设计说明书

——图书管理系统

[引言 2](#_Toc24547800)

[编写目的 2](#_Toc24547801)

[背景 2](#_Toc24547802)

[参考资料 2](#_Toc24547803)

[系统结构 2](#_Toc24547804)

[程序设计说明 3](#_Toc24547805)

[程序描述 3](#_Toc24547806)

[主界面： 3](#_Toc24547807)

[定义： 4](#_Toc24547808)

[查询： 5](#_Toc24547809)

[控件： 6](#_Toc24547810)

[书籍入库： 6](#_Toc24547811)

[测试连接数据库： 6](#_Toc24547812)

[添加书籍： 6](#_Toc24547813)

[借阅/查询： 7](#_Toc24547814)

[查询书库： 9](#_Toc24547815)

[数据库介绍： 9](#_Toc24547816)

[基本信息： 9](#_Toc24547817)

[数据库连接： 10](#_Toc24547818)

[写入数据库： 10](#_Toc24547819)

[根据书名搜索： 10](#_Toc24547820)

[依据书号搜索： 11](#_Toc24547821)

[删除： 12](#_Toc24547822)

[数据库更新： 12](#_Toc24547823)

[查询所有数据库信息： 12](#_Toc24547824)

[业务逻辑层： 13](#_Toc24547825)

[展示： 13](#_Toc24547826)

[功能展示 13](#_Toc24547827)

[添加书籍信息 14](#_Toc24547828)

[借阅/归还 15](#_Toc24547829)

[查询书库 16](#_Toc24547830)

[来访记录 17](#_Toc24547831)

[删除书籍 17](#_Toc24547832)

[储存分配 18](#_Toc24547833)

[内部数据： 18](#_Toc24547834)

[限制条件 19](#_Toc24547835)

[技术约束 19](#_Toc24547836)

[环境约束 19](#_Toc24547837)

[有待改进的地方： 19](#_Toc24547838)

## 引言

## 编写目的

1. 掌握结构化程序设计的基本思想；

2. 熟练掌握系统结构的设计和应用；

3. 熟练掌握数据库的存取方法；

4. 了解综合系统设计的基本原理；

5. 提高C#语言程序设计的综合应用能力。

本需求的编写是为了研究嵌入式图书管理系统软件的开发途径和应用方法。同时它也是进行项目策划、概要设计和详细设计的基础，是维护人员进行内部维护，信息更新，验收和测试的依据。

