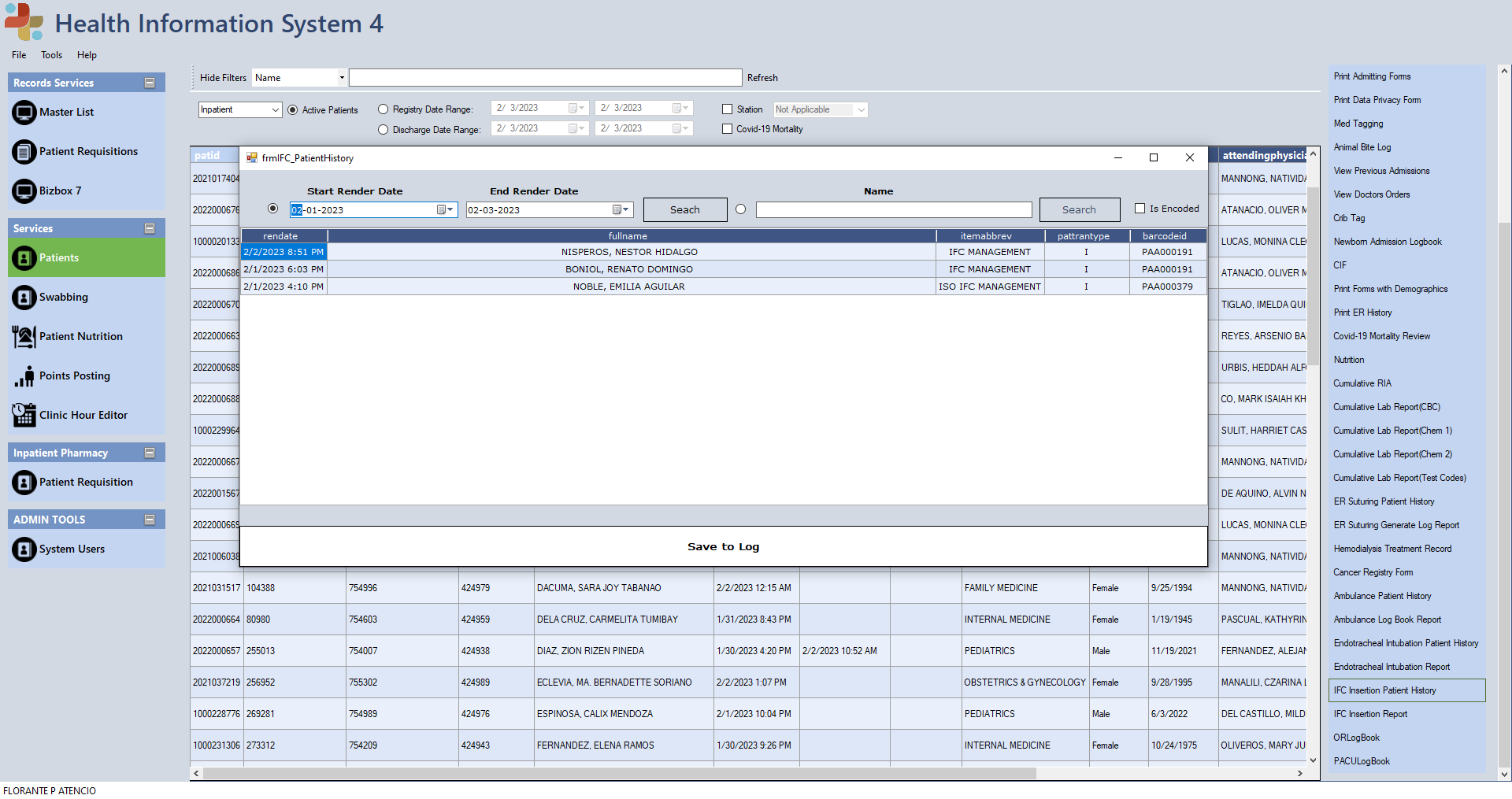
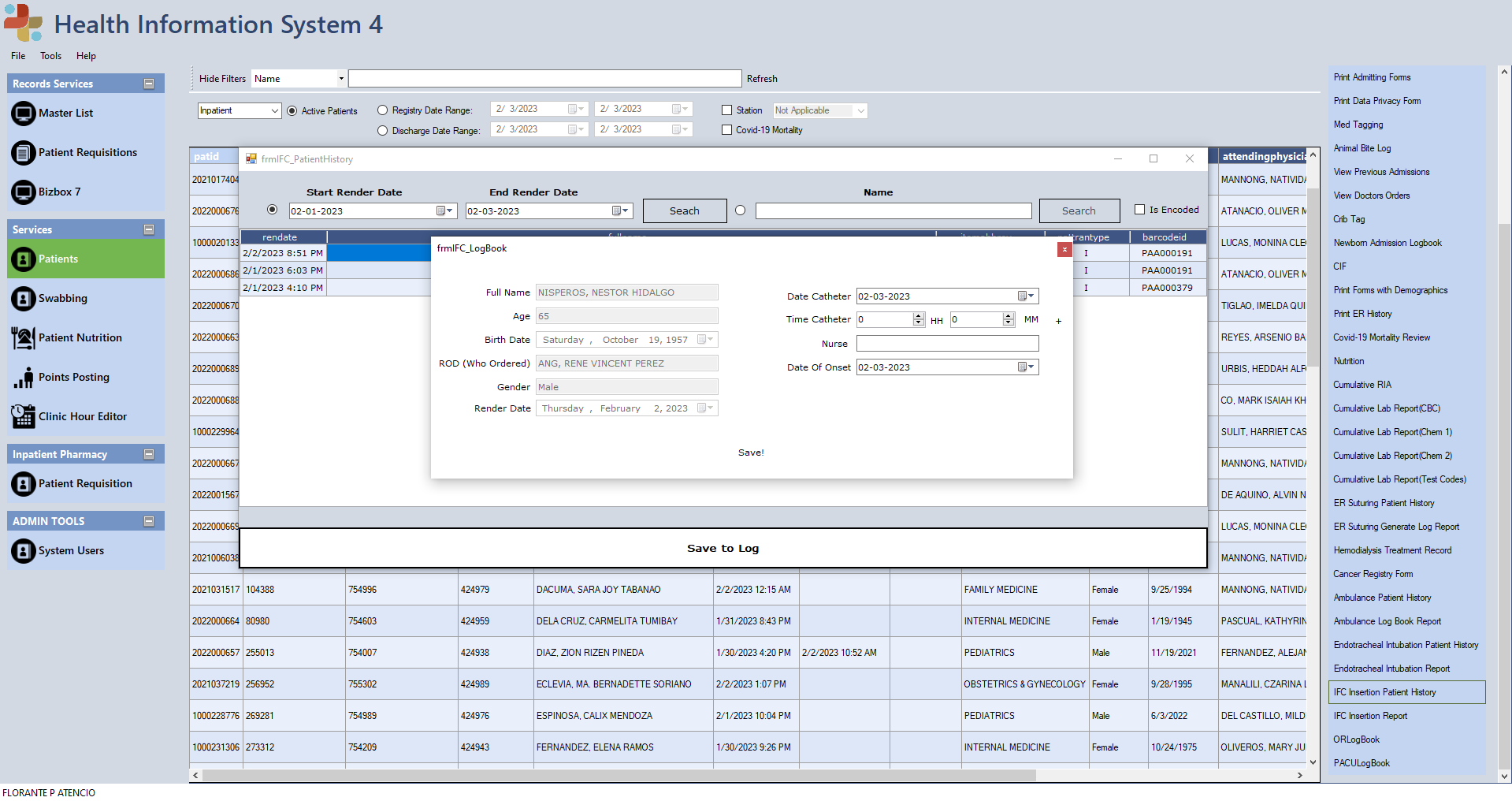
