

**Bangladesh University of Business and Technology**

**(BUBT)**

**Project report on Blood Bucket**

Course No : CSE-200

Course Title : SOFTWARE DEVELOPMENT II

|  |  |  |  |
| --- | --- | --- | --- |
| Submitted by :  Name :Rashedul ISlam  ID No : 16172103183  Name : Nahid Hassan  ID No : 16172103382  Name :Md. Hafiz Al Foisal  ID No : 16172103116  Intake : 35  Section : 06  Program: B.Sc. in CSE |  |  | Submitted To : Name : MR. NAHID ANWAR Lecturer Department of CSE,  Bangladesh University of Business And Technology (BUBT) |
|  |  |  |  |
|  |  |  |  |

Submission date: 27/02/2019

Table of Contents

1. [Introduction: 1](#_Toc2040066)
2. [Objectives: 2](#_Toc2040067)
3. [Scope: 2](#_Toc2040068)
4. [Donor: 3](#_Toc2040069)
5. [Administrator: 3](#_Toc2040070)
6. [Proposed system: 4](#_Toc2040071)
7. [Limitation: 4](#_Toc2040072)
8. [Data-Flow Diagrams: 5](#_Toc2040073)
9. [Context Level – DFD: 6](#_Toc2040074)
   1. [I Level – DFD: 6](#_Toc2040075)
   2. [II Level – DFD: 7](#_Toc2040076)
10. [DATABASE DESIGN: 7](#_Toc2040077)
11. [Data Dictionary: 7](#_Toc2040078)
12. [Register Table: 7](#_Toc2040079)
13. [Detailed Description: 7](#_Toc2040080)
14. [Donor Registration: 8](#_Toc2040081)
15. [Modifying Donor Information: 8](#_Toc2040082)
16. [Donor Search: 8](#_Toc2040083)
17. [Life Saving Contacts: 9](#_Toc2040084)
18. [Interface Design: 9](#_Toc2040085)
    1. [Landing page: 9](#_Toc2040086)
       1. [Code: 10](#_Toc2040087)
    2. [Blood Search: 12](#_Toc2040088)
       1. [CODE: 12](#_Toc2040089)
    3. [Registration Form: 15](#_Toc2040090)
       1. [Code: 15](#_Toc2040091)
    4. [Admin login: 17](#_Toc2040092)
       1. [Code: 18](#_Toc2040093)
    5. [Admin view: 21](#_Toc2040094)
       1. [Code: 21](#_Toc2040095)
    6. [Update Information: 24](#_Toc2040096)
       1. [Code: 24](#_Toc2040097)
19. [Introduction to System Design: 28](#_Toc2040098)
    1. [Code design: 29](#_Toc2040099)
    2. [Input Design: 30](#_Toc2040100)
20. [Implementation and Evaluation: 31](#_Toc2040101)
21. [CONCLUSION: 32](#_Toc2040102)
    1. [Reference websites: 32](#_Toc2040103)

**Blood Bucket**

A Blood Bank Management System

# Introduction:

The number of persons who are in need of blood are increasing in large number day by day. In order to help people who are in need of blood, my Blood Bank can be used effectively for getting the details of blood donors having the same blood group and within the same city. With the help of our Blood Bank people who are having the thought of donating blood gets registered in our Blood Bank giving their total details.

Our Blood Bank site is available to everyone easily. A person who likes to donate blood gives his entire details i.e., fill in the registration form and can create a username with a password by which he can modify his details if at all there are any changes in his information given before.

Our Program also helps people who are in need of blood by giving the details of the donors by searching, if at all there are no donors having the same group and within their own city they will be given the addresses with phone numbers of some contact persons in major cities who represent a club or an organization with free of cost.

The present project elucidates the following features.

 Registering the Donors

 Modification of Donor Information

 Searching a Donor

 Life Saving Contacts (in major cities)

# Objectives:

The goal of the project is to develop a web application for blood banks to manage information about their donors and blood stock.

The main objectives of this website development can be defined as follows:

1. To develop a system that provides functions to support donors to view and manage their information conveniently.

2. To maintain records of blood donors, blood donation information and blood stocks in a centralized database system.

3. To inform donors of their blood result after their donation.

4. To support searching, matching and requesting for blood convenient for administrators.

5. To provide a function to send an e-mail directly to the donor for their user account and the hospital, the availability of the blood bag.