本需求的预期读者是与图书管理系统软件开发有联系的决策人，开发组成人员，扶助开发者，支持本项目的领导和公司人员，软件验证者

### 背景

本项目的名称：图书馆管理应用系统系统

本项目的任务提出者是RFID教师，开发者是上RFID课的学生,用户是潜在用户

本产品是针对电脑管理图书的需求设计的，借用嵌入式RFI，可以完成读者来访记录、读者借还书、图书登记、图书查询，删除，书库查看功能。

### 参考资料

来自csdn，github，博客园下附链接。

<https://download.csdn.net/download/qq_37056790/9888121>

<https://download.csdn.net/download/zjg076000/10444942>

<https://download.csdn.net/download/liliuling/2890193>

<https://download.csdn.net/download/a87462272/2131196>

<https://download.csdn.net/download/linlizhq/3380358>

<https://download.csdn.net/download/liuchang0117/2024661>

<https://github.com/ygsama/BooK-Library-System>

<https://github.com/raojun/BookManage>

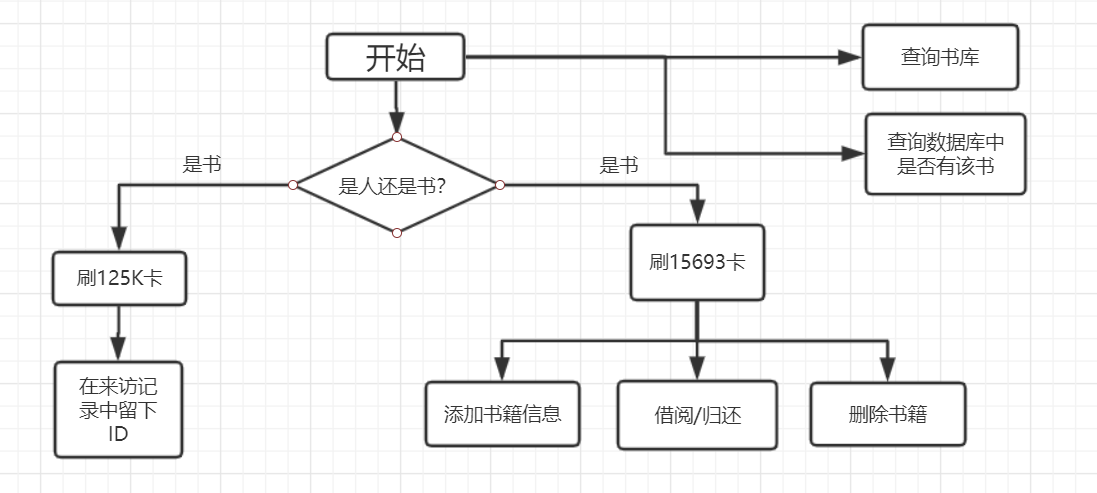
<https://github.com/kgdpevj/LibraryManagementProfessional>

<https://www.cnblogs.com/go2bed/p/4166183.html>

### 系统结构

我们用两层体系结构作为图书馆管理信息系统软件解决方案的核心，将系统的操作人员划分为三类：读者，工作人员，系统管理员

依据软件工程的基本原理，详细设计阶段的根本任务是确定应该怎样具体实现所要求的系统，也就是说，经过这个阶段的设计工作，应该得出对目标系统的精确描述，从而在系统实现阶段可以把这个描述直接翻译成用某种程序设计语言书写的程序。具体来说就是把经过总体设计得到的各个模块详细的加以描述。



## 程序设计说明

### 程序描述

假设每个学生都携带一张125K卡，每次进入图书馆进行一次登记记录。

图书馆里的图书用15693卡做标记，每本书对应一张15693卡，书籍信息存储在数据库，所存储书籍信息如下所示（Borrow用于记录图书状态，表示该书是否存于图书馆，是否可以借出）：

功能包括：

添加书籍信息

借阅/归还

查询书库

来访记录

删除图书

支持查阅图书馆是否存在该书籍，可以由书号查询，也可以由书名查询。

注：以下粘贴部分代码

#### 主界面：

基本界面的加载，串口号，波特率。

以及各用户控件的调用按钮。

单一书籍查询

##### 定义：

string strID;

string BookNo;

string BookName;

string strBookNo;

bool IsOpen\_125k;

string strWGData;

private Thread thread\_125k;

private SerialPort serialPort;

Panel panel = new Panel();

KV\_ISO15693.Reader read = new KV\_ISO15693.Reader();

//private Thread thread\_125k;

private KV\_125K.Reader reader = new KV\_125K.Reader();

/// <summary>

/// 打开按钮

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void btnOpen\_Click(object sender, EventArgs e)

{

if (btnOpen.Text == "打开")

{

String PortName = cmbportID.Text.Trim();

int BaudRate = Convert.ToInt32(txtBuand.Text.Trim());

Byte value = read.OpenSerialPort(PortName, BaudRate);

String[] TagNumber = new String[1];

int TagCount = 0;

if (value == 0x00)

{

MessageBox.Show("15693串口打开成功！");

value = read.Inventory(ModulateMethod.ASK, InventoryModel.Single, ref TagCount, ref TagNumber);

if (value == 0x00)

{

BookNo = TagNumber[0];

}

}

else

{

MessageBox.Show("15693串口打开失败！");

}

}

}

private void but\_People\_Click(object sender, EventArgs e)

{

if (reader.OpenPort(comboBox1.Text, Convert.ToInt32(textBox1.Text)) == "")

{

MessageBox.Show("125K串口打开成功!");

thread\_125k = new Thread(ReceiveData);

thread\_125k.IsBackground = true;

thread\_125k.Start();

}

else

{

MessageBox.Show("125K的串口打开失败!");

}

}

##### 查询：

private void button1\_Click(object sender, EventArgs e)

{

txt\_MessageShow.Clear();

BookBLL bookbll = new BookBLL();

Book book = new Book();

book = bookbll.SearchByBookNo(BookNo);

textBox3.Text = book.BookNo;

if (textBox3.Text == book.BookNo)