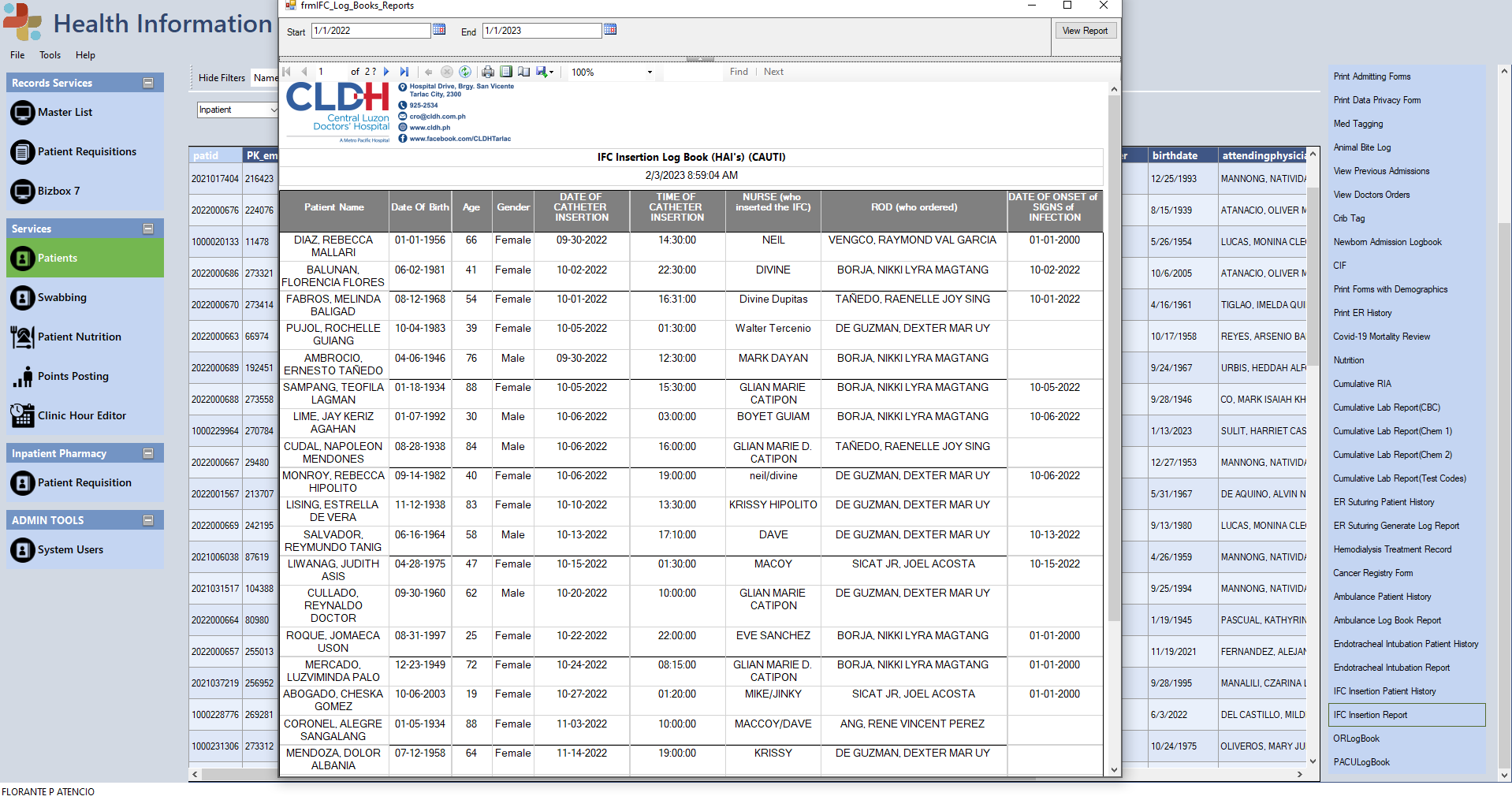
IFC insertion Module in HIS 4

