|  |
| --- |
|  |
| Blood Donor Management System |
| Synopsis |
|  |

**DIPANWITA DEY(105136520)**

|  |
| --- |
|  |

Table of Contents

[1. Introduction & Objective 3](#_Toc344229886)

[1.1 Introduction 3](#_Toc344229887)

[1.2 Objective 3](#_Toc344229888)

[3. Project Category 4](#_Toc344229889)

[4. Hardware and Software Specification 4](#_Toc344229890)

[4.1 Hardware Requirement 4](#_Toc344229891)

[4.2 Software Requirement 4](#_Toc344229892)

[5. REQUIREMENTS AND ANALYSIS 4](#_Toc344229893)

[5.1 Problem Definition 4](#_Toc344229894)

[5.2 Requirements Specification 6](#_Toc344229895)

[5.2.1 Functional Requirement 6](#_Toc344229896)

[5.2.2 Technical Specification 8](#_Toc344229897)

[5.3 Planning and Scheduling 9](#_Toc344229898)

[5.3.1 Gantt chart 9](#_Toc344229899)

[5.3.2 Tracking Gantt 9](#_Toc344229900)

[5.3.3 Pert Chart 10](#_Toc344229901)

[6. Scope of the Solution 10](#_Toc344229902)

[7. Analysis 11](#_Toc344229903)

[7.1 Context Diagram 11](#_Toc344229904)

[7.2 0-Level DFD 11](#_Toc344229905)

[7.3 1-Level DFD 12](#_Toc344229906)

[7.4 2-Level DFD 13](#_Toc344229907)

[7.5 E-R Diagram 13](#_Toc344229908)

[7.6 Class Diagram 15](#_Toc344229909)

[8. Database & Table Details 16](#_Toc344229910)

[9. Complete Structure 16](#_Toc344229911)

[9.1 Module Description 16](#_Toc344229912)

[9.2 estimation 17](#_Toc344229913)

[9.2 Data Structure 17](#_Toc344229914)

[9.3 Process Logic 17](#_Toc344229915)

[9.4 Implementation Methodology 17](#_Toc344229916)

[9.5 List of Reports 17](#_Toc344229917)

[11. Implementation of Security Mechanism at Various Levels 18](#_Toc344229918)

[12. Future Scope & Further Enhancement of the Project 18](#_Toc344229919)

[13. Bibliography 18](#_Toc344229920)

# Introduction & Objective

## Introduction

Now a days “Blood donation camp”, “Blood crisis” are very well-known term. We all know that the supply of blood is lower than the demand. As a result naturally a crisis arises. In the same time it is also true that blood bank dispose large amount of blood as because it is impossible to store or preserve blood after a certain period properly. To overcome this crisis we came up with a suggestion of having a parallel eco-system of blood bank - “Blood Donor Management System (**BDMS**)”. In this eco-system the willing blood donor will enroll with **BDMS** and whenever a patient will need blood, member of this community will go and donate blood. We will develop a software application to maintain the whole process.

## Objective



Blood Donor Management System (will be referred as **BDMS** in this document) will create a bridge between blood donor and patient which will cater critical blood requirements, hence save life. It is a software solution for managing Blood Donor in any organization. It is very inefficient to use paper books, ledgers or excel sheets, word documents to track and manage donor, patient’s requirements. Blood Donor Management System will enable users to maintain computerized records and manage patients’ needs more efficiently with help of sophisticated data management techniques and technologies. **BDMS** will have a web based client and mobile client also to enable update & query from anywhere.

# 3. Project Category

This software will follow Object Oriented Programming Paradigm and use below mentioned areas.