{

txt\_MessageShow.Text = "Name: " + book.BookName + " \t\t" + " Author:" + book.Author + " \t\t Publisher: " + book.Publisher + " \t\tFormat:" + book.Format + " \t\t\t Page:" + book.Page + " \t\t\t\t\tPrice:" + book.Price + " \t\t EnterDate:" + book.EnterDate;

}

else

{

MessageBox.Show("not found!");

}

}

##### 控件：

/// <summary>

/// 书籍入库

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void btn\_AddBook\_Click(object sender, EventArgs e)

{

UserControl\_BookIn fb = new UserControl\_BookIn(BookNo);

splitContainer1.Panel2.Controls.Clear();

splitContainer1.Panel2.Controls.Add(fb);

}

#### 书籍入库：

##### 测试连接数据库：

private void TestDBConnection()

{

try

{

SqlConnection con = DBConnection.CreateDBConnection();

MessageBox.Show("数据库连接测试成功！");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

##### 添加书籍：

private void btn\_AddBook\_Click(object sender, EventArgs e)

{

BookBLL Bookbll = new BookBLL();

Book book = new Book();

book.BookNo = \_strBookNo;

book.BookName = txt\_bookName.Text.Trim();

book.Author = txt\_author.Text.Trim();

book.Publisher = txt\_publisher.Text.Trim();

book.Format = Convert.ToInt32(comb\_format.Text.Trim());

book.Page = Convert.ToInt32(txt\_page.Text.Trim());

book.Price = Convert.ToSingle(txt\_price.Text.Trim());

book.EnterDate = dateTimePicker1.Text.Trim();

book.Borrow = "1";

try

{

int row = Bookbll.AddBook(book);

if (row > 0)

MessageBox.Show("添加信息成功！");

}

#### 借阅/查询：

/// <summary>

/// 借阅

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void btn\_borrow\_Click(object sender, EventArgs e)

{

int row;

if (book.Borrow == "1")

{

book.Borrow = "2";

row = bookbll.UpdateBook(book);

if (row > 0)

{

MessageBox.Show("借阅成功！");

txt\_borrow.Text = "已借出";

}

}

else if (book.Borrow == "2")

{

MessageBox.Show("已经借出！");

}

else

{

MessageBox.Show("尚未入库！");

}

}

private void User\_borrow\_return\_Load(object sender, EventArgs e)

{

book = bookbll.SearchByBookNo(BookNo);

txt\_bookNo.Text = book.BookNo;

txt\_bookName.Text = book.BookName;

txt\_author.Text = book.Author;

txt\_publisher.Text = book.Publisher;

comb\_format.Text = Convert.ToString(book.Format);

txt\_page.Text = Convert.ToString(book.Page);

txt\_price.Text = Convert.ToString(book.Price);

dateTimePicker1.Text = book.EnterDate;

if (book.Borrow == "1")

{

txt\_borrow.Text = "可借阅";

}

else if (book.Borrow == "2")

{

txt\_borrow.Text = "已借出";

}

else

{

txt\_borrow.Text = "书籍不存在";

}

}

private void btn\_return\_Click(object sender, EventArgs e)

{

int row;

if (book.Borrow == "2")

{

book.Borrow = "1";

row = bookbll.UpdateBook(book);

if (row > 0)

{

MessageBox.Show("归还成功！");

txt\_borrow.Text = "可借阅";

}

}

else if (book.Borrow == "1")

{

MessageBox.Show("尚未借出！");

}

else if (book.Borrow == "0")

{

MessageBox.Show("尚未入库！");

}

}

#### 查询书库：

private void DBtable\_Load(object sender, EventArgs e)

{

List<Book> users = bookbll.SearchAllBooks();

dataGridView1.DataSource = users;

//DataGridView默认指向第一行

book.BookNo = dataGridView1.Rows[0].Cells[0].Value.ToString();

book.BookName = dataGridView1.Rows[0].Cells[1].Value.ToString();

book.Author = dataGridView1.Rows[0].Cells[2].Value.ToString();

book.Publisher = dataGridView1.Rows[0].Cells[3].Value.ToString();

book.Format = Convert.ToInt32(dataGridView1.Rows[0].Cells[4].Value);

book.Page = Convert.ToInt32(dataGridView1.Rows[0].Cells[5].Value);

book.Price=Convert.ToInt32(dataGridView1.Rows[0].Cells[6].Value);

