Project Plastic Card Registration and Printing System

OPD Patient - Civil Hospital Karachi - 01-11-2012

GitHub Link: https://github.com/saqibazam/PlasticCardPrintMachine

---

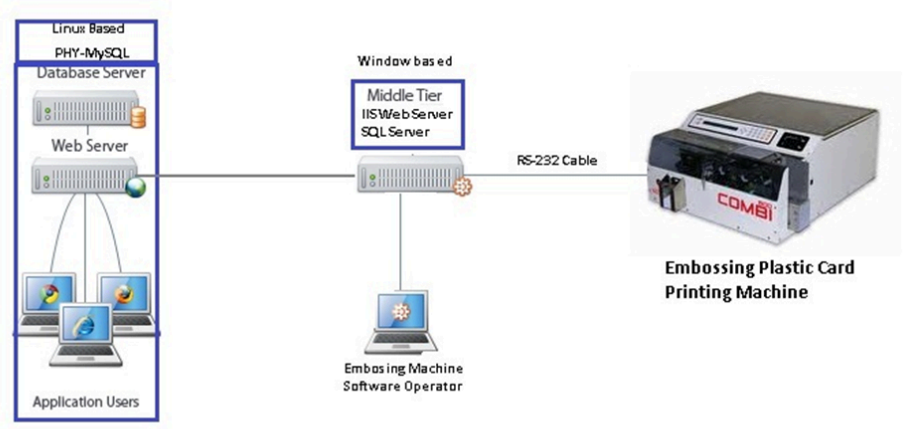
This project outlines a comprehensive system for automating patient plastic card registration and printing at Civil Hospital Karachi's Out-Patient Department (OPD). The system focuses on integrating web-based patient information capture with a specialized printing machine through a robust middle-tier Windows application, ensuring efficient and accurate card issuance.

1. Project Overview

Civil Hospital Karachi: OPD Patient Plastic Card Registration and Printing System

Object: To develop a **middle-tier Windows application** for the Civil Hospital Karachi's OPD to automate **plastic card printing** using an embossing machine. This application bridges the gap between the existing **PHP/MySQL core application (on Linux)** and the embossing machine's **DOS/Windows98/XP and data stored in DBase file.**

Project Visualization



Description:

1. **Patient Data Capture:** Hospital staff at **10 OPD counters** collect patient information using a **PHP web form**. Upon submission, this card data is sent for printing.
2. **Data Transmission: SOAP Web Services** on the PHP server transmit the card data in **XML format** to both the **MySQL database** and the **middle-tier IIS Web Server**.
3. **Data Processing & Transfer:** The IIS Web Server inserts the XML data into a **SQL Server database**. This data is then retrieved on a **First-In, First-Out (FIFO)** basis and inserted into the **Embossing Card Machine's native DBase file**.
4. **Automated Printing:** A **threading mechanism** continuously monitors the DBase file, triggering the embossing machine to print cards in **FIFO order** as new records become available.

Required Forms:

- **Windows Forms:**
  - Card Data Management (Add, Edit, Update in SQL Server)
  - Card Data Posting
  - Card Start / Stop Form (Manages FIFO threading)
- **ASP.NET Pages:**
  - View Pending Card Job
  - View Printed Card
  - Card Printing Report

Required Reports:

- Daily Card Printed Report
- Monthly Card Printed Report

Tools & Technologies:

- **Development Environment:** Visual Studio 2008
- **Databases:** SQL Server 2008, MySQL, Microsoft Visual FoxPro (for DBase)
- **Frameworks:** .NET Framework 3.5, ASP.NET 3.5
- **Web Server:** IIS Web Server 7.4
- **Reporting:** Crystal Report 10.5
- **Integration:** SOAP Web Services, Microsoft Visual FoxPro OLEDB Provider
- **Installer:** Windows Installer 3.1.0
-

6. Project Directory Structure (C:/root)

The project files are organized under the following root directories on the C: drive:

- Asp
- Cp
- Inetpub
- CARDWIN
- Sqldb
- WindowCard

7. Project Duration

The estimated project completion time is **2 months**.

8. Software Installation Steps

A. Windows XP Installation:

- Install Windows XP from the provided CD.
- During installation, set the computer name to **p1-8b050d32a27d**.
- Configure TCP/IP settings:
  - **IP Address:** 172.20.34.5
  - **Subnet Mask:** 255.255.255.0
  - **Default Gateway:** 172.20.34.250
- Install IIS from the Windows XP installation CD.

B. Installation from Setup Folder:

- Install the **.NET Framework**.
- Install **SQL Server** and attach the card database from the c:\sqldb folder.
- Install **Crystal Report Viewer**.
- Install **VFPOLEDBSetup**.