**OOP Language:** C#

**RDBMS**: MySQL 5.5.15

**Networking**: TCP/IP

**Applications**: Expert Systems

# 4. Hardware and Software Specification

## 4.1 Hardware Requirement

* **Disc capacity :** 10 MB of available hard disk space
* **RAM :** 1 GB (32 Bit) or 2 GB (64 Bit)
* **Processor :** 1.6GHz or faster
* DVD-ROM Drive / USB **Port**

## 4.2 Software Requirement

* Windows XP (x86) with Service Pack 3 / Windows Vista (x86 & x64) with

Service Pack 2 / Windows 7 (x86 & x64)

* Microsoft .NET 4.0

# 5. REQUIREMENTS AND ANALYSIS

## 5.1 Problem Definition

In our country blood crisis is a problem. In this situation we are trying to find out some alternative ways to solve this crisis. Blood Donor Management System is an effort to alleviate this issue.

Generally it takes long time to give service to the needy patients in traditional system, to avoid this time consuming process we are coming up with **BDMS**. Computerized **BDMS** solution will be more efficient and it will save time by enabling search, querying the information faster. We can gather information through JAVA enabled mobile devices also and store into the main Server system. We can communicate with large number of people through the web client any time anywhere. Register and Update process will take less effort.

Storing blood in blood bank is expensive and sometimes blood is wasted due to lack of maintenance. **BDMS** will manage donors and patient such a way that we can save and utilize blood for saving life.

Blood gets expired after certain duration being properly stored also. Our **BDMS** will overcome this issue by managing the need and supply in an efficient manner

The main purpose of Blood Donor Management System is to bring up a unified process of blood donation and better utilize the donated blood. Below diagram shows the different stake holders of **BDMS**.



The main features of Blood Donor Management System:

1. Secure user login and data access
2. Computerized records of Donor details and Patient requirements
3. Query & update donor information from various clients like Desktop, Mobile, Web client.
4. Generate testimonial to encourage the donor.
5. Manage the financial transactions& donations.
6. Share and update in social networking site.

.

## 5.2 Requirements Specification

### 5.2.1 Functional Requirement

#### 5.2.1.1 Add Donor

**Introduction:**

Register a new Donor.

**Input:**

Relevant Donor data like name, address, contact number, blood group, date of birth.

**Processing:**

Admin will enter the data in the **BDMS** and create a new Donor entry.

**Output:**

The **BDMS** will generate a donor I for future reference.

#### 5.2.1.2 Patient Requirement Registration for Blood

**Introduction:**

Patient can register for blood.

**Input:**

Patient will enter data like name, address, blood group, admitted hospital address, date of need.

**Processing:**

The **BDMS** will check for availability and create the registration confirmation.

**Output:**

The **BDMS** will generate a Case ID for future reference.

#### 5.2.1.3 Search donor

**Introduction:**

Anyone can search for blood.

**Input:**

He will enter data like Blood group, area, date of need.

**Processing:**

The **BDMS** will search for the requirement.

**Output:**

The **BDMS** will display the search result.

#### 5.2.1.4 Make a donation

**Introduction:**

People can donate money to encourage the volunteers and bear the expenses.

**Input:**

Admin will enter the name, amount, and address etc. data in the **BDMS**.

**Processing:**

The **BDMS** will create a new financial donor entry.

**Output:**

The **BDMS** will generate financial donation details.

#### 5.2.1.5 Create an Event

**Introduction:**

In **BDMS** blood donation events can be created.

**Input:**

Admin will enter the Event name, venue, requirements and date etc. data in the **BDMS**.

**Processing:**

The **BDMS** will create a new event entry.

**Output:**

The **BDMS** will generate an upcoming event ID and details.

#### 5.2.1.6 Mobile data entry & query

**Introduction:**

**BDMS** data can be entered and queried using a mobile device.

**Input:**

User will enter the registration details, donor information in the mobile device.

**Processing:**

The device will stored the entered data and sync with Server while manual sync operation. While querying device will search its internal storage for the query and display the result.

**Output:**

The mobile device will display the search result.

#### 5.2.1.7 Web data entry & query

**Introduction:**

**BDMS** data can be entered and queried using a web interface.

**Input:**

Admin will new donor details, event details as well as search query.

**Processing:**

Web interface will store new entry in the Google doc storage and while searching it will search its internal storage. Web interface will sync with main server while manual sync.