//book.EnterDate = dataGridView1.Rows[0].Cells[7].Value.ToString();

}

#### 数据库介绍：

##### 基本信息：

#region 字段

private string \_BookNO;

private string \_BookName;

private string \_Author;

private string \_Publisher;

private int \_Format;

private int \_Page;

private float \_Price;

private string \_EnterDate;

private string \_Borrow;

#endregion

public string BookNo

{

get { return \_BookNO; }

set { \_BookNO = value; }

}

##### 数据库连接：

private static SqlConnection con;

private static string constr = @"Data Source=(LocalDB)\v11.0;AttachDbFilename=G:\bigproject\bigproject\MyDataBase.mdf;Integrated Security=True";

public static SqlConnection CreateDBConnection()

{

try

{

con = new SqlConnection(constr);

con.Open();

}

##### 写入数据库：

public int Addbook(Book usr)

{

int row;

try

{

con = DBConnection.CreateDBConnection();

string sql = "insert into T\_User (BookNo,BookName,Author,Publisher,Format,Page,Price,EnterDate,Borrow) values(@BookNo,@BookName,@Author,@Publisher,@Format,@Page,@Price,@EnterDate,@Borrow)";

cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@BookNo", usr.BookNo);

cmd.Parameters.AddWithValue("@BookName", usr.BookName);

cmd.Parameters.AddWithValue("@Author", usr.Author);

cmd.Parameters.AddWithValue("@Publisher", usr.Publisher);

cmd.Parameters.AddWithValue("@Format", usr.Format);

cmd.Parameters.AddWithValue("@Page", usr.Page);

cmd.Parameters.AddWithValue("@Price", usr.Price);

cmd.Parameters.AddWithValue("@EnterDate", usr.EnterDate);

cmd.Parameters.AddWithValue("@Borrow", usr.Borrow);

row = cmd.ExecuteNonQuery();

##### 根据书名搜索：

public Book SelectBookName(string BookName)//查询书名

{

Book users = new Book();

try

{

con = DBConnection.CreateDBConnection();

string sql = "select \* from T\_User where BookName like '%" + BookName + "%'";

cmd = new SqlCommand(sql, con);

rd = cmd.ExecuteReader();

while (rd.Read())

{

Book usr = new Book();

usr.BookNo = rd.GetString(0);

usr.BookName = rd.GetString(1);

usr.Author = rd.GetString(2);

usr.Publisher = rd.GetString(3);

usr.Format = rd.GetInt32(4);

usr.Page = rd.GetInt32(5);

usr.Price = Convert.ToSingle(rd.GetValue(6));

usr.EnterDate = rd.GetString(7);

usr.Borrow = rd.GetString(8);

##### 依据书号搜索：

public Book SelectByBookNo(string BookNo)

{

Book book = new Book();

try

{

con = DBConnection.CreateDBConnection();

string sql = "select \* from T\_User where BookNo like '%" + BookNo + "%'";

cmd = new SqlCommand(sql, con);

rd = cmd.ExecuteReader();

while (rd.Read())

{

book.BookNo = rd.GetString(0);

book.BookName = rd.GetString(1);

book.Author = rd.GetString(2);

book.Publisher = rd.GetString(3);

book.Format = rd.GetInt32(4);

book.Page = rd.GetInt32(5);

book.Price = Convert.ToSingle(rd.GetValue(6));

book.EnterDate = rd.GetString(7);

book.Borrow = rd.GetString(8);

}

##### 删除：

public int DeleteByNo(string No)

{

int row;

try

{

con = DBConnection.CreateDBConnection();

string sql = string.Format("delete from T\_User where BookNo='{0}'", No);

cmd = new SqlCommand(sql, con);

row = cmd.ExecuteNonQuery();

}

##### 数据库更新：

public int UpdateByNo(Book book)//借阅更新

{

int row;

try

{

con = DBConnection.CreateDBConnection();

string sql = "update T\_User set BookNo=@BookNo,BookName=@BookName,Author=@Author,Publisher=@Publisher,Format=@Format,Page=@Page,Price=@Price,EnterDate=@EnterDate,Borrow=@Borrow where BookNo=@BookNo";

cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@BookNo", book.BookNo);

cmd.Parameters.AddWithValue("@BookName", book.BookName);

cmd.Parameters.AddWithValue("@Author", book.Author);

cmd.Parameters.AddWithValue("@Publisher", book.Publisher);

cmd.Parameters.AddWithValue("@Format", book.Format);

cmd.Parameters.AddWithValue("@Page", book.Page);

cmd.Parameters.AddWithValue("@Price", book.Price);

cmd.Parameters.AddWithValue("@EnterDate", book.EnterDate);

cmd.Parameters.AddWithValue("@Borrow", book.Borrow);

row = cmd.ExecuteNonQuery();