# Scope:

This system plays a vital role in the development life cycle & it describes the complete requirement of the system. It is meant for use by the developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process.

User Requirements:

There are two internal users involved in this system. The user requirements are considered as follows:

# Donor:

1. To be able to view their donation records, including where and when they made donations, and the blood results for each, to learn of their donated blood quality and schedule their next donations.

2. To be able to view and update their personal information, including name, contact address, and phone number, to keep their donor’s information record upto-date with the blood bank.

3. To be notified of the blood results of their previous donation by e-mail, to know the success of their donation.

# Administrator:

1. To be able to create, update, delete, and query donor’s records in order to manage donor information.

2. To be able to deposit donated blood into inventory when donations are made.

3. To be able to withdraw blood from the inventory and keep a record of blood stocks to always keep count of the blood bags.

Benefits:

Blood Bank donation system can collect blood from many donators in short from various sources and distribute that blood to needy people who require blood.

To do all this we require high quality Web Application to manage those jobs.

1. Users do not have to contact the hospital to know the results of their blood donation. They can view their results through the website by logging-in with their user email and password.

2. The reports and information are kept in electronic form and can be easily maintained by the administrators, and donors may access their donation records whenever they want to.

3. The reports of donations are kept in electronic files so that they may last longer and have less chance of being lost or damaged

4. Administrators of the system can easily manage blood stock and blood withdrawals that have been requested by the hospitals.

# Proposed system:

The proposed system, Online Blood Bank site overcomes the drawbacks of the present system. The Blood Bank helps the people who are in need of a blood by giving them overall details regarding the donors with the same blood group and within their city.

The advantages of the proposed system are listed below.

 The people in need of blood can search for the donors by giving their blood group and location.

 It is very flexible and user friendly.

 The person’s time and work is reduced very much which prevails in the present system.

 Easy and Helpful.

 The people are not limited to receive or provide services in working hours of the branch only; he is serviced 24 hours a day, 7 days of week and 365 days of the year.

# Limitation:

 In this project the searching can be done for donors for majority of cities but not for every city.

 In this project the contact person’s details are given for the limited cities only.

 All the search box is defined and fixed but not custom search.

# Data-Flow Diagrams:

Blood

Bank

Blood

Donors

Blood

Seekers

Registration

Login Check

Details

Modification

Seekers

Log in first

Donor

Search

# Context Level – DFD:

## I Level – DFD:

Blood Donors

Life Saving Contacts

Donor

Registration

Donating

Blood

Donor

Search

Blood Seekers

Receiving

Services

**Register**

Donor

Not Found

Receive Donor Details

Donor Found

Contact Person Found

Getting Registered

Searching For Donor

## II Level – DFD:

# 

# DATABASE DESIGN:

Database is an integrated collection of data and provides a centralized access to the data and makes possible to treat data as a separate resource. Usually centralized data managing software is called a Relational Database Management System (RDBMS). The most significant different between RDBMS and other type of Data Management is the separation of data as seen by the program and data as store of on the direct access storage device. This is the difference between logical and physical data.

# Data Dictionary:

The efficiency of an application developed using RDBMS mainly depend upon the database tables, the fields in each table and the way the tables are opened using the contents in them to retrieve the necessary information. Hence a careful selection of tables and their fields are imperative. The database tables used in this system are created keeping the above points in mind. The tables used are given below.

# Register Table:

Description: Used to Register the donor information and helpful for the people in need of blood to get details regarding the person with matching blood group and city.

Field Name Field Type Field Length

# Detailed Description:

My project Online Blood Bank is to provide services for the people who are in need of blood by getting help from the donors who are interested in donating blood for the people. This project mainly elucidates the modules such as:

 Donor Registration

 Modifying Donor Information

 Donor Search

 Life Saving Contacts (in major cities)

These modules can be explained in detail as follows:

# Donor Registration:

In this module, people who are interested in donating blood get registered in my site and give his overall details related to him, i.e. he fills in a registration form by giving the total details such as name, location, weight, blood\_group, e-mail address, password and phone number. He was also given two fields’ e-mail address and password to fill such that he was a registered donor and he can enter the login form with his e-mail address and password and can modify his details if needed.

