**C语言项目报告**

**项目名称：\_\_疫情期间的人员管理系统\_\_**

**项目成员：\_\_\_\_\_\_\_\_\_\_高士钦\_\_\_\_\_\_\_\_\_\_**

**填写日期：\_\_\_\_\_\_\_\_2020/09/08\_\_\_\_\_\_\_\_**

1. **摘要（Abstract）**

项目概述，突出创新点

本次我组设计的项目名称为疫情期间的社区人员管理系统，旨在设计出一个能够对社区人员基本信息和社区人员出行信息等信息数据进行合理地、系统地收集、处理和显示。通过与老师和同学们多次的交流与探讨，本项目除了具备人员管理系统最基本的三大主要功能（包括人员基本/人员出行信息的录入、人员基本/人员出行信息的修改、人员基本/人员出行信息的删除）以外，还通过MFC(Microsoft基础类库)设计了更为人性化的交互界面、并引入Access数据库技术来改进我们输入、修改、删除数据的方式途径、还增加了我们日常生活中随处可见的二维码功能（包括二维码的生成、二维码的解码），这几项技术和功能的引入令本组设计出来的疫情期间的社区人员管理系统更具时代性、美观性、新颖性和实用性，也是本程序的亮点所在。

1. **问题描述（Problem Statement）**

项目背景和拟解决的问题

项目背景与拟解决的现实问题：2020年初的新冠肺炎疫情给所有人的学习和生活带来了巨大的影响，在疫情期间，对于每一个社区每一个单位，做好对于人员的信息的记录、统计是极为重要的一件事，这次我组设计的C语言项目就是基于疫情这个大背景，希望通过合理的方式来设计出一个较为全面的个人信息登记、查询收集与汇总的管理系统，可以及时性、全面性地确定小区各个成员的健康状况和流动人员信息的登记与记录，并且通过二维码的功能引入，以图片的方式来存储个体的所有信息，更加方便管理人员对社区人员信息的分类管理和大量文字以及数字内容的存储和汇集。本系统的设计与实现可以帮助社区管理者及时掌握大量的人员信息数据、对异常情况的及时报备，可以很好地解决社区登记人员人手不足且工作量大地局面，实时汇总数据，能够为疫情防控提供决策支持数据以及更完善的指导防控举措。

1. **组内分工（Group Division）**

包含小组成员信息、具体分工及所占工作量百分比

小组成员🡪

姓名：高士钦 年级：2018级 专业：唐班化学方向 学号：12180717

分工及工作量百分比🡪

高士钦----100%工作量

5%的工作量-----搜集多种对于项目的想法，研究各种方法下面操作的工程量和对项目可操作性的分析

30%的工作量-----确定项目主要内容，研究好项目中对每一个附属功能的的代码分析和实现

65%的工作量-----落实好每一个子功能下的子函数代码的编写（包括用户基本信息/用户出行信息的增加、删除和修改），Access数据库的学习和引入，ADO技术的学习和引入，QR-code源码的查找、学习和引入）

1. **分析（Analysis）**

项目的总体框架设计逻辑

我们需要一个能够帮助社区管理人员通过账号和密码进行登陆的登陆系统。登陆成功后，我们可以进入主菜单，进入菜单后，用户可以执行以下功能：添加用户基本信息/出行信息、修改用户基本信息/出行信息、删除用户基本信息/出行信息、根据（姓名、年龄、出行方式、是否患病、是否被隔离）来进行查询、统计并显示所有用户基本/出行信息、且在每一个界面都能够找到退出系统的按钮返回到初始界面。

我们所进行的所有操作（包括增加、修改、删除各种用户信息数据）都存储在Access数据库中，我们在引入Access数据库时，分别设计了三个表：管理员登陆所用的数据库表、小区用户信息数据库表、小区用户出行信息数据库表，按顺序包括2个字段、7个字段、9个字段。我们通过引入Access数据库以表的方式对数据信息进行写入、修改和删除等操作。我们引入Access数据库不仅仅是因为他存储方式简单、易于维护管理，还因为Access支持ODBC(开发数据库互联)，利用Access优秀的动态数据交换和对象的链接和嵌入特性，可以在一个数据表中嵌入位图、建立动态的数据库报表和窗体、还可以将程序应用于网络、并与网络上的动态数据相连接为我们以后程序的改进提供良好的基础。除了这三种基本功能以外，还成功地在主菜单界面上实现了二维码的生成和解码功能。

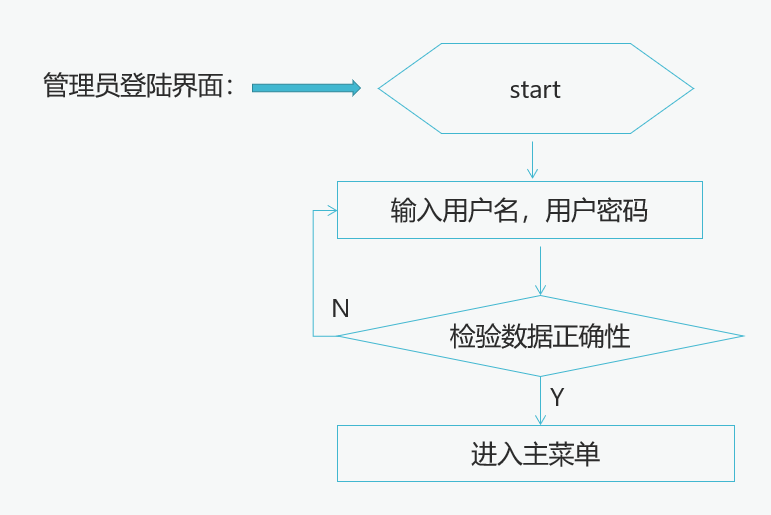
本项目的二维码功能的实现依赖于Zxing, Zxing是一个开放源码的，用Java实现的多种格式一维或者二位的条码图像处理库，它包含了联系到其它语言的端口，通常使用Zxing生成相应的二维码以及解码。生成二维码的编码技术是指基于给定内容生成二维条码图片，Zxing可以对生成的二维码的图片格式、各项参数以及二维码类型进行设置，生成的图片包含给定的内容，当使用识别设备进行扫描时，能够读出给定的内容。使用Zxing进行编码的步骤主要如下：首先，将所需的Zxing类库中的包、导入工程；其次，对需要生成二维码的给定内容进行编码方式处理，防止在显示中文时出现乱码问题，并指定所生成二位码图片的路径、名称、文件格式；然后，找到Zxing中二维码QR码所对应的编码类QR-codeWriter,调用encode方法生成给定内容对应的比特矩阵；最后将比特矩阵转化为制定的图片格式。

我们二维码的处理流程中主要包括五个预处理过程（灰度化、加噪平滑、二值化、旋转矫正、畸变矫正），这里灰度化处理的原因是由于我们日常生活中使用数码相机等摄像设备获取的QR码是彩色的，和灰度图像相比，彩色图像含有的信息量较大，所占存储空间更多，通过灰度化处理可以将彩色图像转换成灰度图像，减少了存储空间降低运算量。二值化处理同样也便于数据的压缩、特征的突出和图像的识别。图像二值化的方法是通过预先设定的一个阈值，将灰度图像分为小于这个阈值和大于这个阈值两个部分，通过阈值的选取可以将图像的背景区和条码符号分开，这里我们利用自适应阈值法对图像进行二值化处理。自适应阈值法的基本原理是采用某一假定的灰度值t将图像的灰度分为两组，当两组的方差最大时，该灰度值就是二值化的最佳阈值。同样的，在二维码图像的采集过程中，由于拍摄角度的问题会导致图像产生一定的旋转畸变，这里我们采用霍夫变换的方法对该情况进行矫正。QRcode解码技术是指从二维码中读取出所包含的信息，本项目直接调用Zxing完成解码识别的过程，主要包括以下几个步骤：首先，导入Zxing类库中的包；其次，指定二维条码图片的路径；然后，找到Zxing中二维码QR码所对应的解码类QR-codeReader,调用其decode方法胜出二维码所包含信息对应的对象；最后，读取出对象中的信息。

