Online Blood Bank

This Project report has been submitted in fulfillment of the requirements for the Degree of Bachelor of Science in Software Engineering.

Submitted By

MD. Mojadded Hossain Khan

ID: 142 - 35 - 695

Department of Software Engineering

Daffodil International University

APPROVAL

This project titled on "**Online Blood Bank**", submitted by **Md. Mojadded Hossain Khan**, **ID:142-35-695** to the Department of Software Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Software Engineering and approval as to its style and contents.

BOARD OF EXAMINERS

28/10/18

Prof. Dr. Touhid Bhuiyan

Professor and Head

Chairman

Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

28/10/2018

Dr. Md. Asraf Ali
Associate Professor

Internal Examiner 1

Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

28.10.18

Ms. Manan Binth Taj Noor

Lecturer

Internal Examiner 2

Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

Dr. Md. Nasim Akhtar
Professor

External Examiner

Department of Computer Science and Engineering
Faculty of Electrical and Electronic Engineering

Dhaka University of Engineering & Technology, Gazipur

DECLARATION

I hereby declare that I have taken this project under the supervision of **Mr. Md. Maruf Hassan, Assistant Professor, Department of Software Engineering, Daffodil International University**. I also declare that neither this project nor any part of this report has been submitted elsewhere for award of any degree.

Submitted By:

Md Mojadded Hossain Khan

.....

Name of the Student: Md. Mojadded Hossain Khan
ID: **142 - 35 - 695**
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

Certified By:

.....

Name of the Supervisor: Mr. Md. Maruf Hassan
Designation: **Assistant Professor**
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

ACKNOWLEDGEMENT

First of all, I express our gratitude to the Almighty Allah for giving us the ability to complete this Project. I also grateful to my parents who always support me and encourage to do something for the mankind. Now I would like to express the gratitude and appreciation to all those who gave me the possibility to make my project, project documentation more effective and also completed. A special thanks to my supervisor and our honorable teacher "**Mr. Md. Maruf Hassan**", whose help, simulation and encouragement, helped me to coordinate my project specially writing this documentation.

A special thank goes to my friend "Md Zahid Hasan Shaikot", who help me to assemble the parts and gave suggestion to make my project in different processes.

From my sincere thanks to friends who have supported my work on the project. Specially, Daffodil International University's family members, friends and brothers, Shakil, Moshiur Islam, Faysal Ahmed, Muktadir Soyeb and also **MD Maruf Hassan Sir** for their valuable and important ideas.

Finally, I would like to thank my family and friend for their support. I wouldn't have been able to get here without them.

Md. Mojadded Hossain Khan

Department of software engineering

Daffodil International University

ABSTRACT

Online Blood Bank are used widely in any situation where anyone need to contact or find the Donor. This system is online base, so, anyone can find the donor through internet. This project's perspective, I would have given a name and that's "**Online Blood Bank**". Using this application seeker can view the post, create post, view notification, search.

TABLE OF CONTAINS

Contents

LIST OF FIGURE

LIST OF TABLE

Chapter-1
Introduction

1 INTRODUCTION

1.1 OVERVIEW

This system is fully online based. Donor information are record in the online. Before online, we need to record donor information in paper like register book, form and all information are written by hand. So, it is so difficult to maintain donor information. We also find a donor through a Blood Bank. As a result, some people take advantage from this site. I want to break this system and I want to easy, hassle-free find donor system. Naturally we saw that people are towards here and there for searching Blood in case of emergency.

This system records the donor information who want to donate blood for mankind. By using this system, a seeker who find the donor for his/her emergency can register to the system and create post for expected Blood Group with location. Than the system send notification to all of the donor who have that blood group in that location to donate blood that person. When the donor accepts the request, the system send donor name, Contact info to the blood seeker.

1.2 Purpose

Online Blood Bank Records Donor information. Anyone can find donor according to the Blood group and location. So, we can define the purpose for Donation Blood for mankind.

1.3 Background

As usual Blood Bank records their Donor information in a Register books, paper etc. as a results, it is difficult to find a Donor easily. It also difficult to maintain These paper books. Besides these, a group of people take advantage from this site. So, I want to build up this system fully mankind so that people can easily find the Donor at their emergency time. The Donor also donate their Blood as the right place with their satisfaction.

1.4 Objectives

- Easy to communication with Donor.
- Seeker can easily find the Donor.
- Donor can notice when, where he/she Donate Blood.
- Seeker can find Donor in the right place.
- Donor can manage Blood Donation Request.
- Easy to use.

1.5 Stakeholder

There are many members associate with this project. They have helped to develop the system directly or indirectly.

Internal Stakeholders:

1. Admin
2. Donor
3. Seeker

External Stakeholders:

1. Patient.
2. Visitor.

1.6 Proposed System

To develop this Online Blood Bank, I proposed a model for this system. I clear the system briefly here.

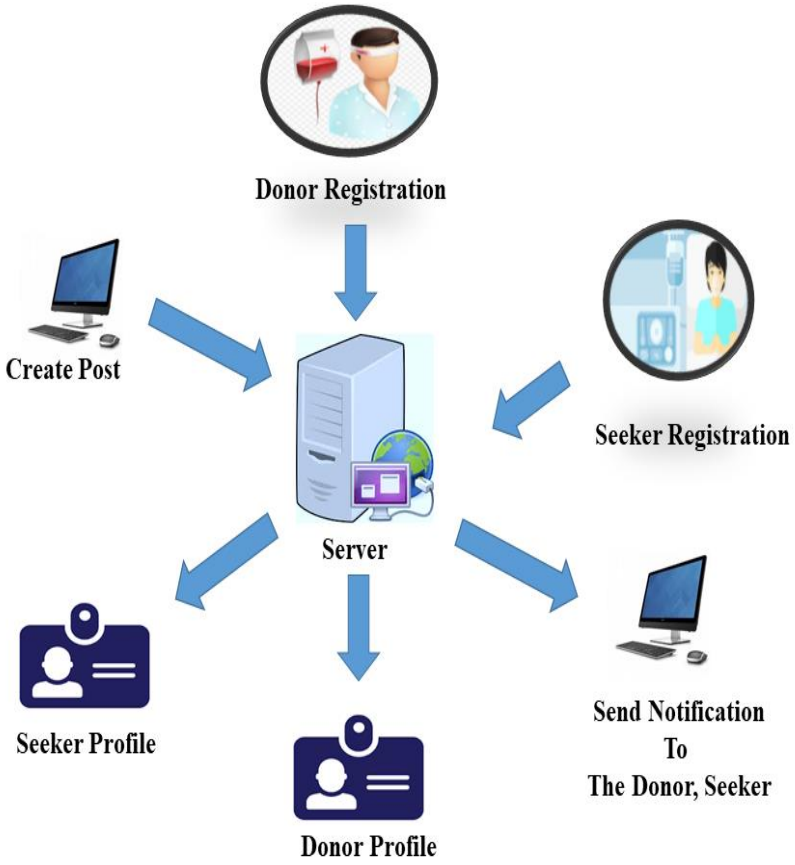


Figure-1.1: Propose System Model

In this system anyone can register as a Donor who want to Donate Blood for mankind. Who want to registration as a Donor, first he/she give his/her information. After register, a donor can view his/her profile through login using user name and password. Anyone who want to Blood, also should register as a seeker. Than the person can create post for Blood. The seeker can see the notification also that the donor response.

1.7 System process:

Figure 1.2: System Process Model

Using this system anyone can registration as a donor or blood seeker. Donor can view profile, handle request, login. Blood seeker can create post for blood, view notification etc.

1.8 PROJECT PLAN:

To full fill the requirements and complete the project the at the right time, project schedule helps for proper planning. I also make a project schedule to complete my project properly.

1.8.1 Gantt Chart

In project planning, I use Gantt chart to manage my project properly. To use this tools, I can track all the task which is not done or not. Also track which one is schedule for the next task. I control my project duration by this tools.

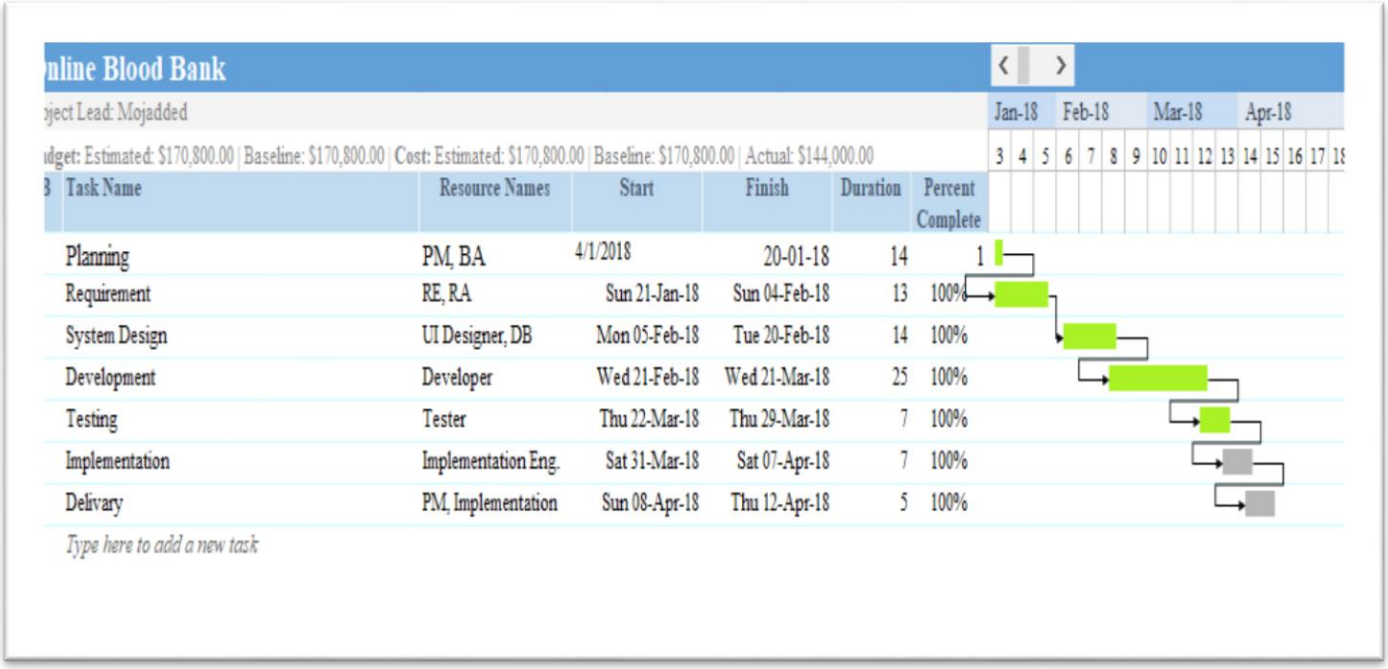


Figure-1.3: Gantt Chart

1.8.2 Risk:

In Gantt chart tools, I show the project duration. I took 25 days for developing, so that I could mitigate my risk. In developing page, I find out a risk that risk is developer. Who may sick or leave.