**Output:**

Website will show all the related information.

### 5.2.2 Technical Specification

**Front End/ GUI Tools:** Windows Presentation Framework (WPF)

**IDE:** Visual Studio 2010

**Framework:** Microsoft .NET 4.0

**Database:** MySQL

**Database Tool:** MySQL workbench CE

**Operating Systems**: Windows XP, Windows 7

**Cloud Technology**: Google Drive, Google forms

## 5.3 Planning and Scheduling

### 5.3.1 Gantt chart

****

### 5.3.2 Tracking Gantt

****

### 5.3.3 Pert Chart



# 6. Scope of the Solution

Blood Donor Management System is not a substitute of blood bank or blood donation camps. This is a parallel eco-system along with other existing system to improve & utilize the blood donation process.

This application will be developed for Windows operating system (Win7, Win XP) only; in future we are planning to make it runnable under LINUX, MAC operating system also.

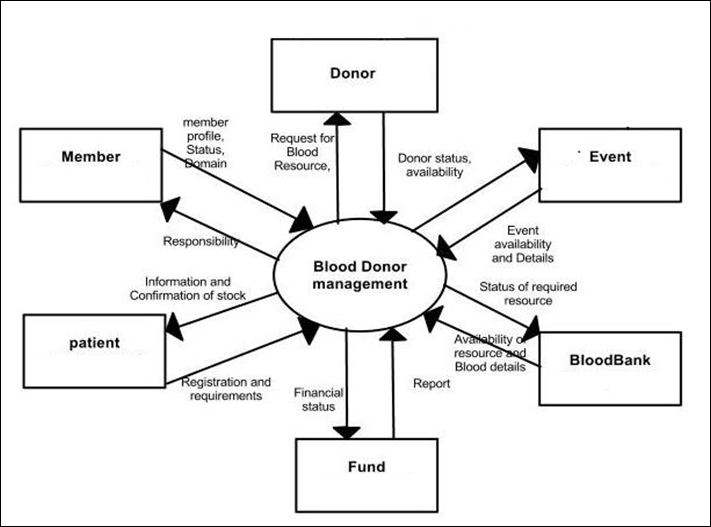
In case of mobile client we would develop it for java supported mobiles only; in future we would extend it to make it runnable under other mobile operating systems like Android, iOS or Windows Mobile OS.

Our web client will be developed using Google App Framework& Google Doc interface. Web client & Mobile client will not be synced automatically; it will require a manual sync with the server.

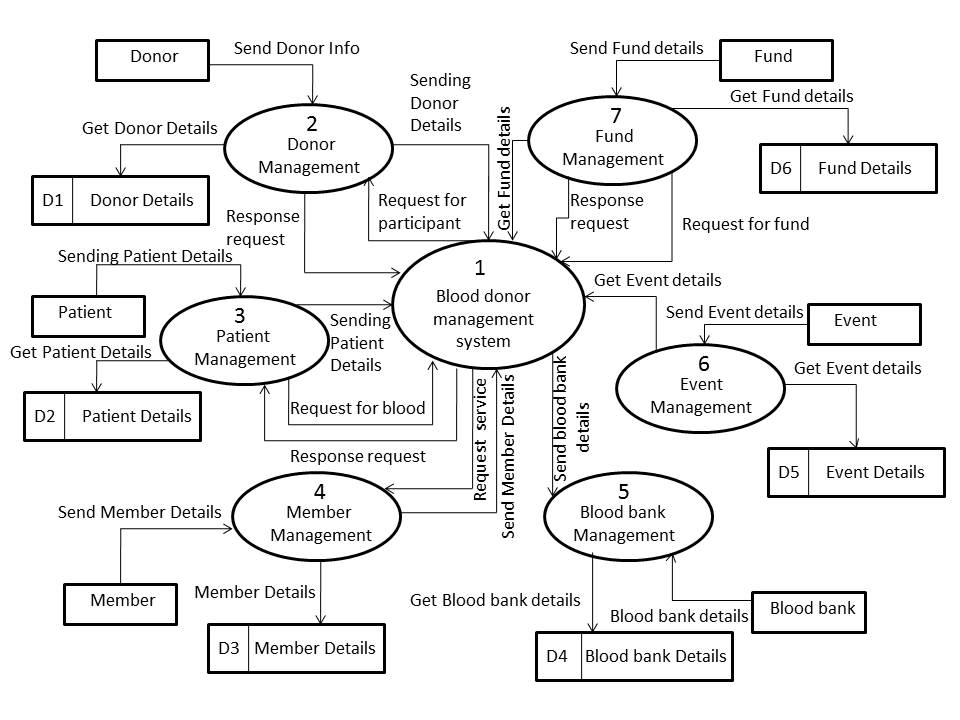
It is under a continuous process of development and we are working hard to make it perfect and error free project.