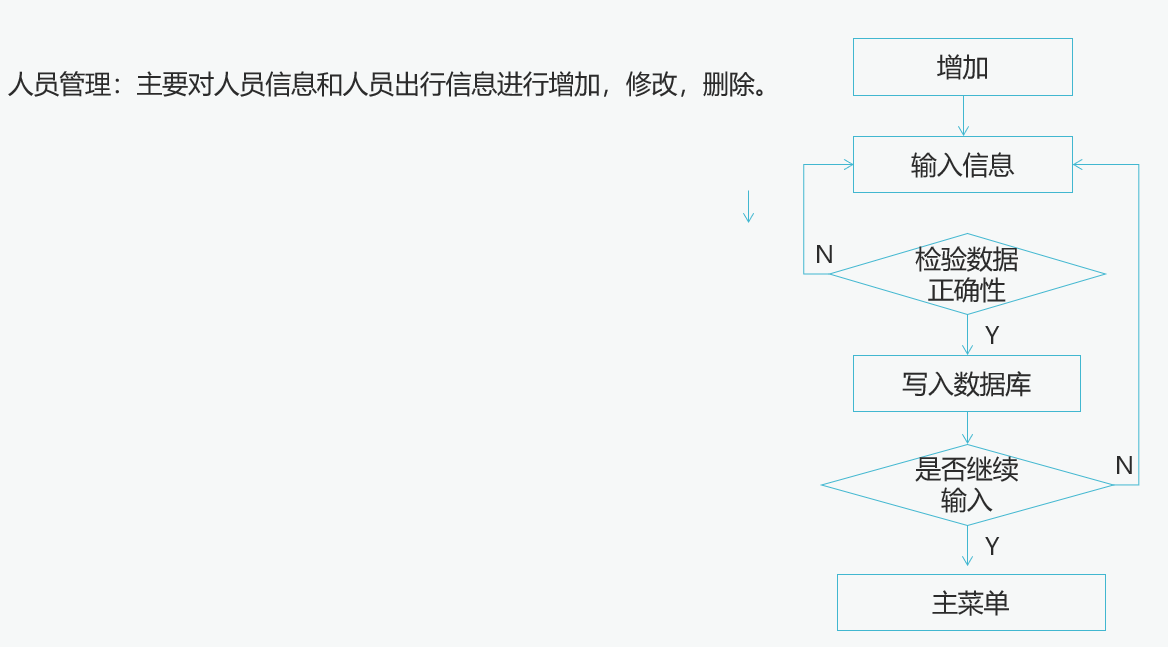
1. **设计（Design）**

各功能模块的介绍及流程图

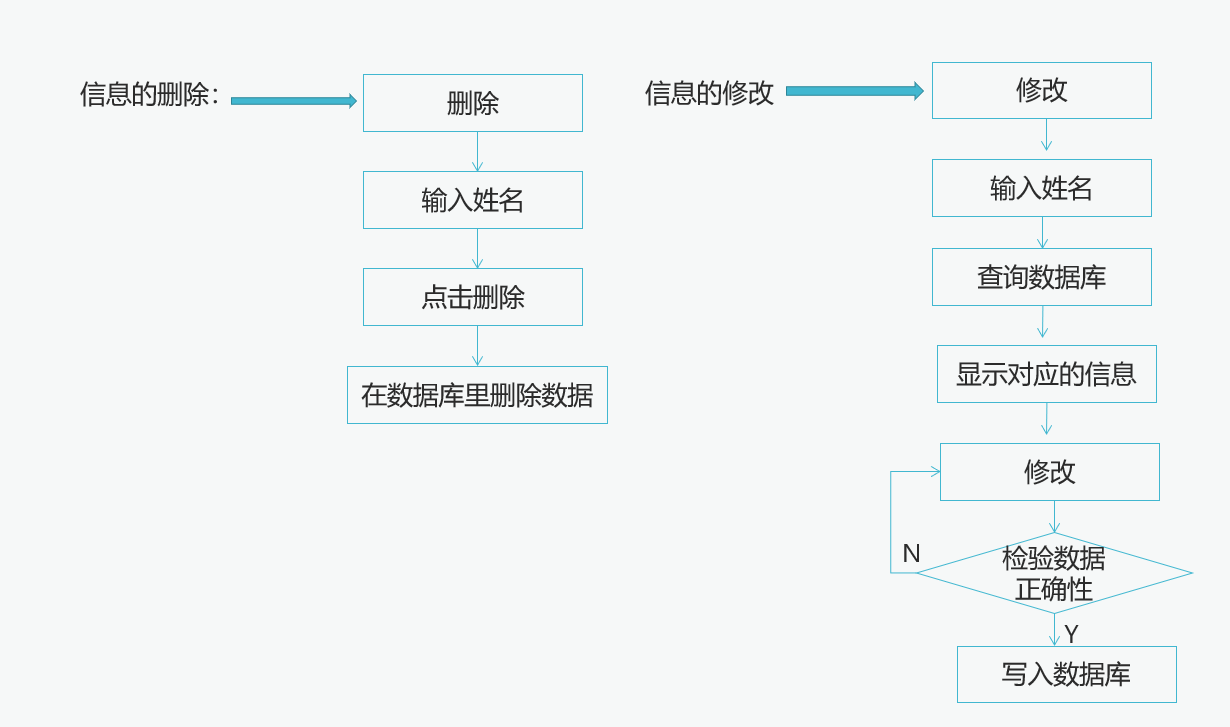
**🡪登录功能流程图：**



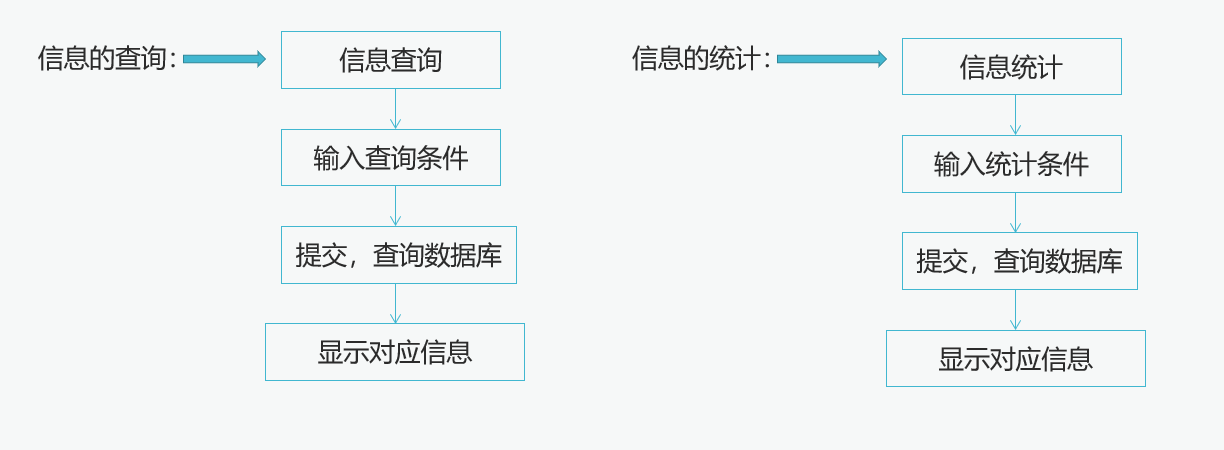
**🡪增加人员信息功能流程图：**



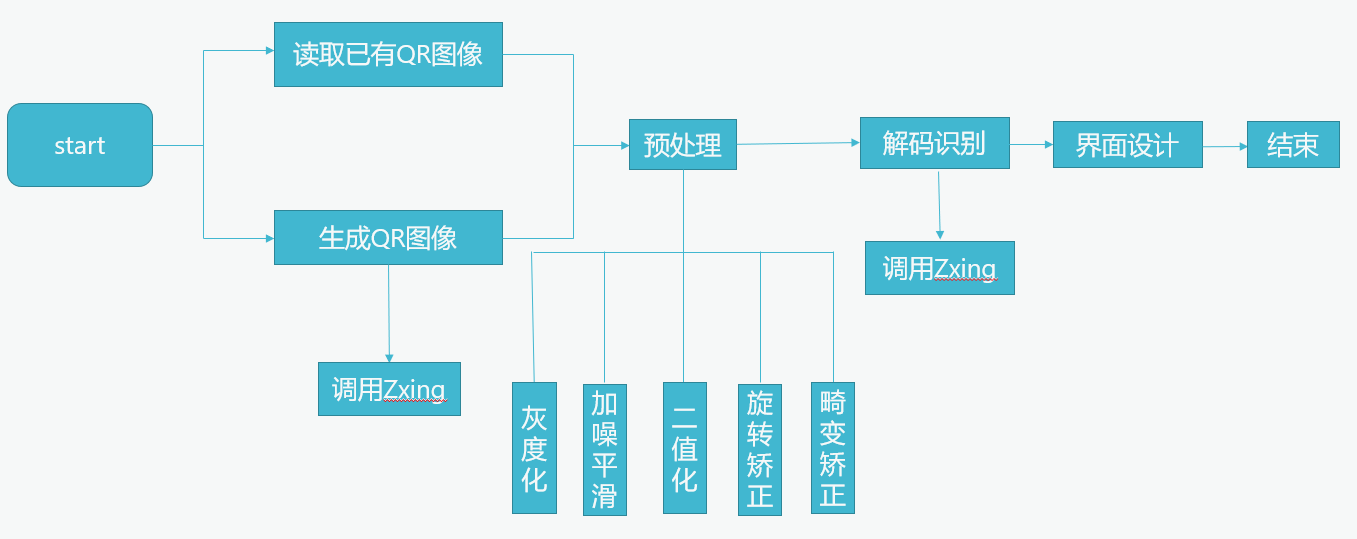
**🡪删除人员信息&修改人员信息功能流程图：**



**🡪人员信息的查询功能&人员信息统计功能流程图：**



**🡪二维码的生成和解码功能的实现流程图：**



1. **实施（Implementation）**

核心模块函数的具体实现代码

**🡪对话框的实现：**

IMPLEMENT\_DYNAMIC(CUserAdd, CDialogEx)

CUserAdd::CUserAdd(CWnd\* pParent /\*=NULL\*/)

: CDialogEx(CUserAdd::IDD, pParent)

, m\_user\_name(\_T(""))

, m\_age(0)

, m\_sfz(\_T(""))

, m\_phone(\_T(""))

{

}

**🡪增加用户信息功能的实现和对消息处理的程序的实现：**

CUserAdd::~CUserAdd()

{

}

void CUserAdd::DoDataExchange(CDataExchange\* pDX)

{

CDialogEx::DoDataExchange(pDX);

DDX\_Text(pDX, IDC\_EDIT1, m\_user\_name);

DDX\_Text(pDX, IDC\_EDIT3, m\_age);

DDX\_Control(pDX, IDC\_COMBO1, m\_sex);

DDX\_Text(pDX, IDC\_EDIT4, m\_sfz);

DDX\_Text(pDX, IDC\_EDIT5, m\_phone);

DDX\_Control(pDX, IDC\_COMBO2, m\_status);

DDX\_Text(pDX, IDC\_EDIT6, m\_addr);

}

BEGIN\_MESSAGE\_MAP(CUserAdd, CDialogEx)

ON\_BN\_CLICKED(IDOK, &CUserAdd::OnBnClickedOk)

END\_MESSAGE\_MAP()

BOOL CUserAdd::OnInitDialog()

{

CDialogEx::OnInitDialog();

m\_user\_name = TEXT("");

m\_sfz = TEXT("");

m\_phone = TEXT("");

m\_addr = TEXT("");

m\_age = 0;

m\_sex.AddString(TEXT("男"));

m\_sex.AddString(TEXT("女"));

m\_sex.SetCurSel(0);

m\_status.AddString(TEXT("未出行"));

m\_status.AddString(TEXT("出行"));

m\_status.SetCurSel(0);

UpdateData(FALSE);

return TRUE; // return TRUE unless you set the focus to a control

}

// CUserAdd 消息处理程序

void CUserAdd::OnBnClickedOk()