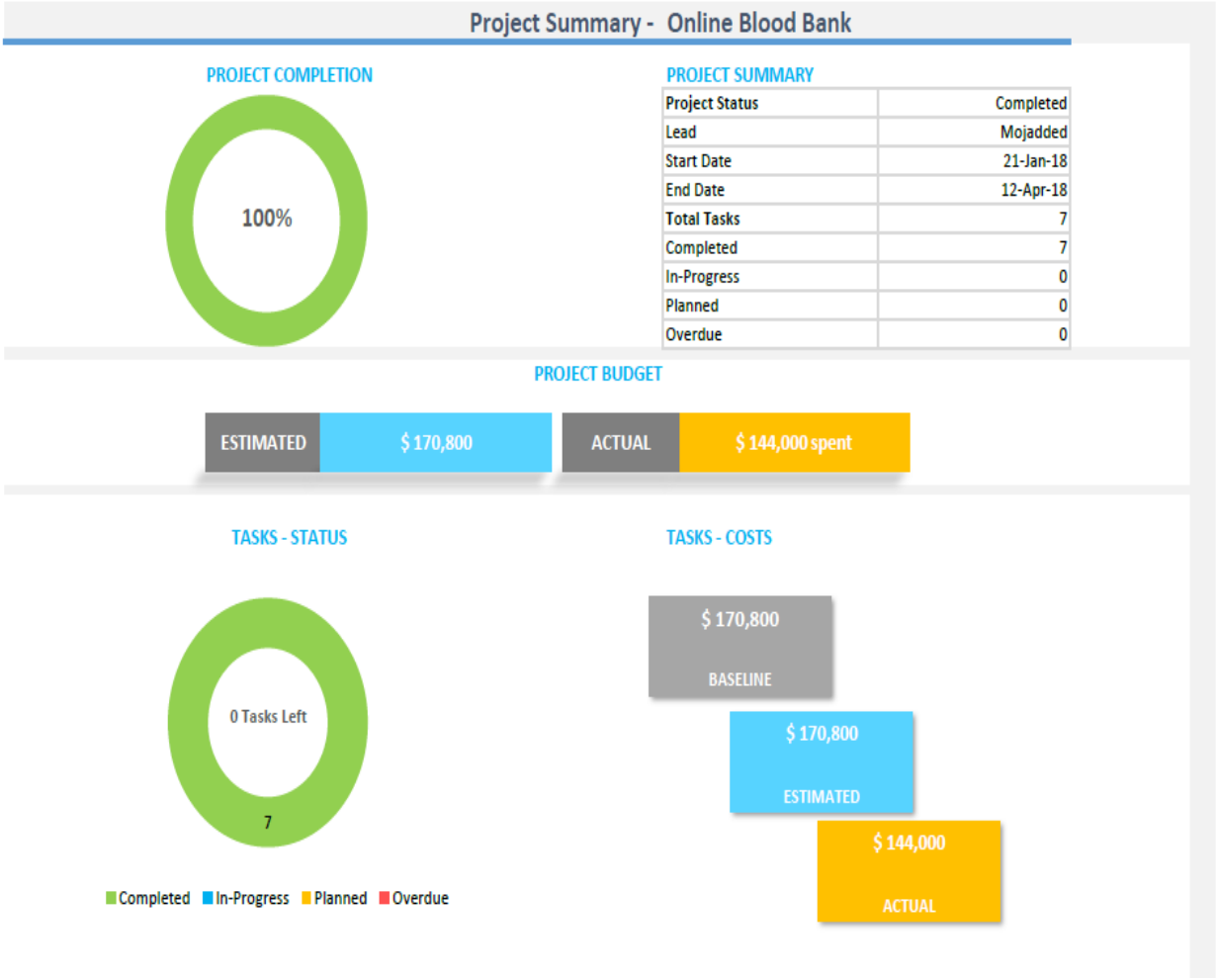


Figure-1.4: Project Summary

1.8.3 Milestones:

Milestones, a time frame of a project, will define the task. These project milestones are as follows:

Task No	Task Name	Duration
01	**Planning**	**14 days**
02	**Requirement gathering & Analysis**	**13 Days**
03	**System Design**	**14 Days**
04	**Development**	**25 Days**
05	**Testing**	**7 Days**
06	**Implementation**	**7 Days**
07	**Relies**	**5 Days**
	Total	**85 Days**

Chapter-2
Requirement Specification

2 SOFTWARE REQUIREMENTS SPECIFICATION

Requirements analysis is the process of identifying the user satisfaction form the System. So, Requirements analysis is an important part of project management.

When I selected this project I thought about some specific Software requirement, like as…

- Who is the stakeholder of this system?
- Is it helpful for them or not?
- Functional & Non- functional requirements
- Maintenance of the system

2.1 FUNCTIONAL REQUIREMENT:

The functional requirements of the system are like below---

2.1.1 DONOR REGISTRATION:

FR-01	**Donor Registration**
Description	In this system there are many users like seeker, visitor etc. But Donor registration page is only for those person who want to donate Blood for mankind and help the people. This page is required some information like as NID No, Phone No etc. and store the date as a donor information.
Stakeholder	**Blood Donor.**

2.1.2 SEEKER REGISTRATION:

FR-02	**Seeker Registration**
Description	In this system there are many users like Donor, visitor etc. But Seeker registration page is only for those person who search Blood for patient. This page is required some information like as Name, Phone No etc and store the date as a Seeker information.
Stakeholder	**Blood seeker, patient, Anyone who's need Blood.**

2.1.3 LOGIN

FR-03	Login
Description	In this system there are many users like seeker, visitor, Donor etc. But Donor and Blood seeker can login after complete Registration.
Stakeholder	**Blood Donor, Seeker.**

2.1.4 CREATE POST:

FR-04	Create Post
Description	In this system only seeker can create post. To create a post for Blood, a person should have registered as a seeker. Hospital name, Location, Blood Group, phone, Date, Time are must to create a post for Blood.
Stakeholder	**Blood Donor, Seeker, patient.**

2.1.5 VIEW NOTIFICATION:

FR-05	View Notification
Description	Both Donor and Seeker can view notification. According to the seeker post donor get a notification to help the patient by donating Blood. If the donor response the notification, then seeker get donor information notification.
Stakeholder	**Blood Donor, Seeker.**

2.1.6 VIEW PROFILE:

FR-06	View Profile
Description	Donor can view his/her Profile. Seeker can see notification.
Stakeholder	**Blood Donor, Seeker.**

2.1.7 ACCEPT, REJECT AND REPLY REQUEST:

FR-07	Accept, Reject and reply request
Description	Donor can Accept, Reject request against the notification. Seeker can see notification.
Stakeholder	**Blood Donor.**

2.2 PERFORMANCE REQUIREMENTS:

It's very necessary to sustain the performance of the project. To assure the better performance, this project has to meet some requirements which will provide the better performance.

2.2.1 SPEED AND LATENCY REQUIREMENTS:

While inserting or viewing the system in the browser, system need a minimum amount of speed to perform the task.

SLR-01	The system will be faster
Description	When the user browsing, it depends on their internet speed. It also depends on server bandwidth speed.
Stakeholders	**Donor, Blood seeker, visitor.**

2.2.2 LEGIBILITY AND ACCURACY REQUIREMENTS:

System have to confirm the Legibility and Accuracy of the data.

LAR-01	**Data accuracy**
Description	The input data should be correct and right pattern data, otherwise the input information never save. Like NID No, Phone etc the input information is not valid, the data never save. Or the input data pattern is not match, the system never saves or accept the data.
Stakeholders	**Donor, Blood seeker.**

2.2.3 CAPACITY REQUIREMENTS

The system should maintain the all inserting data.

CR-01	**Manage the all data in database system.**
Description	All registration data like Donor registration data, Blood seeker registration data, Post information are store in the database in right format.
Stakeholders	**Donor, Blood seeker.**

2.3 DEPENDABILITY REQUIREMENTS:

Dependability means, it measures of a system availability, reliability, security etc. Here, dependability means the running time of this project.

2.3.1 RELIABILITY AND AVAILABILITY:

RA-01	The system must be available 24x7
Description	➢ It's available 24 hours in a day ➢ The system must be updated regularly
Stakeholders	**Donor, Blood seeker, visitor, patient.**

2.3.2 SAFETY CRITICAL REQUIREMENTS:

There are no specific safety critical requirements.

2.4 MAINTAINABILITY AND SUPPORTABILITY:

For Maintenance The system and support the system, some people associate the project.