C. Folder Copy and Shortcut Creation:

- Copy all relevant project folders from the installation source to the C: drive root.
- In IIS, create a **Virtual Directory** and name it ASP.
- Create desktop shortcuts for:
  - card.exe located in the CARDWIN folder.
  - CardPrint.exe from c:\cp\CardPrint\CardPrint\bin\Debug\CardPrint.exe.
  - WindowCard.exe from c:\WindowCard\WindowCard\WindowCard\bin\Debug\WindowCard.exe.

>Embossing Machine Command Line parameters:

```
/SET=fileset
/JOB=filejob
/DATA=filedata
/HOWMANY=howmanyrecords
/START=startrecord
/GOAUTO
```

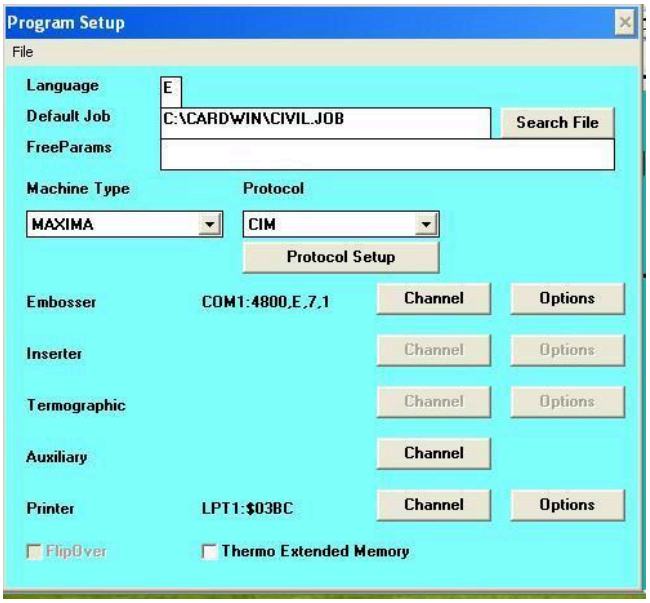
Card.bat for print Card on Embossing Machine

```
@echo off
set par1=%1
C:\CARDWIN\card /job=civil.job /set=civil.set /data=c:/asp/cimeng.dbf /start=%par1% /howmany=1 /copies=1 /goauto
```

SOAP WebService File

```
using System;
using System.Web;
using System.Web.Services;
using System.Web.Services.Protocols;
using System.Data;
using System.Data.SqlClient;
//using System.Data.OleDb;

[WebService(Namespace = "http://PrintCard/")]
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
// To allow this Web Service to be called from script, using ASP.NET AJAX, uncomment the following line.
// [System.Web.Script.Services.ScriptService]
public class Service : System.Web.Services.WebService
{
    public Service () {

        //Uncomment the following line if using designed components
        //InitializeComponent();
    }

    [WebMethod]
```

```
    //public string HelloWorld() {
    //    return "Hello World";
    //}
    public string InsertData(string CardNo, string Name, string DOB, string DeptID, string ip)
    {
        try
        {
            string str = "Data Source =P1-8B050D32A27D\\SQLEXPRESS; Initial Catalog = card ;Integrated Security=True";
            SqlConnection con = new SqlConnection(str);
            SqlCommand cmd = new SqlCommand("sp_insertData", con);
            con.Open();
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@_cardnum", SqlDbType.VarChar).Value = CardNo;
            cmd.Parameters.Add("@_cname", SqlDbType.VarChar).Value = Name;
            cmd.Parameters.Add("@_dob", SqlDbType.VarChar).Value = DOB;
            cmd.Parameters.Add("@_deptid", SqlDbType.VarChar).Value = DeptID;
            cmd.Parameters.Add("@_pdate", SqlDbType.DateTime).Value = DateTime.Now.Date;
            cmd.Parameters.Add("@_cid", SqlDbType.VarChar).Value = ip;
            cmd.Parameters.Add("@_cstatus", SqlDbType.Bit).Value = false;
            cmd.ExecuteNonQuery();
            con.Close();
            return "Data has been Inserted";
        }
        catch (Exception)
        {

            return "Error: Card Already Send";

        }
        }


}
```



9. Database Schema and View

The primary table for tracking card printing statistics is **card_dailytotal**:

SQL

```
USE [card]
GO

SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[card_dailytotal](
   [id] [int] IDENTITY(1,1) NOT NULL,
   [card_date] [datetime] NOT NULL,
   [total_card] [int] NULL,
   [excess_amount] [int] NULL,
   [short_amount] [int] NULL,
CONSTRAINT [PK_card_dailytotal] PRIMARY KEY CLUSTERED
(
   [id] ASC
)WITH (PAD_INDEX = OFF,
STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
```

