bookName = bookName;

}

public String getPress() {

return press;

}

public void setPress(String press) {

this.press = press;

}

public String getAuthor() {

return author;

}

public void setAuthor(String author) {

this.author = author;

}

public int getLend() {

return lend;

}

public void setLend(int lend) {

this.lend = lend;

}

}

c、Util类  
连接和关闭数据库

package com.book.util;

import java.sql.Connection;

import java.sql.DriverManager;

public class DbUtil {

private String url = "jdbc:mysql://localhost:3306/work?serverTimezone=UTC";

private String username = "root";

private String password = "mysql1234";

//连接数据库方法

public Connection getCon() throws Exception {

//Class.forName("com.mysql.jdbc.Driver");

Class.forName("com.mysql.cj.jdbc.Driver");

Connection con = (Connection) DriverManager.getConnection(url, username, password);

return con;

}

//关闭数据库方法

public void closeCon(java.sql.Connection con) throws Exception {

if(con != null) {

con.close();

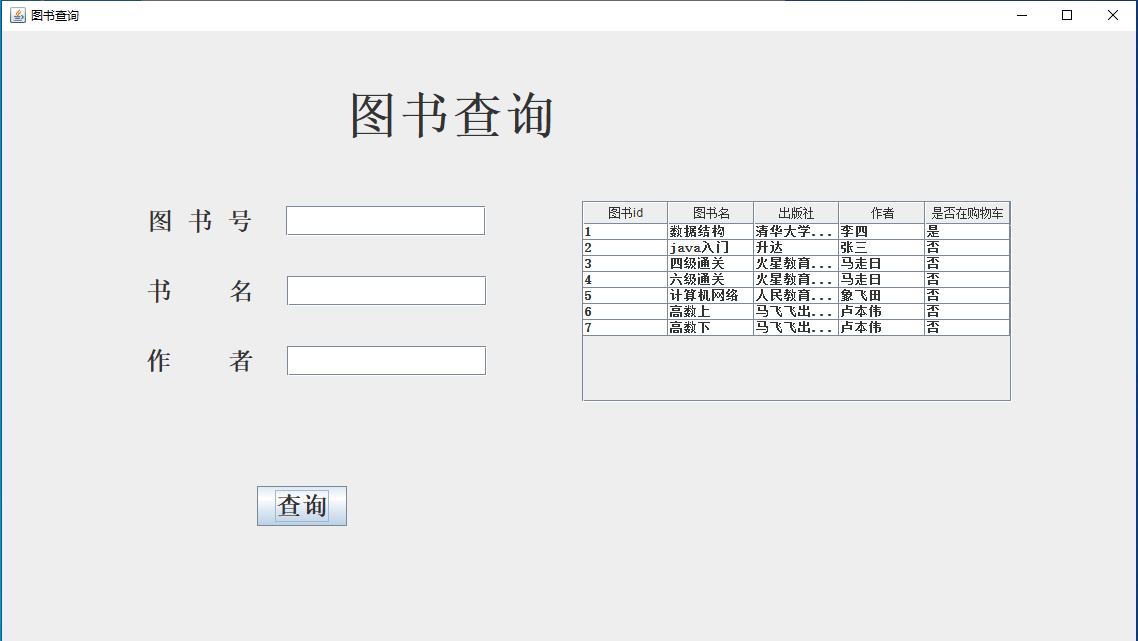
}

}

**5.图片实例**



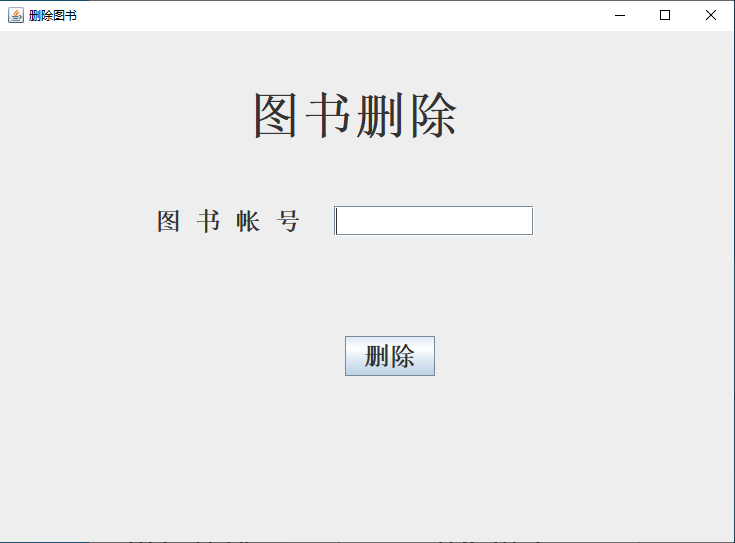
登录界面



图书查询



图书修改界面



删除图书



加购



从购物车删除



查询用户