2.4.1 SUPPORTABILITY REQUIREMENTS SPECIFICATION:

- **SRS-1.** To understand the system's behavior on a technical support is required by the system operator. The reason for reading them might be
- **SRS-2.** System malfunction has occurred and the system operator has to find the exact point of time when this happened
- **SRS-3.** System produces wrong results and the developers must be able to reproduce the data flow through the system
- **SRS-4**. Hacker tried to breach the system's security mechanisms and the system operator must understand what he did.

2.4.2 ADAPTABILITY REQUIREMENTS:

There are no specific adaptability Requirements.

2.5 SECURITY REQUIREMENTS:

- **SR-1.** Log in as a Donor
- **SR-2.** Log in as a Blood seeker

To get access to this system or a specific module the system must provide an authentication mechanism. To prevent anyone to exploit stolen Data all user's password must be encrypted in hash process.

2.5.1 ACCESS REQUIREMENTS:

This system provides accesses the different module, by access the authentication way the authentic user.

2.5.2 INTEGRITY REQUIREMENTS:

To prevent credentials information of user from being stolen, all passwords are stored in encrypted form. The Requirements significantly reduces the value of stolen user credentials, it's not easy to decrypt the password.

2.6 USABILITY AND HUMAN INTEGRITY REQUIREMENTS

This system easy to use and all of the people who wants to donate blood and who need blood.

2.7 Data Validation

In this stage I have try to validate almost all input field

2.8 User Interface Design

It is important to consult the system users and their necessities while designing the user interface.

Chapter-3

Requirements Analysis

3.1 Use Case Diagram:

In this system a user (Donor) what things he/she can do, is describe in this picture that provide in below. A Donor can login in the system. But before login he/she must registration in this system as a Blood Donor. Than he/she can access the login option. Donor also view Blood donation request, profile etc. Donor also handle the request.

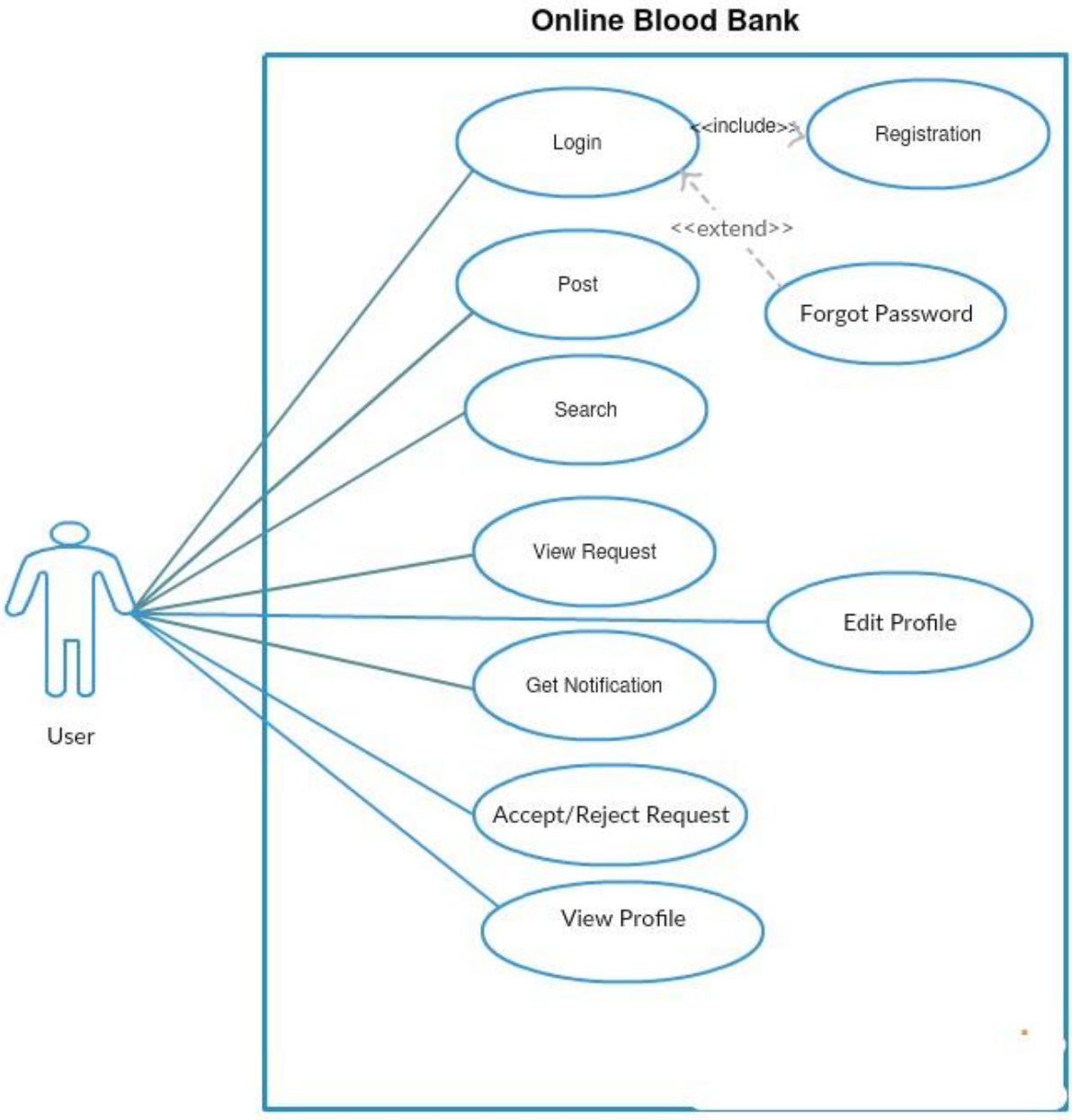


Figure-3.1: Use-Case Diagram user (Donor)

3.1.1 Donor

Use Case Title	**Donor**
Goal	Insert Donor Information to the database.
Precondition	User must NID No, Phone.
Success & End Condition	System store the Donor information.
Failed End Condition	Database can't store the data.
Primary Actors: **Secondary Actors:**	Donor
Tigger	Donor Registration
Description	Who want to Donate Blood is must to registration in the system insert his/her information.
Alternative Flows	N/A
Quality Requirements	N/A

3.2 Use Case Diagram:

A blood seeker, what option that he/she can access or can do by this system is describe in this picture that shown in the below. A blood seeker can create post. But to create a post for blood, he/she must login in this system. To login in this system he/she must be registered in this system. Seeker can view post response, search.

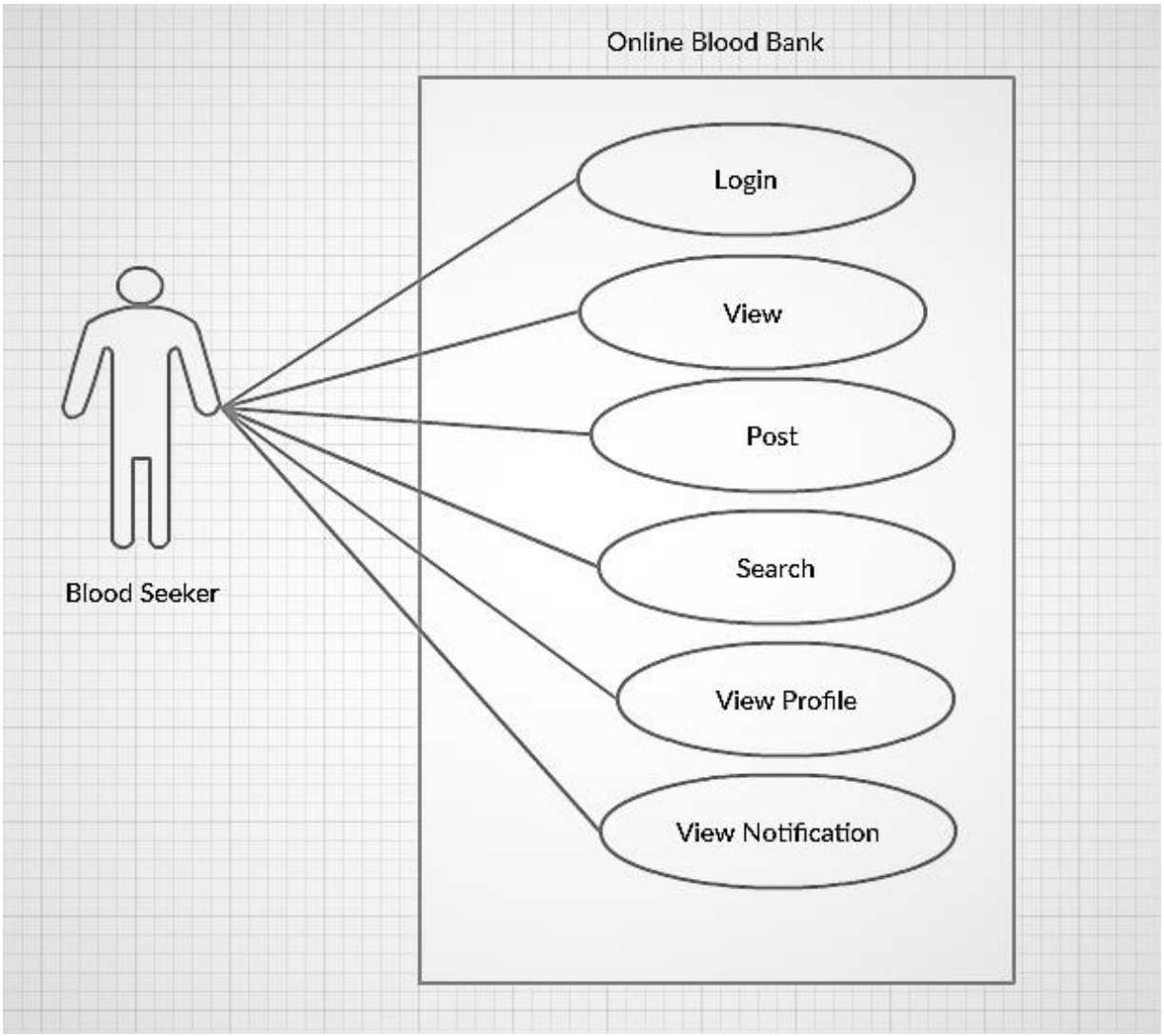


Figure-3.2: Use-Case Diagram user (Blood seeker)

3.2.1 Blood Seeker:

Use Case Title	**Blood Seeker**
Goal	Insert Blood Seeker Information to the database.
Precondition	User must have a Phone number.
Success & End Condition	System store the Donor information.
Failed End Condition	Database can't store the data.
Primary Actors: **Secondary Actors:**	Blood Seeker
Tigger	Blood seeker registration, Create post
Description	To search Blood in the specific location and create a post for Blood, a Blood seeker must be registration in the system.
Alternative Flows	N/A
Quality Requirements	N/A

3.3 Activity Diagram

Following activity diagrams are exactly describing the flow of the different state of the project.