}

##### 查询所有数据库信息：

public List<Book> SelectAllBooks()//查询所有书籍

{

List<Book> users = new List<Book>();

try

{

con = DBConnection.CreateDBConnection();

string sql = "select \* from T\_User";

cmd = new SqlCommand(sql, con);

rd = cmd.ExecuteReader();

while (rd.Read())

{

Book usr = new Book();

usr.BookNo = rd.GetString(0);

usr.BookName = rd.GetString(1);

usr.Author = rd.GetString(2);

usr.Publisher = rd.GetString(3);

usr.Format = rd.GetInt32(4);

usr.Page = rd.GetInt32(5);

usr.Price = Convert.ToSingle(rd.GetValue(6));

usr.EnterDate = rd.GetString(7);

usr.Borrow = rd.GetString(8);

users.Add(usr);

}

##### 业务逻辑层：

BookDAL bookdb = new BookDAL();

调用BookDAL中的函数，用来对数据进行操作

基本结构，以下是查询全部数据库的语句，查询单一信息，在括号传参即可

public List<Book> SearchAllBooks()

{

List<Book> books = bookdb.SelectAllBooks();

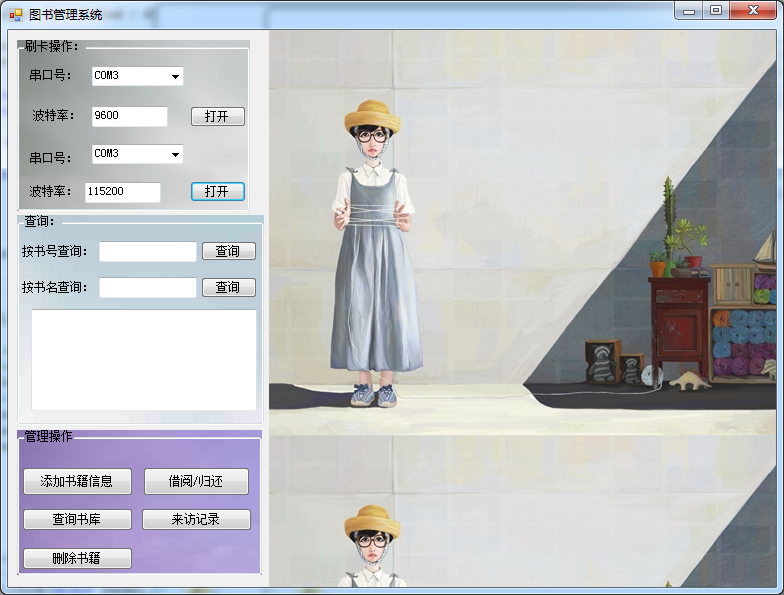
return books;