{

UpdateData(TRUE);

if (m\_user\_name.IsEmpty()) /\*判断用户名信息是否为空\*/

{

AfxMessageBox(TEXT("请输入用户名！"));

return;

}

if (m\_sfz.IsEmpty()) /\*判断信息是否为空\*/

{

AfxMessageBox(TEXT("请输入用户密码！"));

return;

}

if (m\_phone.IsEmpty()) /\*判断信息是否为空\*/

{

AfxMessageBox(TEXT("请输入用户身份证信息！"));

return;

}

if (m\_addr.IsEmpty()) /\*判断信息是否为空\*/

{

AfxMessageBox(TEXT("请输入用户地址！"));

return;

}

if (CheckID(m\_sfz) == false)

{

AfxMessageBox(TEXT("身份证信息不正确,请重新输入！"));

return;

}

CString strSqlText;

strSqlText.Format(TEXT("insert into UserInfo(UserName,UserAge,UserSex,UserSfz,UserPhone,UserStatus,UserAddr) values ('%s',%d,%d,'%s','%s',%d,'%s')"),

m\_user\_name, m\_age, m\_sex.GetCurSel(), m\_sfz, m\_phone, m\_status.GetCurSel(), m\_addr);

//strSqlText.Format(TEXT("insert into UserInfo(UserName) values ('%s')"),m\_user\_name);

AdoWorker \*ado = NULL;

ado = new AdoWorker();

BOOL bExec = FALSE;

//先删除原有数据

bExec = ado->ExecuteSql(strSqlText);//将数据插入到了表格中

if (bExec)

{

m\_nSex = m\_sex.GetCurSel();

m\_nStatus = m\_status.GetCurSel();

CDialogEx::OnOK();

}

else

AfxMessageBox(TEXT("用户名字有冲突，请检查！"));

delete ado;

}

bool CUserAdd::CheckID(CString ID\_num)/\*身份证校验\*/

{

int weight[] = { 7, 9, 10, 5, 8, 4, 2, 1, 6, 3, 7, 9, 10, 5, 8, 4, 2 };

char validate[] = { '1', '0', 'X', '9', '8', '7', '6', '5', '4', '3', '2' };

int sum = 0;

int mode = 0;

int i = 0;

int length = ID\_num.GetLength();

if (length == 18)

{

for (i = 0; i<length - 1; i++)

sum = sum + (ID\_num[i] - '0') \* weight[i];

mode = sum % 11;

if (validate[mode] == ID\_num[17])

return true;

}

return false;

}

**🡪删除用户信息功能的实现：**

CUserDel::~CUserDel()

{

}

void CUserDel::DoDataExchange(CDataExchange\* pDX)

{

CDialogEx::DoDataExchange(pDX);

DDX\_Text(pDX, IDC\_EDIT1, m\_user\_name);

}

BEGIN\_MESSAGE\_MAP(CUserDel, CDialogEx)

ON\_BN\_CLICKED(IDOK, &CUserDel::OnBnClickedOk)

END\_MESSAGE\_MAP()

// CUserDel 消息处理程序

void CUserDel::OnBnClickedOk()

{

UpdateData(TRUE);

if (m\_user\_name.IsEmpty()) /\*判断用户名信息是否为空\*/

{

AfxMessageBox(TEXT("请输入用户名！"));

return;

}

CString strSqlText;

strSqlText.Format(TEXT("delete from UserInfo where UserName='%s'"), m\_user\_name);

AdoWorker \*ado = NULL;

ado = new AdoWorker();

BOOL bExec = FALSE;

//先删除原有数据

bExec = ado->ExecuteSql(strSqlText);//将数据插入到了表格中

if (bExec)

{

CDialogEx::OnOK();

}

else

AfxMessageBox(TEXT("用户名字不正确，请检查！"));

CDialogEx::OnOK();

}

**🡪修改用户信息功能的实现：**

CUserEdit::~CUserEdit()

{

}

void CUserEdit::DoDataExchange(CDataExchange\* pDX)

{

CDialogEx::DoDataExchange(pDX);

DDX\_Text(pDX, IDC\_EDIT1, m\_user\_name);

DDX\_Text(pDX, IDC\_EDIT3, m\_age);

DDX\_Control(pDX, IDC\_COMBO1, m\_sex);

DDX\_Text(pDX, IDC\_EDIT4, m\_sfz);

DDX\_Text(pDX, IDC\_EDIT5, m\_phone);

DDX\_Control(pDX, IDC\_COMBO2, m\_status);

DDX\_Text(pDX, IDC\_EDIT6, m\_addr);

}

BEGIN\_MESSAGE\_MAP(CUserEdit, CDialogEx)

ON\_BN\_CLICKED(IDC\_QUERY, &CUserEdit::OnBnClickedButton1)

ON\_BN\_CLICKED(IDOK, &CUserEdit::OnBnClickedOk)

END\_MESSAGE\_MAP()

// CUserEdit 消息处理程序

void CUserEdit::OnBnClickedButton1()

{

// TODO: 在此添加控件通知处理程序代码

UpdateData(TRUE);

if (m\_user\_name.IsEmpty()) /\*判断用户名信息是否为空\*/

{

AfxMessageBox(TEXT("请输入用户名！"));

return;

}

CString strSqlText;

strSqlText.Format(TEXT("select \* from userinfo where UserName='%s'"), m\_user\_name);

//////////////////////////////////创建SQL语句执行类实例

AdoWorker \*ado = NULL;

\_UserInfo \*pUserNameInfo = NULL;

ado = new AdoWorker();

pUserNameInfo = (\_UserInfo \*)ado->ExecuteSql(strSqlText, 0);

if (pUserNameInfo == NULL)

{

MessageBox(TEXT("没有这样的账号，请先检查账号的正确性！"), NULL, MB\_ICONERROR);

delete pUserNameInfo;

delete ado;

return;

}

else//存在这样的账号，在此设定表格中第二行的姓名

{

m\_user\_name = pUserNameInfo->UserName;

m\_age = pUserNameInfo->UserAge;

m\_sex.SetCurSel(pUserNameInfo->UserSex);

m\_sfz = pUserNameInfo->UserSfz;

m\_phone = pUserNameInfo->UserPhone;

m\_addr = pUserNameInfo->UserAddress;

m\_status.SetCurSel(pUserNameInfo->UserStatus);

delete pUserNameInfo;

delete ado;

}

UpdateData(FALSE);

}

void CUserEdit::OnBnClickedOk()

{

UpdateData(TRUE);

if (m\_user\_name.IsEmpty()) /\*判断用户名信息是否为空\*/

{

AfxMessageBox(TEXT("请输入用户名！"));

return;

}

if (m\_sfz.IsEmpty()) /\*判断信息是否为空\*/

{

AfxMessageBox(TEXT("请输入用户密码！"));

return;

}

if (m\_phone.IsEmpty()) /\*判断信息是否为空\*/

{

AfxMessageBox(TEXT("请输入用户身份证信息！"));

return;

}

if (m\_addr.IsEmpty()) /\*判断信息是否为空\*/

{

AfxMessageBox(TEXT("请输入用户地址！"));

return;

}

if (CheckID(m\_sfz) == false)

{

AfxMessageBox(TEXT("身份证信息不正确,请重新输入！"));

return;

}

CString strSqlText;

strSqlText.Format(TEXT("update UserInfo set UserAge=%d,UserSex=%d,UserSfz='%s',UserPhone='%s',UserStatus=%d,UserAddr='%s' where UserName='%s'"), m\_age, m\_sex.GetCurSel(), m\_sfz, m\_phone, m\_status.GetCurSel(), m\_addr, m\_user\_name);

AdoWorker \*ado = NULL;

ado = new AdoWorker();

BOOL bExec = FALSE;

//先删除原有数据

bExec = ado->ExecuteSql(strSqlText);//将数据插入到了表格中

if (bExec)

{

m\_nSex = m\_sex.GetCurSel();

m\_nStatus = m\_status.GetCurSel();

CDialogEx::OnOK();

}

else

AfxMessageBox(TEXT("用户名字有冲突，请检查！"));

CDialogEx::OnOK();

}

BOOL CUserEdit::OnInitDialog()

{

CDialogEx::OnInitDialog();

m\_user\_name = TEXT("");

m\_sfz = TEXT("");

m\_phone = TEXT("");

m\_addr = TEXT("");

m\_age = 0;

m\_sex.AddString(TEXT("男"));

m\_sex.AddString(TEXT("女"));

m\_sex.SetCurSel(0);

m\_status.AddString(TEXT("未出行"));

m\_status.AddString(TEXT("出行"));

m\_status.SetCurSel(0);

UpdateData(FALSE);

return TRUE; // return TRUE unless you set the focus to a control

}

bool CUserEdit::CheckID(CString ID\_num)/\*身份证校验\*/

{

int weight[] = { 7, 9, 10, 5, 8, 4, 2, 1, 6, 3, 7, 9, 10, 5, 8, 4, 2 };

char validate[] = { '1', '0', 'X', '9', '8', '7', '6', '5', '4', '3', '2' };

int sum = 0;

int mode = 0;

int i = 0;

int length = ID\_num.GetLength();

if (length == 18)

{

for (i = 0; i<length - 1; i++)

sum = sum + (ID\_num[i] - '0') \* weight[i];

mode = sum % 11;

if (validate[mode] == ID\_num[17])

return true;

}

return false;

}

**🡪用户信息的查询功能的实现：**

CUserQuery::~CUserQuery()

{

}

void CUserQuery::DoDataExchange(CDataExchange\* pDX)

{

CDialogEx::DoDataExchange(pDX);

DDX\_Control(pDX, IDC\_COMBO1, m\_select\_type);

DDX\_Text(pDX, IDC\_USER\_NAME, m\_user\_name);

DDX\_Control(pDX, IDC\_USER\_LIST, m\_UserList);

DDX\_Control(pDX, IDC\_USER\_LIST2, m\_UserStatusList);

DDX\_Control(pDX, IDC\_COMBO2, m\_tool\_select);

DDX\_Control(pDX, IDC\_USER\_NAME, m\_edit\_user\_name);

DDX\_Control(pDX, IDC\_USER\_AGE\_LIST, m\_age\_list);