3.3.1 System Activity

By this figure I explain my system. If anyone enter the system, he/she see the all the option. And who are registered user and he/she can login in the system. After login criteria he/she can access different option.

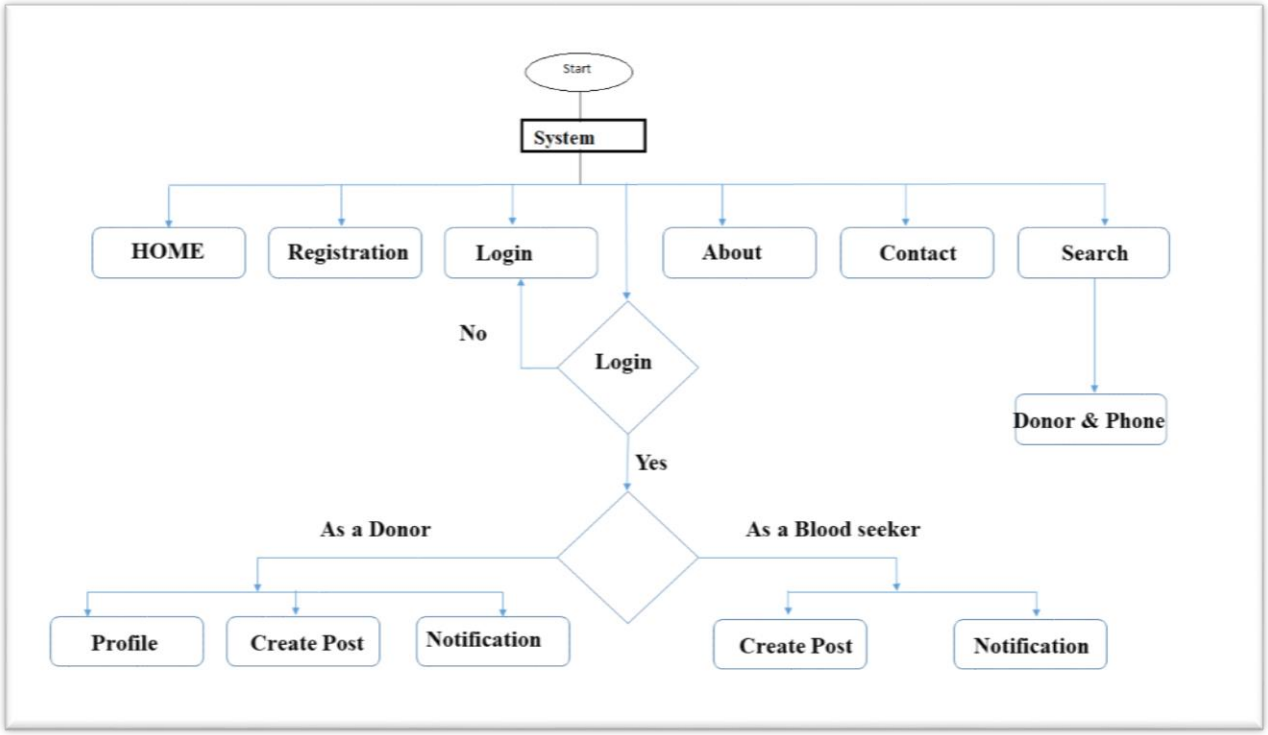


Figure-3.3.1: System Activity

3.3.2 Registration Activity

By this figure I explain Registration process. If anyone enter the system, he/she see the all the option. And who are want to registration, he/she might do follow the instruction. After successful registration he/she can login in the system and access different option.

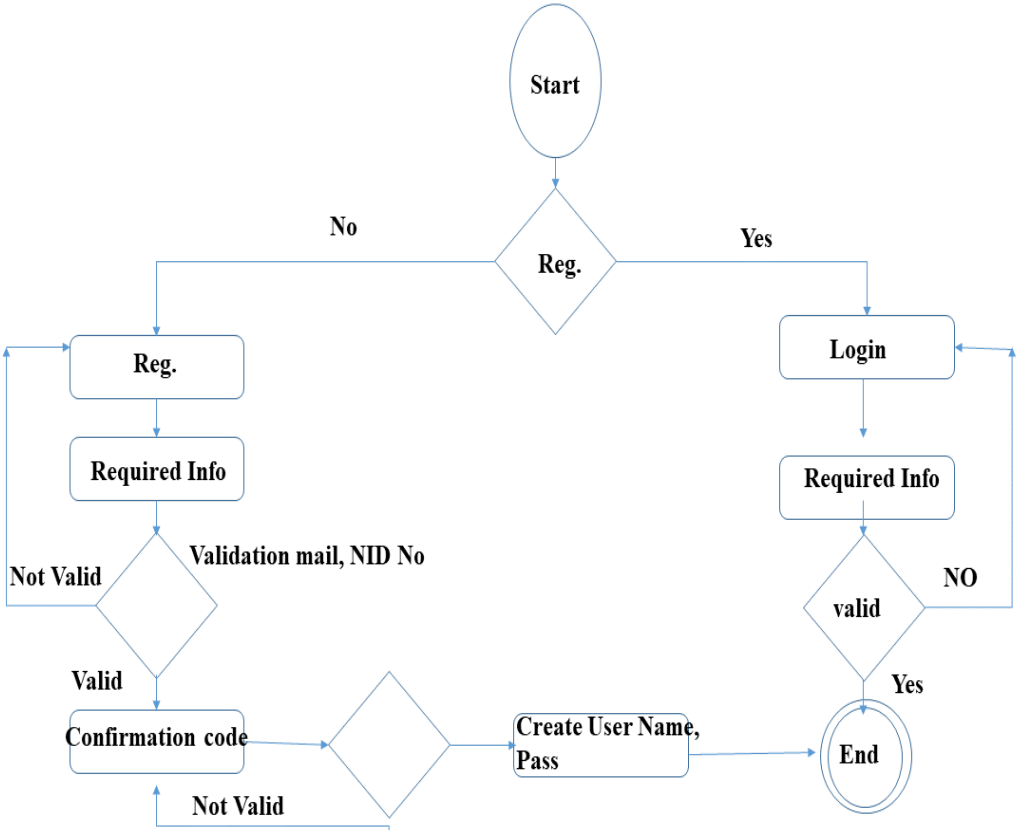


Figure-3.3.2: Registration Activity

3.3.3 Create Post Activity

In this figure I show the create post process. If anyone enter the system, he/she see the all the option. And who are registered user as a seeker, he/she can login in the system. After login he/she can create post for blood.

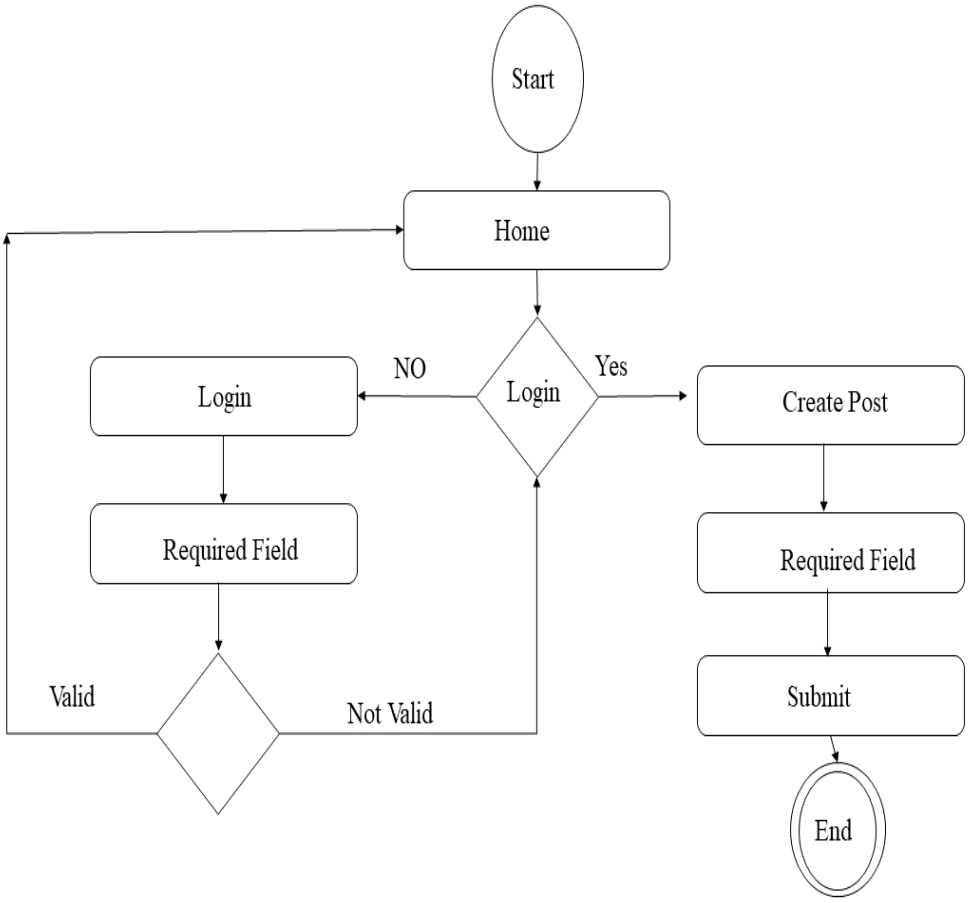


Figure-3.3.3: Create Post Activity

3.4 Sequence Diagram:

Sequence Diagram show the process in sequential way that it's actor done.