# Analysis

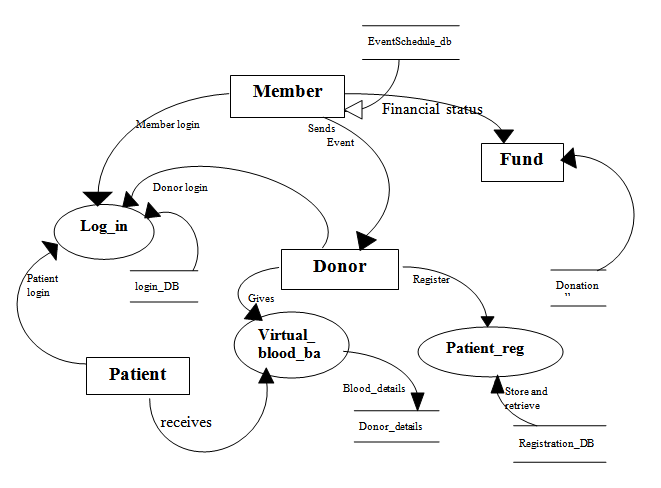
## Context Diagram



## 0-Level DFD



## 1-Level DFD



## 2-Level DFD

## E-R Diagram

We will design a RDBMS for Blood Donor Management System. The entities and their attributes are listed below. Attributes in Bold letter is the unique key.

|  |  |
| --- | --- |
| **Entities** | **Attributes** |
| Donor | **donor Id**, Name, Address , Availability, Details |
| Blood Donor Management System | **ORG Id** , Name, Address, Registered no |
| Machine | **Machine Id**, Name, Software |
| Event | **event Id, place**, Time, patient\_id |
| Admin | **Admin Id**, Name, address, contact number |
| User Preference | **Preference Id,** Type, Description |
| Fund | Amount,dateOfdonation,Serial\_no |
| Patient | Patient\_id,name,contact no,photo id no,address, |

**Relationship between Entities:**

* Blood Donor Management System has donors  1 : N
* Blood Donor Management System has Machine 1 : N
* Users System uses Session 1 : 1
* Blood Donor Management System avails donation 1 : N
* Users provide Preferences  M : N
* Donor donates blood to patient→1:1
* Admin organizes events→1:1



## Class Diagram

# 8. Database & Table Details

The database used for this software is called **bdmsdb**. A screenshot from the MySQl workbench is given below. It shows the tables and its columns. The first column is the primary key.

# 9. Complete Structure

## 9.1 Module Description



The main components of Blood Donation Management system are:

* **BDMS Server**

This is the backbone of BDMS. This dedicated server stores all the data and handles the business logic of the software. It also handles interaction with clients and data syncing. This is divided into two major blocks.

* + **BDMS Engine**: This is the controller of BDMS. This takes decision based on the business logic and employs other components.
  + **BDMS Database**: This is the centralized storage of all BDMS data.
* **BDMS Desktop Client**

This is the full functionality client of BDMS which will be deployed on desktop computers and laptops.

* **BDMS Web Client**

This is the limited functionality client which can be accessed using any web browser.

* **BDMS Mobile Client**

This is the limited functionality client which can be accessed using any mobile devices such as Mobile phone, tabs & pads.