# Modifying Donor Information:

The registered donor only is able to modify his details; only by admin. no other person can modify his details as there was a login form which restricts others from entering the e-mail address and password providing high security for the details given by the donor. If at all the donor wants to modify his details, he was forced to give his e-mail address and password to enter in. After giving the e-mail address and password it checks for the donor whether he is an existing donor or not and if the e-mail address and password matches, his information, then able to modify his total details.

# Donor Search:

The people who are in need of blood can search in our site for getting the details of donors having the same blood group and within the same city. They can directly click on the link search a donor and can select a city name as well as the blood group which he needs. He then gets the details of the donors who exist within the city and the same blood group that he has selected. If no match was are found for the city and group selected by him he gets a message ‘SORRY DONORS ARE NOT AVAILABE WITH THE FOLLOWING BLOOD GROUP AND AREA’. Before the search of donor Secker must be login first and this verified from donor database.

# Life Saving Contacts:

If at all the people in search of a donor doesn’t get any match for their area and group then they will be provided a service i.e. he will be given a Contact Person details for their nearby cities who have the details of many other donors with him. The people in search can call him and can get the details of the donors and can be provided services in this manner. But this life saving contact persons can be available only for a limited number of cities but not for all. These contact persons are the authorized persons of my blood bank.

# 

# Interface Design:

## Landing page:



### Code:

**using System;**

**using System.Collections.Generic;**

**using System.ComponentModel;**

**using System.Data;**

**using System.Drawing;**

**using System.Linq;**

**using System.Text;**

**using System.Threading.Tasks;**

**using System.Windows.Forms;**

**using System.Data.OleDb;**

**namespace foisal**

**{**

**public partial class Form1 : Form**

**{**

**public Form1()**

**{**

**InitializeComponent();**

**}**

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

**{**

**this.Hide();**

**be\_a\_donor donor = new be\_a\_donor();**

**donor.Show();**

**}**

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

**{**

**this.Hide();**

**log\_blood\_taker taker = new log\_blood\_taker();**

**taker.Show();**

**}**

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

**{**

**this.Hide();**

**Admin\_show admin = new Admin\_show();**

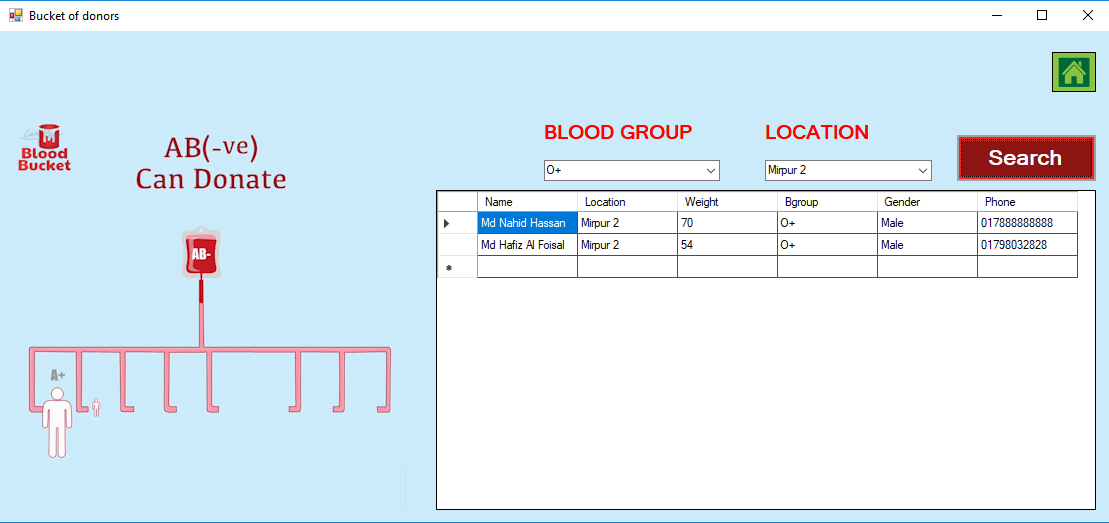
**admin.Show();**

**}**

**}**

**}**

## Blood Search:



### CODE:

**using System;**

**using System.Collections.Generic;**

**using System.ComponentModel;**

**using System.Data;**

**using System.Drawing;**

**using System.Linq;**

**using System.Text;**

**using System.Threading.Tasks;**

**using System.Windows.Forms;**

**using System.Data.OleDb;**

**namespace foisal**

**{**

**public partial class find\_donor : Form**

**{**

**OleDbConnection con = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=c:\users\user\source\repos\foisal\foisal\nahid.accdb");**