}

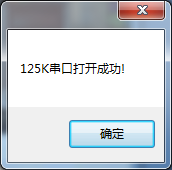
### 展示：

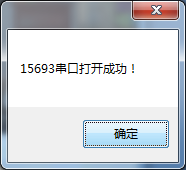
#### 功能展示

初始化页面：



打开串口：

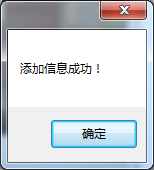




##### 添加书籍信息



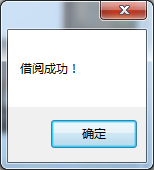


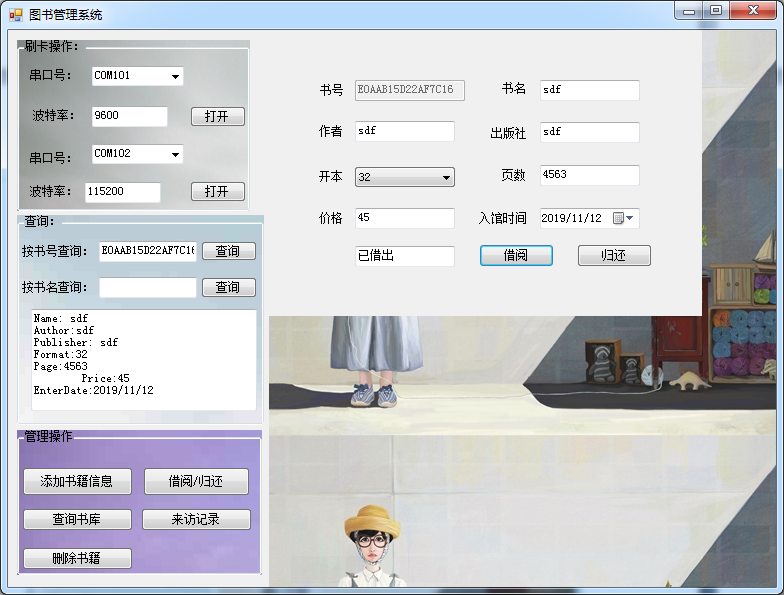


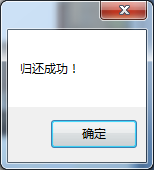
##### 借阅/归还

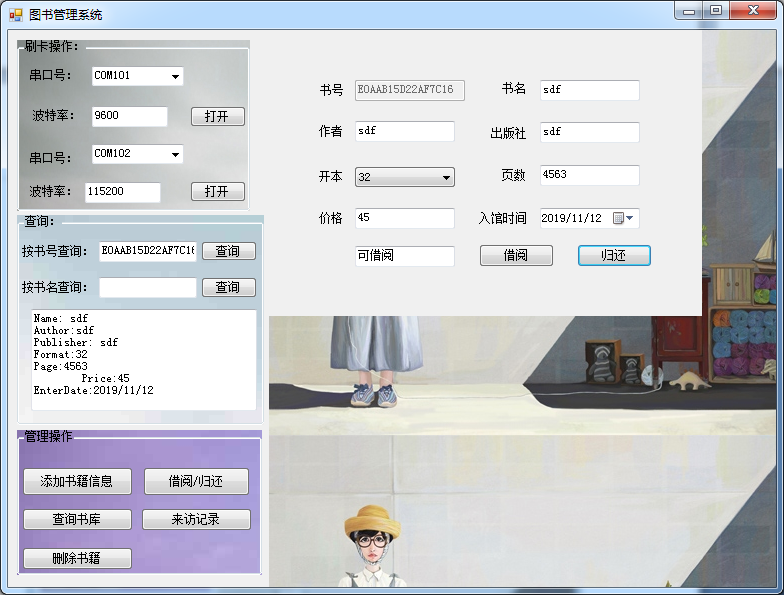
左下角是图书状态，其余是书籍信息





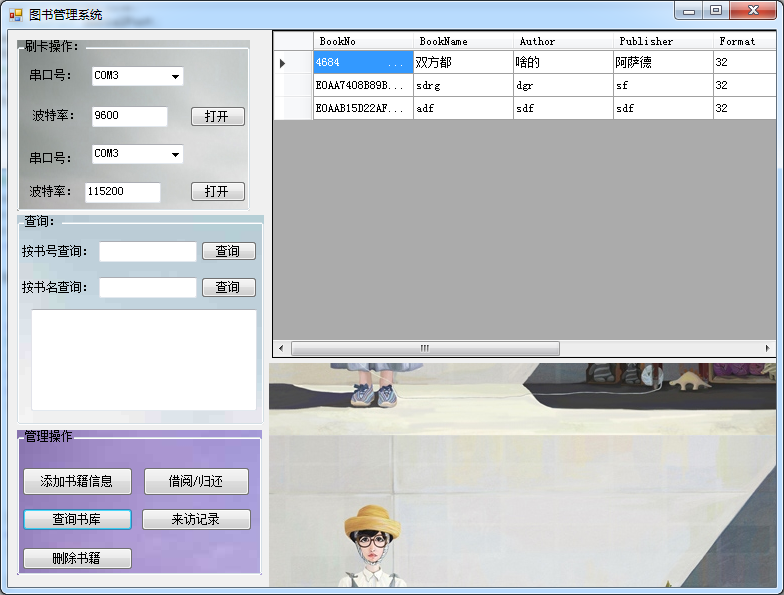






##### 查询书库

查看整个书库里的图书信息

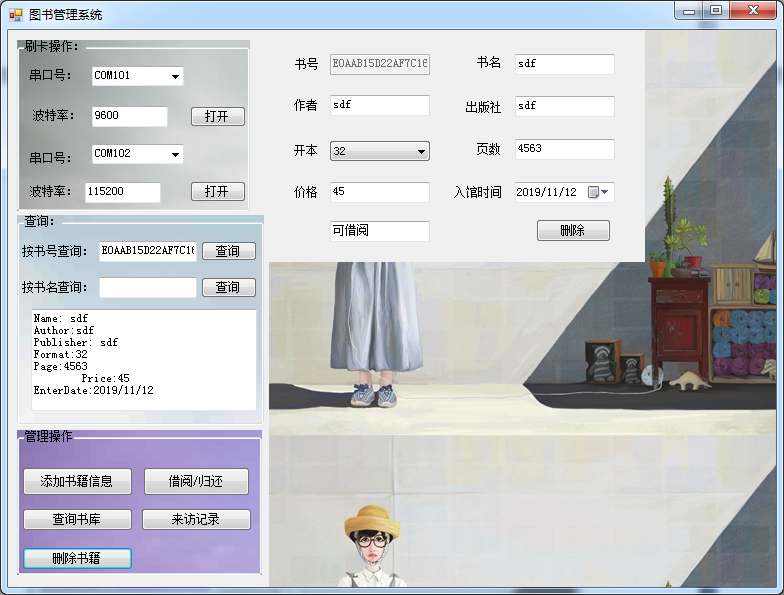


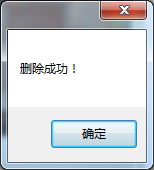