* **BDMS Sync Manager**

This component handles the data syncing between server and web & mobile clients.

## 9.2 estimation



## 9.2 Data Structure

|  |
| --- |
| **DonorInfo** |
| public class DonorInfo  {  public string id { get; set; }  public string name { get; set; }  public string bloodGroup { get; set; }  public string phone { get; set; }  public string address { get; set; }  public DateTime lastDonateDate { get; set; }  public DateTime dob { get; set; }  } |

|  |
| --- |
| **EventInfo** |
| public class EventInfo  {  public string id { get; set; }  public string eventTitle { get; set; }  public DateTime eventDoe { get; set; }  public string eventVenue { get; set; }  public string eventGoal { get; set; }  } |

|  |
| --- |
| **ExpenseInfo** |
| public class ExpenseInfo  {  public string id { get; set; }  public string purpose { get; set; }  public DateTime doe { get; set; }  public string expensed\_by { get; set; }  public double amount { get; set; }  } |

|  |
| --- |
| **FundInfo** |
| public class FundInfo  {  public string id { get; set; }  public string wellwisher\_name { get; set; }  public string contact { get; set; }  public DateTime dod { get; set; }  public string received\_by { get; set; }  public double amount { get; set; }  } |

|  |
| --- |
| **MemberInfo** |
| public class MemberInfo  {  public string id { get; set; }  public string name { get; set; }  public DateTime doj { get; set; }  public string address { get; set; }  public string phone { get; set; }  } |

|  |
| --- |
| **PatientInfo** |
| public class PatientInfo  {  public string id { get; set; }  public string name { get; set; }  public string bloodGroup { get; set; }  public int age { get; set; }  public string address { get; set; }  public string phone { get; set; }  public string admittedAddress { get; set; }  public DateTime expectedDate { get; set; }  public string assignedDonor { get; set; }  public string donorContact { get; set; }  } |

|  |
| --- |
| **TodoInfo** |
| public class TodoInfo  {  public string id { get; set; }  public DateTime date { get; set; }  public string details { get; set; }  } |

|  |
| --- |
| **WellWisherInfo** |
| public class WellWisherInfo  {  public string id { get; set; }  public string name { get; set; }  public string address { get; set; }  public DateTime doj { get; set; }  public string phone { get; set; }  public string remarks { get; set; }  } |

## 9.4 Implementation Methodology

* Object Oriented Programming methodology will be adopted
* User interface development will be done in MVC architecture using Windows Presentation Framework.
* Relational DBMS MySQL will be used to implement & execute SQL query to database.
* Agile Software Development model will be used while developing this software.

## 9.5 List of Reports

List of reports that are likely to be generated in this software are given below:

* Blood donors will be given a certificate as a token of appreciation
* List of donors can be generated
* List of patients can be generated
* List of members can be generated
* Event details can be generated
* Fund details can be generated
* Yearly donation report can be generated
* Case study report can be created

# 11. Implementation of Security Mechanism at Various Levels

* This software requires a valid password to login and then it allows using any of its features.
* The login password will be saved in encrypted format in database.
* This software will use Google open-id authentication for web interface.

# 12. Future Scope & Further Enhancement of the Project

This application will be developed for Windows operating system (Win7, Win XP) only; in future we are planning to make it runnable under LINUX, MAC operating system also.

In case of mobile client we would develop it for java supported mobiles only; in future we would extend it to make it runnable under other mobile operating systems like Android, iOS or Windows Mobile OS.

Our web client will be developed using Google App Framework& Google Doc interface. Web client & Mobile client will not be synced automatically; it will require a manual sync with the server. It is under a continuous process of development and we are working hard to make it perfect and error free project

# 13. Bibliography

* http://en.wikipedia.org
* http://msdn.microsoft.com/en-us/
* http://www.microsoft.com/en-us/default.aspx
* http://www.codeplex.com/
* http://stackoverflow.com/
* http://www.codeguru.com/
* http://www.w3schools.com
* www.mysql.org