**int count = 0;**

**public find\_donor()**

**{**

**InitializeComponent();**

**}**

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

**{**

**this.Hide();**

**Form1 form1 = new Form1();**

**form1.Show();**

**}**

**private void search\_box\_Click(object sender, EventArgs e)**

**{**

**count = 0;**

**con.Open();**

**OleDbCommand cmd = con.CreateCommand();**

**cmd.CommandType = CommandType.Text;**

**cmd.CommandText = " select Name, Location, Weight, Bgroup, Gender, Phone from Table1 where Bgroup = '" + bg.Text + "' and Location = '"+ loc.Text +"' ";**

**cmd.ExecuteNonQuery();**

**DataTable dt = new DataTable();**

**OleDbDataAdapter da = new OleDbDataAdapter(cmd);**

**da.Fill(dt);**

**count = Convert.ToInt32(dt.Rows.Count.ToString());**

**dataGridView1.DataSource = dt;**

**con.Close();**

**if(count==0)**

**{**

**MessageBox.Show("Record is not found");**

**}**

**}**

**private void button3\_Click\_1(object sender, EventArgs e)**

**{**

**this.Hide();**

**Form1 form1 = new Form1();**

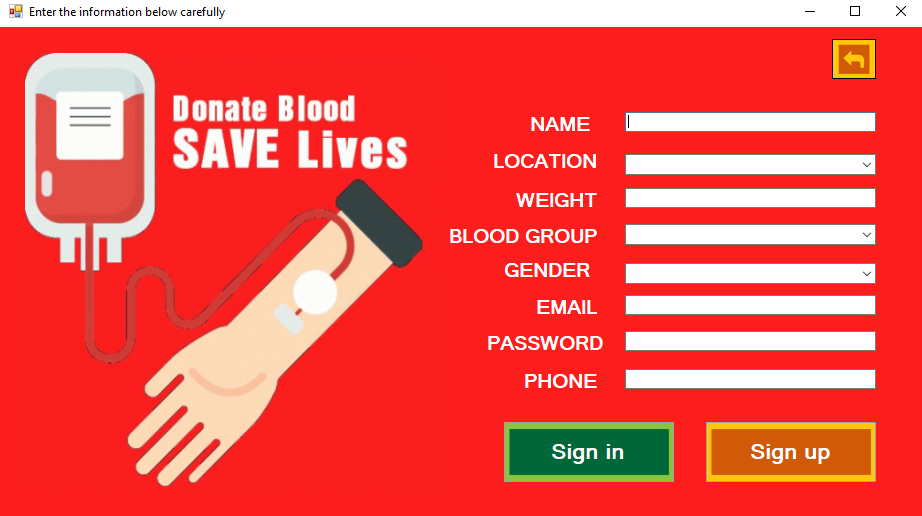
**form1.Show();**

**}**

**}**

**}**

## Registration Form:



### Code:

**using System;**

**using System.Collections.Generic;**

**using System.ComponentModel;**

**using System.Data;**

**using System.Drawing;**

**using System.Linq;**

**using System.Text;**

**using System.Threading.Tasks;**

**using System.Windows.Forms;**

**using System.Data.OleDb;**

**namespace foisal**

**{**

**public partial class be\_a\_donor : Form**

**{**

**OleDbConnection con = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=c:\users\user\source\repos\foisal\foisal\nahid.accdb");**

**public be\_a\_donor()**

**{**

**InitializeComponent();**

**}**

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

**{**

**this.Hide();**

**log\_in go = new log\_in();**

**go.Show();**

**}**

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

**{**

**con.Open();**

**OleDbCommand cmd = con.CreateCommand();**

**cmd.CommandType = CommandType.Text;**

**cmd.CommandText = "insert into Table1 values('"+ name.Text+ "','" + location.Text + "','" + weight.Text + "','" + b\_g.Text + "'," +**