##### 来访记录



##### 删除书籍

每刷一张125K卡上面会显示卡信息





### 储存分配

#### 内部数据：

数据库存储了书籍的八个信息：

BookNo：书号

BookName：书名

Author：作者

Publisher：出版社

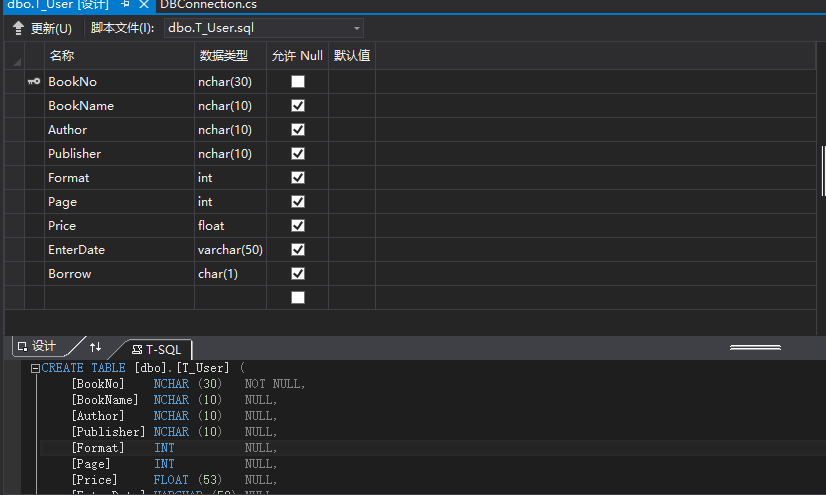
Format：版式

Page：页数

Price：价格

EnterDate：存入日期

Borrow：存储状态



### 限制条件

##### 技术约束

本项目的设计是在汉语程序设计语言（C#）的条件下进行的，技术设计采用软硬一体化的设计方法。

##### 环境约束

运行该软件所适用的具体环境windows使用2012版Visual Studio，基于.NET Framework 4.5框架，采用vs自带的数据库，尚不清楚版本号。

## 有待改进的地方：

实现功能比较简单，有待优化

125K刷卡数据传输受技术限制，没有实现

显示整个表数据时时间无法显示，是格式还是其他问题，尚不清楚

用书名查询书籍时显示的所有数据为0，尚未解决

可以在<https://github.com/shiyuekitty/RFID-Bigproject>上看到我上传的详细信息，以及原码

175044223

施玥