3.4.1 BLOOD SEEKER SEQUENCE DIAGRAM

In this picture describe the Blood seeker work sequence system to database. A blood seeker can request for registration, login, create post, view notification and all these 1st go to the system than check in database. If the request is valid than system get confirmation and system confirm the seeker.

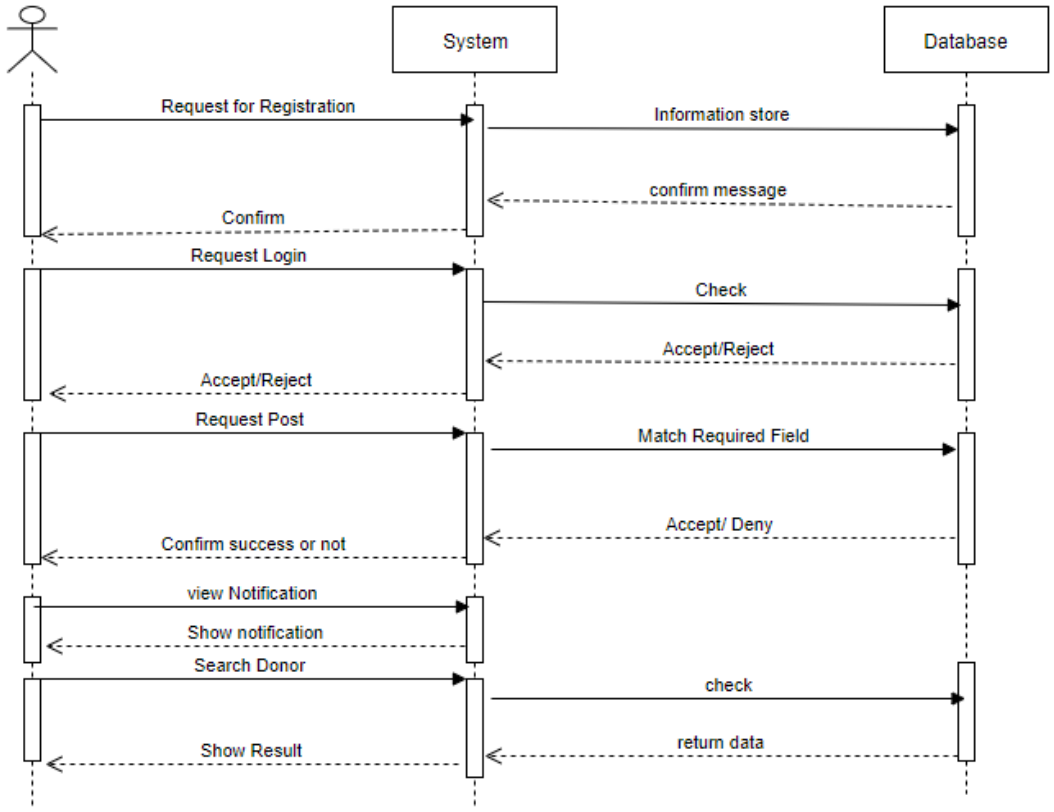


Figure-3.4.1: Blood Seeker Sequence Diagram

3.4.2 DONOR SEQUENCE DIAGRAM:

In this picture describe the Blood Donor work sequence system to database. A blood Donor can request for registration, login, create post, view notification, view profile and all these 1st go to the system than check in database. If the request is valid than system get confirmation and system confirm the seeker.

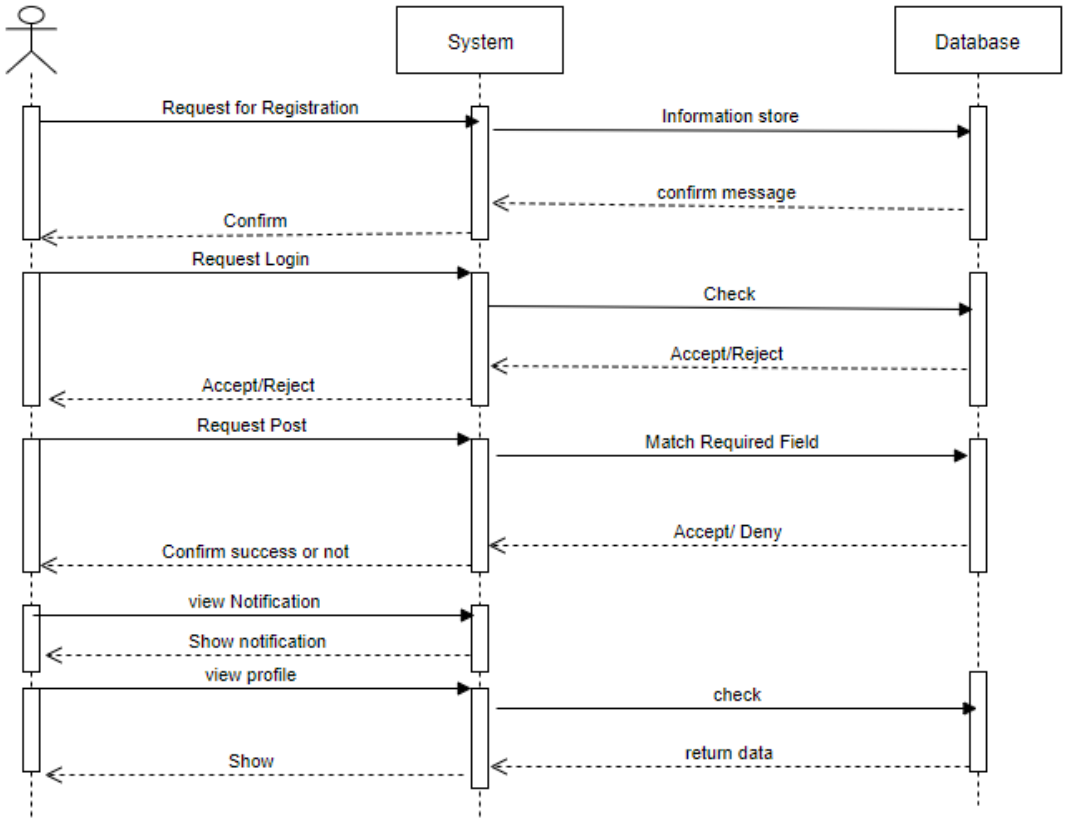


Figure-3.4.2: Blood Seeker Sequence Diagram

Chapter 4

Design and Development

4. Technology & Tools

For developing this project, I have used some tools and technology that's are talking in below.

4.1 User Interface Technology:

User interface (UI) is everything designed into a system view that a person's associates with this system may like the interface of this system.

4.2 TECHNOLOGY

- Programming language: Asp.Net
- Web server: IIS
- Design: html, CSS, bootstrap, JavaScript, Ajax
- Database server: SQL Server

4.3 TOOLS:

- Visual Studio
- SQL Server Management
- Windows 10

4.4 ERD:

In this figure show the database relation. DonorInfo table and Blood Group table is one to one relationship. Because One Donor have one Blood Group. Seeker table and Blood group table also one to one relationship, because one seeker can post only for one blood group. Donor and user table is one to one relationship because NID information only one donor and it is a unique number. Donor and notification table one to many relationships because, one donor can get more than one notification. Same as notification and seeker.