**"'" + gender.Text + "','" + email.Text + "','" + password.Text + "','" + phone.Text + "','donor')";**

**cmd.ExecuteNonQuery();**

**con.Close();**

**MessageBox.Show("You are successfully registered as a blood donor.");**

**this.Hide();**

**log\_blood\_taker go = new log\_blood\_taker();**

**go.Show();**

**}**

**private void button7\_Click(object sender, EventArgs e)**

**{**

**this.Hide();**

**Form1 form1 = new Form1();**

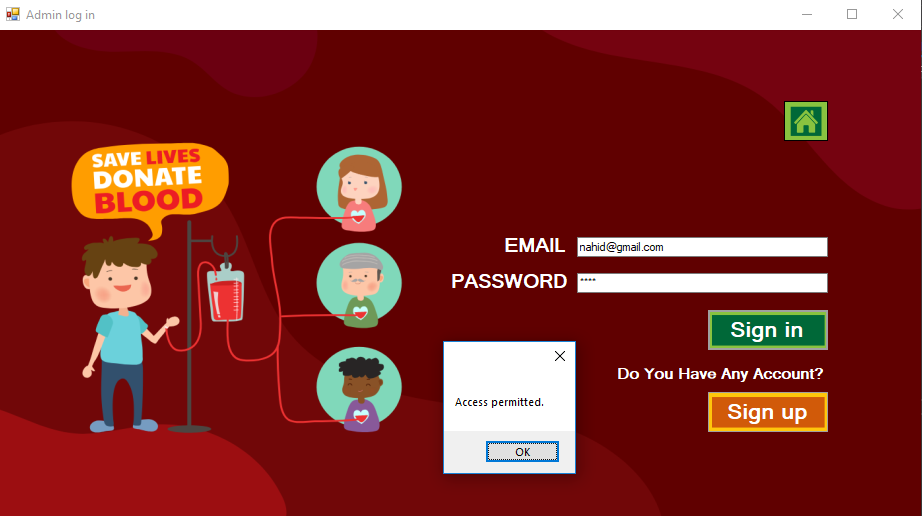
**form1.Show();**

**}**

**}**

**}**

## Admin login:



### Code:

**using System;**

**using System.Collections.Generic;**

**using System.ComponentModel;**

**using System.Data;**

**using System.Drawing;**

**using System.Linq;**

**using System.Text;**

**using System.Threading.Tasks;**

**using System.Windows.Forms;**

**using System.Data.OleDb;**

**namespace foisal**

**{**

**public partial class log\_in : Form**

**{**

**OleDbConnection con = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=c:\users\user\source\repos\foisal\foisal\nahid.accdb");**

**int count = 0;**

**public log\_in()**

**{**

**InitializeComponent();**

**}**

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

**{**

**this.Hide();**

**be\_a\_donor donor = new be\_a\_donor();**

**donor.Show();**

**}**

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

**{**

**con.Open();**

**OleDbCommand cmd = new OleDbCommand();**

**cmd.Connection = con;**

**cmd.CommandText = " select \* from Table1 where Email = '"+ email.Text+"' and Password = '"+ password.Text +"' and Type = 'admin'";**