In Services tag under Patients module IFC Insertion patient history click

IFC insertion patient history Window will pop up

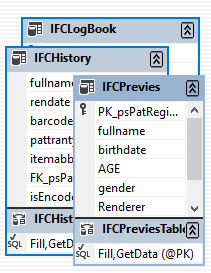
In order to save into log First select a patient inside the data table then click the save to log button. It will proceed to encoding date and time. It will required you to fill all the necessary field and click the save button. Then you can repeat the process.

To Access the Report Viewer from Patient module under Services Tab in action tab find the IFC Insertion Report click. Report server 10.4.1.3/reportserver/JUN/frmEndotrachealReport.rdl

**ADO.NET (Table Adapter and DataBindsource)**

Accessing 10.4.1.3/sqlcldh02 (HealthInfoDB) and 10.4.1.3/sqlcldh01 (LIVE\_CLDH)

**ADO.NET (Table Adapter and Data Bind Source)**



**Demo Graphics**

**IFCPrevies**

**IFCHistory**

**CRUD**

**IFCLogBook**

**Module Source Code**

**SQL SERVER:**

USE [HealthInfoDB]

GO

/\*\*\*\*\*\* Object: Table [dbo].[IFCLogBook] Script Date: 2/3/2023 10:49:01 AM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[IFCLogBook](

[PK\_IFCLogBook] [int] IDENTITY(1,1) NOT NULL,

[FK\_psPatRegisters] [int] NULL,

[dateCatheter] [date] NULL,

[timeCatheter] [time](7) NULL,

[nurseAssist] [nvarchar](max) NULL,

[dateofOnset] [date] NULL,

[isEncode] [bit] NULL,

PRIMARY KEY CLUSTERED

(

[PK\_IFCLogBook] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

**Visual Studio 2019 C#**

frmIFC\_PatientHistory: using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace HealthInformationSystem4.Forms