A SQL view, **View_CardTotal**, is defined to retrieve all records from card_dailytotal, ordered by card_date:

SQL

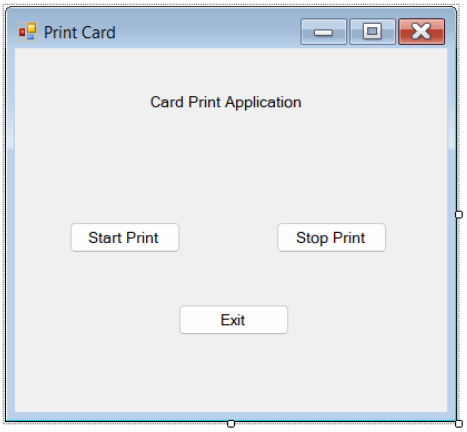
```
USE [card]
GO
```

```
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

ALTER VIEW [dbo].[View_CardTotal]
AS
SELECT TOP (100) PERCENT id, card_date, total_card,
excess_amount, short_amount
FROM dbo.card_dailytotal
ORDER BY card_date
GO
```

10. Application Code Snippets

10.1. Card Print Start and Stop Form (C#)

This C# code demonstrates the core logic for the card printing service, managing the FIFO processing and interaction with the embossing machine:



C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Threading;
using System.Data.OleDb;
using System.Diagnostics;

namespace CardPrint
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
```

```
        // Initialization code for the form
    }

    private void btn_Start_Click(object sender, EventArgs e)
    {
        StartJob();
    }

    private void btn_Stop_Click(object sender, EventArgs e)
    {
        StopJob();
    }

    private bool _isRunning = false;
    private int _interval = 1; // Thread sleep interval in milliseconds

    // Starts the card printing job in a new thread
    public void StartJob()
    {
        btn_Start.Enabled = false;
        ThreadStart oThreadStart = new ThreadStart(DoWork);
        Thread t = new Thread(oThreadStart);
        _isRunning = true;
        t.Start();
    }

    // Stops the card printing job
    public void StopJob()
    {
        btn_Stop.Enabled = false;
        _isRunning = false;
    }

    // The main work loop for the printing thread
    private void DoWork()
    {
        while (_isRunning)
        {
            Thread.Sleep(_interval);
            getCard(); // Retrieve the next card to print

            if (_sno != null && (_cstatus == false))
            {
                insertDBF(); // Insert card data into the DBF for the embossing machine
                PrintBatchCard(); // Execute batch file to print the card
                updatePrintCard(); // Update card status in SQL Server to 'printed'
                _cstatus = true; // Mark current card as processed
            }
        }
        Thread.CurrentThread.Join(); // Ensure thread exits gracefully
    }

    // Private fields to hold card data
    private string _sno;
    private string _cardnum;
    private string _cname;
    private string _dob;
    private string _deptid;
    private string _pdate;
    private string _cid;
```

```
        private bool _cstatus;

        // Fetches card data from SQL Server based on FIFO and status
        private void getCard()
        {
            string str = "Data Source=p1-8b050d32a27d\\SQLEXPRESS;Initial Catalog=card;Integrated Security=True";
            using (SqlConnection con = new SqlConnection(str))
            {
                con.Open();
                SqlCommand countMin = new SqlCommand("select min(sno) from card_Data where cid like 'C-%' and cstatus = 0 and pdate = '" +
DateTime.Now.Date + "'", con);
                string _countMin = countMin.ExecuteScalar()?.ToString();

                if (!string.IsNullOrEmpty(_countMin))
                {
                    SqlCommand cmd = new SqlCommand("select sno, cardnum,cname, dob,deptid,pdate,cid,cstatus from card_Data where sno = '" +
_countMin + "' and cstatus = 0 and pdate = '" + DateTime.Now.Date +"'", con);
                    using (SqlDataReader reader = cmd.ExecuteReader())
                    {
                        if (reader.Read())
                        {
                            _sno = reader["sno"].ToString();
                            _cardnum = reader["cardnum"].ToString();
                            _cname = reader["cname"].ToString();
                            _dob = reader["dob"].ToString();
                            _deptid = reader["deptid"].ToString();
                            _pdate = reader["pdate"].ToString();
                            _cid = reader["cid"].ToString();
                            _cstatus = Convert.ToBoolean(reader["cstatus"]);
                        }
                    }
                }
            }
        }

        // Updates the card status in the SQL Server database to 'printed'
        public void updatePrintCard()
        {
            string str = "Data Source=p1-8b050d32a27d\\SQLEXPRESS;Initial Catalog=card;Integrated Security=True";
            using (SqlConnection con = new SqlConnection(str))
            {
                con.Open();
                SqlCommand cmd = new SqlCommand("update card_Data set cstatus = 1 where cstatus = 0 and pdate = '" + DateTime.Now.Date + "'
and sno = " + _sno + " ", con);
                cmd.ExecuteNonQuery();
            }
        }