}

BEGIN\_MESSAGE\_MAP(CUserQuery, CDialogEx)

ON\_BN\_CLICKED(IDC\_QUERY, &CUserQuery::OnBnClickedQuery)

ON\_CBN\_SELCHANGE(IDC\_COMBO1, &CUserQuery::OnSelchangeCombo1)

END\_MESSAGE\_MAP()

// CUserQuery 消息处理程序

BOOL CUserQuery::OnInitDialog()

{

CDialogEx::OnInitDialog();

m\_age\_list.InsertColumn(0, TEXT("姓名"), LVCFMT\_LEFT, 80);

m\_age\_list.InsertColumn(1, TEXT("年龄"), LVCFMT\_LEFT, 40);

m\_age\_list.InsertColumn(2, TEXT("性别"), LVCFMT\_LEFT, 40);

m\_age\_list.InsertColumn(3, TEXT("身份证"), LVCFMT\_LEFT, 160);

m\_age\_list.InsertColumn(4, TEXT("电话"), LVCFMT\_LEFT, 90);

m\_age\_list.InsertColumn(5, TEXT("状态"), LVCFMT\_LEFT, 50);

m\_age\_list.InsertColumn(6, TEXT("地址"), LVCFMT\_LEFT, 200);

m\_age\_list.SetExtendedStyle(LVS\_EX\_GRIDLINES);

m\_age\_list.InsertItem(0, NULL);

m\_UserList.InsertColumn(0, TEXT("姓名"), LVCFMT\_LEFT, 80);

m\_UserList.InsertColumn(1, TEXT("年龄"), LVCFMT\_LEFT, 40);

m\_UserList.InsertColumn(2, TEXT("性别"), LVCFMT\_LEFT, 40);

m\_UserList.InsertColumn(3, TEXT("身份证"), LVCFMT\_LEFT, 160);

m\_UserList.InsertColumn(4, TEXT("电话"), LVCFMT\_LEFT, 90);

m\_UserList.InsertColumn(5, TEXT("状态"), LVCFMT\_LEFT, 50);

m\_UserList.InsertColumn(6, TEXT("地址"), LVCFMT\_LEFT, 200);

m\_UserList.SetExtendedStyle(LVS\_EX\_GRIDLINES);

m\_UserList.InsertItem(0, NULL);

m\_UserStatusList.InsertColumn(0, TEXT("姓名"), LVCFMT\_LEFT, 80);

m\_UserStatusList.InsertColumn(1, TEXT("开始时间"), LVCFMT\_LEFT, 80);

m\_UserStatusList.InsertColumn(2, TEXT("结束时间"), LVCFMT\_LEFT, 80);

m\_UserStatusList.InsertColumn(3, TEXT("出行地址"), LVCFMT\_LEFT, 100);

m\_UserStatusList.InsertColumn(4, TEXT("结束地址"), LVCFMT\_LEFT, 100);

m\_UserStatusList.InsertColumn(5, TEXT("出行工具"), LVCFMT\_LEFT, 60);

m\_UserStatusList.InsertColumn(6, TEXT("工具信息"), LVCFMT\_LEFT, 120);

m\_UserStatusList.InsertColumn(7, TEXT("隔离"), LVCFMT\_LEFT, 40);

m\_UserStatusList.InsertColumn(8, TEXT("患病"), LVCFMT\_LEFT, 40);

m\_UserStatusList.SetExtendedStyle(LVS\_EX\_GRIDLINES);

m\_UserStatusList.InsertItem(0, NULL);

m\_select\_type.AddString(TEXT("按姓名查询"));

m\_select\_type.AddString(TEXT("按目的地查询"));

m\_select\_type.AddString(TEXT("按交通工具查询"));

m\_select\_type.AddString(TEXT("按隔离人员查询"));

m\_select\_type.AddString(TEXT("按患病人员查询"));

m\_select\_type.AddString(TEXT("按年龄查询"));

m\_select\_type.SetCurSel(0);

m\_tool\_select.AddString(TEXT("飞机"));

m\_tool\_select.AddString(TEXT("火车"));

m\_tool\_select.AddString(TEXT("汽车"));

m\_tool\_select.AddString(TEXT("自驾"));

m\_tool\_select.AddString(TEXT("步行"));

m\_tool\_select.AddString(TEXT("其他"));

m\_tool\_select.SetCurSel(0);

m\_edit\_user\_name.ShowWindow(SW\_SHOW);

m\_tool\_select.ShowWindow(SW\_HIDE);

m\_age\_list.ShowWindow(SW\_HIDE);

m\_UserList.ShowWindow(SW\_SHOW);

m\_UserStatusList.ShowWindow(SW\_SHOW);

// TODO: 在此添加额外的初始化

return TRUE; // return TRUE unless you set the focus to a control

// 异常: OCX 属性页应返回 FALSE

}

void CUserQuery::OnBnClickedQuery()

{

// TODO: 在此添加控件通知处理程序代码

UpdateData(TRUE);

int nSelect = m\_select\_type.GetCurSel();

if (nSelect == 0)

{

if (m\_user\_name.IsEmpty()) /\*判断用户名信息是否为空\*/

{

AfxMessageBox(TEXT("请输入用户名！"));

return;

}

WriteUserList(m\_user\_name);

WriteUserStateList(m\_user\_name);

}

else if (nSelect == 1)

{

if (m\_user\_name.IsEmpty()) /\*判断用户名信息是否为空\*/

{

AfxMessageBox(TEXT("请输入目的地信息！"));

return;

}

CString strSqlText;

strSqlText.Format(TEXT("select \* from Recordinfo where EndAddr='%s'"), m\_user\_name);

WriteQueryList(strSqlText);

}

else if (nSelect == 2)

{

int nToosSelect = m\_tool\_select.GetCurSel();

CString strSqlText;

strSqlText.Format(TEXT("select \* from Recordinfo where UserTool=%d"), nToosSelect);

WriteQueryList(strSqlText);

}

else if (nSelect == 3)

{

CString strSqlText;

strSqlText.Format(TEXT("select \* from Recordinfo where HasIsOlace=%d"), 1);

WriteQueryList(strSqlText);

}

else if (nSelect == 4)

{

CString strSqlText;

strSqlText.Format(TEXT("select \* from Recordinfo where HasIsInfect=%d"), 1);

WriteQueryList(strSqlText);

}

else if (nSelect == 5)

{

CString strSqlText;

strSqlText.Format(TEXT("select \* from userinfo order by UserAge"));

//////////////////////////////////创建SQL语句执行类实例

AdoWorker \*ado = NULL;

\_RecordUserInfo \*pUserNameInfo = NULL;

ado = new AdoWorker();

pUserNameInfo = (\_RecordUserInfo \*)ado->ExecuteSql(strSqlText, 4);

if (pUserNameInfo == NULL)

{

delete pUserNameInfo;

delete ado;

return;

}

else//存在这样的账号，在此设定表格中第二行的姓名

{

WriteAllUserList(pUserNameInfo);

delete ado;

}

}

}

void CUserQuery::WriteQueryList(CString QuerySql)

{

//////////////////////////////////创建SQL语句执行类实例

AdoWorker \*ado = NULL;

\_RecordStatusInfo \*pUserNameInfo = NULL;

ado = new AdoWorker();

pUserNameInfo = (\_RecordStatusInfo \*)ado->ExecuteSql(QuerySql, 3);

if (pUserNameInfo == NULL)

{

delete pUserNameInfo;

delete ado;

return;

}

else//存在这样的账号，在此设定表格中第二行的姓名

{

WriteList(pUserNameInfo);

delete ado;

}

}

void CUserQuery::WriteUserList(CString UserName)

{

CString strSqlText;

strSqlText.Format(TEXT("select \* from userinfo where UserName='%s'"), UserName);

//////////////////////////////////创建SQL语句执行类实例

AdoWorker \*ado = NULL;

\_UserInfo \*pUserNameInfo = NULL;

ado = new AdoWorker();

pUserNameInfo = (\_UserInfo \*)ado->ExecuteSql(strSqlText, 0);

if (pUserNameInfo == NULL)

{

delete pUserNameInfo;

delete ado;

return;

}

else//存在这样的账号，在此设定表格中第二行的姓名

{

WriteList(pUserNameInfo);

delete pUserNameInfo;

delete ado;

}

}

void CUserQuery::WriteUserStateList(CString UserName)

{

CString strSqlText;

strSqlText.Format(TEXT("select \* from Recordinfo where UserName='%s'"), UserName);

//////////////////////////////////创建SQL语句执行类实例

AdoWorker \*ado = NULL;

\_RecordStatusInfo \*pUserNameInfo = NULL;

ado = new AdoWorker();

pUserNameInfo = (\_RecordStatusInfo \*)ado->ExecuteSql(strSqlText, 3);

if (pUserNameInfo == NULL)

{

delete pUserNameInfo;

delete ado;

return;

}

else//存在这样的账号，在此设定表格中第二行的姓名

{

WriteList(pUserNameInfo);

delete ado;

}

}

void CUserQuery::WriteList(\_UserInfo \*pRecordInfo)

{

m\_UserList.DeleteAllItems();

m\_UserList.InsertItem(0, NULL);//新插入一行

//AfxMessageBox(m\_UserAddDlg.m\_user\_name);

m\_UserList.SetItemText(0, 0, pRecordInfo->UserName);

CString strAge = \_T("");

strAge.Format(TEXT("%d"), pRecordInfo->UserAge);

m\_UserList.SetItemText(0, 1, strAge);

if (pRecordInfo->UserSex == 0)

m\_UserList.SetItemText(0, 2, TEXT("男"));

else

m\_UserList.SetItemText(0, 2, TEXT("女"));

m\_UserList.SetItemText(0, 3, pRecordInfo->UserSfz);

m\_UserList.SetItemText(0, 4, pRecordInfo->UserPhone);

if (pRecordInfo->UserStatus == 0)

m\_UserList.SetItemText(0, 5, TEXT("未出行"));

else

m\_UserList.SetItemText(0, 5, TEXT("出行"));

m\_UserList.SetItemText(0, 6, pRecordInfo->UserAddress);