{

public partial class frmIFC\_PatientHistory : Form

{

string isEncoded = "(isEncode is null OR isEncode = 0)";

public frmIFC\_PatientHistory()

{

InitializeComponent();

}

private void frmIFC\_PatientHistory\_Load(object sender, EventArgs e)

{

radioButton1.Checked = true;

int MM = int.Parse(DateTime.Now.ToString("MM"));

int yyyy = int.Parse(DateTime.Now.ToString("yyyy"));

int dd = int.Parse(DateTime.Now.ToString("dd"));

dateTimePicker1.Value = new DateTime(yyyy, MM, 01);

dateTimePicker2.Value = new DateTime(yyyy, MM, dd);

this.iFCHistoryBindingSource.Filter = "rendate >= '" + dateTimePicker1.Value.ToString("MM-dd-yyyy 00:00:00") + "' and rendate <= ' " + dateTimePicker2.Value.ToString("MM-dd-yyyy 23:59:59") + "' AND " + isEncoded;

this.iFCHistoryTableAdapter.Fill(this.cancerRegistry.IFCHistory);

}

private void button2\_Click(object sender, EventArgs e)

{

CancerRegistryTableAdapters.IFCLogBookTableAdapter log = new CancerRegistryTableAdapters.IFCLogBookTableAdapter();

CancerRegistry cancer = new CancerRegistry();

log.Fill(cancer.IFCLogBook, (int)psPatitemDataGridView.CurrentRow.Cells[1].Value);

if (cancer.IFCLogBook.Rows.Count > 0)

{

frmIFC\_LogBook ifc = new frmIFC\_LogBook();

ifc.pk = psPatitemDataGridView.CurrentRow.Cells[1].Value.ToString();

ifc.ShowDialog();

}

else {

log.Insert((int)psPatitemDataGridView.CurrentRow.Cells[1].Value, null, null, null, null,false);

frmIFC\_LogBook ifc = new frmIFC\_LogBook();

ifc.pk = psPatitemDataGridView.CurrentRow.Cells[1].Value.ToString();

ifc.ShowDialog();

}

this.iFCHistoryTableAdapter.Fill(this.cancerRegistry.IFCHistory);

}

private void radioButton1\_CheckedChanged(object sender, EventArgs e)

{

iFCHistoryBindingSource.Filter = null;

dateTimePicker1.Enabled = true;

dateTimePicker2.Enabled = true;

button1.Enabled = true;

textBox1.Enabled = false;

button3.Enabled = false;

}

private void radioButton3\_CheckedChanged(object sender, EventArgs e)

{

iFCHistoryBindingSource.Filter = null;

dateTimePicker1.Enabled = false;

dateTimePicker2.Enabled = false;

button1.Enabled = false;

textBox1.Enabled = true;

button3.Enabled = true;

}

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

{

iFCHistoryBindingSource.Filter = "rendate >= '"+dateTimePicker1.Value.ToString("MM-dd-yyyy 00:00:00")+"' and rendate <= ' "+ dateTimePicker2.Value.ToString("MM-dd-yyyy 23:59:59") + "'AND " + isEncoded;

this.iFCHistoryTableAdapter.Fill(this.cancerRegistry.IFCHistory);

}

private void button3\_Click(object sender, EventArgs e)

{

iFCHistoryBindingSource.Filter = "fullName like '%"+textBox1.Text+ "%' AND " + isEncoded;

this.iFCHistoryTableAdapter.Fill(this.cancerRegistry.IFCHistory);

}

private void KeyPressDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

button3.PerformClick();

}

}

private void checkBox1\_CheckedChanged(object sender, EventArgs e)

{

if (!checkBox1.Checked)

{

iFCHistoryBindingSource.Filter = "(isEncode is null OR isEncode = 0)";

isEncoded = "(isEncode is null OR isEncode = 0)";

}

else

{

iFCHistoryBindingSource.Filter = "(IsEncode = 1)";

isEncoded = "(IsEncode = 1)";

}

}

}

}

frmIFC\_Log\_Books\_Reports:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace HealthInformationSystem4.Forms

{

public partial class frmIFC\_Log\_Books\_Reports : Form

{

public frmIFC\_Log\_Books\_Reports()

{

InitializeComponent();