        // Inserts card data into the Visual FoxPro DBF file
        private void insertDBF()
        {
            string strLogConnectionString = @"Provider=vfpoledb;Data Source= C:\ASP\CIMENG.DBF;Collating Sequence=machine;Mode=ReadWrite;";
            using (OleDbConnection strConLog = new OleDbConnection(strLogConnectionString))
            {
                strConLog.Open();
                string currentDate = DateTime.Now.ToString("dd/MM/yyyy");
                OleDbCommand oComm = new OleDbCommand("Insert into CIMENG(CARDNUM,NAME,DOB,DEPTID,PDATE,STATUS,USER) values('" +
                                        _cardnum + "','" + _cname + "','" + _dob + "','"
                                        + _deptid +"','" + currentDate + "', 1,'" + _cid + "')",
                                        strConLog);
```

```
            oComm.ExecuteNonQuery();
        }
    }

    // Counts the number of cards in the DBF file
    public int countCard()
    {
        string strLogConnectionString = @"Provider=vfpoledb;Data Source= C:\ASP\CIMENG.DBF;Collating Sequence=machine;Mode=ReadWrite;";
        using (OleDbConnection strConLog = new OleDbConnection(strLogConnectionString))
        {
            strConLog.Open();
            OleDbCommand countquery = new OleDbCommand("select count(CARDNUM) from CIMENG", strConLog);
            string count = countquery.ExecuteScalar()?.ToString();
            return string.IsNullOrEmpty(count) ? 0 : Convert.ToInt32(count);
        }
    }

    // Executes a batch file to trigger the card printing machine
    public void PrintBatchCard()
    {
        Process p = new Process();
        p.StartInfo.UseShellExecute = false;
        p.StartInfo.WindowStyle = ProcessWindowStyle.Hidden;
        p.StartInfo.FileName = @"c:\asp\card.bat";
        p.StartInfo.WorkingDirectory = @"c:\asp";
        int count = countCard();
        p.StartInfo.Arguments = count.ToString();
        p.StartInfo.CreateNoWindow = true;
        p.Start();
        p.StartInfo.WindowStyle = ProcessWindowStyle.Minimized; // Ensure the window is minimized
        p.WaitForExit();
    }