}

void CUserQuery::WriteAllUserList(\_RecordUserInfo \*pRecordInfo)

{

m\_age\_list.DeleteAllItems();

\_RecordUserInfo \*pDelete = NULL;

pDelete = pRecordInfo;

for (int i = 0; pRecordInfo != NULL; i++)

{

m\_age\_list.InsertItem(i, NULL);//新插入一行

//AfxMessageBox(m\_UserAddDlg.m\_user\_name);

m\_age\_list.SetItemText(i, 0, pRecordInfo->UserName);

CString strAge = \_T("");

strAge.Format(TEXT("%d"), pRecordInfo->UserAge);

m\_age\_list.SetItemText(i, 1, strAge);

if (pRecordInfo->UserSex == 0)

m\_age\_list.SetItemText(i, 2, TEXT("男"));

else

m\_age\_list.SetItemText(i, 2, TEXT("女"));

m\_age\_list.SetItemText(i, 3, pRecordInfo->UserSfz);

m\_age\_list.SetItemText(i, 4, pRecordInfo->UserPhone);

if (pRecordInfo->UserStatus == 0)

m\_age\_list.SetItemText(i, 5, TEXT("未出行"));

else

m\_age\_list.SetItemText(i, 5, TEXT("出行"));

m\_age\_list.SetItemText(i, 6, pRecordInfo->UserAddress);

pRecordInfo = pRecordInfo->Next;

delete pDelete;

pDelete = pRecordInfo;

}

}

void CUserQuery::WriteList(\_RecordStatusInfo \*pRecordInfo)

{

m\_UserStatusList.DeleteAllItems();

\_RecordStatusInfo \*pDelete = NULL;

pDelete = pRecordInfo;

for (int i = 0; pRecordInfo != NULL; i++)

{

m\_UserStatusList.InsertItem(i, NULL);

m\_UserStatusList.SetItemText(i, 0, pRecordInfo->UserName);

m\_UserStatusList.SetItemText(i, 1, pRecordInfo->BeginTime);

m\_UserStatusList.SetItemText(i, 2, pRecordInfo->EndTime);

m\_UserStatusList.SetItemText(i, 3, pRecordInfo->BeginAddr);

m\_UserStatusList.SetItemText(i, 4, pRecordInfo->EndAddr);

if (pRecordInfo->UserTool == 0)

m\_UserStatusList.SetItemText(i, 5, TEXT("飞机"));

else if (pRecordInfo->UserTool == 1)

m\_UserStatusList.SetItemText(i, 5, TEXT("火车"));

else if (pRecordInfo->UserTool == 2)

m\_UserStatusList.SetItemText(i, 5, TEXT("汽车"));

else if (pRecordInfo->UserTool == 3)

m\_UserStatusList.SetItemText(i, 5, TEXT("自驾"));

else if (pRecordInfo->UserTool == 4)

m\_UserStatusList.SetItemText(i, 5, TEXT("步行"));

else

m\_UserStatusList.SetItemText(i, 5, TEXT("其他"));

m\_UserStatusList.SetItemText(i, 6, pRecordInfo->ToolInfo);

if (pRecordInfo->Hasisolace == 0)

m\_UserStatusList.SetItemText(i, 7, TEXT("无"));

else

m\_UserStatusList.SetItemText(i, 7, TEXT("有"));

if (pRecordInfo->Hasinfect == 0)

m\_UserStatusList.SetItemText(i, 8, TEXT("无"));

else

m\_UserStatusList.SetItemText(i, 8, TEXT("有"));

pRecordInfo = pRecordInfo->Next;

delete pDelete;

pDelete = pRecordInfo;

}

}

void CUserQuery::OnSelchangeCombo1()

{

// TODO: 在此添加控件通知处理程序代码

m\_age\_list.ShowWindow(SW\_HIDE);

m\_UserList.ShowWindow(SW\_SHOW);

m\_UserStatusList.ShowWindow(SW\_SHOW);

int nSelect = m\_select\_type.GetCurSel();

if (nSelect==0)

{

m\_edit\_user\_name.ShowWindow(SW\_SHOW);

m\_tool\_select.ShowWindow(SW\_HIDE);

}

else if (nSelect == 1)

{

m\_edit\_user\_name.ShowWindow(SW\_SHOW);

m\_tool\_select.ShowWindow(SW\_HIDE);

}

else if (nSelect == 2)

{

m\_edit\_user\_name.ShowWindow(SW\_HIDE);

m\_tool\_select.ShowWindow(SW\_SHOW);

}

else if (nSelect == 5)

{

m\_age\_list.ShowWindow(SW\_SHOW);

m\_UserList.ShowWindow(SW\_HIDE);

m\_UserStatusList.ShowWindow(SW\_HIDE);

m\_edit\_user\_name.ShowWindow(SW\_HIDE);

m\_tool\_select.ShowWindow(SW\_HIDE);

}

else

{

m\_edit\_user\_name.ShowWindow(SW\_HIDE);

m\_tool\_select.ShowWindow(SW\_HIDE);

}

}

**🡪用户信息的统计功能的实现：**

CUserTotal::~CUserTotal()

{

}

void CUserTotal::DoDataExchange(CDataExchange\* pDX)

{

CDialogEx::DoDataExchange(pDX);

DDX\_Control(pDX, IDC\_COMBO1, m\_type\_select);

DDX\_Control(pDX, IDC\_USER\_LIST, m\_UserList);

DDX\_Control(pDX, IDC\_USER\_STATE\_LIST, m\_UserStatusList);

}

BEGIN\_MESSAGE\_MAP(CUserTotal, CDialogEx)

ON\_BN\_CLICKED(IDC\_QUERY, &CUserTotal::OnBnClickedQuery)

ON\_CBN\_SELCHANGE(IDC\_COMBO1, &CUserTotal::OnSelchangeCombo1)

END\_MESSAGE\_MAP()

// CUserTotal 消息处理程序

BOOL CUserTotal::OnInitDialog()

{

CDialogEx::OnInitDialog();

// TODO: 在此添加额外的初始化

m\_UserList.InsertColumn(0, TEXT("姓名"), LVCFMT\_LEFT, 80);

m\_UserList.InsertColumn(1, TEXT("年龄"), LVCFMT\_LEFT, 40);

m\_UserList.InsertColumn(2, TEXT("性别"), LVCFMT\_LEFT, 40);

m\_UserList.InsertColumn(3, TEXT("身份证"), LVCFMT\_LEFT, 160);

m\_UserList.InsertColumn(4, TEXT("电话"), LVCFMT\_LEFT, 90);

m\_UserList.InsertColumn(5, TEXT("状态"), LVCFMT\_LEFT, 50);

m\_UserList.InsertColumn(6, TEXT("地址"), LVCFMT\_LEFT, 200);

m\_UserList.SetExtendedStyle(LVS\_EX\_GRIDLINES);

m\_UserList.InsertItem(0, NULL);

m\_UserStatusList.InsertColumn(0, TEXT("姓名"), LVCFMT\_LEFT, 80);

m\_UserStatusList.InsertColumn(1, TEXT("开始时间"), LVCFMT\_LEFT, 80);

m\_UserStatusList.InsertColumn(2, TEXT("结束时间"), LVCFMT\_LEFT, 80);

m\_UserStatusList.InsertColumn(3, TEXT("出行地址"), LVCFMT\_LEFT, 100);

m\_UserStatusList.InsertColumn(4, TEXT("结束地址"), LVCFMT\_LEFT, 100);

m\_UserStatusList.InsertColumn(5, TEXT("出行工具"), LVCFMT\_LEFT, 60);

m\_UserStatusList.InsertColumn(6, TEXT("工具信息"), LVCFMT\_LEFT, 120);

m\_UserStatusList.InsertColumn(7, TEXT("隔离"), LVCFMT\_LEFT, 40);

m\_UserStatusList.InsertColumn(8, TEXT("患病"), LVCFMT\_LEFT, 40);

m\_UserStatusList.SetExtendedStyle(LVS\_EX\_GRIDLINES);

m\_UserStatusList.InsertItem(0, NULL);

m\_type\_select.AddString(TEXT("人员信息"));

m\_type\_select.AddString(TEXT("人员出行信息"));

m\_type\_select.SetCurSel(0);

m\_UserList.ShowWindow(SW\_SHOW);

m\_UserStatusList.ShowWindow(SW\_HIDE);

return TRUE; // return TRUE unless you set the focus to a control

// 异常: OCX 属性页应返回 FALSE

}

void CUserTotal::OnBnClickedQuery()

{

// TODO: 在此添加控件通知处理程序代码

UpdateData(TRUE);

int nSelect = m\_type\_select.GetCurSel();

if (nSelect == 0)

{

CString strSqlText;

strSqlText.Format(TEXT("select \* from userinfo"));

//////////////////////////////////创建SQL语句执行类实例

AdoWorker \*ado = NULL;

\_RecordUserInfo \*pUserNameInfo = NULL;

ado = new AdoWorker();

pUserNameInfo = (\_RecordUserInfo \*)ado->ExecuteSql(strSqlText, 4);

if (pUserNameInfo == NULL)

{

delete pUserNameInfo;

delete ado;

return;

}

else//存在这样的账号，在此设定表格中第二行的姓名

{

WriteAllUserList(pUserNameInfo);

delete ado;

}

}

else if (nSelect == 1)

{

CString strSqlText;

strSqlText.Format(TEXT("select \* from Recordinfo"));

//////////////////////////////////创建SQL语句执行类实例

AdoWorker \*ado = NULL;

\_RecordStatusInfo \*pUserNameInfo = NULL;

ado = new AdoWorker();

pUserNameInfo = (\_RecordStatusInfo \*)ado->ExecuteSql(strSqlText, 3);

if (pUserNameInfo == NULL)

{

delete pUserNameInfo;

delete ado;

return;

}

else//存在这样的账号，在此设定表格中第二行的姓名

{

WriteAllUserStateList(pUserNameInfo);

delete ado;

}

}

}

void CUserTotal::WriteAllUserList(\_RecordUserInfo \*pRecordInfo)

{

m\_UserList.DeleteAllItems();

\_RecordUserInfo \*pDelete = NULL;

pDelete = pRecordInfo;

for (int i = 0; pRecordInfo != NULL; i++)