**OleDbDataReader reader = cmd.ExecuteReader();**

**//cmd.ExecuteNonQuery();**

**count = 0;**

**while(reader.Read())**

**{**

**count++;**

**}**

**if(count == 1)**

**{**

**MessageBox.Show("Access permitted.");**

**con.Close();**

**this.Hide();**

**Admin\_show admin = new Admin\_show();**

**admin.Show();**

**}**

**else**

**{**

**MessageBox.Show("Access denide");**

**con.Close();**

**}**

**}**

**private void button4\_Click(object sender, EventArgs e)**

**{**

**this.Hide();**

**Form1 form1 = new Form1();**

**form1.Show();**

**}**

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

**{**

**this.Hide();**

**Form1 form1 = new Form1();**

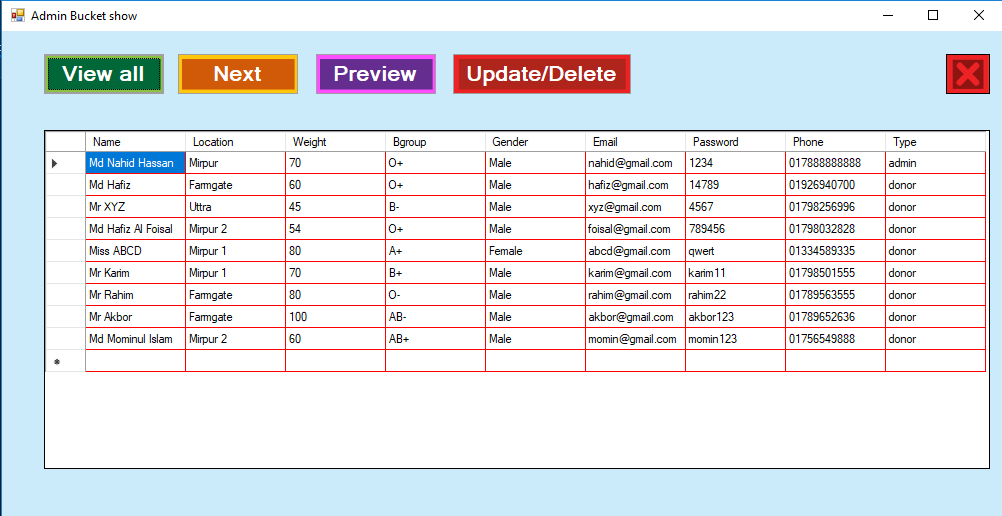
**form1.Show();**

**}**

**}**

**}**

## Admin view:

****

### Code:

**using System;**

**using System.Collections.Generic;**

**using System.ComponentModel;**

**using System.Data;**

**using System.Drawing;**

**using System.Linq;**

**using System.Text;**

**using System.Threading.Tasks;**

**using System.Windows.Forms;**

**using System.Data.OleDb;**

**namespace foisal**

**{**

**public partial class Admin\_show : Form**

**{**

**OleDbConnection con = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=c:\users\user\source\repos\foisal\foisal\nahid.accdb");**