    private void btn_Exit_Click(object sender, EventArgs e)
    {
        Application.ExitThread();
        Application.Exit();
    }
  }
}
```

10.2. WindowCard Login Form (C#)

This code snippet manages the user login process for the Windows Card application:

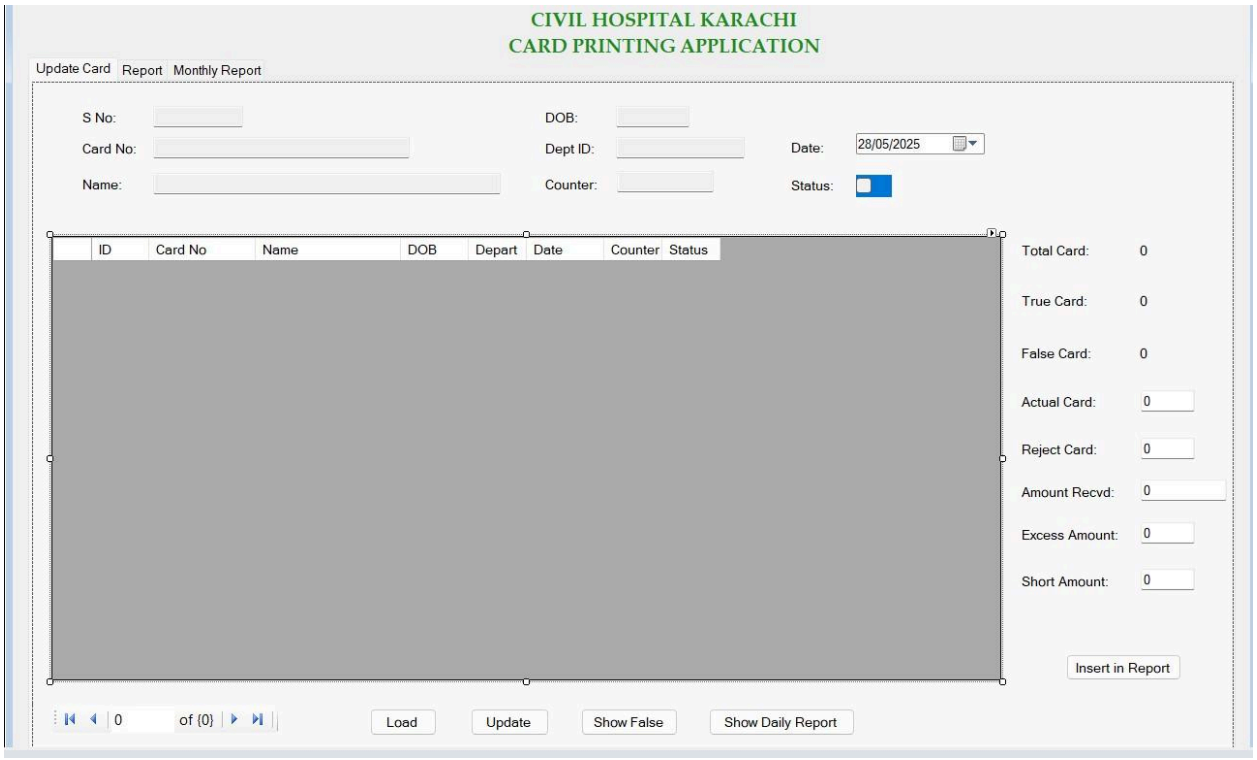
C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace WindowCard
{
    public partial class Form_Login : Form
    {
        public Form_Login()
        {
            InitializeComponent();
            txtName.Focus(); // Set focus to the username field
        }

        private void bttnLogin_Click(object sender, EventArgs e)
        {
            LoginCheck();
        }

        // Checks user credentials for login
        private void LoginCheck()
        {
            if (txtName.Text == "Admin" && txtPass.Text =="SunBeamImpact")
            {
                this.Visible = false; // Hide the login form
                Form_CardWindow fw = new Form_CardWindow();
                fw.Show(); // Show the main card window
            }
            else
            {
                MessageBox.Show("INVALID USER NAME OR PASSWORD", "INVALID PASSWORD", MessageBoxButtons.OK,
MessageBoxIcon.Information);
                txtName.Focus();
                txtName.SelectionStart = 0;
                txtName.SelectionLength = txtName.Text.Length;
                txtPass.Text = ""; // Clear password field
            }
        }

        private void txtPass_KeyPress(object sender, KeyPressEventArgs e)
        {
            if (e.KeyChar == (char)Keys.Enter) // Check for Enter key press
            {
                LoginCheck();
            }
        }

        private void button1_Click(object sender, EventArgs e)
        {
            Application.Exit(); // Exit the application
        }

        private void txtName_KeyPress(object sender, KeyPressEventArgs e)
        {
            if (e.KeyChar == (char)Keys.Enter) // Check for Enter key press
```

```
        {
            txtPass.Focus(); // Move focus to password field
        }
    }

    private void Form_Login_Load(object sender, EventArgs e)
    {
        txtName.Focus();
    }
  }
}
```

10.3. WindowCard Data Report Form (C#)

This C# code manages the data display and report generation for the WindowCard application:



C#

```
using System;
using System.Windows.Forms;
using CrystalDecisions.CrystalReports.Engine;// Required for Crystal Reports

namespace WindowCard
{
  public partial class Form_CardWindow : Form
  {
    private DialogResult dlgResult;

    public Form_CardWindow()
    {
        InitializeComponent();
    }

    private void Form_CardWindow_Load(object sender, EventArgs e)
    {
        try
```

```
        {
            // Fill data for the current date
            this.card_dataTableAdapter.FillByCdate(this.cardDataSet1.card_data, pdateDateTimePicker.Value);

            // Update card count labels
            label_TotalCard.Text = this.card_dataTableAdapter.CountTotalCard(pdateDateTimePicker.Value)?.ToString() ?? "0";
            label_TrueCard.Text = this.card_dataTableAdapter.CountCard(true, pdateDateTimePicker.Value)?.ToString() ?? "0";
            label_FalseCard.Text = this.card_dataTableAdapter.CountCard(false, pdateDateTimePicker.Value)?.ToString() ?? "0";
        }
        catch (Exception ex)
        {
            MessageBox.Show(Convert.ToString(ex), "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }

    private void card_dailytotalBindingNavigatorSaveItem_Click(object sender, EventArgs e)
    {
        try
        {
            this.Validate();
            this.card_dailytotalBindingSource.EndEdit();
            this.tableAdapterManager.UpdateAll(this.cardDataSet);
        }
        catch (Exception ex)
        {
            MessageBox.Show(Convert.ToString(ex), "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }

    private void button_Load_Click(object sender, EventArgs e)
    {
        try
        {
            label_heading.Text = "Load Current Date Record";
            this.card_dataTableAdapter.FillByCdate(this.cardDataSet1.card_data, pdateDateTimePicker.Value);

            // Update card count labels
            label_TotalCard.Text = this.card_dataTableAdapter.CountTotalCard(pdateDateTimePicker.Value)?.ToString() ?? "0";
            label_TrueCard.Text = this.card_dataTableAdapter.CountCard(true, pdateDateTimePicker.Value)?.ToString() ?? "0";
            label_FalseCard.Text = this.card_dataTableAdapter.CountCard(false, pdateDateTimePicker.Value)?.ToString() ?? "0";
        }
        catch (Exception ex)
        {
            MessageBox.Show(Convert.ToString(ex), "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }

    private void button_Update_Click(object sender, EventArgs e)
    {
        try
        {
            label_heading.Text = "Updated Record";
            this.card_dataTableAdapter.UpdateQuery_Status(cstatusCheckBox.Checked, pdateDateTimePicker.Value,
cardnumTextBox.Text);
            this.card_dataTableAdapter.FillByCdate(this.cardDataSet1.card_data, pdateDateTimePicker.Value);

            // Recalculate and update card counts
            label_TotalCard.Text = this.card_dataTableAdapter.CountTotalCard(pdateDateTimePicker.Value)?.ToString() ?? "0";
            label_TrueCard.Text = this.card_dataTableAdapter.CountCard(true, pdateDateTimePicker.Value)?.ToString() ?? "0";
            label_FalseCard.Text = this.card_dataTableAdapter.CountCard(false, pdateDateTimePicker.Value)?.ToString() ?? "0";
```

```
        }
        catch (Exception ex)
        {
            MessageBox.Show(Convert.ToString(ex),"Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }

    private void button_FalseCard_Click(object sender, EventArgs e)
    {
        try
        {
            label_heading.Text = "False Record";
            this.card_dataTableAdapter.FillByFalse(this.cardDataSet1.card_data, pdateDateTimePicker.Value);
            label_FalseCard.Text = this.card_dataTableAdapter.CountCard(false, pdateDateTimePicker.Value)?.ToString() ?? "0";
        }
        catch (Exception ex)
        {
            MessageBox.Show(Convert.ToString(ex), "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }

    private void button_PrintDailyReport_Click(object sender, EventArgs e)
    {
        try
        {
            CrystalReport_DailyReport CDR = new CrystalReport_DailyReport();
            CDR.Load("C:\\WindowCard\\WindowCard\\WindowCard\\CrystalReport_DailyReport.rpt");
            crystalReportViewer1.ReportSource = CDR;
            crystalReportViewer1.DisplayToolbar = true;
            crystalReportViewer1.DisplayGroupTree = false;
            CDR.SetParameterValue("_pdate", pdateDateTimePicker.Value);
            tabControl1.SelectTab(1); // Assuming tabControl1 has a tab for reports
        }
        catch (Exception Ex)
        {
            MessageBox.Show(Convert.ToString(Ex), "Report Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }

    private void button_InsertMonthlyReport_Click(object sender, EventArgs e)
    {
        dlgResult = MessageBox.Show("Do you want to insert this record?", "Insert Record", MessageBoxButtons.YesNo,
MessageBoxIcon.Warning);
        if (dlgResult == DialogResult.Yes)
        {
            try
            {
                // Attempt to insert the record
                this.card_dailytotalTableAdapter.Insert_Card_DailyTotal(
                    pdateDateTimePicker.Value,
                    Convert.ToInt32(label_TotalCard.Text),
                    Convert.ToInt32(textBox_ExcessAmount.Text),
                    Convert.ToInt32(textBox_ShortAmount.Text),
                    Convert.ToInt32(textBox_tAmount.Text),
                    Convert.ToInt32(textBox_ActualCard.Text),
                    Convert.ToInt32(textBox_RejectCard.Text)
                );
                MessageBox.Show("Record has been inserted, click OK to continue...", "Insert Record", MessageBoxButtons.OK,
MessageBoxIcon.Information);
            }
```

```
            catch
            {
                // If insertion fails (e.g., record already exists), offer to update
                dlgResult = MessageBox.Show("This record already exists. Do you want to update it?", "Already Inserted",
MessageBoxButtons.YesNo, MessageBoxIcon.Warning);
                if (dlgResult == DialogResult.Yes)
                {
                    try
                    {
                        // Attempt to update the record
                        this.card_dailytotalTableAdapter.UpdateCardDailyTotal(
                            Convert.ToInt32(label_TotalCard.Text),
                            Convert.ToInt32(textBox_ExcessAmount.Text),
                            Convert.ToInt32(textBox_ShortAmount.Text),
                            Convert.ToInt32(textBox_tAmount.Text),
                            Convert.ToInt32(textBox_ActualCard.Text),
                            Convert.ToInt32(textBox_RejectCard.Text),
                            pdateDateTimePicker.Value
                        );
                        MessageBox.Show("Record Updated Successfully", "Update Record", MessageBoxButtons.OK,
MessageBoxIcon.Information);
                    }
                    catch (Exception Ex)
                    {
                        MessageBox.Show(Ex.ToString(), "Update Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
                    }
                }
            }
        }
    }

    private void button_MonthlyReport_Click(object sender, EventArgs e)
    {
        try
        {
            CrystalReport_MonthlyReport CMR = new CrystalReport_MonthlyReport();
            CMR.Load("C:\\WindowCard\\WindowCard\\WindowCard\\CrystalReport_MonthlyReport.rpt");
            crystalReportViewer_MonthlyReport.ReportSource = CMR;
            crystalReportViewer_MonthlyReport.DisplayToolbar = true;
            crystalReportViewer_MonthlyReport.DisplayGroupTree = false;
            CMR.SetParameterValue("DateTo", dateTimePickerTo.Value);
            CMR.SetParameterValue("DateFrom", dateTimePickerFrom.Value);
        }
        catch (Exception Ex)
        {
            MessageBox.Show(Convert.ToString(Ex), "Report Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
  }
}
```

