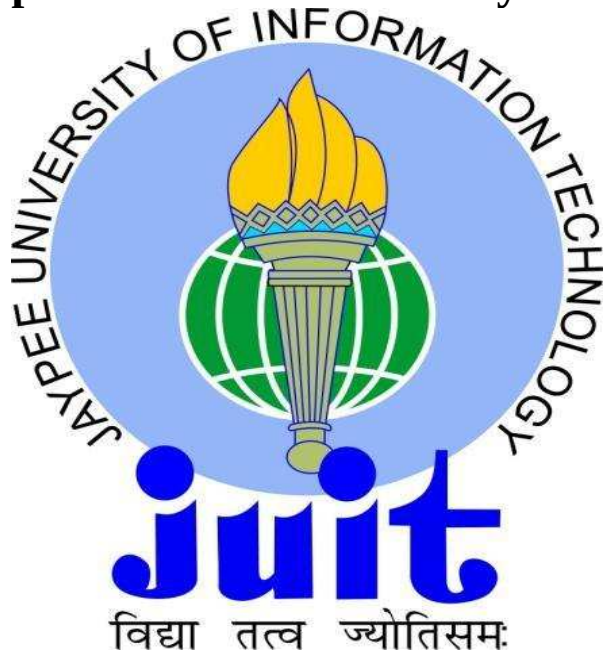
TO IMPLEMENT HEALTH CARE PRODUCT IN ASP.NET

Submitted by :
Sidharth Mittal
101316
Under the Supervision of Prof. Ansuyia Makroo



MAY 2014

Submitted in partial fulfillment of the requirement for the degree of
BACHELOR OF TECHNOLOGY
Department of Computer Science & Engineering
JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY
WAKNAGHAT SOLAN , HIMACHAL PRADESH

CERTIFICATE

This is to certify that the work titled "Health Care Product" submitted by SIDHARTH MITTAL, in partial fulfillment for the award of degree of B.Tech of Jaypee University of Information Technology, Waknaghat, Himachal Pradesh has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Signature of Supervisor: ……

Name of Supervisor: Prof Ansuyia Makroo

Designation: Lecturer

Date:

ACKNOWLEDGMENT

Working on this project has been a great learning experience for me, provided a brisk and valuable experience to me. My experience while working on this project ranges from moments of anxiety to ecstasy when after working for several days I was able to successfully implement this project. It would not have been possible for me to work on this project without the help of many people and I would like to express my gratitude to them.

This project "**Health Care Product"** undertaken at "JUIT" has been a useful and enlightening experience. I express my sincere thanks to **Prof. Ansuyia Makroo**, for her guidance, endless hours of help and tremendous support during the development of this project. Her cooperation and suggestions have greatly aided to the timely completion of my project possible.

I am also grateful to **Prof. Dr. S.P Grera** H.O.D of CSE department, for his support and guidance.

I also express my gratitude towards all the people associated with project for their support, co-operation and cheerful readiness in reviewing this project.

Finally, I would like to acknowledge and thank all my colleagues for cheerfully providing assistance whenever need.

Last but not least, I am very thankful to my parents who are my source of inspiration in every field of life.

Name of the Student: Date:

Sidharth Mittal(101316)

ABSTRACT

Heath Care product provides the benefits of streamlined operations, enhanced administration and control, superior patient care, strict cost control and at the same time improves profitability.

The "**Health Care Product"** application enables an end user to view the details of doctors, patients, ward numbers etc from the hospital database. End user place a query to gather particular information at the enquiry counter. The operator sitting there then provide correct information to the users by using this product. The application also enables to the Admin, manage the data stored in a database for the hospital, such as adding a record, modifying record, and deleting a particular record. Users can also make request for the hospital bill and computer generated bill will be issued to the users by using this product.

CONTENTS

1. INTRODUCTION

"Health Care Product" is a desktop application. **Health Care Product** project is mainly useful for any individual that wants to get current information about patients, doctors and other useful information from hospital like blood donor details. The aim is to provide up-to-date information to the end user. It also enables the admin to add or modify new entries about both patients and the staff members into the database.

The primary objective of the project is to provide the users with an integrated hospital management system which works on a variety of different platforms and provides him with an easy access to the hospital records.

About the Project

Objective:

The "**Health Care Product"** application enables an end user to view the details of doctors, patients, ward numbers etc from the hospital database. End user place a query to gather particular information at the enquiry counter. The operator sitting there then provide correct information to the users by using this product. The application also enables to the Admin, manage the data stored in a database for the hospital, such as adding a record, modifying record, and deleting a particular record. Users can also make request for the hospital bill and computer generated bill will be issued to the users by using this product. The aim is to provide up-to-date information to the end user.

The **Health Care Product** has been designed for multispecialty hospitals, to cover a wide range of hospital administration and management processes. It is an itegrated end to end hospital management system which provides relevant decision making for patient care, hospital administration and critical financial accounting.

2. Project Analysis

(a) Feasibility study:

Feasibility study encompasses the following things:

1. Technical Feasibility
2. Operational Feasibility
3. Economical Feasibility

1. Technical Feasibility:

Technical feasibility determines whether the organization has the technology and skills necessary to carryout the project and how the technology and skills necessary to carryout the project and how should this is obtained. The system can be technically feasible because of the following grounds.

- All necessary technology exists to develop the system.
- The existing resources are capable and can hold all the necessary data.
- The system is too flexible and it can be expanded further.
- The system can give guarantees of accuracy, ease of use,
- Reliability and the data security.
- The system can give instant responses to inquiries.

So "*e-MAGAZINE*" can conclude that the system is technically feasible.

Hardware & Software Specifications :

Software Requirement:

Front-end Tools	Microsoft C# .Net
Back-end Tools	Ms Access

Hardware Requirement:

Processor	Minimum P3@ 700 MHz Recommended: P4@2.0 Recommended: 256MB
RAM	Minimum 128 MB.
Hard Disk Space	Minimum100 MB Recommended: 500MB
Video	800*600,256 Colors Recommended: high color 16-bit
Operating System	Windows 2000 Professional Windows XP Professional

2. Operational Feasibility:

Operational feasibility determines if the proposed system satisfied user objectives and can be fitted into the current system operation. The present system Distribution management system can be justified as operationally feasible based on the following grounds.

- The methods of processing and presentation are completely accepted to the manager since they can meet all the requirements.
- The proposed system will not cause any problem under any circumstances.

Is proposed system will certainly satisfy the user objectives and it will also enhance their capability. The proposed system can be best fitted into current operation. Also there is no need to replace any existing staff. Therefore the system is operationally feasible.

3. Economical Feasibility:

Economical feasibility determines whether projects goal can be with in the resource limits allocated to it. It must determines whether it is worthwhile to process with the project all or whether the benefits obtained from the new system is not worth the costs. After conducting cost benefit analysis, it reveals that the objectives of the proposed system can be achieved within the allocated resources. Proposed system requires no extra manpower, cost almost nil. Also the cash invested to implement the proposed system can be easily recoverable. So the system is economically feasible.

(b). REQUIREMENT ANALYSIS :

FUNCTIONAL REQURIMENTS:

The requirement phase basically consists of three activities:

1. Requirement Analysis
2. Requirement Specification
3. Requirement Validation

1. Requirement Analysis:

Requirement Analysis is a software engineering task that bridges the gap between system level software allocation and software design. It provides the system engineer to specify software function and performance indicate software's interface with the other system elements and establish constraints that software must meet.

The basic aim of this stage is to obtain a clear picture of the needs and requirements of the end-user and also the organization. Analysis involves interaction between the clients and the analysis. Usually analysts research a problem from any questions asked and reading existing documents. The analysts have to uncover the real needs of the user even if they don't know them clearly. During analysis it is essential that a complete and consistent set of specifications emerge for the system. Here it is essential to resolve the contradictions that could emerge from information got from various parties.

This is essential to ensure that the final specifications are consistent.

It may be divided into 5 areas of effort.

1. Request recognition
2 Evaluation and synthesis
3. Modeling
4. Specification
5. Review

Each Requirement analysis method has a unique point of view. However all analysis methods are related by a set of operational principles. They are

- The information domain of the request must be represented and understood.
- The functions that the software is to perform must be defined.
- The behavior of the software as a consequence of external events must be defined.
- The models that depict information function and behavior must be partitioned in a hierarchical or layered fashion.
- The analysis process must move from essential information to implementation detail.

2. REQUIREMENTS SPECIFICATION

Specification Principles:

Software Requirements Specification plays an important role in creating quality software solutions. Specification is basically a representation process. Requirements are represented in a manner that ultimately leads to successful software implementation.

Requirements may be specified in a variety of ways. However there are some guidelines worth following: -

- Representation format and content should be relevant to the request
- Information contained within the specification should be nested
- Diagrams and other notational forms should be restricted in number and consistent in use.
- Representations should be revisable.

Software Requirements Specifications:

The software requirements specification is produced at the culmination of the analysis task. The function and performance allocated to the software as a part of system engineering are refined by establishing a complete information description.