{

m\_UserList.InsertItem(i, NULL);//新插入一行

//AfxMessageBox(m\_UserAddDlg.m\_user\_name);

m\_UserList.SetItemText(i, 0, pRecordInfo->UserName);

CString strAge = \_T("");

strAge.Format(TEXT("%d"), pRecordInfo->UserAge);

m\_UserList.SetItemText(i, 1, strAge);

if (pRecordInfo->UserSex == 0)

m\_UserList.SetItemText(i, 2, TEXT("男"));

else

m\_UserList.SetItemText(i, 2, TEXT("女"));

m\_UserList.SetItemText(i, 3, pRecordInfo->UserSfz);

m\_UserList.SetItemText(i, 4, pRecordInfo->UserPhone);

if (pRecordInfo->UserStatus == 0)

m\_UserList.SetItemText(i, 5, TEXT("未出行"));

else

m\_UserList.SetItemText(i, 5, TEXT("出行"));

m\_UserList.SetItemText(i, 6, pRecordInfo->UserAddress);

pRecordInfo = pRecordInfo->Next;

delete pDelete;

pDelete = pRecordInfo;

}

}

void CUserTotal::WriteAllUserStateList(\_RecordStatusInfo \*pRecordInfo)

{

m\_UserStatusList.DeleteAllItems();

\_RecordStatusInfo \*pDelete = NULL;

pDelete = pRecordInfo;

for (int i = 0; pRecordInfo != NULL; i++)

{

m\_UserStatusList.InsertItem(i, NULL);

m\_UserStatusList.SetItemText(i, 0, pRecordInfo->UserName);

m\_UserStatusList.SetItemText(i, 1, pRecordInfo->BeginTime);

m\_UserStatusList.SetItemText(i, 2, pRecordInfo->EndTime);

m\_UserStatusList.SetItemText(i, 3, pRecordInfo->BeginAddr);

m\_UserStatusList.SetItemText(i, 4, pRecordInfo->EndAddr);

if (pRecordInfo->UserTool == 0)

m\_UserStatusList.SetItemText(i, 5, TEXT("飞机"));

else if (pRecordInfo->UserTool == 1)

m\_UserStatusList.SetItemText(i, 5, TEXT("火车"));

else if (pRecordInfo->UserTool == 2)

m\_UserStatusList.SetItemText(i, 5, TEXT("汽车"));

else if (pRecordInfo->UserTool == 3)

m\_UserStatusList.SetItemText(i, 5, TEXT("自驾"));

else if (pRecordInfo->UserTool == 4)

m\_UserStatusList.SetItemText(i, 5, TEXT("步行"));

else

m\_UserStatusList.SetItemText(i, 5, TEXT("其他"));

m\_UserStatusList.SetItemText(i, 6, pRecordInfo->ToolInfo);

if (pRecordInfo->Hasisolace == 0)

m\_UserStatusList.SetItemText(i, 7, TEXT("无"));

else

m\_UserStatusList.SetItemText(i, 7, TEXT("有"));

if (pRecordInfo->Hasinfect == 0)

m\_UserStatusList.SetItemText(i, 8, TEXT("无"));

else

m\_UserStatusList.SetItemText(i, 8, TEXT("有"));

pRecordInfo = pRecordInfo->Next;

delete pDelete;

pDelete = pRecordInfo;

}

}

void CUserTotal::OnSelchangeCombo1()

{

// TODO: 在此添加控件通知处理程序代码

int nSelect = m\_type\_select.GetCurSel();

if (nSelect == 0)

{

m\_UserList.ShowWindow(SW\_SHOW);

m\_UserStatusList.ShowWindow(SW\_HIDE);

}

else if (nSelect == 1)

{

m\_UserList.ShowWindow(SW\_HIDE);

m\_UserStatusList.ShowWindow(SW\_SHOW);

}

}

**🡪二维码编码功能的实现（核心部分）：**

#include "stdafx.h"

#include"ProductQRCode.h"

#include "QR\_Encode.h"

#include<string>

#include "QRDecode\\QRcodeImage.h"

#include "strcoding.h"

// nLevel　级别

//nVersion 版本

//bAutoExtent　是否拉伸

//pScaleFactor　缩放比例

//pCodeStr 内容

//pFilenameStr　保存的文件名

bool ProcessEncode(int nLevel, int nVersion,

BOOL bAutoExtent,

int nMaskingNo,

int pScaleFactor,

const char \* pCodeStr,

const char \* pFilenameStr)

{

if(pScaleFactor < 1)

return false;

int i, j;

CQR\_Encode\* pQR\_Encode = new CQR\_Encode;

if (pQR\_Encode->EncodeData(nLevel, nVersion, bAutoExtent, nMaskingNo, pCodeStr))

{

// 生成原始的编码文件

int originalSize = pQR\_Encode->m\_nSymbleSize + (QR\_MARGIN \* 2);

CBitmap\* saveBmp = new CBitmap;

saveBmp->CreateBitmap(originalSize, originalSize, 1, 1, NULL);

CDC \*saveDC = new CDC;

saveDC->CreateCompatibleDC(NULL);

CBitmap\* pOldBitmap = saveDC->SelectObject(saveBmp);

saveDC->PatBlt(0, 0, originalSize, originalSize, WHITENESS);

// 根据结果赋值

for (i = 0; i < pQR\_Encode->m\_nSymbleSize; ++i)

{

for (j = 0; j < pQR\_Encode->m\_nSymbleSize; ++j)

{

if (pQR\_Encode->m\_byModuleData[i][j])

saveDC->SetPixel(i+ QR\_MARGIN, j+ QR\_MARGIN, RGB(0, 0, 0));

}

}

// 缩放10倍

int scaleSize = originalSize \* pScaleFactor;

int nBmpSize = ((scaleSize + 31) / 32) \* 32 / 8;

nBmpSize \*= scaleSize;

HANDLE hFile = CreateFileA(pFilenameStr, GENERIC\_READ | GENERIC\_WRITE, 0, NULL,

CREATE\_ALWAYS, FILE\_ATTRIBUTE\_NORMAL, NULL);

if (hFile == INVALID\_HANDLE\_VALUE)

return FALSE;

HANDLE hFileMapping = CreateFileMapping(hFile, NULL, PAGE\_READWRITE | SEC\_COMMIT, 0,

sizeof(BITMAPFILEHEADER) + sizeof(BITMAPINFOHEADER) + sizeof(RGBQUAD) \* 2 + nBmpSize, NULL);

LPBYTE lpbyMapView = (LPBYTE)MapViewOfFile(hFileMapping, FILE\_MAP\_WRITE, 0, 0, 0);

ZeroMemory(lpbyMapView, sizeof(BITMAPFILEHEADER) + sizeof(BITMAPINFOHEADER) + sizeof(RGBQUAD) \* 2 + nBmpSize);

LPBITMAPFILEHEADER pbfh = (LPBITMAPFILEHEADER)lpbyMapView;

LPBITMAPINFO pbmi = (LPBITMAPINFO)(lpbyMapView + sizeof(BITMAPFILEHEADER));

LPVOID pbdt = lpbyMapView + sizeof(BITMAPFILEHEADER) + sizeof(BITMAPINFOHEADER) + sizeof(RGBQUAD) \* 2;

pbfh->bfType = (WORD) (('M' << 8) | 'B'); // "BM"

pbfh->bfSize = sizeof(BITMAPFILEHEADER) + sizeof(BITMAPINFOHEADER) + sizeof(RGBQUAD) \* 2 + nBmpSize;

pbfh->bfReserved1 = 0;

pbfh->bfReserved2 = 0;

pbfh->bfOffBits = sizeof(BITMAPFILEHEADER) + sizeof(BITMAPINFOHEADER) + sizeof(RGBQUAD) \* 2;

pbmi->bmiHeader.biSize = sizeof(BITMAPINFOHEADER);

pbmi->bmiHeader.biWidth = scaleSize;

pbmi->bmiHeader.biHeight = scaleSize;

pbmi->bmiHeader.biPlanes = 1;

pbmi->bmiHeader.biBitCount = 1;

pbmi->bmiHeader.biCompression = BI\_RGB;

pbmi->bmiHeader.biSizeImage = nBmpSize;

pbmi->bmiHeader.biXPelsPerMeter = 3780;

pbmi->bmiHeader.biYPelsPerMeter = 3780;

pbmi->bmiHeader.biClrUsed = 0;

pbmi->bmiHeader.biClrImportant = 0;

CDC\* pWorkDC = new CDC;

pWorkDC->CreateCompatibleDC(NULL);

CBitmap\* pWorkBitmap = new CBitmap;

pWorkBitmap->CreateBitmap(scaleSize, scaleSize, 1, 1, NULL);

pOldBitmap = pWorkDC->SelectObject(pWorkBitmap);

pWorkDC->StretchBlt(0, 0, scaleSize, scaleSize, saveDC, 0, 0, originalSize, originalSize, SRCCOPY);

GetDIBits(pWorkDC->m\_hDC, (HBITMAP)\*pWorkBitmap, 0, scaleSize, pbdt, pbmi, DIB\_RGB\_COLORS);

pWorkDC->SelectObject(pOldBitmap);

delete pWorkBitmap;

delete pWorkDC;

UnmapViewOfFile(lpbyMapView);

CloseHandle(hFileMapping);

CloseHandle(hFile);

delete saveDC;

delete saveBmp;

}

else

return false;

delete pQR\_Encode;

pQR\_Encode = NULL;

return true;

}

//生成二维码

// scale 放大比例

// strIn 要生成的二维码的内容

// strFilePathName 文件图片保存的全路径名

// 返回 true成功， false失败

bool ProductQRCodeA(int scale,char \* strIn,char \* strFilePathName)

{

bool bBack = false;

if(!strIn || !strFilePathName)

return false;

int nLen = strlen(strIn);

if(nLen<=0)

return false;

// 编码

bBack = ProcessEncode(0,1,1,-1,scale,strIn,strFilePathName);

return bBack;

}

bool ProductQRCodeW(int scale,wchar\_t \* strIn,wchar\_t \* strFilePathName)