}

private void frmIFC\_Log\_Books\_Reports\_Load(object sender, EventArgs e)

{

this.reportViewer1.RefreshReport();

}

private void reportViewer1\_Load(object sender, EventArgs e)

{

}

}

}

frmIFC\_LogBook:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace HealthInformationSystem4.Forms

{

public partial class frmIFC\_LogBook : Form

{

public string pk { get; set; }

//public bool dateonset = false;

public bool dateCatherer = false;

public frmIFC\_LogBook()

{

InitializeComponent();

}

private void frmIFC\_LogBook\_Load(object sender, EventArgs e)

{

this.iFCLogBookTableAdapter.Fill(this.cancerRegistry.IFCLogBook, int.Parse(pk));

this.iFCPreviesTableAdapter.Fill(this.cancerRegistry.IFCPrevies, int.Parse(pk));

if (timeCatheterTextBox.Text != "")

{

string[] split = timeCatheterTextBox.Text.Split(':');

numericUpDown1.Value = decimal.Parse(split[0].ToString());

numericUpDown2.Value = decimal.Parse(split[1].ToString());

}

dateCatherer = isEncodeCheckBox.Checked ? true : false;

}

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

{

try

{

timecheck();

if (dateCatherer /\*&& dateonset\*/ && timetocheck())

{

isEncodeCheckBox.Checked = true;

this.iFCLogBookBindingSource.EndEdit();

this.iFCLogBookTableAdapter.Update(this.cancerRegistry.IFCLogBook);

MessageBox.Show("Succesful!");

this.Close();

}

else

{

MessageBox.Show("Please Enter a Date And Fill up all the Form", "Date Required And Other Form");

}

}

catch (Exception ) {

MessageBox.Show("Please Try Again or call for an IT");

}

}

private bool timetocheck() {

bool pass = nurseAssistTextBox.Text != "" && timeCatheterTextBox.Text != "" ? true : false;

return pass;

}

private void timecheck()

{

timeCatheterTextBox.Text = numericUpDown1.Value + ":" + numericUpDown2.Value + ":00";

}

private void button2\_Click(object sender, EventArgs e)

{

numericUpDown1.Value = decimal.Parse(DateTime.Now.ToString("HH"));

numericUpDown2.Value = decimal.Parse(DateTime.Now.ToString("mm"));

}

private void DropDownDateCate(object sender, EventArgs e)

{

dateCatherer = true;

}

//private void Dp(object sender, EventArgs e)

//{

// dateonset = true;

//}

private void timeCatheterTextBox\_TextChanged(object sender, EventArgs e)

{

}

}

}

**Report Viewer:**

SELECT a.PK\_psPatRegisters,

e.fullname,

z.birthdate,

dbo.udf\_AgeInYears(z.birthdate, a.registrydate) AS AGE,

z.gender,

y.[dateCatheter],

y.[timeCatheter],

y.[nurseAssist],

dbo.udf\_GetFullName(c.FK\_ASURenUser) as Renderer ,

dbo.udf\_GetAdmittingDoctors(a.pk\_pspatRegisters) AS AdmittingDoctors,

y.[dateofOnset],

b.rendate

--,count(PK\_psPatRegisters)

--format(z.birthdate,'MM-dd-yyyy') as BirthDate,

--z.gender,

--a.dischdiagnosis,

--dbo.udf\_GetFullName(a.FK\_ASURegistry) as AssistBilling,

--a.dischdiagnosis,

--d.itemabbrev

FROM psPatRegisters a

JOIN psDatacenter e

ON e.PK\_psDatacenter = a.FK\_emdPatients

JOIN psPersonaldata z

ON z.PK\_psPersonalData = a.FK\_emdPatients

JOIN psEmergencies x

ON x.FK\_psPatRegisters = a.PK\_psPatRegisters

JOIN psPatitem b

ON a.PK\_psPatRegisters = b.FK\_psPatRegisters

JOIN psPatinv c

ON c.PK\_TRXNO = b.FK\_TRXNO

JOIN iwitems d

ON b.FK\_iwItemsREN = d.PK\_iwItems

JOIN [10.4.1.3\sqlcldh02].[HealthInfoDB].[dbo].[IFCLogBook] y

ON y.FK\_psPatRegisters = a.PK\_psPatRegisters

WHERE d.barcodeid IN ('PAA000379','PAA000191','PAA000444','PAA000494','PAA000581') AND c.doctype <> 'CN' AND a.pattrantype = 'I' AND (y.[dateCatheter] BETWEEN @Start AND @End)