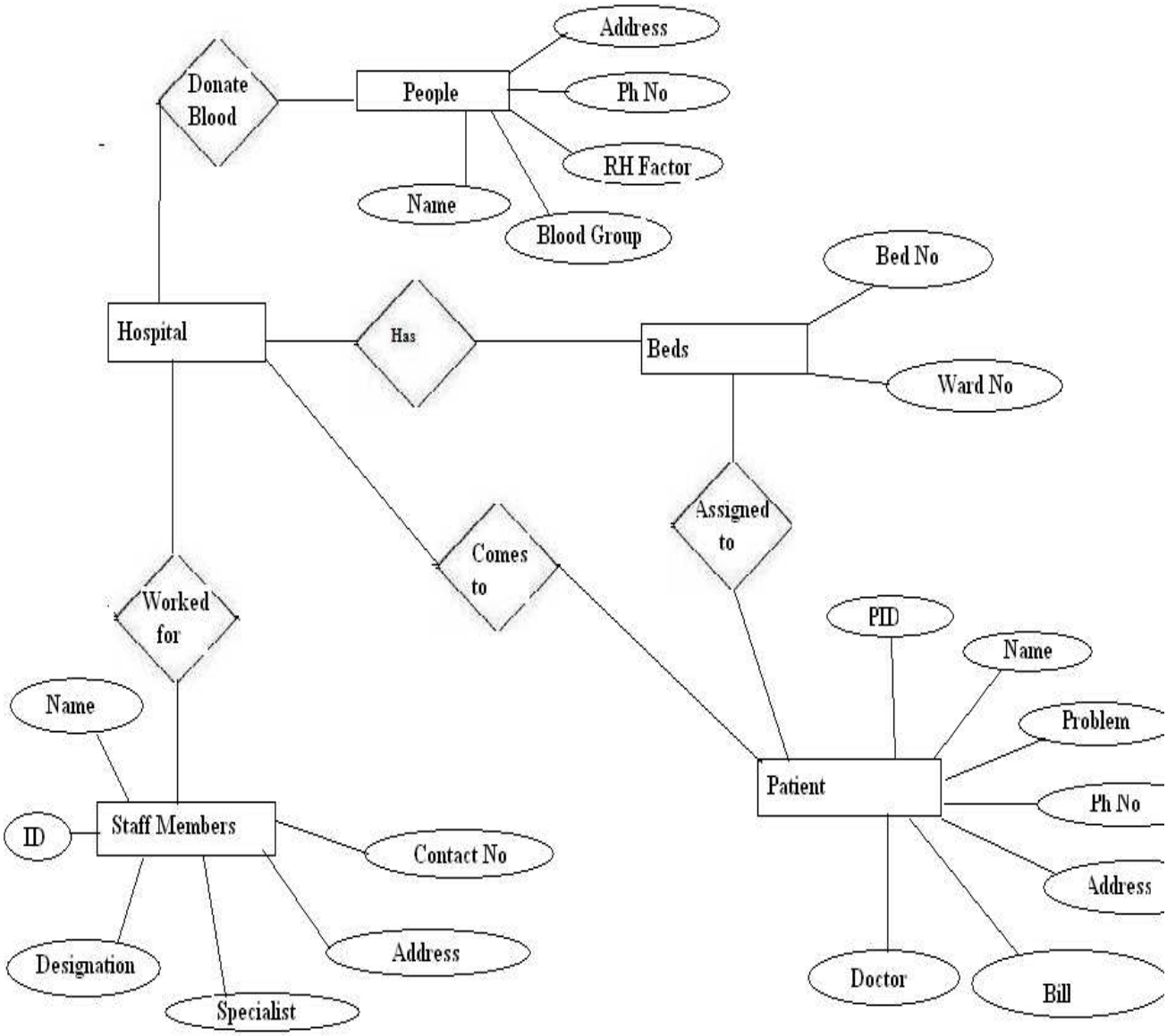
Implementation Language

- C#
- ADO .Net

3.DESIGN

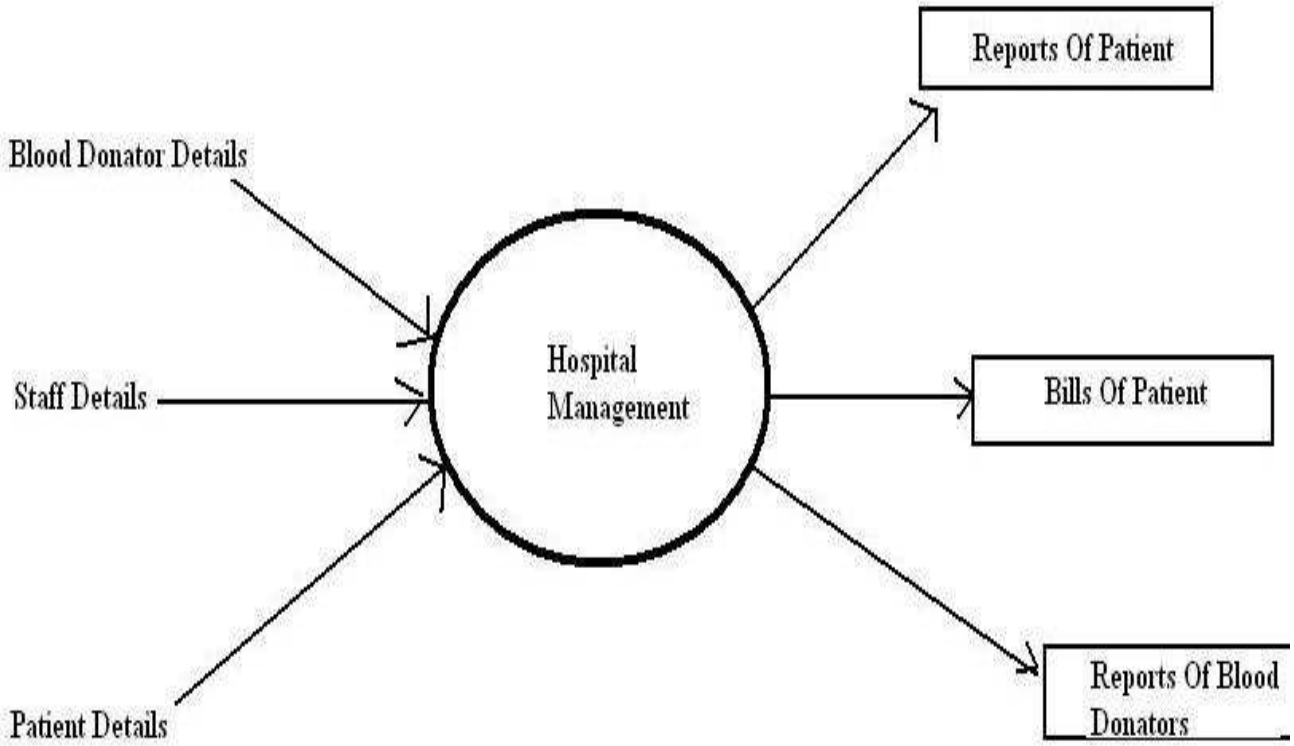
WORKFLOW DIAGRAMS

1. **ER Diagram**

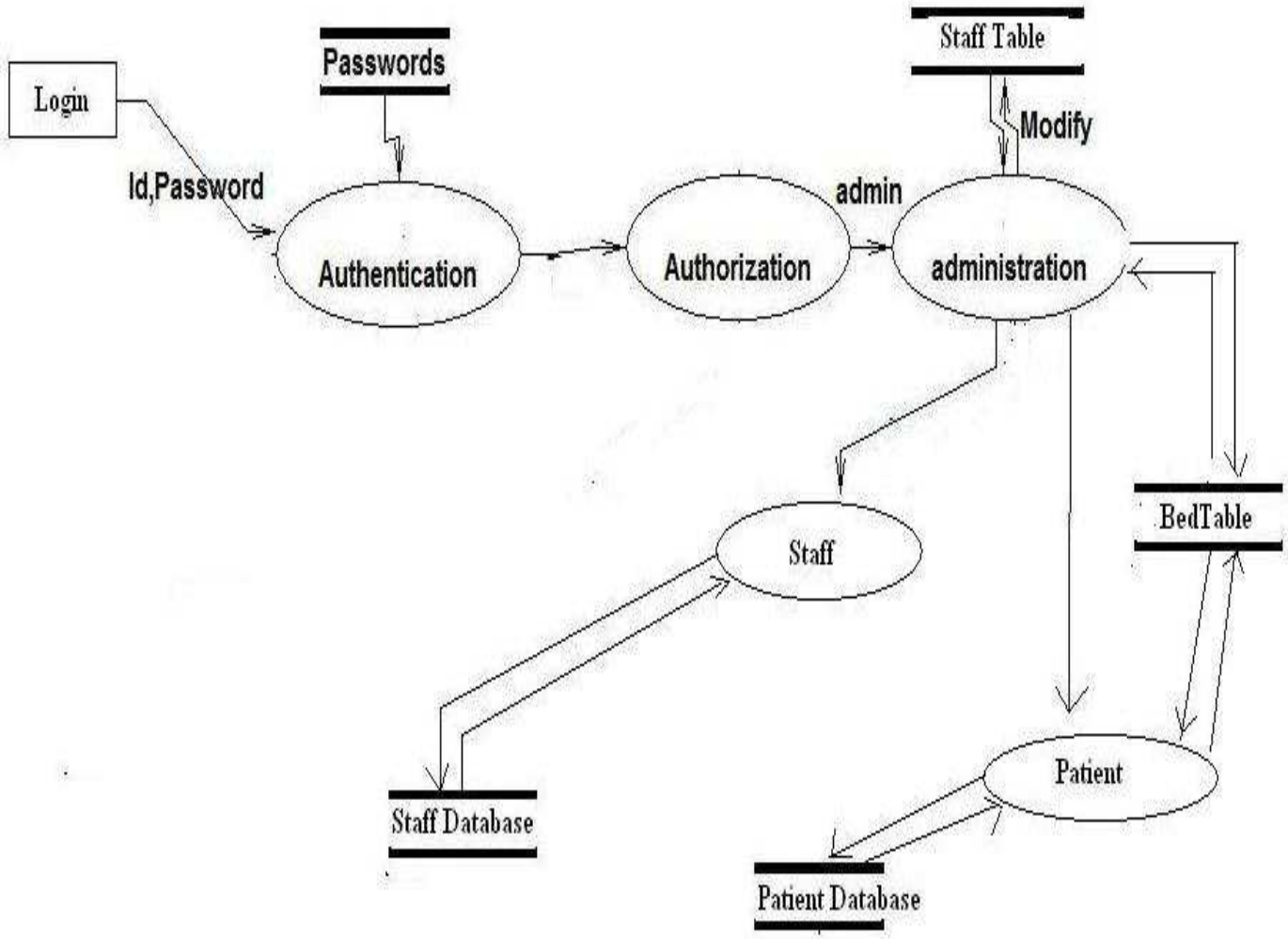


2. Data Flow Diagram

Zero Level DFD(Context Diagram):



Level 1 DFD


Login
Passwords
Staff Table
Modify
Id,Password
admin
Authentication
Authorization
administration
BedTable
Staff
Staff Database
Patient
Patient Database

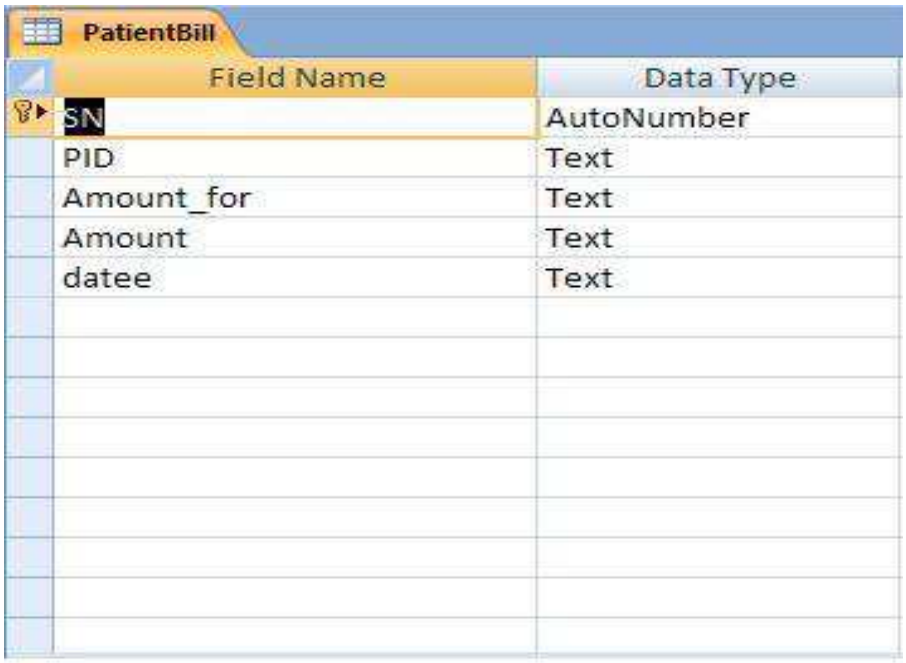
4.DATABASE DESIGN TABLES

Patient

Field Name	Data Type
PID	AutoNumber
Name	Text
Father	Text
Contact	Text
Address	Text
City	Text
PIN	Text
Age	Text
Problem	Text
Doctor	Text
Datee	Text
BedNo	Text

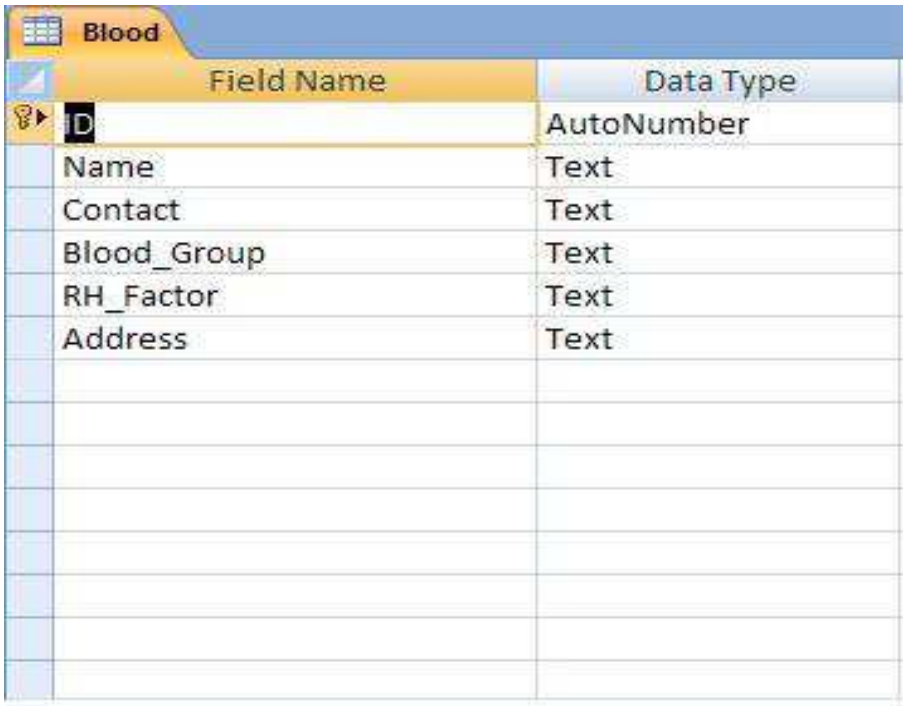
staff

Field Name	Data Type
Emp_ID	AutoNumber
Name	Text
Father	Text
Contact_No	Text
Address	Text
City	Text
PIN	Text
Education	Text
DOBirth	Date/Time
Designation	Text
Specialist	Text
DOJoin	Date/Time



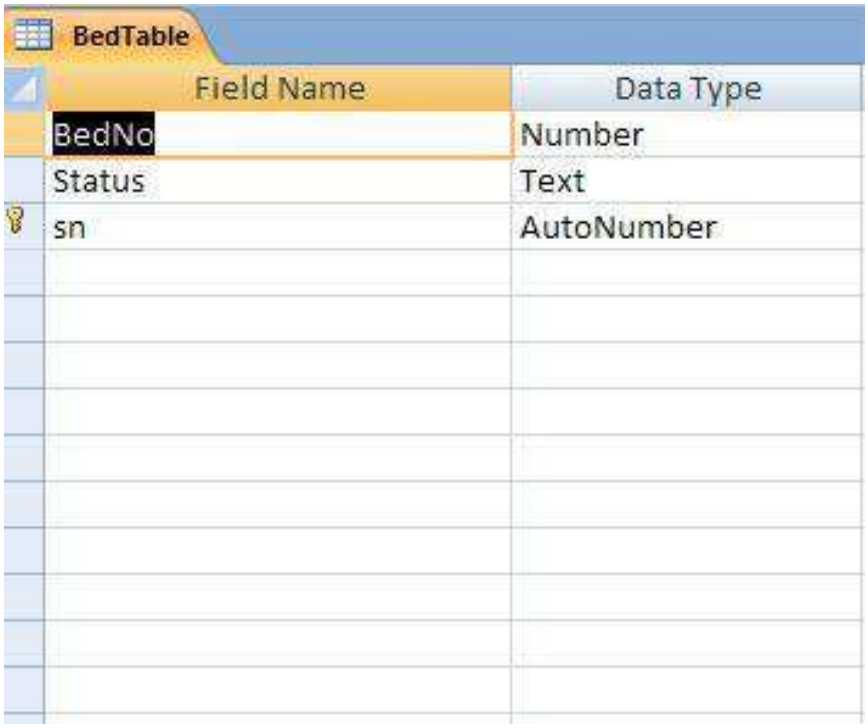
PatientBill

Field Name	Data Type
SN	AutoNumber
PID	Text
Amount_for	Text
Amount	Text
datee	Text



Blood

Field Name	Data Type
ID	AutoNumber
Name	Text
Contact	Text
Blood_Group	Text
RH_Factor	Text
Address	Text


BedTable
Field Name
Data Type
BedNo
Number
Status
Text
sn
AutoNumber

5.CODE GENERATION

1.FORM1.cs

```
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 Health_Care_Product
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void timer1_Tick(object sender, EventArgs e)
        {
            if (progressBar1.Value < 100)
                progressBar1.Value = progressBar1.Value + 1;
            else
            {
                timer1.Stop();
                timer1.Enabled = false;
                Form2 ob = new Form2();
                this.Hide();
                ob.Show();
            }
        }
    }
}
```

2.FORM2.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;

using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Health_Care_Product
{
    public partial class Form2 : Form
    {
        <div style="width:1345px;height:700px;background-
image:url(img14.jpg)">
        public Form2()
        {
            InitializeComponent();
        }

        private void menuStrip1_ItemClicked(object sender,
ToolStripItemClickedEventArgs e)
        {

        }

        private void searchToolStripMenuItem_Click(object sender,
EventArgs e)
        {

        }

        private void Form2_Load(object sender, EventArgs e)
        {

        }

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

```
            Form3 frmob = new Form3(this);
            frmob.Show();

        }

        private void exitToolStripMenuItem_Click(object sender,
EventArgs e)
        {

        }

        private void Form2_FormClosing(object sender,
FormClosingEventArgs e)
        {
            Application.Exit();
        }

        private void patientEntryToolStripMenuItem_Click(object
sender, EventArgs e)
        {
            Form4 ob = new Form4(this);
            ob.Show();
        }

        private void newArrivalToolStripMenuItem_Click(object
sender, EventArgs e)
        {
            Form17 ob = new Form17(this);
            ob.Show();
        }

        private void exitToolStripMenuItem_Click_1(object sender,
EventArgs e)
        {
            Application.Exit();
        }

        private void staffToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            Form5 ob = new Form5(this);
            ob.Show();
        }

        private void allPatientToolStripMenuItem_Click(object
sender, EventArgs e)
```

```
        {
            Form6 ob = new Form6(this);
            ob.Show();
        }

        private void byNameToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            Form7 ob = new Form7(this);
            ob.Show();
        }

        private void byEmployeeIDToolStripMenuItem_Click(object
sender, EventArgs e)
        {
            Form8 ob = new Form8(this);
            ob.Show();
        }

        private void
bySpecializationToolStripMenuItem_Click(object sender, EventArgs
e)
        {
            Form9 ob = new Form9(this);
            ob.Show();
        }

        private void byNameToolStripMenuItem1_Click(object
sender, EventArgs e)
        {
            Form10 ob = new Form10(this);
            ob.Show();
        }

        private void byWardNumberToolStripMenuItem_Click(object
sender, EventArgs e)
        {
            Form11 ob = new Form11(this);
            ob.Show();
        }

        private void staffRecordToolStripMenuItem_Click(object
sender, EventArgs e)
        {
            Form12 ob = new Form12(this);
            ob.Show();
```

```
        }

        private void patientDetailsToolStripMenuItem_Click(object
sender, EventArgs e)
        {
            Form13 ob = new Form13(this);
            ob.Show();
        }

        private void bloodDonateToolStripMenuItem_Click(object
sender, EventArgs e)
        {
            Form14 ob = new Form14(this);
            ob.Show();
        }

        private void
viewAllBloodDonatorsToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            Form15 ob = new Form15(this);
            ob.Show();
        }

        private void searchForBloodToolStripMenuItem_Click(object
sender, EventArgs e)
        {
            Form16 ob = new Form16(this);
            ob.Show();
        }

        private void
supplyMedicineToToolStripMenuItem_Click(object sender, EventArgs
e)
        {
            Form18 ob = new Form18(this);
            ob.Show();
        }

        private void
dischargPatientToolStripMenuItem1_Click(object sender, EventArgs
e)
        {
            Form19 ob = new Form19(this);
            ob.Show();
        }
```

```
        private void aboutProjectToolStripMenuItem_Click(object
sender, EventArgs e)
        {
            Form20 ob = new Form20(this);
            ob.Show();
        }
    }
}
```