{

bool bBack = false;

if(!strIn || !strFilePathName)

return false;

int nLen = wcslen(strIn);

if(nLen<=0)

return false;

USES\_CONVERSION;

std::string strInTemp = W2A(strIn);

std::string strFilePathNameTemp = W2A(strFilePathName);

// 编码

bBack = ProcessEncode(0,1,1,-1,scale,strInTemp.c\_str(),strFilePathNameTemp.c\_str());

return bBack;

}

bool DeQRCodeA(char \* strFilePathName,char \* strQRContent)

{

bool bBack = false;

BYTE m\_CodeData[MAX\_MODULESIZE][MAX\_MODULESIZE]; // 存储条形码位数据

int m\_nCodeSize;//编码区域大小

int m\_nMoudleSize;//模块大小

int m\_nVersion;//版本号

int m\_nLevel;//纠错等级

CString filePathName, tempStr;

filePathName = strFilePathName;

QRcodeImage \*QR\_image = new QRcodeImage(filePathName);

QR\_image->GetImageData();

m\_nVersion = QR\_image->m\_finderPattern->m\_version;

m\_nCodeSize = m\_nVersion \* 4 +17;

for(int j=0;j<m\_nCodeSize;j++)

for(int k=0;k<m\_nCodeSize;k++)

m\_CodeData[j][k]=QR\_image->bitMatrix[j][k];

ContentDecoder \*QR\_Decode = new ContentDecoder;

if(QR\_Decode->DecodeData(m\_nCodeSize, m\_nVersion, m\_CodeData))

{

// m\_nPendingListCtrl.SetItemText(i, 2, \_T("处理完成"));

// m\_nPendingListCtrl.SetItemText(i, 3, QR\_Decode->m\_strData);

// USES\_CONVERSION;

// sprintf(strQRContent,"%s",T2A(QR\_Decode->m\_strData.GetBuffer(0)));

sprintf(strQRContent,"%s",QR\_Decode->m\_strData.c\_str());

switch(QR\_Decode->m\_nLevel)

{

case QR\_LEVEL\_L:tempStr="L(%7)";break;

case QR\_LEVEL\_M:tempStr="M(%15)";break;

case QR\_LEVEL\_Q:tempStr="Q(%25)";break;

case QR\_LEVEL\_H:tempStr="H(%30)";break;

}

// m\_nPendingListCtrl.SetItemText(i, 4, tempStr);

// tempStr.Format("%d", QR\_Decode->m\_nVersion);

// m\_nPendingListCtrl.SetItemText(i, 5, tempStr);

// tempStr.Format("%d", QR\_Decode->m\_nMaskingNo);

// m\_nPendingListCtrl.SetItemText(i, 6, tempStr);

bBack=true;

}

else

{

bBack=false;

}

delete QR\_Decode;

delete QR\_image;

return bBack;

}

bool DeQRCodeW(wchar\_t \* strFilePathName,wchar\_t \* strQRContent)

{

bool bBack = false;

BYTE m\_CodeData[MAX\_MODULESIZE][MAX\_MODULESIZE]; // 存储条形码位数据

int m\_nCodeSize;//编码区域大小

int m\_nMoudleSize;//模块大小

int m\_nVersion;//版本号

int m\_nLevel;//纠错等级

CString filePathName, tempStr;

filePathName = strFilePathName;

QRcodeImage \*QR\_image = new QRcodeImage(filePathName);

QR\_image->GetImageData();

m\_nVersion = QR\_image->m\_finderPattern->m\_version;

m\_nCodeSize = m\_nVersion \* 4 +17;

for(int j=0;j<m\_nCodeSize;j++)

for(int k=0;k<m\_nCodeSize;k++)

m\_CodeData[j][k]=QR\_image->bitMatrix[j][k];

ContentDecoder \*QR\_Decode = new ContentDecoder;

if(QR\_Decode->DecodeData(m\_nCodeSize, m\_nVersion, m\_CodeData))

{

// m\_nPendingListCtrl.SetItemText(i, 2, \_T("处理完成"));

// m\_nPendingListCtrl.SetItemText(i, 3, QR\_Decode->m\_strData);

USES\_CONVERSION;

strCoding clsCoding;

// tempStr = QR\_Decode->m\_strData;

// std::string strTmp = T2A(tempStr.GetBuffer(0));

std::string strTmp = QR\_Decode->m\_strData;

// strTmp = clsCoding.UTF8GB2312 (strTmp);

#ifdef UNICODE

::wsprintf(strQRContent,\_T("%s"),A2T(strTmp.c\_str()));

#endif

switch(QR\_Decode->m\_nLevel)

{

case QR\_LEVEL\_L:tempStr="L(%7)";break;

case QR\_LEVEL\_M:tempStr="M(%15)";break;

case QR\_LEVEL\_Q:tempStr="Q(%25)";break;

case QR\_LEVEL\_H:tempStr="H(%30)";break;

}

// m\_nPendingListCtrl.SetItemText(i, 4, tempStr);

// tempStr.Format("%d", QR\_Decode->m\_nVersion);

// m\_nPendingListCtrl.SetItemText(i, 5, tempStr);

// tempStr.Format("%d", QR\_Decode->m\_nMaskingNo);

// m\_nPendingListCtrl.SetItemText(i, 6, tempStr);

bBack=true;

}

else

{

bBack=false;

}

delete QR\_Decode;

delete QR\_image;

return bBack;

}

**🡪二维码解码功能的实现（核心部分）：**

#include "stdafx.h"

#include "RsDecode.h"

#ifdef \_DEBUG

#undef THIS\_FILE

static char THIS\_FILE[]=\_\_FILE\_\_;

#define new DEBUG\_NEW

#endif

/\*

数组长度

sg0===npar

sg1===npar

sigma===npar/2+2

omega===npar/2+1

syn===npar

\*/

RsDecode::RsDecode()

{

m\_galois = Galois();

}

RsDecode::~RsDecode()

{

}

RsDecode::RsDecode(int npar)

{

m\_galois = Galois();

this->m\_npar = npar;

}

int RsDecode::calcSigmaMBM(int \* sigma, int \* omega, int \* syn)

{

int \* sg0 = new int[m\_npar];

int \* sg1 = new int[m\_npar];

int i;

for(i=0;i<m\_npar;i++)

{

sg0[i]=0;sg1[i]=0;

}

sg0[1] = 1;

sg1[0] = 1;

int jisu0 = 1;

int jisu1 = 0;

int m = -1;

for(int n = 0; n < m\_npar; n++)

{

int d = syn[n];

for(int i = 1; i <= jisu1; i++)

{

d ^= m\_galois.mul(sg1[i], syn[n - i]);

}

if(d != 0)

{

int logd = m\_galois.toLog(d);

int \* wk = new int[m\_npar];

for(i=0;i<m\_npar;i++)

wk[i]=0;

for(int i = 0; i <= n; i++)

{

wk[i] = sg1[i] ^ m\_galois.mulExp(sg0[i], logd);

}

int js = n - m;

if(js > jisu1)

{

m = n - jisu1;

jisu1 = js;

if(jisu1 > m\_npar / 2)

{

delete [] wk;

delete [] sg0;

delete [] sg1;

return -1;

}

for(int i = 0; i <= jisu0; i++)

{

sg0[i] = m\_galois.divExp(sg1[i], logd);

}

jisu0 = jisu1;

}

// sg1 = wk; //gy 改成下面的

for(i=0;i<m\_npar;i++) //gy

{ //gy

sg1[i]=wk[i]; //gy

} //gy

delete [] wk;

}

int cnt=m\_npar-1<jisu0 ? m\_npar-1 : jisu0;

for(i=cnt-1;i>=0;i--)

sg0[i+1]=sg0[i];

sg0[0] = 0;

jisu0++;

}

m\_galois.mulPoly(omega, sg1, syn, m\_npar/2+1, m\_npar, m\_npar);

int cnt=m\_npar<m\_npar/2+2 ? m\_npar : m\_npar/2+2;

for(i=0;i<cnt;i++)

sigma[i]=sg1[i];

delete [] sg0;

delete [] sg1;

return jisu1;

}

int RsDecode::chienSearch(int \* pos, int n, int jisu, int \* sigma)

{

int last = sigma[1];

if(jisu == 1) {

if(m\_galois.toLog(last) >= n) {

return RS\_CORRECT\_ERROR;

}

pos[0] = last;

return 0;

}

int posIdx = jisu - 1;

for(int i = 0; i < n; i++) {

int z = 255 - i;

int wk = 1;

for(int j = 1; j <= jisu; j++) {

wk ^= m\_galois.mulExp(sigma[j], (z \* j) % 255);

}

if(wk == 0) {

int pv = m\_galois.toExp(i);

last ^= pv;

pos[posIdx--] = pv;

if(posIdx == 0) {

if(m\_galois.toLog(last) >= n) {

return RS\_CORRECT\_ERROR;

}

pos[0] = last;

return 0;

}

}

}

return RS\_CORRECT\_ERROR;

}

void RsDecode::doForney(BYTE \* data, int length, int jisu, int \* pos, int \* sigma, int \* omega)

{

int i,j;

for(i = 0; i < jisu; i++)

{

int ps = pos[i];

int zlog = 255 - m\_galois.toLog(ps); int ov = omega[0];

for(j = 1; j < jisu; j++)

{

ov ^= m\_galois.mulExp(omega[j], (zlog \* j) % 255); // ov += ωi \* z^j

}

int dv = sigma[1];

for(j = 2; j < jisu; j += 2)

{

dv ^= m\_galois.mulExp(sigma[j + 1], (zlog \* j) % 255); // dv += σ<j+1> \* z^j

}

data[m\_galois.toPos(length, ps)] ^= m\_galois.mul(ps, m\_galois.div(ov, dv));

}

}

int RsDecode::decode(BYTE \* data, int length, boolean noCorrect)

{

if(length < m\_npar || length > 255) {

return RS\_PERM\_ERROR;

}

int i;

int \* syn = new int[m\_npar];

for(i=0;i<m\_npar;i++)syn[i]=0;

if(m\_galois.calcSyndrome(data, length, syn, m\_npar)) {

if(syn)