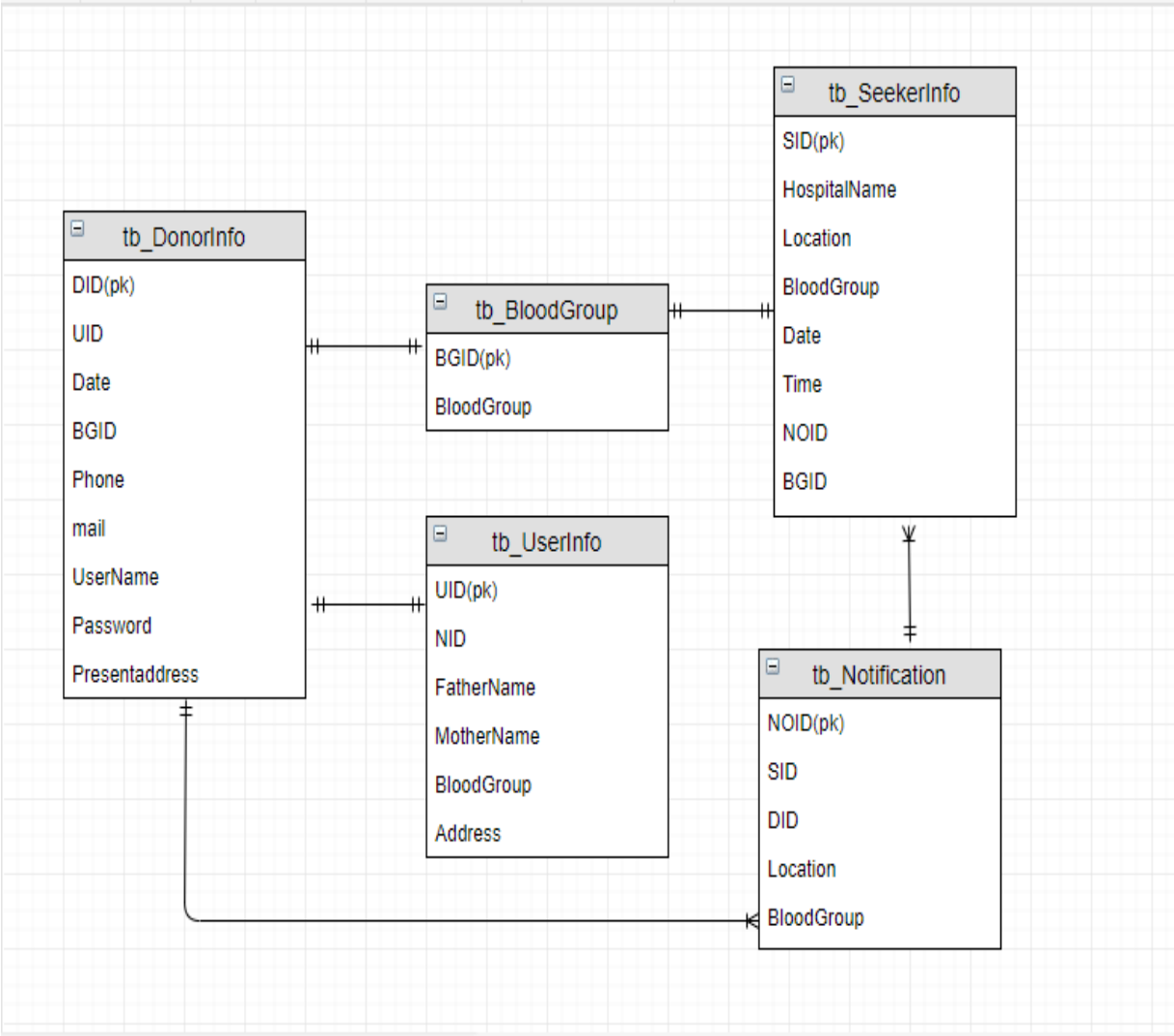


Figure-4.1: ER Diagram (Physical)

4.5 Schema Diagram

This Diagram describe the relationships among the tables.

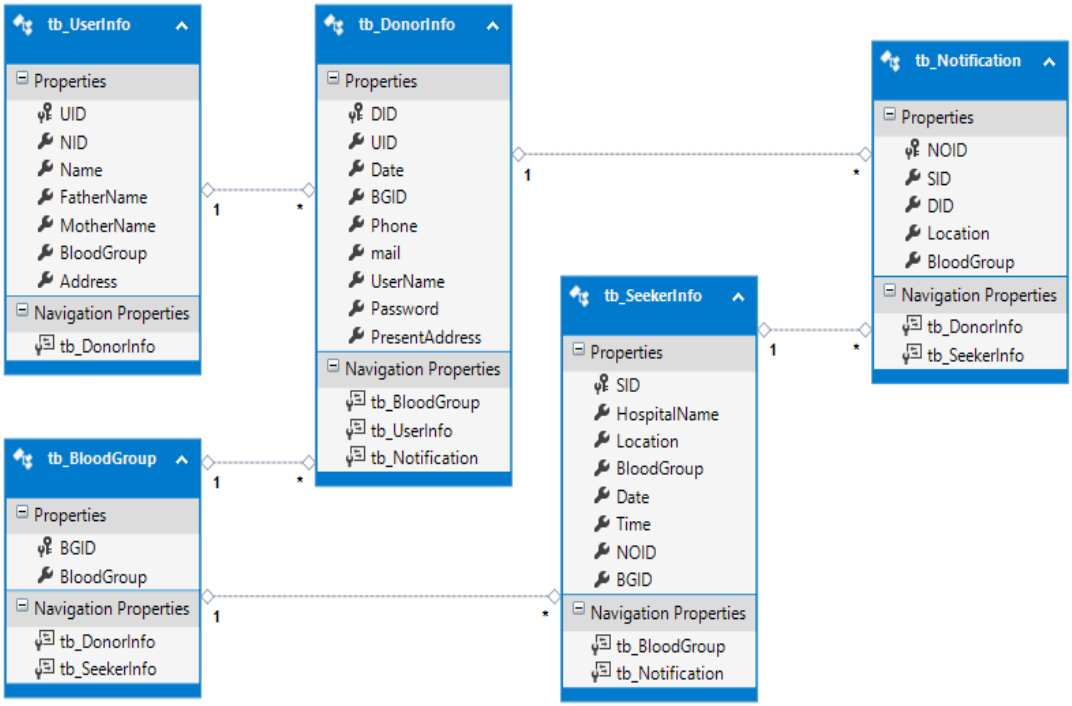


Figure-4.2: Schema Diagram

5. IMPLEMENTATION

5.1 HARDWARE & SOFTWARE SPECIFICATIONS

In this stage I want to describe what's need to build this application.

- **Hardware Requirements:**
  - ✓ PROCESSOR: Dual Core or above
  - ✓ RAM: 2GB or above
  - ✓ Cache Memory: 2MB or above
  - ✓ HDD: 20GB or above

- **Software Requirements:**
  - ✓ **IDE**: Visual Studio.
  - ✓ **Database:** SQL Server Management.
  - ✓ **Web-Server:** IIS

5.2 UI IMPLEMENTATION

5.2.1 Home page

Now I am showing home page of my application. In home page anyone can view.

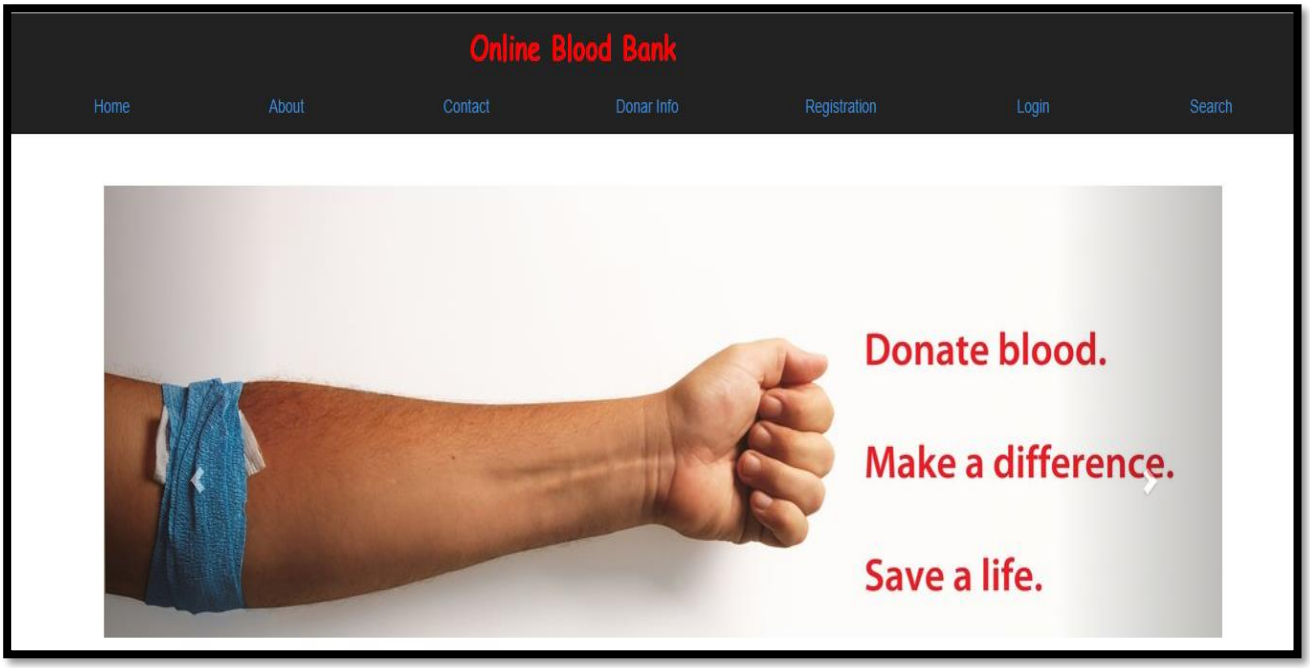


Figure 5.2.1: Home page

5.2.2 Registration page:

Blood seeker and Donor registration form to access the system.

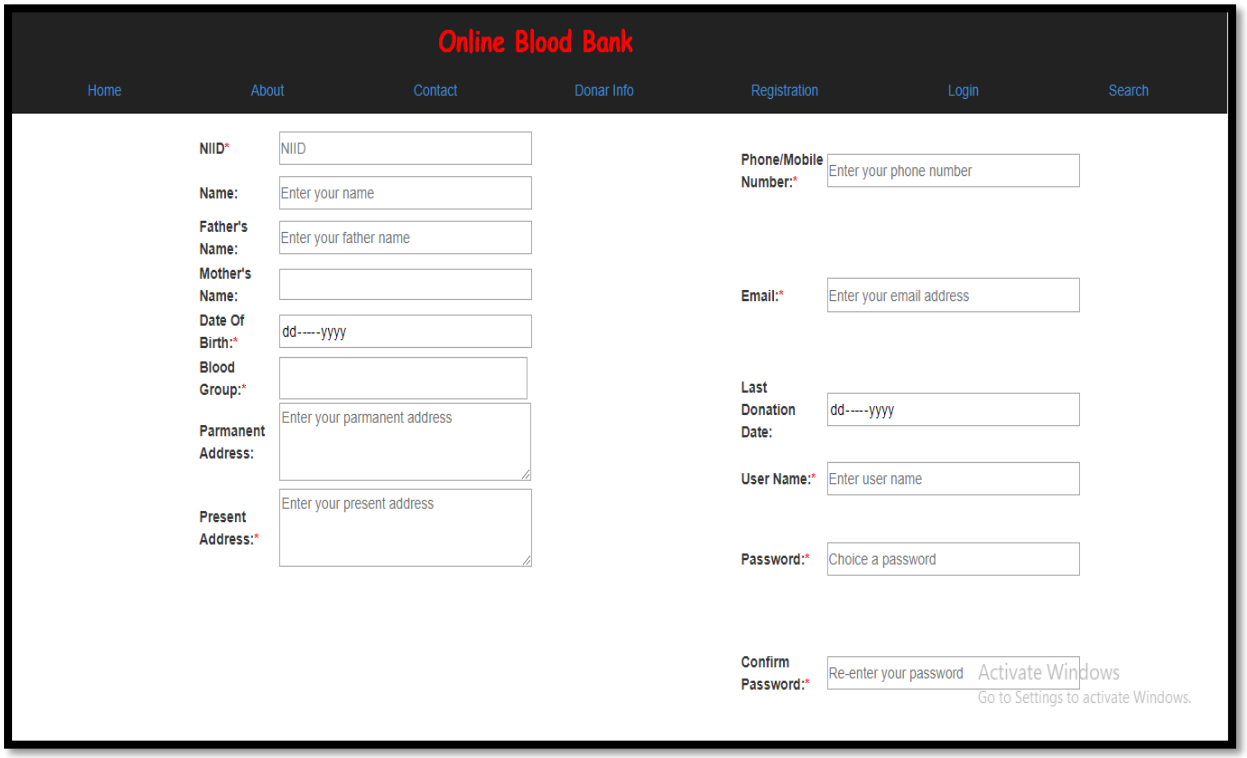


Figure 5.2.2: Registration page (part 1)