3.FORM3.cs

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

namespace Health_Care_Product
{
    public partial class Form3 : Form
    {
        public Form3(Form2 ob)
        {
            InitializeComponent();
            this.MdiParent = ob;
        }

        private void Form3_Load(object sender, EventArgs e)
        {
            this.comboBox2.Visible = false;
            this.label12.Visible = false;
            this.comboBox2.Text = "-";
        }

        private void groupBox1_Enter(object sender, EventArgs e)
```

```
        {

        }

        private void comboBox1_SelectedIndexChanged(object
sender, EventArgs e)
        {
            if (this.comboBox1.SelectedItem== "Doctor")
            {
                this.comboBox2.Visible = true;
                this.label12.Visible = true;
            }
            else
            {
                this.comboBox2.Visible = false;
                this.label12.Visible = false;
                this.comboBox2.Text = "-";
            }
        }

        private void button2_Click(object sender, EventArgs e)
        {
            this.Dispose();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            OleDbConnection con=null;
            if (this.textBox1.Text == "" || this.textBox2.Text ==
"" || this.textBox3.Text == "" || this.textBox4.Text == "" ||
this.textBox5.Text == "" || this.textBox6.Text == "" ||
this.textBox7.Text == "" || this.dateTimePicker1.Text == "" ||
this.comboBox1.Text == "" || this.comboBox2.Text == "" ||
this.dateTimePicker2.Text == "" || textBox3.Text.Length<10)
            {
                MessageBox.Show("Please Fill empty Fields",
"Error", MessageBoxButtons.OK);
            }
            else
            {
                //Code to insert staff record into staff table
                try
                {
                    con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
```

```
                    con.Open();
                    OleDbDataAdapter odp = new
OleDbDataAdapter("select * from staff", con);
                    OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                    DataSet ds = new DataSet();
                    odp.Fill(ds, "staff");
                    DataRow drow = ds.Tables["staff"].NewRow();
                    drow[1] = textBox1.Text;
                    drow[2] = textBox2.Text;
                    drow[3] = textBox3.Text;
                    drow[4] = textBox4.Text;
                    drow[5] = textBox5.Text;
                    drow[6] = textBox6.Text;
                    drow[7] = textBox7.Text;
                    drow[8] = dateTimePicker1.Text;
                    drow[9] = comboBox1.Text;
                    drow[10] = comboBox2.Text;
                    drow[11] = dateTimePicker2.Text;

                    ds.Tables["staff"].Rows.Add(drow);
                    odp.Update(ds, "staff");
                    OleDbCommand com = new OleDbCommand("select
Emp_ID from staff where Name='" + textBox1.Text + "' and
Father='"+textBox2.Text+"' and Designation='"+comboBox1.Text+"'",
con);
                    Int32 idc;
                    idc = (Int32)com.ExecuteScalar();
                    MessageBox.Show("Record Saved.Your Emp ID is
: "+idc.ToString() , "SAVE", MessageBoxButtons.OK);
                    textBox1.Text = "";
                    textBox2.Text = "";
                    textBox3.Text = "";
                    textBox4.Text = "";
                    textBox5.Text = "";
                    textBox6.Text = "";
                    textBox7.Text = "";
                    comboBox2.Text = "-";
                    comboBox2.Visible = false;
                    label12.Visible = false;
                    comboBox1.Text = "";
                }
                catch (Exception err)
                {
                    MessageBox.Show("Connection to database
failed", "Connection Failed", MessageBoxButtons.OK);
```

```
                }
                finally
                {
                    con.Close();
                }

            }
        }
        private void textBox1_KeyPress(object sender,
KeyPressEventArgs e)
        {
            if (e.KeyChar > 47 && e.KeyChar < 58)
                e.Handled = true;
        }

        private void textBox2_KeyPress(object sender,
KeyPressEventArgs e)
        {
            if (e.KeyChar > 47 && e.KeyChar < 58)
                e.Handled = true;
        }

        private void textBox3_KeyPress(object sender,
KeyPressEventArgs e)
        {

            if ((e.KeyChar < 48 || e.KeyChar > 57 ||
textBox3.Text.Length >= 10) && e.KeyChar != 8)
                e.Handled = true;
        }

        private void textBox6_KeyPress(object sender,
KeyPressEventArgs e)
        {
            if ((e.KeyChar < 48 || e.KeyChar > 57 ) && e.KeyChar
!= 8)
                e.Handled = true;
        }

        private void textBox1_TextChanged(object sender,
EventArgs e)
        {

        }
    }
}
```

4.FORM4.cs

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

namespace Health_Care_Product
{
    public partial class Form4 : Form
    {
        OleDbConnection con = null;
        public Form4(Form2 ob)
        {
            InitializeComponent();
            this.MdiParent = ob;
        }

        private void Form4_Load(object sender, EventArgs e)
        {



            try
            {
                con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                con.Open();
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from staff where
Designation='Doctor'", con);
                DataSet ds = new DataSet();
                odp.Fill(ds, "staff");
                foreach (DataRow dr in ds.Tables["staff"].Rows)
                {
                    comboBox2.Items.Add(dr[0]);
                    comboBox1.Items.Add(dr[1]);
```

```
                }

            }
            catch (Exception err)
            {
                MessageBox.Show(err.Message, "Connection Failed",
MessageBoxButtons.OK);

            }
            finally
            {
                con.Close();
            }



            try
            {
                con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                con.Open();
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from BedTable where Status='false'",
con);
                DataSet ds = new DataSet();
                odp.Fill(ds, "BedTable");
                foreach (DataRow dr in
ds.Tables["BedTable"].Rows)
                {
                    comboBox3.Items.Add(dr[0]);
                   }
            }
            catch (Exception err)
            {
                MessageBox.Show(err.Message, "Connection Failed",
MessageBoxButtons.OK);

            }
            finally
            {
                con.Close();

            }
        }
```

```
        private void groupBox1_Enter(object sender, EventArgs e)
        {

        }

        private void button1_Click(object sender, EventArgs e)
        {


            if (this.textBox1.Text == "" || this.textBox2.Text ==
"" || this.textBox3.Text == "" || this.textBox4.Text == "" ||
this.textBox5.Text == "" || this.textBox6.Text == "" ||
this.textBox7.Text == "" || this.textBox8.Text =="" ||
comboBox1.Text =="" || comboBox3.Text=="" ||
textBox3.Text.Length<10)
            {
                MessageBox.Show("Please Fill empty Fields",
"Error", MessageBoxButtons.OK);
            }
            else
            {

                try
                {
                    con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                    con.Open();

                    OleDbDataAdapter odp = new
OleDbDataAdapter("select * from Patient", con);
                    OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                    DataSet ds = new DataSet();
                    odp.Fill(ds,"Patient");
                    DataRow droww =
ds.Tables["Patient"].NewRow();

                    droww[1] = textBox1.Text.ToString();
                    droww[2] = textBox2.Text.ToString();
                    droww[3] = textBox3.Text.ToString();
                    droww[4] = textBox4.Text.ToString();
                    droww[5] = textBox5.Text.ToString();
                    droww[6] = textBox6.Text.ToString();
                    droww[7] = textBox7.Text.ToString();
```

```
                    droww[8] = textBox8.Text.ToString();
                    droww[9] =
comboBox2.Text.ToString().ToString();
                    droww[10] = dateTimePicker2.Text.ToString();
                    droww[11] = comboBox3.Text.ToString();

                    ds.Tables["Patient"].Rows.Add(droww);
                    odp.Update(ds, "Patient");

                    OleDbCommand com = new OleDbCommand("select
PID from Patient where Name='" + textBox1.Text + "' and Father='"
+ textBox2.Text+"' and BedNo='"+comboBox3.Text+"'", con);
                    Int32 idc;
                    idc = (Int32)com.ExecuteScalar();
                    MessageBox.Show("Record Saved.Your Patient ID
is : " + idc.ToString(), "SAVE", MessageBoxButtons.OK);



                        OleDbDataAdapter odp1 = new
OleDbDataAdapter("select * from BedTable", con);
                        OleDbCommandBuilder bul1 = new
OleDbCommandBuilder(odp1);
                        DataSet ds1 = new DataSet();
                        odp1.Fill(ds1, "BedTable");
                        int recno = 0;
                        foreach (DataRow dr1 in
ds1.Tables["BedTable"].Rows)
                        {

                            if (dr1[0].ToString() ==
comboBox3.Text.ToString())
                            {

ds1.Tables["BedTable"].Rows[recno][1] = "true";
                                odp1.Update(ds1, "BedTable");
                                break;
                            }
                            recno++;
                        }



                    this.Close();
                }
                catch (Exception err)
```

```
                {
                    MessageBox.Show(err.Message, "Connection
Failed", MessageBoxButtons.OK);
                }


            }
        }

        private void comboBox1_SelectedIndexChanged(object
sender, EventArgs e)
        {
            comboBox2.SelectedIndex = comboBox1.SelectedIndex;
        }

        private void textBox3_KeyPress(object sender,
KeyPressEventArgs e)
        {
            if ((e.KeyChar < 48 || e.KeyChar > 57 ||
textBox3.Text.Length >= 10) && e.KeyChar != 8)
                e.Handled = true;
        }

        private void textBox1_KeyPress(object sender,
KeyPressEventArgs e)
        {

            if (e.KeyChar > 47 && e.KeyChar < 58)
                e.Handled = true;
        }

        private void textBox6_TextChanged(object sender,
EventArgs e)
        {

        }

        private void textBox6_KeyPress(object sender,
KeyPressEventArgs e)
        {
            if ((e.KeyChar < 48 || e.KeyChar > 57) && e.KeyChar
!= 8)
                e.Handled = true;
        }
```

```
        private void textBox2_KeyPress(object sender,
KeyPressEventArgs e)
        {

            if (e.KeyChar > 47 && e.KeyChar < 58)
                e.Handled = true;
        }

        private void textBox7_KeyPress(object sender,
KeyPressEventArgs e)
        {
            if ((e.KeyChar < 48 || e.KeyChar > 57 ||
textBox7.Text.Length >= 3) && e.KeyChar != 8)
                e.Handled = true;
        }

        private void label1_Click(object sender, EventArgs e)
        {

        }

        private void dateTimePicker2_ValueChanged(object sender,
EventArgs e)
        {

        }

        private void label13_Click(object sender, EventArgs e)
        {

        }

        private void label11_Click(object sender, EventArgs e)
        {

        }

        private void textBox7_TextChanged(object sender,
EventArgs e)
        {

        }

        private void label8_Click(object sender, EventArgs e)
        {
```

```
        }

        private void label7_Click(object sender, EventArgs e)
        {

        }

        private void textBox5_TextChanged(object sender,
EventArgs e)
        {

        }

        private void label6_Click(object sender, EventArgs e)
        {

        }

        private void textBox4_TextChanged(object sender,
EventArgs e)
        {

        }

        private void textBox3_TextChanged(object sender,
EventArgs e)
        {

        }

        private void label5_Click(object sender, EventArgs e)
        {

        }

        private void label4_Click(object sender, EventArgs e)
        {

        }

        private void textBox2_TextChanged(object sender,
EventArgs e)
        {

        }
```

```
        private void label3_Click(object sender, EventArgs e)
        {

        }

        private void textBox1_TextChanged(object sender,
EventArgs e)
        {

        }

        private void button2_Click(object sender, EventArgs e)
        {
            this.Dispose();
        }

        private void comboBox3_SelectedIndexChanged(object
sender, EventArgs e)
        {

        }

        private void label10_Click(object sender, EventArgs e)
        {

        }

        private void comboBox2_SelectedIndexChanged(object
sender, EventArgs e)
        {

        }

        private void textBox8_TextChanged(object sender,
EventArgs e)
        {

        }

        private void label9_Click(object sender, EventArgs e)
        {

        }

        private void label2_Click(object sender, EventArgs e)
        {
```

```
        }
    }
}
```

5.FORM5.cs

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

namespace Health_Care_Product
{
    public partial class Form5 : Form
    {
       public Form5(Form2 ob)
        {
            InitializeComponent();
            this.MdiParent = ob;
        }

        private void Form5_Load(object sender, EventArgs e)
        {
            try
            {
                OleDbConnection con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                con.Open();
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from staff", con);
                OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                DataSet ds = new DataSet();
```

```
                odp.Fill(ds, "staff");
                dataGridView1.DataSource = ds.Tables["staff"];
                con.Close();
            }
            catch (Exception eee)
            {
                MessageBox.Show("Connection to database failed",
"Connection Failed", MessageBoxButtons.OK);
            }
        }
    }
}
```

6.FORM6.cs

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

namespace Health_Care_Product
{
    public partial class Form6 : Form
    {
       public Form6(Form2 ob)
        {
            InitializeComponent();
            this.MdiParent = ob;
        }

        private void Form6_Load(object sender, EventArgs e)
        {
            try
            {
```

```
                OleDbConnection con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                con.Open();
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from Patient", con);
                OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                DataSet ds = new DataSet();
                odp.Fill(ds, "Patient");
                dataGridView1.DataSource = ds.Tables["Patient"];
                con.Close();
            }
            catch (Exception eee)
            {
                MessageBox.Show("Connection to database failed",
"Connection Failed", MessageBoxButtons.OK);
            }
        }
    }
}
```

7.FORM7.cs

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

namespace Health_Care_Product
{
    public partial class Form7 : Form
    {
       public Form7(Form2 ob)
```

```
        {
            InitializeComponent();
            this.MdiParent = ob;
        }

        private void Form7_Load(object sender, EventArgs e)
        {

        }

        private void textBox1_KeyPress(object sender,
KeyPressEventArgs e)
        {

        }

        private void textBox1_TextChanged(object sender,
EventArgs e)
        {

        }

        private void textBox1_KeyUp(object sender, KeyEventArgs
e)
        {
            try
            {
                OleDbConnection con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                con.Open();
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from staff where Name like'" +
textBox1.Text + "%'", con);

                OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                DataSet ds = new DataSet();
                odp.Fill(ds, "staff");
                dataGridView1.DataSource = ds.Tables["staff"];
                con.Close();
            }
            catch (Exception eee)
            {
                MessageBox.Show("Connection to database failed",
"Connection Failed", MessageBoxButtons.OK);
```

```
            }
        }
    }
}
```

8.FORM8.cs

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

namespace Health_Care_Product
{
    public partial class Form8 : Form
    {public Form8(Form2 ob)
        {
            InitializeComponent();
            this.MdiParent = ob;
        }

        private void Form8_Load(object sender, EventArgs e)
        {

        }

        private void textBox1_KeyUp(object sender, KeyEventArgs
e)
        {
            try
            {
                OleDbConnection con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                con.Open();
```

```
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from staff where Emp_ID like'" +
textBox1.Text + "%'", con);

                OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                DataSet ds = new DataSet();
                odp.Fill(ds, "staff");
                dataGridView1.DataSource = ds.Tables["staff"];
                con.Close();
            }
            catch (Exception eee)
            {
                MessageBox.Show("Connection to database failed",
"Connection Failed", MessageBoxButtons.OK);
            }
        }
    }
}
```

9.FORM9.cs

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

namespace Health_Care_Product
{
    public partial class Form9 : Form
    {
      public Form9(Form2 ob)
        {
            InitializeComponent();
            this.MdiParent = ob;
        }

        private void Form9_Load(object sender, EventArgs e)
```

```
        {

        }

        private void textBox1_KeyUp(object sender, KeyEventArgs
e)
        {
            try
            {
                OleDbConnection con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                con.Open();
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from staff where Specialist like'" +
textBox1.Text + "%'", con);

                OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                DataSet ds = new DataSet();
                odp.Fill(ds, "staff");
                dataGridView1.DataSource = ds.Tables["staff"];
                con.Close();
            }
            catch (Exception eee)
            {
                MessageBox.Show("Connection to database failed",
"Connection Failed", MessageBoxButtons.OK);
            }
        }

        private void textBox1_TextChanged(object sender,
EventArgs e)
        {

        }
    }
}
```

10.FORM10.cs

```
using System;
```

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

namespace Health_Care_Product
{
    public partial class Form10 : Form
    {
        public Form10(Form2 ob)
        {
            InitializeComponent();
            this.MdiParent = ob;
        }

        private void Form10_Load(object sender, EventArgs e)
        {

        }

        private void textBox1_KeyUp(object sender, KeyEventArgs
e)
        {

            try
            {
                OleDbConnection con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                con.Open();
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from Patient where Name like'" +
textBox1.Text + "%'", con);

                OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                DataSet ds = new DataSet();
                odp.Fill(ds, "Patient");
                dataGridView1.DataSource = ds.Tables["Patient"];
                con.Close();
            }
            catch (Exception eee)
```

```
            {
                MessageBox.Show("Connection to database failed",
"Connection Failed", MessageBoxButtons.OK);
            }
        }
    }
}
```

11.FORM11.cs

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

namespace Health_Care_Product
{
    public partial class Form11 : Form
    {
       public Form11(Form2 ob)
        {
            InitializeComponent();
            this.MdiParent = ob;
        }

        private void Form11_Load(object sender, EventArgs e)
        {

        }

        private void textBox1_KeyUp(object sender, KeyEventArgs
e)
        {
            try
            {
                OleDbConnection con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
```

```
                con.Open();
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from Patient where BedNo like'" +
textBox1.Text + "%'", con);

                OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                DataSet ds = new DataSet();
                odp.Fill(ds, "Patient");
                dataGridView1.DataSource = ds.Tables["Patient"];
                con.Close();
            }
            catch (Exception eee)
            {
                MessageBox.Show("Connection to database failed",
"Connection Failed", MessageBoxButtons.OK);
            }
        }
    }
}
```

12.FORM12.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;

using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Health_Care_Product
{
    public partial class Form12 : Form
    {
        public Form12(Form2 ob)
        {
            InitializeComponent();
            this.MdiParent = ob;
```

```
        }

        private void Form12_Load(object sender, EventArgs e)
        {

        }

        private void comboBox1_SelectedIndexChanged(object
sender, EventArgs e)
        {
            if (this.comboBox1.SelectedItem == "Doctor")
            {
                this.comboBox2.Visible = true;
                this.label12.Visible = true;
            }
            else
            {
                this.comboBox2.Visible = false;
                this.label12.Visible = false;
                this.comboBox2.Text = "-";
            }
        }

        private void button1_Click(object sender, EventArgs e)
        {
            int recno = 0;
            int flag = 0;
            OleDbConnection con = null;
            if (this.textBox8.Text == "" || this.textBox1.Text ==
"" || this.textBox2.Text == "" || this.textBox3.Text == "" ||
this.textBox4.Text == "" || this.textBox5.Text == "" ||
this.textBox6.Text == "" || this.textBox7.Text == "" ||
this.dateTimePicker1.Text == "" || this.comboBox1.Text == "" ||
this.comboBox2.Text == "" || this.dateTimePicker2.Text == "" ||
textBox3.Text.Length < 10)
            {
                MessageBox.Show("Please Fill empty Fields",
"Error", MessageBoxButtons.OK);
            }
            else
            {
                //Code to insert staff record into staff table
                try
                {
```

```
                    con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                    con.Open();
                    OleDbDataAdapter odp = new
OleDbDataAdapter("select * from staff", con);
                    OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                    DataSet ds = new DataSet();
                    odp.Fill(ds, "staff");
                    foreach (DataRow dr in
ds.Tables["staff"].Rows)
                    {
                        if (dr[0].ToString() == textBox8.Text)
                        {
                            flag = 1;
                            ds.Tables["staff"].Rows[recno][1] =
textBox1.Text;
                            ds.Tables["staff"].Rows[recno][2] =
textBox2.Text;
                            ds.Tables["staff"].Rows[recno][3] =
textBox3.Text;
                            ds.Tables["staff"].Rows[recno][4] =
textBox4.Text;
                            ds.Tables["staff"].Rows[recno][5] =
textBox5.Text;
                            ds.Tables["staff"].Rows[recno][6] =
textBox6.Text;
                            ds.Tables["staff"].Rows[recno][7] =
textBox7.Text;
                            ds.Tables["staff"].Rows[recno][8] =
dateTimePicker1.Text;
                            ds.Tables["staff"].Rows[recno][9] =
comboBox1.Text;
                            ds.Tables["staff"].Rows[recno][10] =
comboBox2.Text;
                            ds.Tables["staff"].Rows[recno][11] =
dateTimePicker2.Text;
                            odp.Update(ds, "staff");
                            break;
                        }
                        recno++;
                    }
                    if (flag == 1)
                    {
```

```
                        MessageBox.Show("Successfully Updated",
"Updated", MessageBoxButtons.OK);
                        this.Close();
                    }
                    else
                    {
                        MessageBox.Show("Record Not Found", "Not
Found", MessageBoxButtons.OK);
                    }
                }
                catch (Exception err)
                {
                    MessageBox.Show("Connection to database
failed", "Connection Failed", MessageBoxButtons.OK);
                }
                finally
                {
                    con.Close();
                }

            }
        }

        private void button2_Click(object sender, EventArgs e)
        {
            this.Dispose();
        }
    }
}
```

13.FORM13.cs

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

namespace Health_Care_Product
```

```
{
    public partial class Form13 : Form
    {
        OleDbConnection con = null;
        public Form13(Form2 ob)
        {
            InitializeComponent();
            this.MdiParent = ob;
        }


        private void Form13_Load(object sender, EventArgs e)
        {



            try
            {
                con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                con.Open();
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from staff where
Designation='Doctor'", con);
                DataSet ds = new DataSet();
                odp.Fill(ds, "staff");
                foreach (DataRow dr in ds.Tables["staff"].Rows)
                {
                    comboBox2.Items.Add(dr[0]);
                    comboBox1.Items.Add(dr[1]);

                }

            }
            catch (Exception err)
            {
                MessageBox.Show(err.Message, "Connection Failed",
MessageBoxButtons.OK);

            }
            finally
            {
                con.Close();
            }
```

```
        }

        private void comboBox1_SelectedIndexChanged(object
sender, EventArgs e)
        {
            comboBox2.SelectedIndex = comboBox1.SelectedIndex;
        }

        private void button1_Click(object sender, EventArgs e)
        {
            int recno = 0;
            int flag = 0;

            if (this.textBox9.Text == "" || this.textBox1.Text ==
"" || this.textBox2.Text == "" || this.textBox3.Text == "" ||
this.textBox4.Text == "" || this.textBox5.Text == "" ||
this.textBox6.Text == "" || this.textBox7.Text == "" ||
this.textBox8.Text == "" || comboBox1.Text == "" ||
textBox3.Text.Length < 10)
            {
                MessageBox.Show("Please Fill empty Fields",
"Error", MessageBoxButtons.OK);
            }
            else
            {
                //Code to insert staff record into staff table
                try
                {
                    con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                    con.Open();
                    OleDbDataAdapter odp = new
OleDbDataAdapter("select * from Patient", con);
                    OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                    DataSet ds = new DataSet();
                    odp.Fill(ds, "Patient");
                    foreach (DataRow dr in
ds.Tables["Patient"].Rows)
                    {
                        if (dr[0].ToString() == textBox9.Text)
```

```
                        {
                            flag = 1;
                            ds.Tables["Patient"].Rows[recno][1] =
textBox1.Text;
                            ds.Tables["Patient"].Rows[recno][2] =
textBox2.Text;
                            ds.Tables["Patient"].Rows[recno][3] =
textBox3.Text;
                            ds.Tables["Patient"].Rows[recno][4] =
textBox4.Text;
                            ds.Tables["Patient"].Rows[recno][5] =
textBox5.Text;
                            ds.Tables["Patient"].Rows[recno][6] =
textBox6.Text;
                            ds.Tables["Patient"].Rows[recno][7] =
textBox7.Text;
                            ds.Tables["Patient"].Rows[recno][8] =
textBox8.Text;
                            ds.Tables["Patient"].Rows[recno][9] =
comboBox2.Text.ToString().ToString();
                            ds.Tables["Patient"].Rows[recno][10]
= dateTimePicker2.Text.ToString();

                            odp.Update(ds, "Patient");
                            break;
                        }
                        recno++;
                    }
                    if (flag == 1)
                    {
                        MessageBox.Show("Successfully Updated",
"Updated", MessageBoxButtons.OK);
                        this.Close();
                    }
                    else
                    {
                        MessageBox.Show("Record Not Found", "Not
Found", MessageBoxButtons.OK);
                    }
                }
                catch (Exception err)
                {
                    MessageBox.Show("Connection to database
failed", "Connection Failed", MessageBoxButtons.OK);
                }
                finally
```

```
                {
                    con.Close();
                }

            }
        }

        private void label1_Click(object sender, EventArgs e)
        {

        }

        private void button2_Click(object sender, EventArgs e)
        {
            this.Dispose();
        }
    }
}
```

14.FORM14.cs

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

namespace Health_Care_Product
{
    public partial class Form14 : Form
    {
        public Form14(Form2 ob)
        {
            InitializeComponent();
            this.MdiParent = ob;
        }

        private void Form14_Load(object sender, EventArgs e)
        {
```

```
        }

        private void textBox3_KeyPress(object sender,
KeyPressEventArgs e)
        {

            if ((e.KeyChar < 48 || e.KeyChar > 57 ||
textBox3.Text.Length >= 10) && e.KeyChar != 8)
                e.Handled = true;
        }

        private void button1_Click(object sender, EventArgs e)
        {
            OleDbConnection con = null;
            if (this.textBox1.Text == "" || this.textBox3.Text ==
"" || this.textBox5.Text == "" || this.comboBox3.Text=="" ||
this.comboBox4.Text=="")
            {
                MessageBox.Show("Please Fill empty Fields",
"Error", MessageBoxButtons.OK);
            }
            else
            {

                try
                {
                    con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                    con.Open();
                    OleDbDataAdapter odp = new
OleDbDataAdapter("select * from Blood", con);
                    OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                    DataSet ds = new DataSet();
                    odp.Fill(ds, "Blood");
                    DataRow drow = ds.Tables["Blood"].NewRow();
                    drow[1] = textBox1.Text;
                    drow[2] = textBox3.Text;
                    drow[3] = comboBox3.Text;
                    drow[4] = comboBox4.Text;
                    drow[5] = textBox5.Text;

                    ds.Tables["Blood"].Rows.Add(drow);
                    odp.Update(ds, "Blood");
```

```
                    MessageBox.Show("Successfully Saved", "SAVE",
MessageBoxButtons.OK);
                    this.Close();
                }
                catch (Exception err)
                {
                    MessageBox.Show(err.Message, "Connection
Failed", MessageBoxButtons.OK);
                }
                finally
                {
                    con.Close();
                }

            }
        }

        private void button2_Click(object sender, EventArgs e)
        {
            this.Dispose();
        }
    }
}
```

15.FORM15.cs

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

namespace Health_Care_Product
{
    public partial class Form15 : Form
    {
        public Form15(Form2 ob)
        {
            InitializeComponent();
            this.MdiParent = ob;
```

```
        }


        private void Form15_Load(object sender, EventArgs e)
        {
            try
            {
                OleDbConnection con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                con.Open();
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from Blood", con);

                OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                DataSet ds = new DataSet();
                odp.Fill(ds, "Blood");
                dataGridView1.DataSource = ds.Tables["Blood"];
                con.Close();
            }
            catch (Exception eee)
            {
                MessageBox.Show("Connection to database failed",
"Connection Failed", MessageBoxButtons.OK);
            }
        }
    }
}
```

16.FORM16.cs

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

namespace Health_Care_Product
```

```
{
    public partial class Form16 : Form
    {
        public Form16(Form2 ob)
        {
            InitializeComponent();
            this.MdiParent = ob;
        }

        private void Form16_Load(object sender, EventArgs e)
        {

        }

        private void button1_Click(object sender, EventArgs e)
        {
            try
            {
                OleDbConnection con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                con.Open();
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from Blood where Blood_Group='" +
comboBox3.Text + "' and RH_Factor='"+comboBox4.Text+"'", con);

                OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                DataSet ds = new DataSet();
                odp.Fill(ds, "Blood");
                dataGridView1.DataSource = ds.Tables["Blood"];
                con.Close();
            }
            catch (Exception eee)
            {
                MessageBox.Show("Connection to database failed",
"Connection Failed", MessageBoxButtons.OK);
            }
        }
    }
}
```

17.FORM17.cs

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

namespace Health_Care_Product
{
    public partial class Form17 : Form
    {
         public Form17(Form2 ob)
        {
            InitializeComponent();
            this.MdiParent = ob;
        }

        private void Form17_Load(object sender, EventArgs e)
        {
            OleDbConnection con = null;

            try
            {
                con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                con.Open();
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from Patient", con);
                DataSet ds = new DataSet();
                odp.Fill(ds, "Patient");
                foreach (DataRow dr in ds.Tables["Patient"].Rows)
                {
                    comboBox3.Items.Add(dr[0]);