{

delete [] syn;

syn=NULL;

}

return 0;

}

int \* sigma = new int[m\_npar/2+2];

for(i=0;i<m\_npar/2+2;i++)sigma[i]=0;

int \* omega = new int[m\_npar/2+1];

for(i=0;i<m\_npar/2+1;i++)omega[i]=0;

int jisu = calcSigmaMBM(sigma, omega, syn);

if(jisu <= 0)

{

if(syn)

{

delete [] syn;

syn=NULL;

}

if(sigma)

{

delete [] sigma;

sigma=NULL;

}

if(omega)

{

delete [] omega;

omega=NULL;

}

return RS\_CORRECT\_ERROR;

}

int \* pos = new int[jisu];

for(i=0;i<jisu;i++)pos[i]=0;

int r = chienSearch(pos, length, jisu, sigma);

if(r < 0) {

if(syn)

{

delete [] syn;

syn=NULL;

}

if(sigma)

{

delete [] sigma;

sigma=NULL;

}

if(omega)

{

delete [] omega;

omega=NULL;

}

return r;

}

if(!noCorrect) {

doForney(data, length, jisu, pos, sigma, omega);

}

if(syn)

{

delete [] syn;

syn=NULL;

}

if(sigma)

{

delete [] sigma;

sigma=NULL;

}

if(omega)

{

delete [] omega;

omega=NULL;

}

return jisu;

}

int RsDecode::decode(BYTE \* data, int length)

{

return decode(data, length, false);

}

1. **测试（Test）**

功能测试和界面展示

登录界面🡪



主菜单以及人员管理系统界面🡪



信息查询系统界面🡪



信息统计系统界面🡪



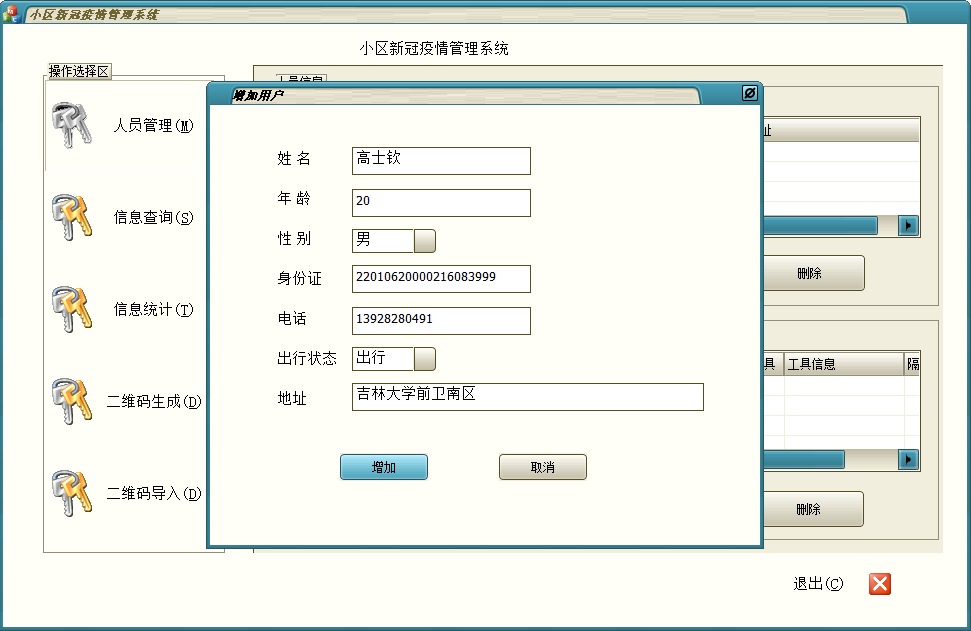
二维码的生成功能🡪



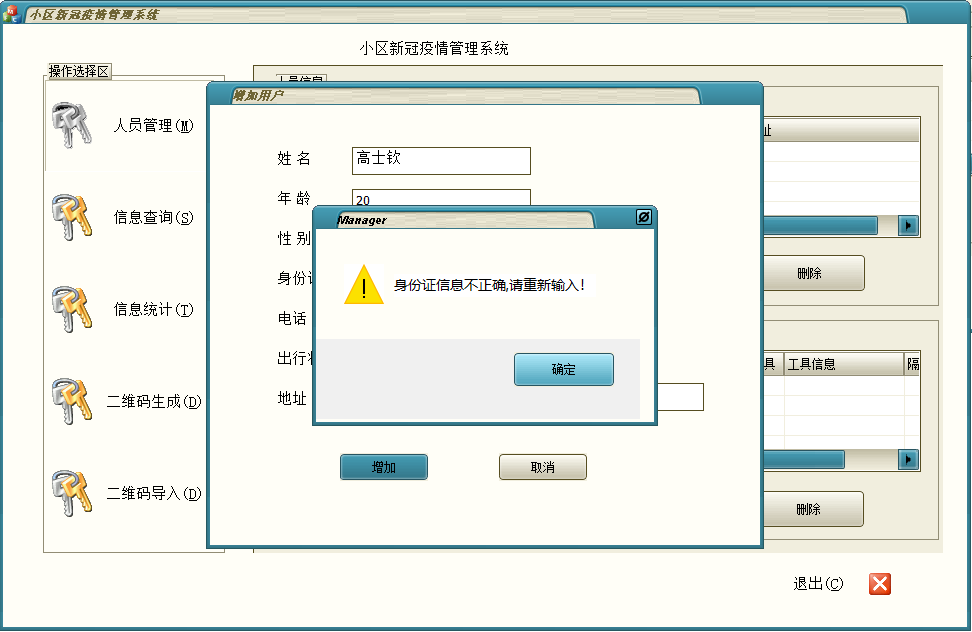
二维码的导入功能🡪



功能测试部分：



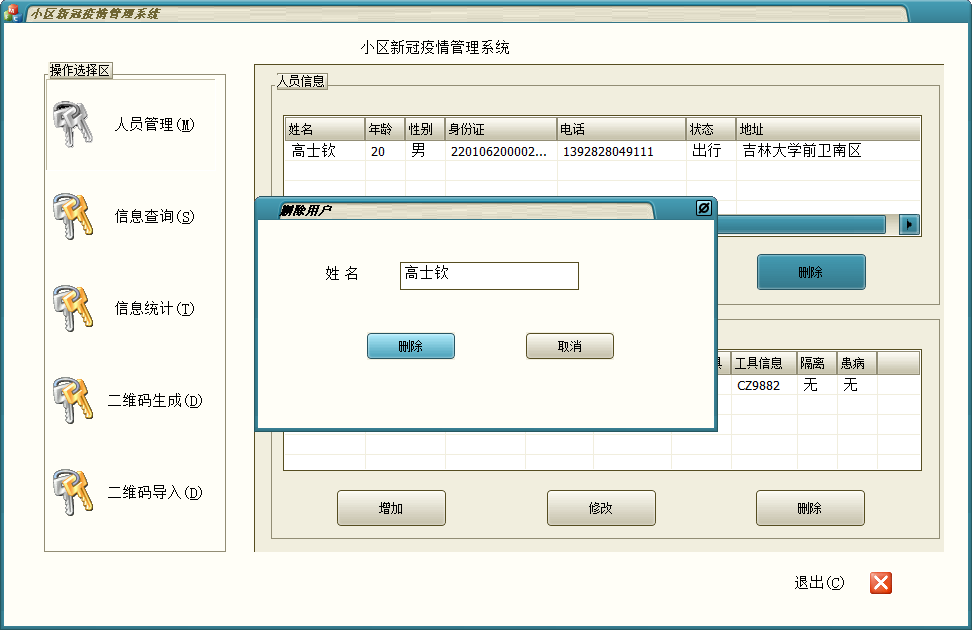
如果填入身份证信息不正确，系统会进行相应报错如下图🡪



人员管理系统正确填入信息后展示🡪



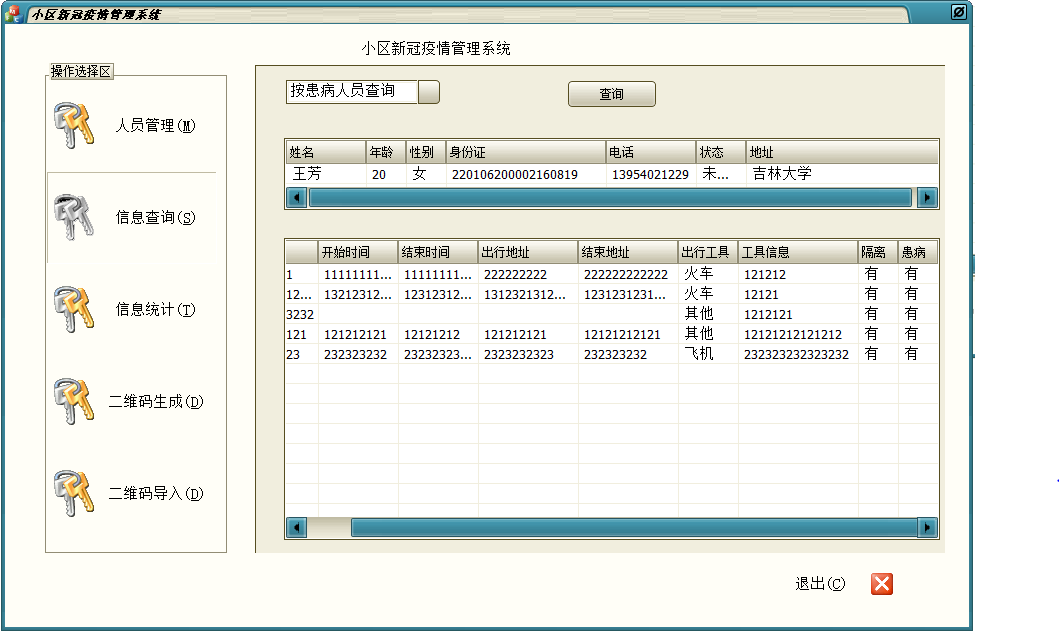
人员信息的删除功能展示🡪



人员信息查询功能展示🡪









人员信息统计系统功能展示🡪