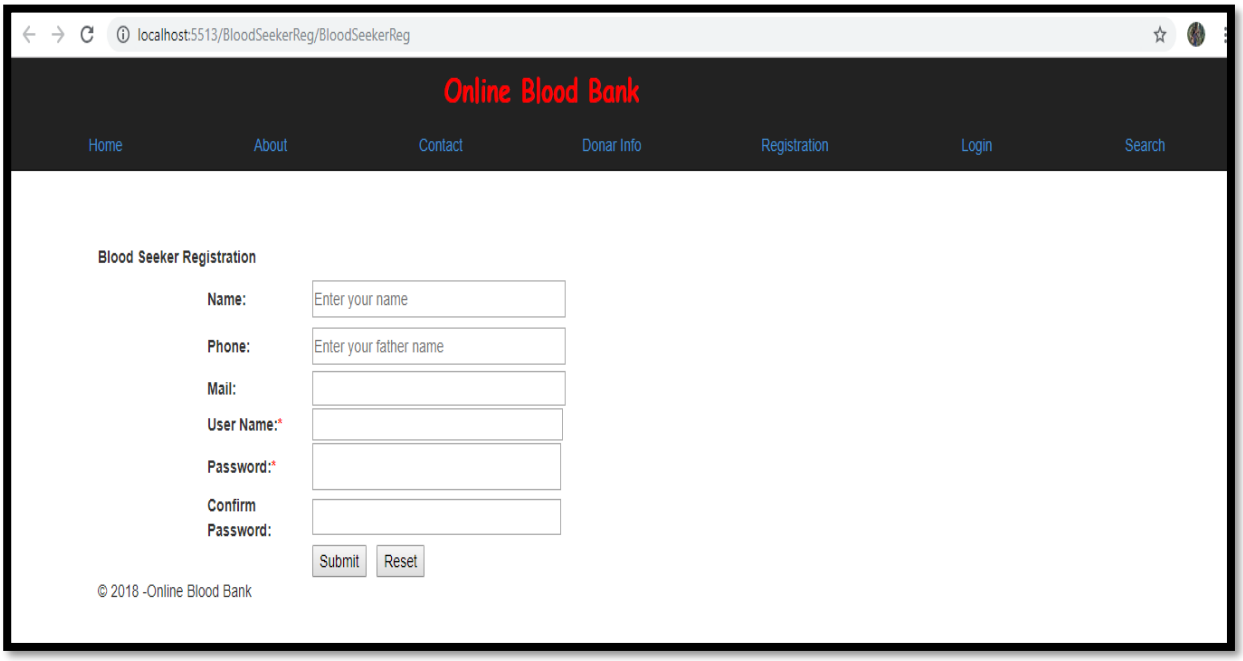


Figure 5.2.3: Registration page (part 2)

5.2.3 LOGIN:

Donor and Blood seeker can login in the system.

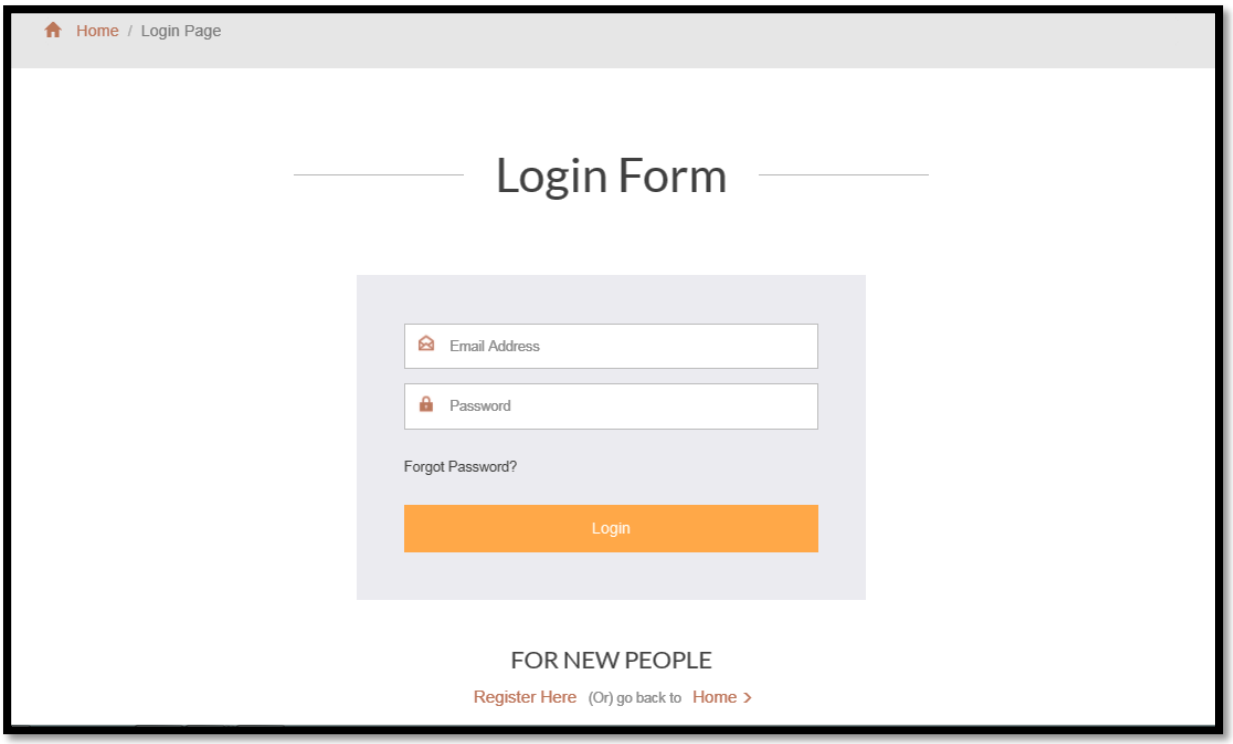


Figure 5.2.4: Login page

5.2.4 CREATE POST:

User can create post for Blood. The fill the form and post it.

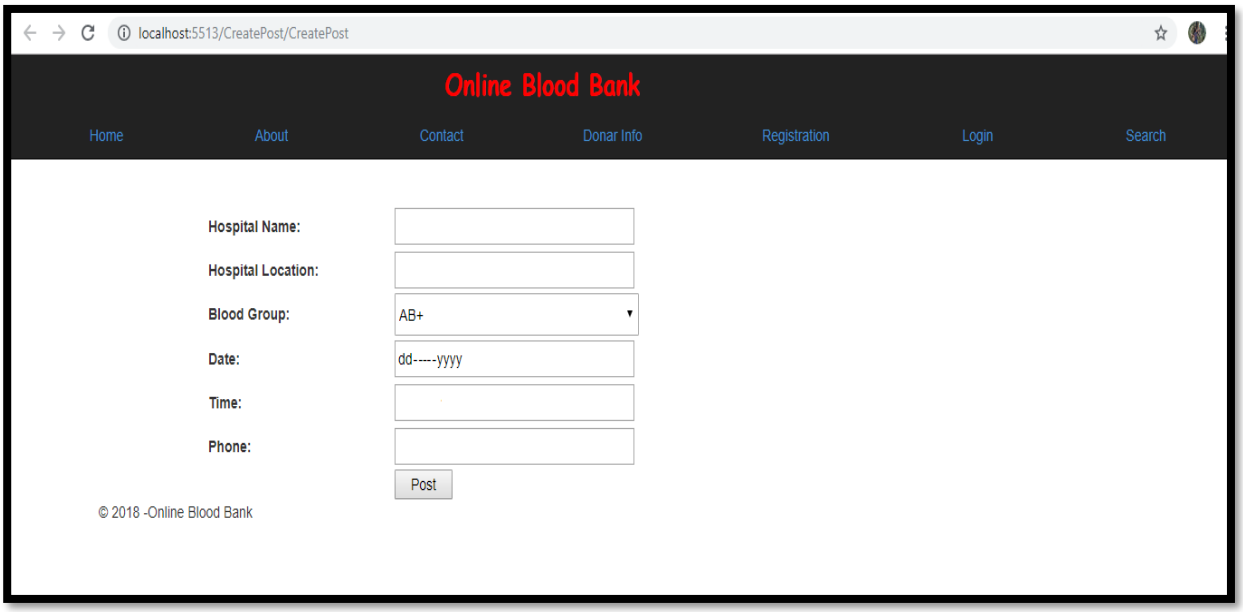


Figure 5.2.5: Create Post

6. TESTING

The testing of the software was done in full manual end user data flow testing style. Due to unavailability of test providing platform, automated software derived testing could not be performed. The testing approach is narrated here omitting the technical details.