                }
```

```
            }
            catch (Exception err)
            {
                MessageBox.Show(err.Message, "Connection Failed",
MessageBoxButtons.OK);

            }
            finally
            {
                con.Close();
            }
        }

        private void button2_Click(object sender, EventArgs e)
        {
            this.Dispose();
        }

        private void textBox5_KeyPress(object sender,
KeyPressEventArgs e)
        {

            if ((e.KeyChar < 48 || e.KeyChar > 57 ) && e.KeyChar
!= 8)
                e.Handled = true;
        }

        private void button1_Click(object sender, EventArgs e)
        {
            OleDbConnection con = null;
            if (comboBox3.Text == "" || textBox1.Text == "" ||
textBox5.Text == "" || dateTimePicker1.Text == "")
            {
                MessageBox.Show("Please Fill empty Fields",
"Error", MessageBoxButtons.OK);
            }
            else
            {
                try
                {
                    con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                    con.Open();
                    OleDbDataAdapter odp = new
OleDbDataAdapter("select * from PatientBill", con);
```

```
                    OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                    DataSet ds = new DataSet();
                    odp.Fill(ds, "PatientBill");
                    DataRow drow =
ds.Tables["PatientBill"].NewRow();
                    drow[1] = comboBox3.Text;
                    drow[2] = textBox1.Text;
                    drow[3] = textBox5.Text;
                    drow[4] = dateTimePicker1.Text;

                    ds.Tables["PatientBill"].Rows.Add(drow);
                    odp.Update(ds, "PatientBill");
                    MessageBox.Show("Successfully added to
patient bill.", "Successfully Added", MessageBoxButtons.OK);
                    textBox1.Text = "";
                    textBox5.Text = "";
                    comboBox3.Text = "";
                }
                catch (Exception err)
                {
                    MessageBox.Show("Connection to database
failed", "Connection Failed", MessageBoxButtons.OK);
                }
                finally
                {
                    con.Close();
                }

            }
        }

        private void groupBox1_Enter(object sender, EventArgs e)
        {

        }
    }
}
```

18.FORM18.cs

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

namespace Health_Care_Product
{
    public partial class Form18 : Form
    {
         public Form18(Form2 ob)
        {
            InitializeComponent();
            this.MdiParent = ob;
        }

        private void Form18_Load(object sender, EventArgs e)
        {
            OleDbConnection con = null;

            try
            {
                con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                con.Open();
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from Patient", con);
                DataSet ds = new DataSet();
                odp.Fill(ds, "Patient");
                foreach (DataRow dr in ds.Tables["Patient"].Rows)
                {
                    comboBox3.Items.Add(dr[0]);


                }
```

```
            }
            catch (Exception err)
            {
                MessageBox.Show(err.Message, "Connection Failed",
MessageBoxButtons.OK);

            }
            finally
            {
                con.Close();
            }
        }

        private void comboBox3_SelectedIndexChanged(object
sender, EventArgs e)
        {
            int recno = 0;
            int s = 0;
            try
            {
                OleDbConnection con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                con.Open();
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from PatientBill where PID='"
+comboBox3.Text+ "'", con);

                OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                DataSet ds = new DataSet();
                odp.Fill(ds, "PatientBill");
                dataGridView1.DataSource =
ds.Tables["PatientBill"];
                foreach (DataRow dr in
ds.Tables["PatientBill"].Rows)
                {


                    s = s +
Convert.ToInt32(ds.Tables["PatientBill"].Rows[recno][3]);
                    recno++;


                }
                label2.Text = "Total Bill : " + s.ToString();
```

```
                con.Close();
            }
            catch (Exception eee)
            {
                MessageBox.Show(eee.Message, "Connection Failed",
MessageBoxButtons.OK);
            }

        }
    }
}
```

19.FORM19.cs

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

namespace Health_Care_Product
{
    public partial class Form19 : Form
    {
        public Form19(Form2 ob)
        {
            InitializeComponent();
            this.MdiParent = ob;
        }

        private void Form19_Load(object sender, EventArgs e)
        {
            OleDbConnection con = null;

            try
            {
```

```
                con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                con.Open();
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from Patient", con);
                DataSet ds = new DataSet();
                odp.Fill(ds, "Patient");
                foreach (DataRow dr in ds.Tables["Patient"].Rows)
                {
                    comboBox3.Items.Add(dr[0]);


                }

            }
            catch (Exception err)
            {
                MessageBox.Show(err.Message, "Connection Failed",
MessageBoxButtons.OK);

            }
            finally
            {
                con.Close();
            }
        }

        private void comboBox3_SelectedIndexChanged(object
sender, EventArgs e)
        {
            int recno = 0;
            int s = 0;
            try
            {
                OleDbConnection con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                con.Open();
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from PatientBill where PID='" +
comboBox3.Text + "'", con);

                OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                DataSet ds = new DataSet();
```

```
                odp.Fill(ds, "PatientBill");
                dataGridView1.DataSource =
ds.Tables["PatientBill"];
                foreach (DataRow dr in
ds.Tables["PatientBill"].Rows)
                {


                    s = s +
Convert.ToInt32(ds.Tables["PatientBill"].Rows[recno][3]);
                    recno++;


                }
                label2.Text = "Total Bill : " + s.ToString();
                con.Close();
            }
            catch (Exception eee)
            {
                MessageBox.Show(eee.Message, "Connection Failed",
MessageBoxButtons.OK);
            }


        }

        private void button1_Click(object sender, EventArgs e)
        {
            int recno = 0;
            string ss = "false";
            try
            {

                OleDbConnection con = new
OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Hospital.mdb;");
                con.Open();
                OleDbCommand com = new OleDbCommand("select BedNo
from Patient where PID="+comboBox3.Text, con);
               string idc=(string)com.ExecuteScalar();
                OleDbCommand DBCom1 = new OleDbCommand("Delete
from Patient where PID="+comboBox3.Text, con);
                DBCom1.ExecuteNonQuery();
                OleDbCommand DBCom2 = new OleDbCommand("update
BedTable set Status='" + ss + "' where BedNo="+idc, con);
                DBCom2.ExecuteNonQuery();
```

```
                /*
                OleDbDataAdapter odp = new
OleDbDataAdapter("select * from Patient where PID='" +
comboBox3.Text + "'", con);

                OleDbCommandBuilder bul = new
OleDbCommandBuilder(odp);
                DataSet ds = new DataSet();
                odp.Fill(ds, "Patient");
                 OleDbCommand com = new OleDbCommand("select
BedNo from Patient where PID='"+comboBox3.Text+"'", con);
                    Int32 idc;
                    idc = (Int32)com.ExecuteScalar();

                foreach (DataRow dr in ds.Tables["Patient"].Rows)
                {


                    if (dr[0].ToString()
==comboBox3.Text.ToString())
                    {


ds.Tables["Patient"].Rows[recno].Delete();
                        odp.Update(ds, "Patient");
                        break;
                    }
                    recno++;


                }
            OleDbDataAdapter odp1 = new OleDbDataAdapter("select
* from BedTable where BedNo='" + idc.ToString() + "'", con);

                OleDbCommandBuilder bul1 = new
OleDbCommandBuilder(odp1);
                DataSet ds1 = new DataSet();
                odp1.Fill(ds1, "Patient");
                recno = 0;
                foreach (DataRow dr in
ds1.Tables["BedTable"].Rows)
                {
                    if (dr[0].ToString() == idc.ToString())
                    {
```

```
                        ds1.Tables["BedTable"].Rows[recno][1]
="false";
                        odp1.Update(ds1, "BedTable");
                        break;
                    }
                    recno++;
                }
                */
                con.Close();
                MessageBox.Show("Patient Successfully
Discharged", "Successfully Discharged", MessageBoxButtons.OK);
            }
            catch (Exception eee)
            {
                MessageBox.Show(eee.Message, "Connection Failed",
MessageBoxButtons.OK);
            }

        }
    }
}
```

20.FORM20.cs

```
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 Health_Care_Product
{
    public partial class Form20 : Form
    {
        public Form20(Form2 ob)
        {
            InitializeComponent();
            this.MdiParent = ob;
        }
```