**int count = 0;**

**public Admin\_show()**

**{**

**InitializeComponent();**

**}**

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

**{**

**count = 0;**

**con.Open();**

**OleDbCommand cmd = con.CreateCommand();**

**cmd.CommandType = CommandType.Text;**

**cmd.CommandText = " select \* from Table1 ";**

**cmd.ExecuteNonQuery();**

**DataTable dt = new DataTable();**

**OleDbDataAdapter da = new OleDbDataAdapter(cmd);**

**da.Fill(dt);**

**count = Convert.ToInt32(dt.Rows.Count.ToString());**

**dataGridView1.DataSource = dt;**

**con.Close();**

**}**

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

**{**

**int next = this.dataGridView1.CurrentRow.Index + 1;**

**this.dataGridView1.CurrentCell = this.dataGridView1.Rows[next].Cells[this.dataGridView1.CurrentCell.ColumnIndex];**

**}**

**private void button4\_Click(object sender, EventArgs e)**

**{**

**int prev = this.dataGridView1.CurrentRow.Index - 1;**

**this.dataGridView1.CurrentCell = this.dataGridView1.Rows[prev].Cells[this.dataGridView1.CurrentCell.ColumnIndex];**

**}**

**private void button6\_Click(object sender, EventArgs e)**

**{**

**this.Hide();**

**operation ope = new operation();**

**ope.Show();**

**}**

**private void button5\_Click(object sender, EventArgs e)**

**{**

**//CloseSockets();**

**this.Close();**

**log\_in login = new log\_in();**

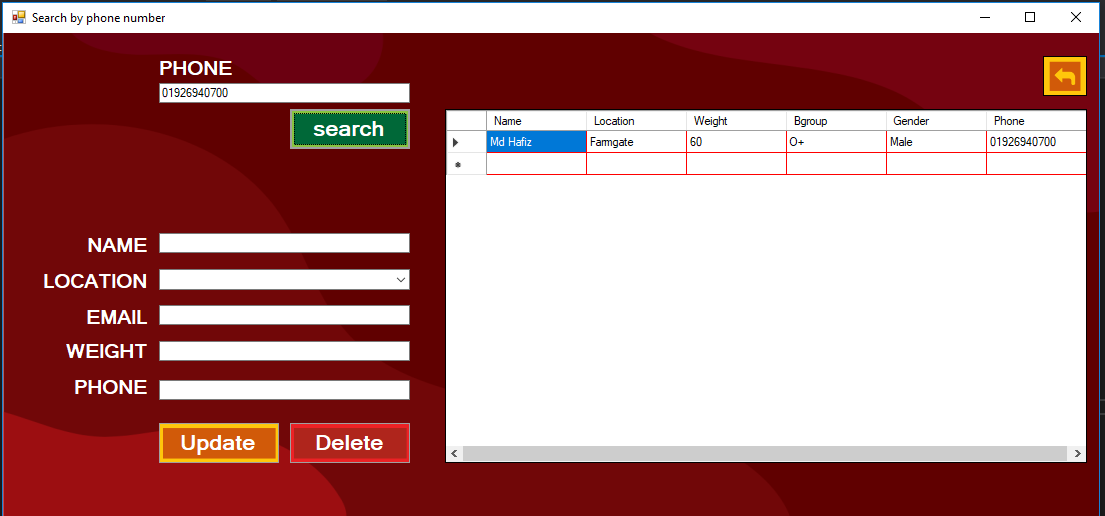
**login.Show();**

**}**

**}**

**}**

## Update Information:



### Code:

**using System;**

**using System.Collections.Generic;**

**using System.ComponentModel;**

**using System.Data;**

**using System.Drawing;**

**using System.Linq;**

**using System.Text;**

**using System.Threading.Tasks;**

**using System.Windows.Forms;**

**using System.Data.OleDb;**

**namespace foisal**

**{**

**public partial class operation : Form**

**{**

**OleDbConnection con = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=c:\users\user\source\repos\foisal\foisal\nahid.accdb");**

**int count = 0;**

**public operation()**

**{**

**InitializeComponent();**

**}**

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

**{**

**this.Hide();**

**Admin\_show a\_s = new Admin\_show();**

**a\_s.Show();**

**}**

**private void search\_Click(object sender, EventArgs e)**

**{**

**count = 0;**

**con.Open();**

**OleDbCommand cmd = con.CreateCommand();**

**cmd.CommandType = CommandType.Text;**

**cmd.CommandText = " select Name, Location, Weight, Bgroup, Gender, Email, Phone from Table1 where Phone = '" + pnumber.Text + "' ";**

**cmd.ExecuteNonQuery();**

**DataTable dt = new DataTable();**

**OleDbDataAdapter da = new OleDbDataAdapter(cmd);**

**da.Fill(dt);**

**count = Convert.ToInt32(dt.Rows.Count.ToString());**

**dataGridView1.DataSource = dt;**

**con.Close();**

**if (count == 0)**

**{**

**MessageBox.Show("Record is not found");**

**}**

**}**

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

**{**

**con.Open();**

**OleDbCommand cmd = con.CreateCommand();**

**cmd.CommandType = CommandType.Text;**

**cmd.CommandText = " delete from Table1 where Phone = '"+ pnumber.Text +"' ";**

**cmd.ExecuteNonQuery();**

**MessageBox.Show("Donor deletion successful.");**

**con.Close();**

**}**

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

**{**

**con.Open();**

**//MessageBox.Show(u\_n.Text, "Message", MessageBoxButtons.OK, MessageBoxIcon.Information);**

**OleDbCommand cmd = con.CreateCommand();**

**cmd.CommandType = CommandType.Text;**

**cmd.CommandText = " update Table1 set Name = '" + u\_n.Text + "', Location = '" + u\_l.Text + "', Email = '" + u\_e.Text + "', Weight = '" + u\_w.Text + "', Phone = '" + u\_p.Text + "' where Phone = '" + pnumber.Text + "' ";**

**cmd.ExecuteNonQuery();**

**MessageBox.Show("Donor information updated successful.");**

**con.Close();**

**u\_n.Text = "";**

**u\_l.Text = "";**

**u\_e.Text = "";**

**u\_w.Text = "";**

**u\_p.Text = "";**

**}**

**private void button7\_Click(object sender, EventArgs e)**

**{**

**this.Hide();**

**Admin\_show a\_s = new Admin\_show();**

**a\_s.Show();**

**}**

**}**

**}**

# Introduction to System Design:

System design is the process of planning a new system or to replace the existing system. Simply, system design is like the blueprint for building, it specifies all the features that are to be in the finished product.

System design phase follows system analysis phase. Design is concerned with identifying functions, data streams among those functions, maintaining a record of the design decisions and providing a blueprint the implementation phase.

Design is the bridge between system analysis and system implementation. Some of the essential fundamental concepts involved in the design of application software are:

• Abstraction

• Modularity

• Verification

Abstraction is used to construct solutions to problem without having to take account of the intricate details of the various component sub problems. Abstraction allows system designer to make step-wise refinement, which at each stage of the design may hide, unnecessary details associated with representation or implementation from the surrounding environment.