ASP.NET Code in C:/ASP Folder:

Card Batch File Auto Print Code:

Code snippet

```
@echo off
```

```
set par1=%1
C:\CARDWIN\card /job=civil.job /set=civil.set /data=c:/asp/cimeng.dbf /start=%par1% /howmany=1 /copies=1 /goauto
```

Default.aspx.cs Code:

C#

```
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;


public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        Service ws = new Service();
        //TextBox1 .Text = ws.HelloWorld();

        //TextBox1.Text = ws.HelloWorld2("Hello ", "PHP WebServices");

        string CardNo = "11111111111";
        string Name = "ABDUL WAHAB KHAN";
        string DOB = "M-55";
        string DeptID = "SERGICAL-01";
        string ip = "C-2";

        TextBox1.Text = ws.InsertData(CardNo, Name, DOB, DeptID,ip);
    }
}
```

PrintCard.aspx.cs Code:

C#

```
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
```

```
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;

using System.Threading;
using System.Data.OleDb;
using System.Diagnostics;

public partial class PrintCard : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void Start_Print_Click(object sender, EventArgs e)
    {
        processCard();
    }

    public int countCard()
    {

        string strLogConnectionString = @"Provider=vfpoledb;Data Source= C:\ASP\CIMENG.DBF;Collating Sequence=machine;Mode=ReadWrite;";

        OleDbConnection strConLog = new OleDbConnection(strLogConnectionString);
        strConLog.Open();
        //string pdate = Convert .ToString (DateTime.Now);
        //string pdate = DateTime.Now.Date.ToShortDateString();
        OleDbCommand countquery = new OleDbCommand("select count(CARDNUM) from CIMENG", strConLog);
        countquery.ExecuteScalar();
        string count = countquery.ExecuteScalar().ToString();
        int a = Convert.ToInt32(count);
        return a;
    }


    public void PrintBatchCard()
    {

        Process p = new Process();
        //p.StartInfo.FileName = "CMD.exe"; //Execute command
        p.StartInfo.FileName = @"c:\asp\card.bat";
        p.StartInfo.WorkingDirectory = @"c:\asp";
        int count = countCard();
        p.StartInfo.Arguments = count.ToString();
        p.Start();
        p.WaitForExit();

    }

    public void callCard()
    {
        int a = countCard();
        Thread.Sleep(10000);
        int b = countCard();
        if (b > a)
        {
            PrintBatchCard();

        }
```

```
    }

    public void processCard()
    {
        //Print Card on Thread with Time Interval
        ThreadStart operation = new ThreadStart(callCard);
        Thread[] theThreads = new Thread[500];

        for (int x = 0; x < 500; ++x)
      {
         // Creates, but does not start, a new thread
         theThreads[x] = new Thread(operation);

         // Starts the work on a new thread
         theThreads[x].Start();
            Thread.Sleep(10000);

        }

        // Wait for each thread to complete
        foreach (Thread t in theThreads)
        {
            t.Join();
            //t.Abort();

        }


    }

}
```

ViewCard.aspx.cs Code:

C#

```
using System;
using System.Collections.Generic;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;

using System.Data.SqlClient;
using System.Data.OleDb;

using CrystalDecisions.CrystalReports.Engine;
using CrystalDecisions.Shared;

public partial class ViewCard : System.Web.UI.Page
{

    protected void Page_Load(object sender, EventArgs e)
    {

    }
// protected System.Timers.Timer _timer;

// private void timer_Elapsed(object sender, System.Timers.ElapsedEventArgs e)
```

```
// {
// // Do whatever you want to do on each tick of the timer
// TextBox1.Text = "OKKK";
//}

    protected void Btn_Pending_job_Click(object sender, EventArgs e)
  {
    CrystalReportViewer1.Visible = false;
    GridView1.Visible = true;
    callParaFunction(DateTime.Now.Date, false);
//Total Card
    int tn = countCard();
    lbl_tcard.Text = tn.ToString();
    int tam = tn * 20;
    lbl_tamount.Text = tam.ToString();

  }
  protected void Btn_Completed_Click(object sender, EventArgs e)
  {
    CrystalReportViewer1.Visible = false;
    GridView1.Visible = true ;
    callParaFunction(DateTime.Now.Date, true);
  //Total Card
    int tn = countCard();
    lbl_tcard.Text = tn.ToString();
    int tam = tn * 20;
    lbl_tamount.Text = tam.ToString();

  }
  protected void Btn_Print_Record_Click(object sender, EventArgs e)
  {
    GridView1.Visible = false;
    CrystalReportViewer1.Visible = true ;
    showReportSQL();
    //Total Card
    int tn = countCard();
    lbl_tcard.Text = tn.ToString();
    int tam = tn * 20;
    lbl_tamount.Text = tam.ToString();

  }


  protected void Button1_Click(object sender, EventArgs e)
  {
    CrystalReportViewer1.Visible = false;
    GridView1.Visible = true;
    callParaFunction(DateTime.Now.Date, false);
    //GridView1.Columns..HeaderText = "Sno";
  }

  public void showReportSQL()
  {
    string str = "Data Source=p1-8b050d32a27d\\sqlexpress;Initial Catalog=card;Integrated Security=True";
    SqlConnection con = new SqlConnection(str);
    con.Open();

    //string currentDate;
    //currentDate = TextBox1.Text;

    ReportDocument rptDoc = new ReportDocument();
```

```
        DataSet ds = new DataSet();

        DataTable dt = new DataTable();

        // dt.TableName = "Crystal";

        SqlCommand cmd = new SqlCommand("select * from card_Data where cid like 'C-%' and cstatus = 1 and pdate = '" + DateTime.Now.Date + "' ", con);
        cmd.CommandType = CommandType.Text;
        SqlDataAdapter da = new SqlDataAdapter();
        da.SelectCommand = cmd;
        da.Fill(dt);
        ds.Merge(dt);
        rptDoc.Load("C://asp//CrystalReport_CP.rpt");
        // rptDoc.Load(Server.MapPath("http://172.20.10.46/PRINT/CrystalReport1.rpt"));
        rptDoc.SetDataSource(ds);
        CrystalReportViewer1.ReportSource = rptDoc;
        rptDoc.SetParameterValue("pdate", DateTime .Now .Date );

        //rptDoc.SetParameterValue("curDate", "18/08/2012");

        con.Close();

    }


    //public void showReportDB()
    //{
    // string strLogConnectionString = @"Provider=vfpoledb;Data Source= C:\ASP\CIMENG.DBF;Collating Sequence=machine;Mode=ReadWrite;";

    // OleDbConnection strConLog = new OleDbConnection(strLogConnectionString);
    // strConLog.Open();

    // // string currentDate;
    // // currentDate = string.Format("{0:dd/MM/yyyy}", DateTime.Now);
    // string currentDate;
    // currentDate = TextBox1 .Text;

    // ReportDocument rptDoc = new ReportDocument();
    // DataSet ds = new DataSet();

    // DataTable dt = new DataTable();

    // // dt.TableName = "Crystal";

    // OleDbCommand cmd = new OleDbCommand("select * from CIMENG where pdate = '" + currentDate + "' ", strConLog);
    // cmd.CommandType = CommandType.Text;
    // OleDbDataAdapter da = new OleDbDataAdapter();
    // da.SelectCommand = cmd;
    // da.Fill(dt);
    // ds.Merge(dt);
    //rptDoc.Load("C://asp//CrystalReport_CP.rpt");
    // // rptDoc.Load(Server.MapPath("http://172.20.10.46/PRINT/CrystalReport1.rpt"));
    // rptDoc.SetDataSource(ds);
    // CrystalReportViewer1.ReportSource = rptDoc;
    // rptDoc.SetParameterValue("pdate", currentDate);
    //rptDoc.SetParameterValue("curDate", "18/08/2012");

    // strConLog.Close();
```

```
//}


public void callParaFunction(DateTime cuDate, bool cuStatus)
{
    string str = "Data Source=p1-8b050d32a27d\\sqlexpress;Initial Catalog=card;Integrated Security=True";
    SqlConnection con = new SqlConnection(str);
    SqlCommand cmd = new SqlCommand("sp_getData", con);
    con.Open();
    cmd.CommandType = Comm
```

Code Developed by: Muhammad Saqib Azam