```
        private void label7_Click(object sender, EventArgs e)
        {

        }

        private void Form20_Load(object sender, EventArgs e)
        {

        }
    }
}
```

6.TESTING

Introduction

The aim of the testing process is to identify all defects existing in a software product. For most practical systems, even after satisfactorily carrying out the testing phase, it is not possible to guarantee that the software is error free. This is because of the fact that it is not possible to guarantee that the software is error free. This is because of the fact that the input data domain of most software products is very large. It is not practical to test the software exhaustively with respect to each value that the input data may assume. We must remember that testing does expose many defects existing in a software product. Therefore, we can safely conclude that testing provides a practical way of reducing defects in a system and increasing the user' confidence in a developed system.

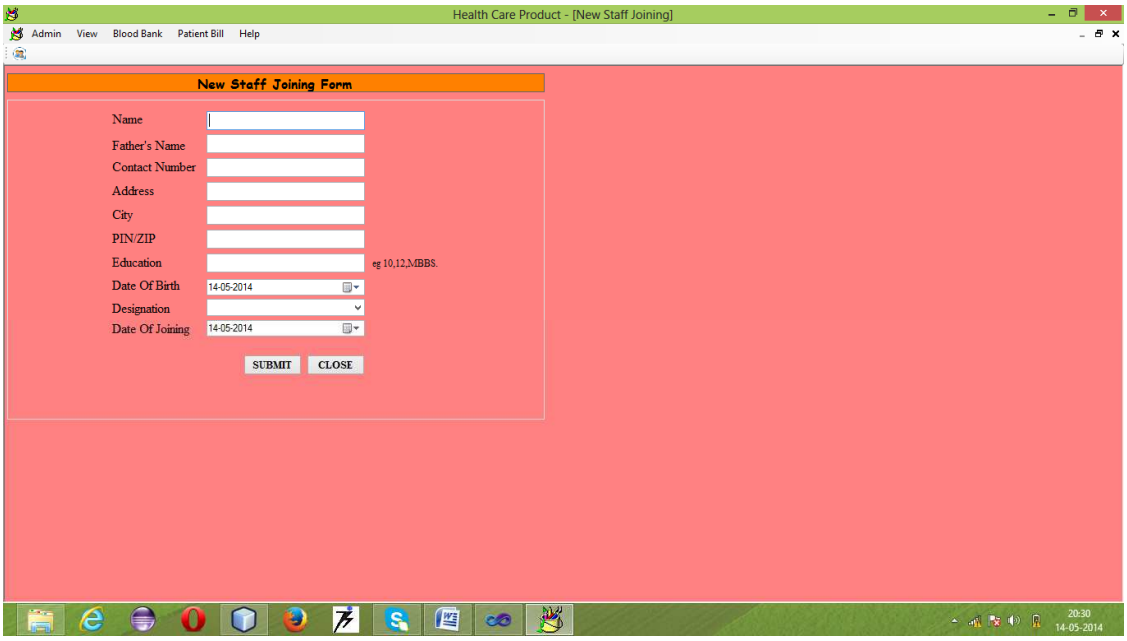
Testing a program consists of subjecting the program to a set of test inputs (or test cases) and observing if the program behaves as expected. If the program fails to behaves as expected, then the conditions under which failure occurs are noted for later debugging and correction.

The following are some commonly used terms associated with testing–

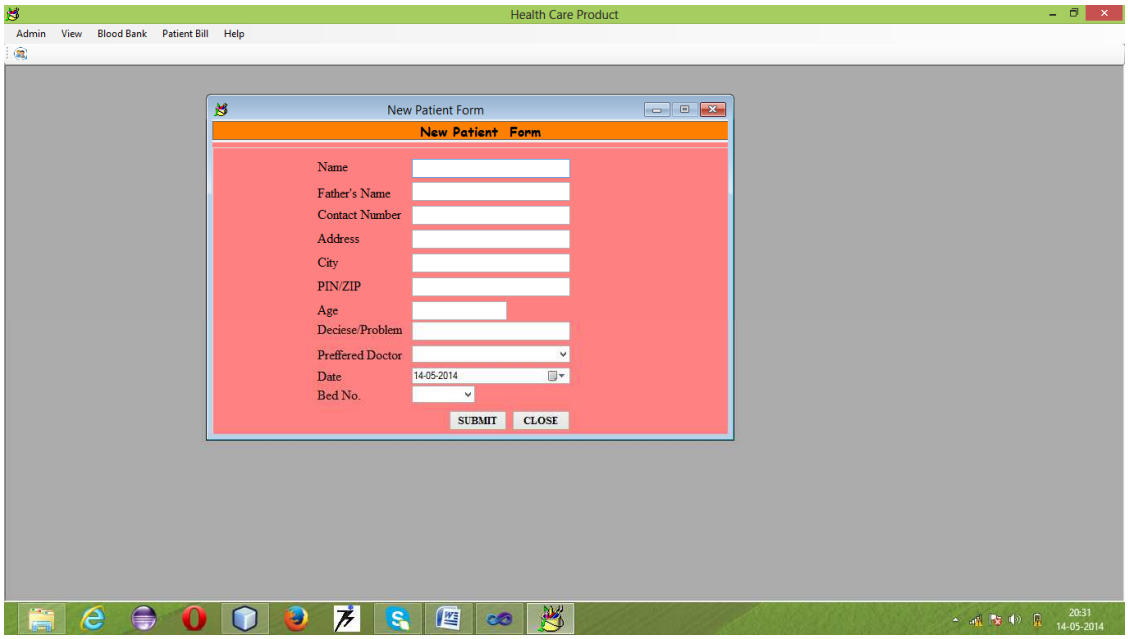
- A failure is a manifestation of an error (or defect or bug). But the mere presence of an error may not necessarily lead to a failure.
- A test case is the Triplet [I,S,O], where [I] is the data input to the system, [S] is the state of the system at which the data is Input, [O] is the expected Output of the system.
- A test suit is the set of all cases with which a given software product is to be tested.

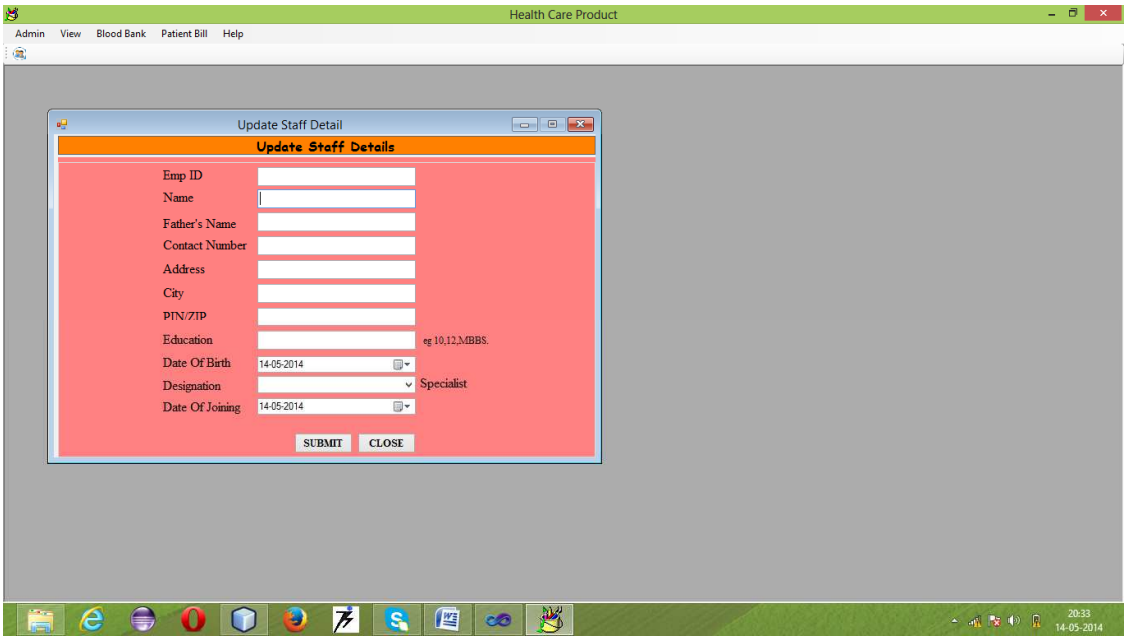
7.SNAPSHOTS

Staff Joining Form

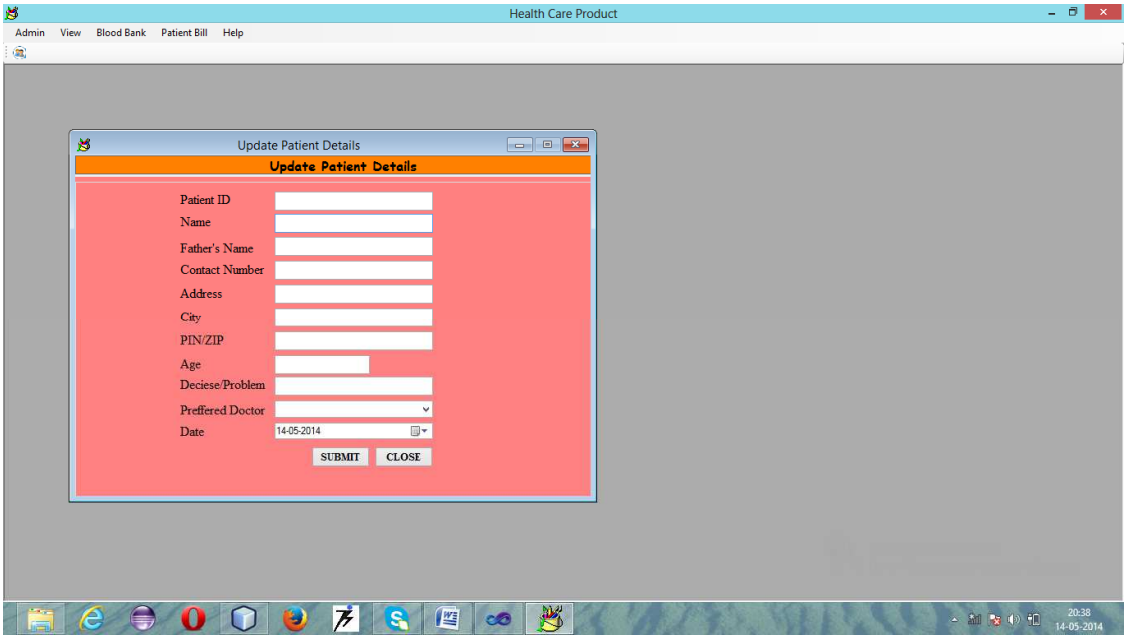


New Patient Form

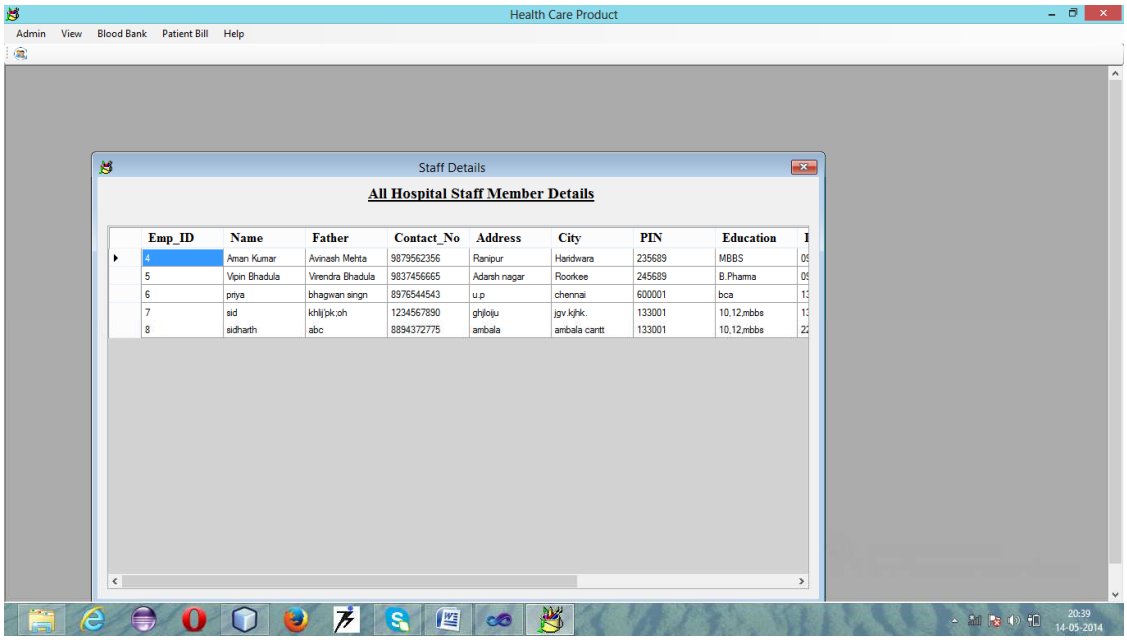




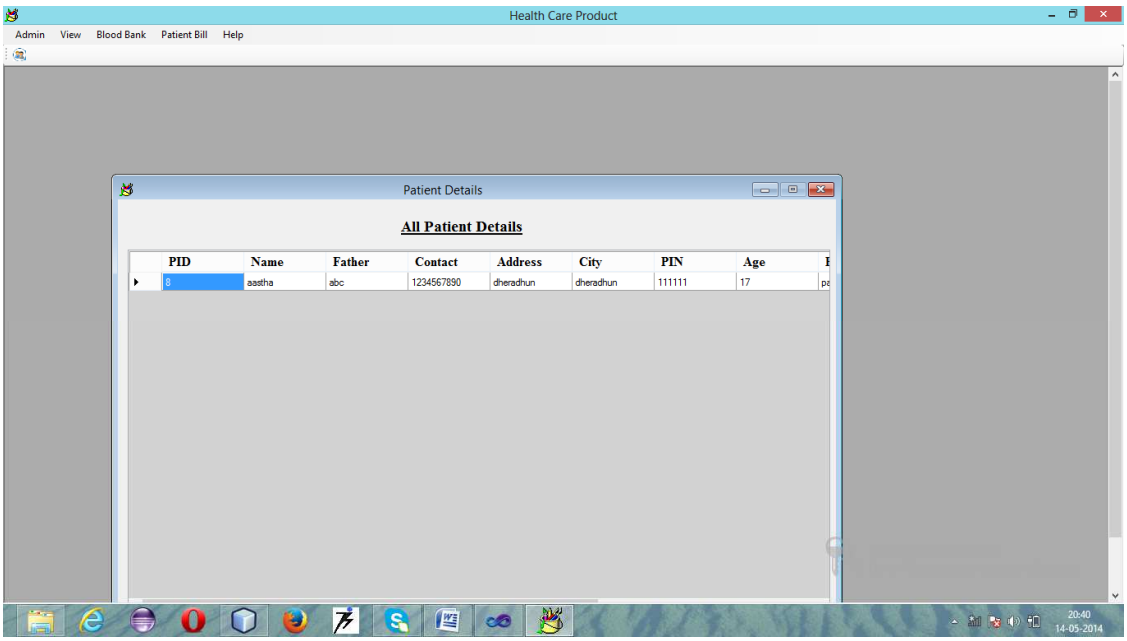
Update Staff Form



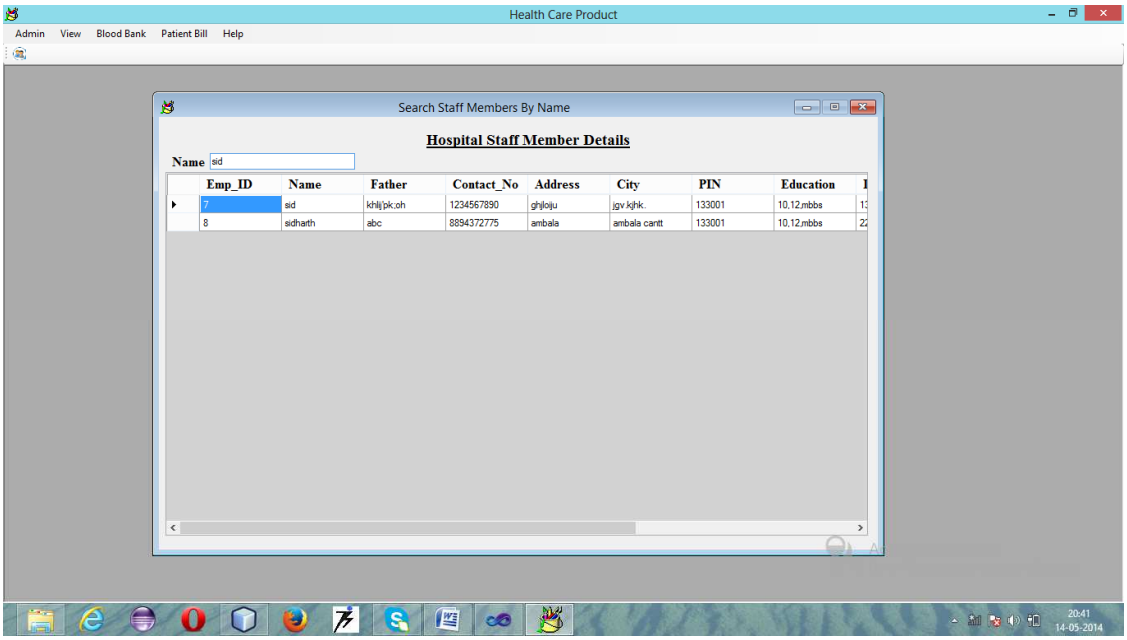
Update Patient Form



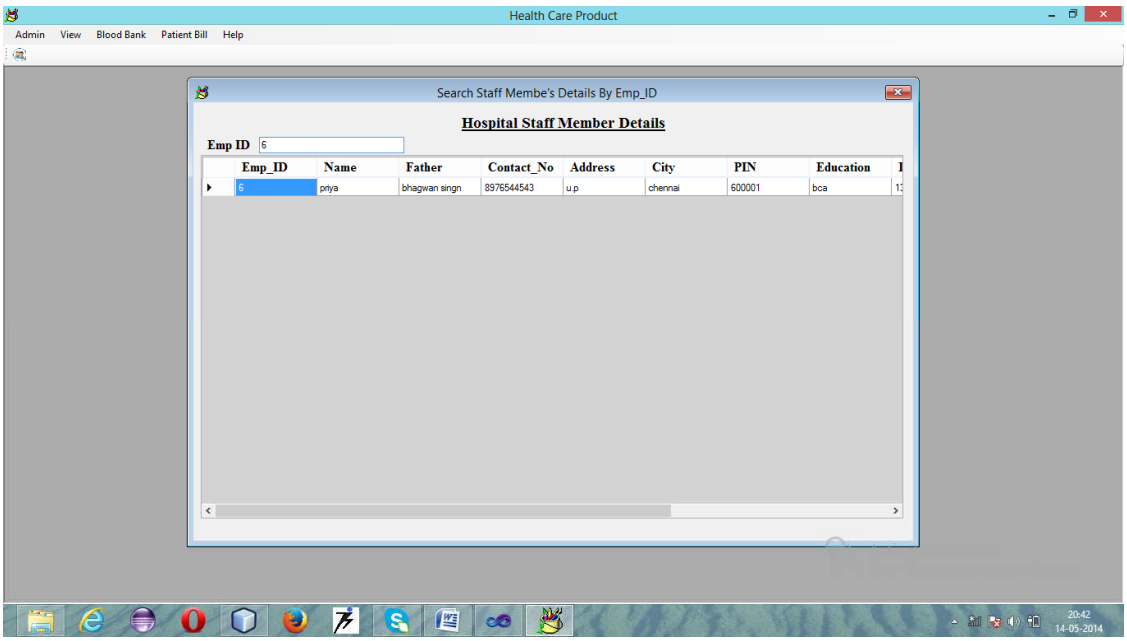
Staff Member Details



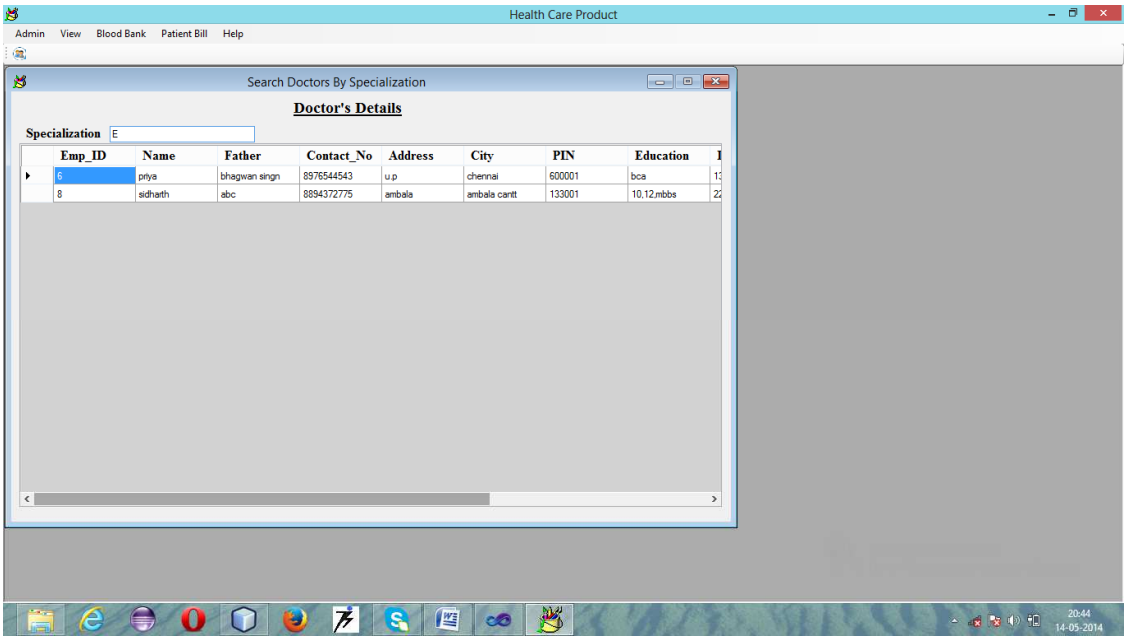
Patient Details



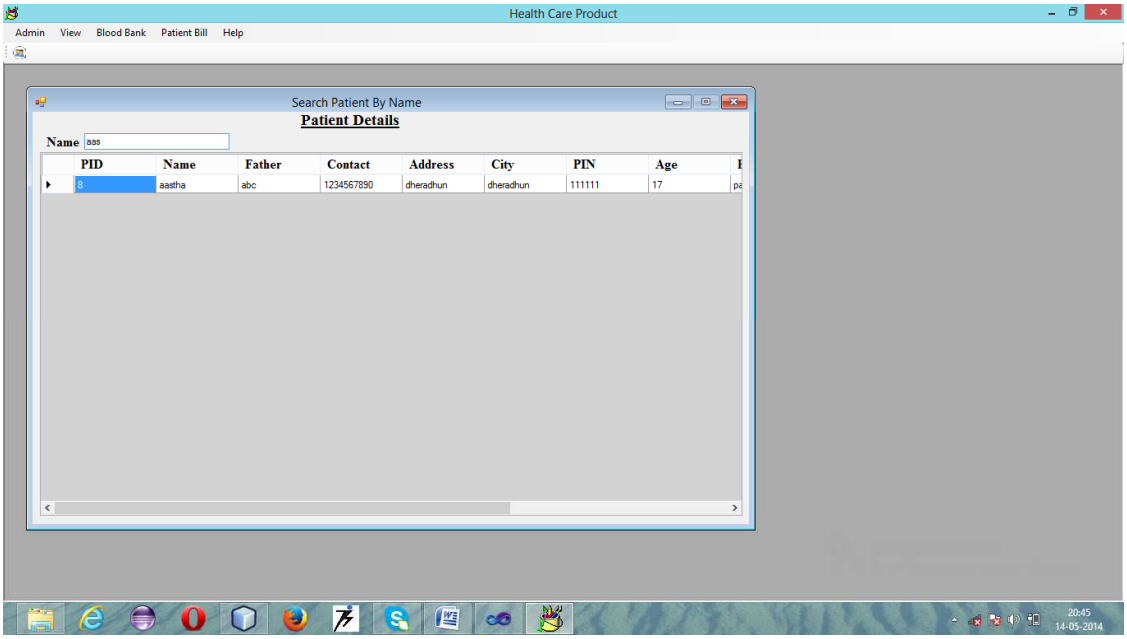
Search Staff Members by name



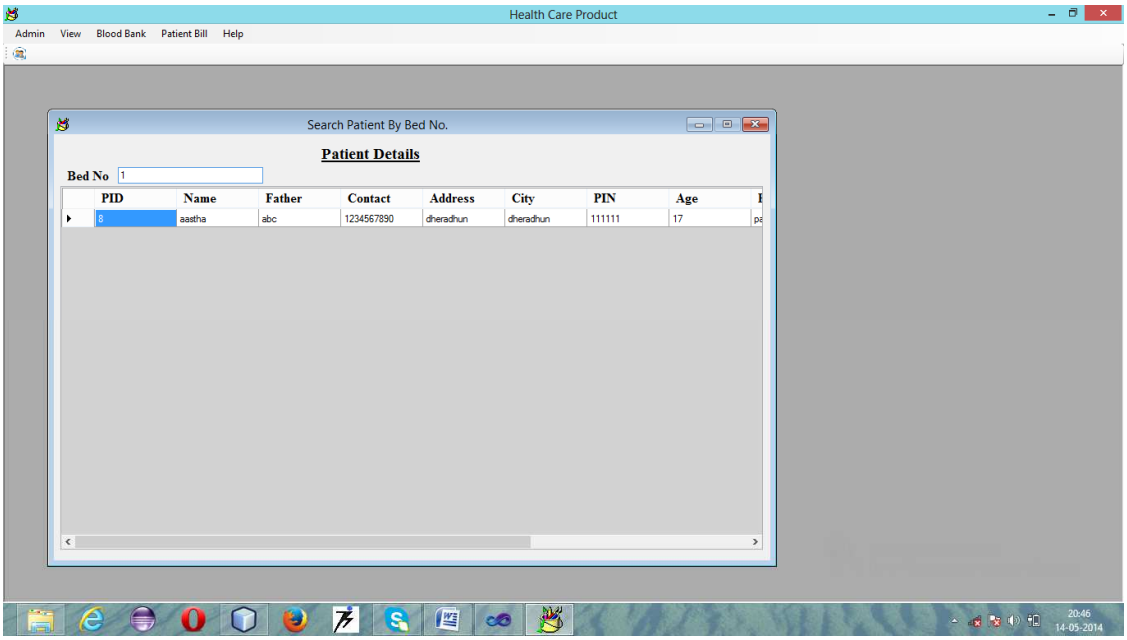
Search Staff Members by Emp id



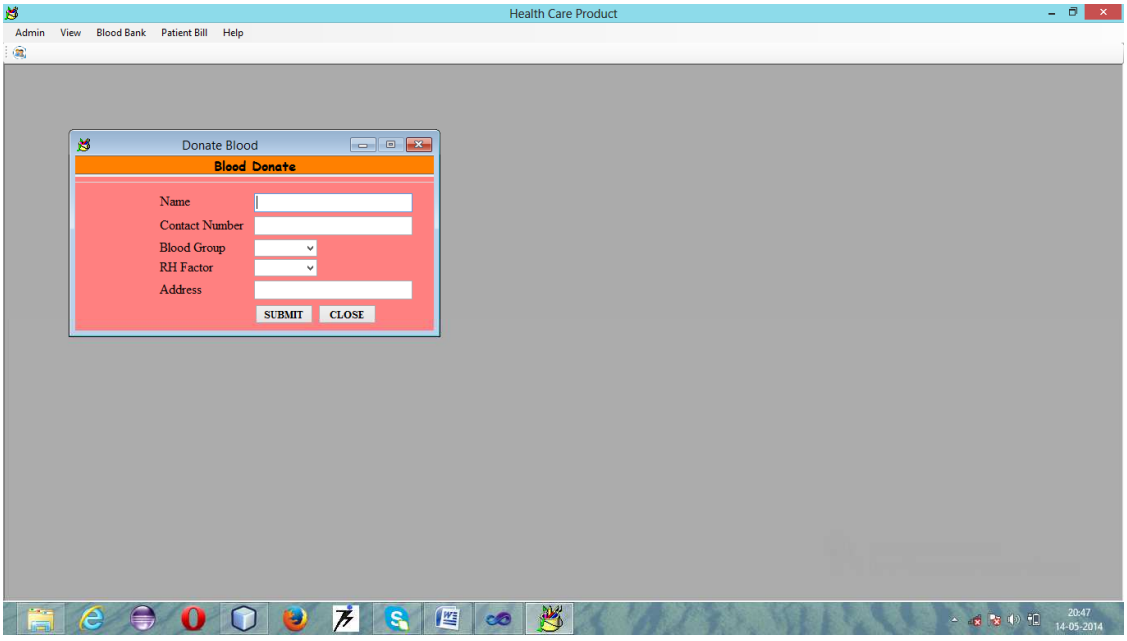
Search Staff Members by spelization



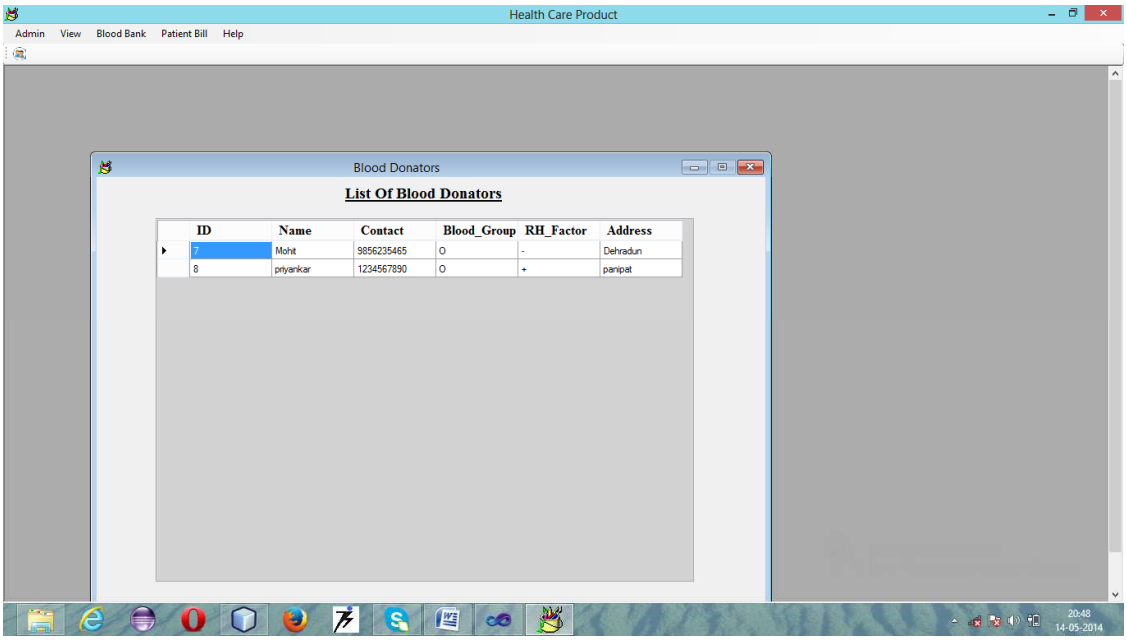
Search Patients by name



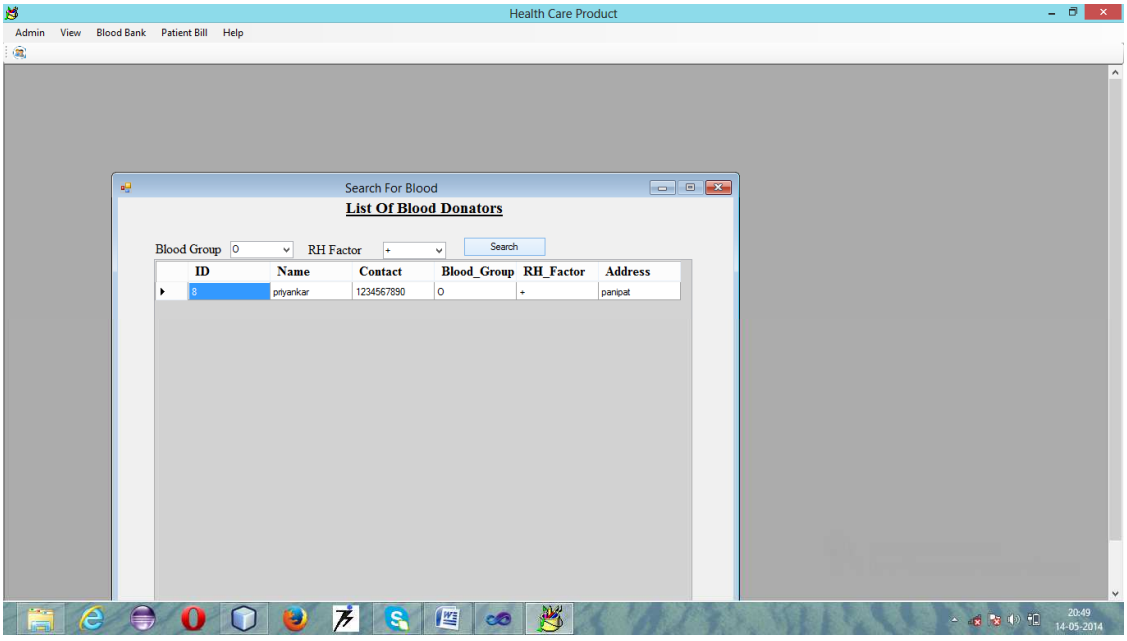
Search Patients by bed no.



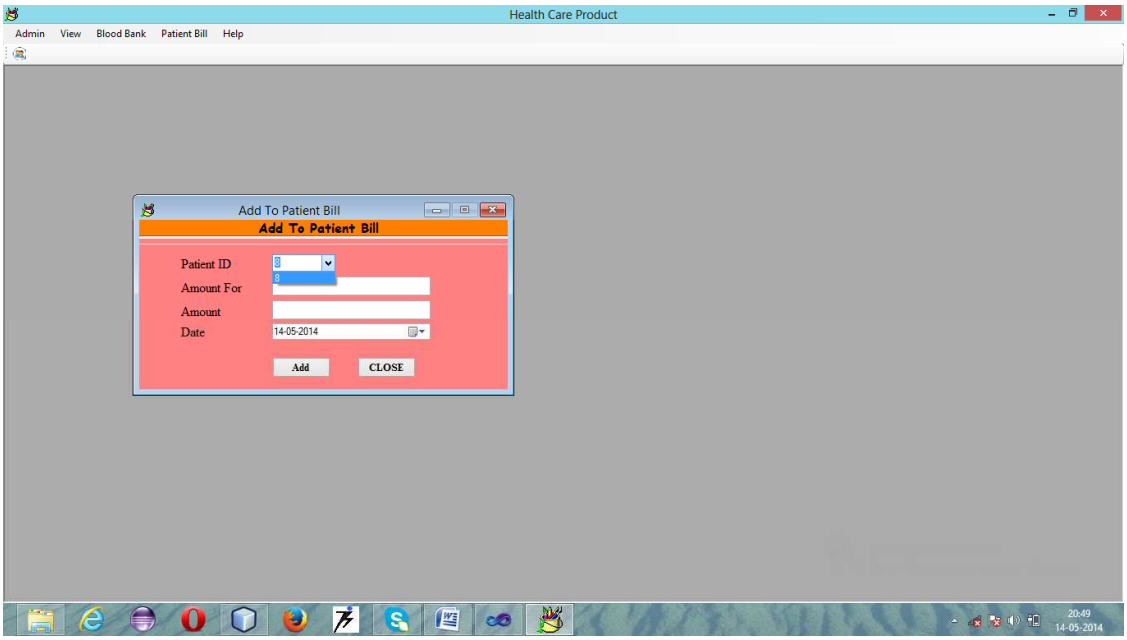
Blood DonateForm



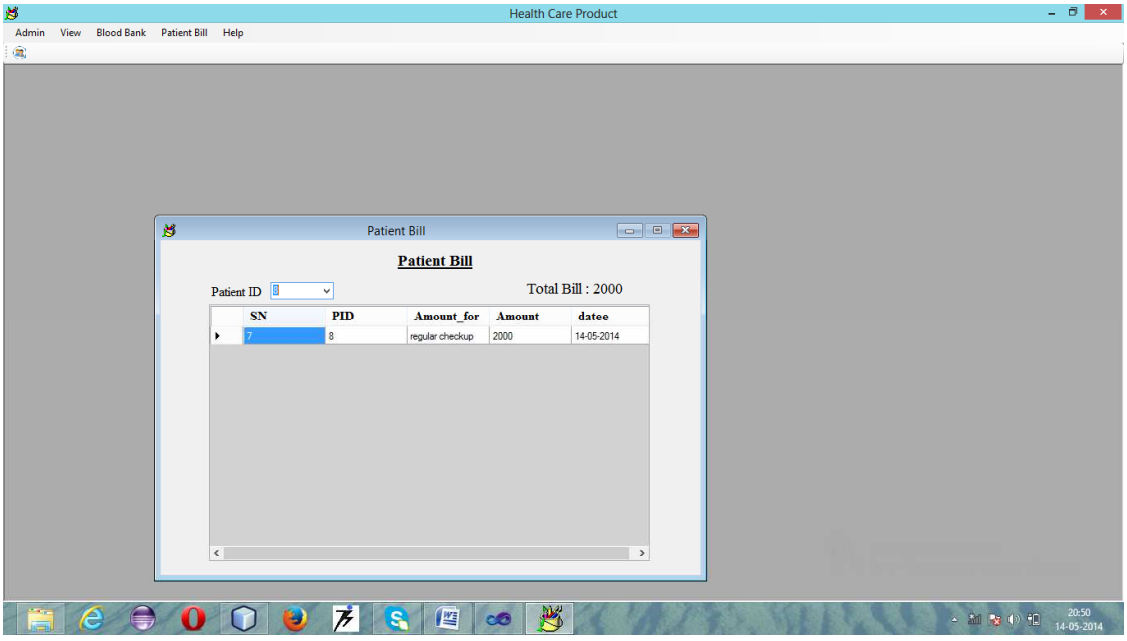
View Blood Donators



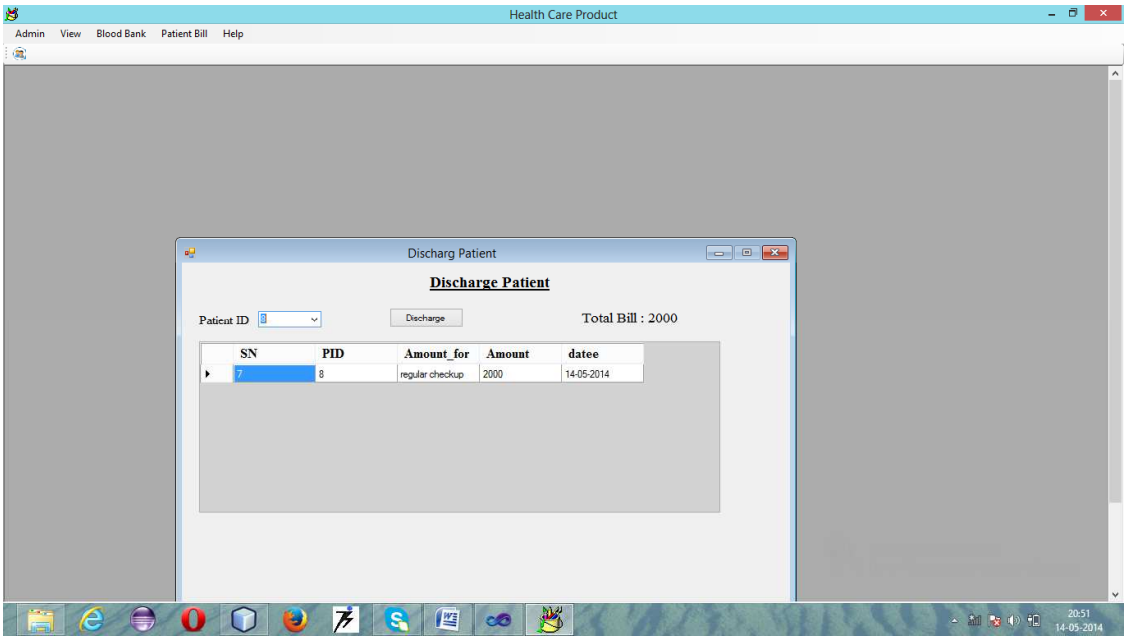
Search for Blood



Add to Patients Bill Form



View Patient Bill



Discharge Patient

8.1.Introduction to ASP.NET

ASP.NET is more than the next version of Active Server Pages (ASP); it provides a unified Web development model that includes the services necessary for developers to build enterprise-class Web applications. While ASP.NET is largely syntax compatible with ASP, it also provides a new programming model and infrastructure for more scalable and stable applications that help provide greater protection. You can feel free to augment your existing ASP applications by incrementally adding ASP.NET functionality to them.
ASP.NET is a compiled, .NET-based environment; you can author applications in any .NET compatible language, including Visual Basic .NET, C#, and JScript .NET. Additionally, the entire .NET Framework is available to any ASP.NET application. Developers can easily access the benefits of these technologies, which include the managed common language runtime environment, type safety, inheritance, and so on. ASP.NET has been designed to work seamlessly with WYSIWYG HTML editors and other programming tools, including Microsoft Visual Studio .NET. Not only does this make Web development easier, but it also provides all the benefits that these tools have to offer, including a GUI that developers can use to drop server controls onto a Web page and fully integrated debugging support.

9.2) Major tools of Asp.Net:
1) ActiveX Data Objects (ADO.NET)
2) AJAX
3) Cascading style sheet
4) Site Map
5) Session
6) Application
7) Master Page

ActiveX Data Objects (ADO.NET)
ADO.NET (ActiveX Data Object for.NET) is a set of computer software components that programmers can use to access data and data services. It is a part of the base class library that is included with the Microsoft .NET Framework. It is commonly used by programmers to access and modify data stored in relational database systems, though it can also access data in non-relational sources. ADO.NET is sometimes considered an evolution of ActiveX Data Objects (ADO) technology, but was changed so extensively that it can be considered an entirely new product.
1) The ADO.NET Data Architecture