Modularity is concerned with decomposing of main module into well-defined manageable units with well-defined interfaces among the units. This enhances design clarity, which in turn eases implementation, Debugging, Testing, Documenting and Maintenance of the software product. Modularity viewed in this sense is a vital tool in the construction of large software projects.

Verification is fundamental concept in software design. A design is verifiable if it can be demonstrated that the design will result in implementation that satisfies the customer’s requirements. Verification is of two types namely.

 Verification that the software requirements analysis satisfies the customer’s needs.

 Verification that the design satisfies the requirement analysis.

Some of the important factors of quality that are to be considered in the design of application software are:

## Code design:

The purpose of code is to facilitate the identification and retrieval for items of information. A code is an ordered collection of symbols designed to provide unique identification of an entity or attribute. To achieve unique identification there must be only one place where the identified entity or the attribute can be entered in the code; conversely there must be a place in the code for everything that is to be identified. This mutually exclusive feature must be built into any coding system.

The codes for this system are designed with two features in mind. Optimum human oriented use and machine efficiency.

Length of the code range from length of one to length of five characteristics:

 The code structure is unique; ensuring that only one value of the code with a single meaning may be correctly applied to a given entity or attributes.

 The code structure is expansible allowing for growth of its set of entities and attributes.

 The code is concise and brief for recording, communication, transmission and storage efficiencies.

 They have a uniform size and format.

 The codes are simple so that the user can easily understand it.

 The codes are also versatile i.e., it is easy to modify to reflect necessary changers in condition, chart eristic and relationships of the encode entities.

 The codes are also easily storable for producing reports in a predetermined order of format.

 The codes are also stable and do not require being frequently updated thereby promoting user efficiency.

 The codes are also meaningful.

 They are also operable i.e., they are adequate for present and anticipate data processing both for machine and human use.

## Input Design:

Input design is a part of overall system design, which requires very careful attention. The main objectives of input design are:

 To produce a cost-effective method of input.

 To achieve the highest possible level of accuracy.

 To ensure that the input is acceptable to and understood by the user staff.

In this system input screens are designed very carefully so that no inaccurate data will enter the database. The data is made as easy as possible. For simplifying the data entry many facilities are given.

Each and every screen in this system is facilitated by many pushbuttons so that the user can easily work with this system.

Output Design:

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also to provide a permanent hard copy of these results for later consultation.

The various types of outputs are required by this system are given below:

 External outputs, whose destination is outside the concern and which require special attention because they, project the image of the concern.

 Internal outputs, whose destination is within the concern and which require careful design because they are the user’s main interface within the computer.

 Operation outputs, whose use is purely within the computer department, E.g., program listings, usage statistics etc,

 Interactive outputs, which involves the user in communicating directly with the computers.

# Implementation and Evaluation:

During the software-testing phase each module of software is thoroughly tested for bugs and for accuracy of output. The system developed is very user-friendly and the detailed documentation is also given to the user as online help wherever necessary. The implementation phase normally ends with the formal test involving all the components.

The entire system was developed using the C#, ASP & mysql as back end. Hence the design of the entire system is user-friendly and simple the implementation has been quite easy.

# CONCLUSION:

The Blood Donation Management System is a 24×7 system which is essential for different kinds of people like blood donation system personnel, doctors, donors, recipients and other general users. Here any person who has undergone blood test can be registered in any authorized blood bank as donor. That person can get facilities like information about blood donation system, donors and recipients. This paper facilitates services like direct access to the site to get donor’s information if there is an emergency. The goal of the paper is to present an edge for bringing mutually giving blood donors and patients (blood requesters) who need blood. The primary objective of the paper is to create an interactive blood donor, blood requesters and blood bank clinics. This web application is to be conceived in its current form as a dynamic site requiring constant updates both from the blood donors as well as the blood requesters and is to enable blood donors (volunteer) to place their profile and blood requesters (patients) to publish their requests. In future, we will develop the mobile application which will provide the users (with multimedia cell phones) the service of finding a blood donor with map interface. Here the application will consist of a map which will highlight the various blood donor’s locations and also it will give information about particular blood donors.

## Reference websites:

www.bloodzone.com

www.bloodbankservices.com