Data Access in ADO.NET relies on two components: DataSet and Data Provider.
DataSet: The dataset is a disconnected, in-memory representation of data. It can be considered as a local copy of the relevant portions of the database. The DataSet is persisted in memory and the data in it can be manipulated and updated independent of the database. When the use of this DataSet is finished, changes can be made back to the central database for updating. The data in DataSet can be loaded from any valid data source like Microsoft SQL server database, an Oracle database or from a Microsoft Access database.

Data Provider: The Data Provider is responsible for providing and maintaining the connection to the database. A Data Provider is a set of related components that work together to provide data in an efficient and performance driven manner. The .NET Framework currently comes with two Data Providers: the SQL Data Provider which is designed only to work with Microsoft's SQL Server 7.0 or later and the OleDb Data Provider which allows us to connect to other types of databases like Access and Oracle. Each Data Provider consists of the following component classes: The Connection object which provides a connection to the database. The Command object which is used to execute a command. The Data Reader object which provides a forward-only, read only, connected recordset.The Data Adapter object which populates a disconnected DataSet with data and performs updateData access with ADO.NET can be summarized as follows: A connection object establishes the connection for the application with the database. The command object provides direct execution of the command to the database. If the command returns more than a single value, the command object returns a DataReader to provide the data. Alternatively, the DataAdapter can be used to fill the Dataset object.

Component classes that make up the Data Providers

Connection Object

The Connection object creates the connection to the database. Microsoft Visual Studio .NET provides two types of Connection classes: the SqlConnection object, which is designed specifically to connect to Microsoft SQL Server 7.0 or later, and the OleDbConnection object, which can provide connections to a wide range of database types like Microsoft Access and Oracle. The Connection object contains all of the information required to open a connection to the database.

The Command Object

The Command object is represented by two corresponding classes: SqlCommand and OleDbCommand. Command objects are used to execute commands to a database across a data connection. The Command objects can be used to execute stored procedures on the database, SQL commands, or return complete tables directly. Command objects provide three methods that are used to execute commands on the database:

ExecuteNonQuery: Executes commands that have no return values such as *INSERT, UPDATE or DELETE*

ExecuteScalar: Returns a single value from a database query

ExecuteReader: Returns a result set by way of a DataReader object

The DataReader Object: The DataReader object provides a forward-only, read-only, connected stream recordset from a database. Unlike other components of the Data Provider, DataReader objects cannot be directly instantiated. Rather, the DataReader is returned as the result of the Command object's ExecuteReader method. The SqlCommand.ExecuteReader method returns a SqlDataReader object, and the OleDbCommand.ExecuteReader method returns an OleDbDataReader object. The DataReader can provide rows of data directly to application logic when you do not need to keep the data cached in memory.

The DataAdapter Object: The DataAdapter is the class at the core of ADO .NET's disconnected data access. It is essentially the middleman facilitating all communication between the database and a DataSet. The DataAdapter is used either to fill a DataTable or DataSet with data from the database with it's Fill method. After the memory-resident data has been manipulated, the DataAdapter can commit the changes to the database by calling the Update method. The DataAdapter provides four properties that represent database commands:

SelectCommand
InsertCommand
DeleteCommand
UpdateCommand

When the Update method is called, changes in the DataSet are copied back to the database and the appropriate InsertCommand, DeleteCommand, or UpdateCommand is executed.

AJAX

AJAX = Asynchronous JavaScript and XML.

AJAX is a technique for creating fast and dynamic web pages.

AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page.

Classic web pages, (which do not use AJAX) must reload the entire page if the content should change.

Examples of applications using AJAX: Google Maps, Gmail, Youtube, and Facebook tabs

.**Cascading style sheet CSS**

- CSS stands for Cascading Style Sheets
- Styles define how to display HTML elements
- Styles were added to HTML 4.0 to solve a problem
- External Style Sheets can save a lot of work
- External Style Sheets are stored in CSS files

Syntax<asp:webcontrol id="id" CssClass="style" runat="server" />

CAPTCHA CODE

CAPTCHA stands for "completely automated public Turing test to tell computers and humans apart."

Enter the code shown above:

Submit

What it means is, a program that can tell humans from machines using some type of generated test. A test most people can easily pass but a computer program cannot.

You've probably encountered such tests when signing up for an online email or forum account. The form might include an image of distorted text, like that seen above, which you are required to type into a text field.

The idea is to prevent spammers from using web bots to automatically post form data in order to create email accounts (for sending spam) or to submit feedback comments or guestbook entries containing spam messages. The text in the image is usually distorted to prevent the use of OCR (optical character reader) software to defeat the process. Hotmail, PayPal, Yahoo and a number of blog sites have employed this technique.

This article demonstrates how to create such an image and employ it within an ASP.NET web form.

SITEMAP

Sitemaps and breadcrumbs (SiteMapPath) are useful and easy to implement for a static site with a sitemap file. For dynamic sites, something as simple seems to get much more complicated. When I started reading about sitemaps for dynamic sites, I found a common approach: generate a static site map for the whole website from, for example, a data source. Re-generate periodically. Use the XmlSiteMapProvider. Cache. The technique is described in this CodeProject article.

The SiteMap has several functions:

- It provides the root node of the site navigation hierarchy (there can be only one root node).
- It identifies which site map provider is the principal, or default, provider.
- It keeps track of all the provider objects that are used to create the SiteMap.

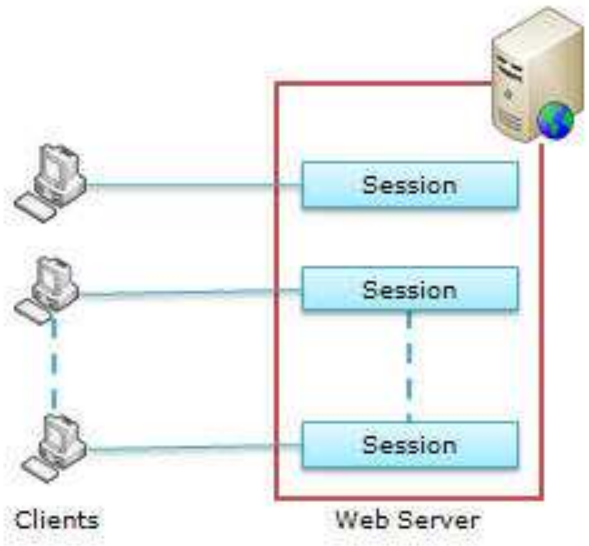
Session

ASP.NET Session state provides a place to store values that will persist across page requests. Values stored in Session are stored on the server and will remain in memory until they are explicitly removed or until the Session expires.

Storing and retrieving a value in the Session is as simple as:VB

Session("Name") = "John Doe"'orSession.Add("Name","John Doe")'retrieving

Dim Name As String = Session("Name")



Application

The life cycle of an ASP.NET application starts with a request sent by a browser to the Web server (for ASP.NET applications, typically IIS). ASP.NET is an ISAPI extension under the Web server. When a Web server receives a request, it examines the file-name extension of the requested file, determines which ISAPI extension should handle the request, and then passes the request to the appropriate ISAPI extension. ASP.NET handles file name extensions that have been mapped to it, such as .aspx, .ascx, .ashx, and .asmx.

GridView

The GridView control is a feature rich and versatile control used to accept, display, and edit data on a web page. It is a commonly used control in ASP.Net web applications.
To use a GridView control a DataSource control has to be attached to the GridView control. The property DataSourceID of the GridView control binds the GridView control to the DataSource control and allows paging, sorting and database operations with the DataSource.
There are four popular DataSource controls and Accessdatasource control is one of them. We use Accessdatasource control, to attach the GridView control to the Access database. The SqlDataSource control contains ConnectionString, SelectCommand, UpdateCommand and DeleteCommand properties. The ConnectionString property connects the DataSource (SqlDataSource) control to the database and properties such as SelectCommand, InsertCommand, UpdateCommand, DeleteCommand contains SQL statements that are executed when we display, insert and delete records using the GridView control.

9.CONCLUSION

The task given to us performed by keeping in mind the goals we have to achieve, these are to provide service-friendliness. This project "**Health Care Product**" is mainly useful for any individual that wants to get current information about patients, doctors and other useful information from hospital like blood donor details. It also makes it easy to keep a track of data of both the patients and the doctors and makes it easy to search for any patient or doctor using their ids, names, departments or bed numbers.

The "**Health Care Product**" is excepted to function as per the project requirements and we expect that it will satisfy requirements of a hospital in general.

10. REFRENCES

11.1) Internet:

1. http://www.w3schools.com/asp/default.asp
2. http://www.softpedia.com/get/Programming/Coding-languages-Compilers/Visual-Studio-Express-Editions.shtml

11.2) Books

1.Using Black Book ASP.Net
2. Database Systems by C.J. Date, Pearson, Edi 8th
3. Software Engineering by Roger S. Pressman, McGraw- Hill Publications.