Two common type of testing is black box testing and white box testing. Black box testing is also called functional testing. In this stage we test only functionality, input, and output. White box testing is structure level testing. For this project I have used black box testing method.

6.1 Testing Strategy:

A testing strategy is a general approach to the testing process rather than a method of devising particular system or component tests. Different testing strategies may be adopted depending on the type of system to be tested and the development process used.

6.2 Test approach:

A test approach is the test strategy implementation of a project, defines how testing would be carried out. Test approach has two techniques:

- **Proactive -** An approach in which the test design process is initiated as early as possible in order to find and fix the defects before the build is created.
- **Reactive -** An approach in which the testing is not started until after design and coding are completed.

6.3 Black Box Testing

Black box testing also called functional testing that ignores the internal mechanism of a system or component and focuses on the outputs generated in response to selected inputs and execution conditions. We have decided to perform equivalence partitioning and Boundary value analysis for this system.

6.4 Equivalence Class Partitioning

In considering the inputs for our equivalence testing, the following types will be used: Legal Input values: Test values within boundaries of the specification equivalence classes. This will be input data the program expects and is programmed to transform into usable values. Illegal Input Values: Test equivalence classes outside the boundaries of the specification. This will be input data the program may be presented, but that will not produce any meaningful output

6.5 White Box Testing

White box testing is a software testing method in which the internal structure /implementation of the item being tested is known to the tester. The tester chooses inputs to exercise paths through the code and determines the appropriate outputs. Programming knowhow and the implementation knowledge is essential.

6.6 Pass / Fail Criteria

The entrance criteria for each phase of testing must be met before the next phase can commence. Now the criteria for pass and fail are given below.

- ✓ According to the given scenario the expected result need to take place then the scenario will be considered as pass otherwise that criteria should be failed.
- ✓ If an item tested 10 times, 9 times perfectly worked and single time do not work properly then it will consider as fail case.
- ✓ System crash will be considered as fail case.
- ✓ After submitting a query in the system, if expected page won't appear then it will be considered as fail case.

6.7 Testing Environment

Testing environment is a setup of software and hardware for the testing teams to execute test cases. In other words, it supports test execution with hardware, software and network configured.

For test environment, key area to set up includes.

- System and applications
- Test data
- Database server
- Front end running environment
- Client operating system
- Browser
- Hardware includes Server Operating system
- Network
- Documentation required like reference documents/configuration guides/installation guides/ user manuals

6.8 Test case:

A test case is a document, which has a set of test data, preconditions, expected results and post conditions, developed for a particular test scenario in order to verify compliance against a specific requirement

Table 6.8.1: Test case for Donor Registration

Test Case ID: TC 01	**Module Name: Registration**
Sub Module: Donor Registration	**Test Designed by:** Md Mojadded
Test Priority (Low/Medium/High): High	**Test Designed date:** 05.08.2018
Test Title:Donor Registration with valid information	**Test Executed by:** Md Mojadded
Description: Test the system's on registration page	**Test Execution date:** 10.08.2018

Pre-conditions: The user navigate to registration page and input the required filled. And click on the Register button.

Step	**Test Steps**	**Test Data**	**Code Module**	**Expected Result**	**Actual Result**	**Pass/Fail**
1	**Navigate to Registration Page**	Click on Registration Tab	**RegInfo.aspx**	Donor should be registered successfully	Donor navigate or redirect to home page.	Pass
2	**Input NID**	1994131496500005 4				
3	**Auto Input Name**	Md Mojadded				
4	**Auto Input Father's Name**	XYZ				
5	**Auto Input**	M				
6	**Address**	Dhanmondi 32, Dhaka 1207				
7	**Date of birth**	17-05-1976				
8	**Select Registration as**	Donor				
9	**Input Phone**	01632053996				

10	**Blood Group**	AB+				
11	**Present Address Address**	Dhanmondi 32, Dhaka 1207				
13	**UserName**	mojaddedkawsar				
14	**Email**	mojaddedkawsar@gmail.com				
15	**Password**	123@123				
16	**Confirm password**	123@123				
17	**Confirmation code**	875634				
Post-conditions: if the user information's are valid then the information will saved in the database otherwise show the invalid fields.						

6.8.2 Test Case for Blood Seeker Registration:

Test Case ID: TC 02	**Module Name: Registration**
Sub Module: Blood Seeker Registration	**Test Designed by:** Md Mojadded
Test Priority (Low/Medium/High): High	**Test Designed date:** 05.08.2018
Test Title:Donor Registration with valid information	**Test Executed by:** Md Mojadded
Description: Test the system's on registration page	**Test Execution date:** 10.08.2018

Pre-conditions: The user navigate to registration page and input the required filled. And click on the Register button.

Step	**Test Steps**	**Test Data**	**Code Module**	**Expected Result**	**Actual Result**	**Pass/Fail**
1	**Input Name**	Md Mojadded	BloodSeekerReg.aspx	Blood Seeker should be registered successfully	Blood Seeker redirect to the Seeker profile	Pass
2	**Input Phone**	01632053996				
3	**Input Mail**	mojaddedkawsar@gmail.com				
4	**UserName**	mojaddedkawsar				
5	**Password**	123@123				
6	**Confirm password**	123@123				
7	**Confirmation code**	875634				

Post-conditions: if the user information's are valid then the information will saved in the database otherwise show the invalid fields.

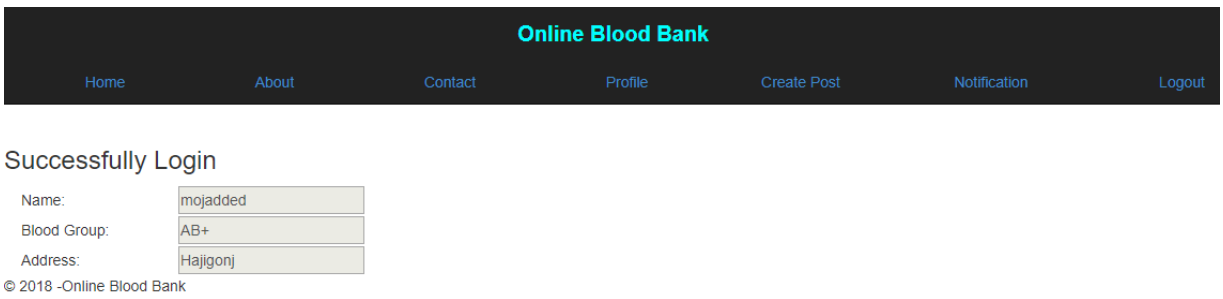
6.8.3 Test Case for Donor Login:

Test Case ID: TC 03	**Module Name: Login**
Sub Module: Donor Login	**Test Designed by:** Md Mojadded
Test Priority (Low/Medium/High): High	**Test Designed date:** 05.08.2018
Test Title: Donor Login with valid email/user name and password	**Test Executed by:** Md Mojadded
Description: Test the system's Login page	**Test Execution date: 1**0.08.2018

Pre-conditions: The user has valid email and password.
The current email is mojadded695@diu.edu.bd and password 123@123.
The system navigates to Login page. And click on the Login button.

Step	Test Steps	Test Data	Code Module	Expected Result	Actual Result	Pass/Fail
1	**Navigate to Login Page**	Click on Login Tab	**Login.aspx**	Donor should be able to login successfully.	Donor navigate or Home Page successfully.	Pass
2	**Input Email and it must be Unique**	Mojadded695@diu.edu.bd				
3	**Input Password**	123@123				
4	**Click on Signup Button**					

Post-conditions: Donor is validated with database and successfully login to account. The account session details are logged in database.



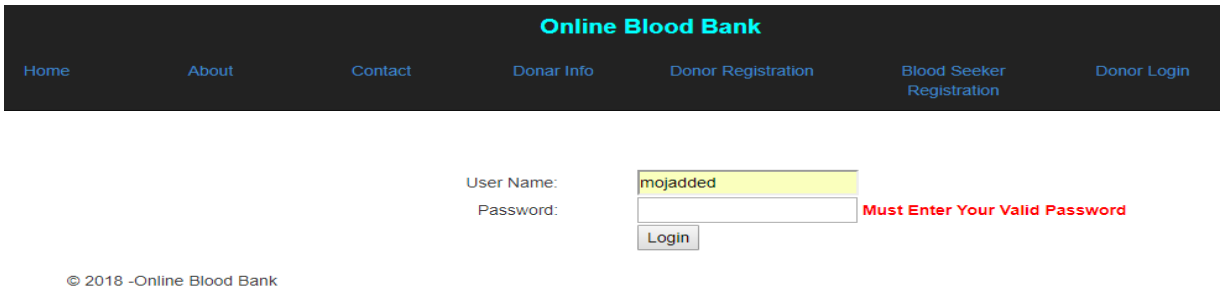
6.8.4 Test Case for Donor Login failed:

Test Case ID: TC 04	**Module Name: Login**
Sub Module: Donor Login	**Test Designed by:** Md Mojadded
Test Priority (Low/Medium/High): High	**Test Designed date:** 05.08.2018
Test Title: Donor Login with valid email/user name and password	**Test Executed by:** Md Mojadded
Description: Test the system's Login page	**Test Execution date: 1**0.08.2018

Pre-conditions: The user has valid email and password.
The current email is mojadded695@diu.edu.bd and password 123@123.
The system navigates to Login page. And click on the Login button.

Step	Test Steps	Test Data	Code Module	Expected Result	Actual Result	Pass/Fail
1	**Navigate to Login Page**	Click on Login Tab	**Login.aspx**	Donor should not be able to login successfully .	Donor navigate or redirect to the login page with error message	Fail
2	**Input Email and it must be Unique**	Mojadded@diu.edu.bd				
3	**Input Password**	123@890				
4	**Click on Signup Button**					

Post-conditions: Donor is validated with database and successfully login to account. The account session details are logged in database.



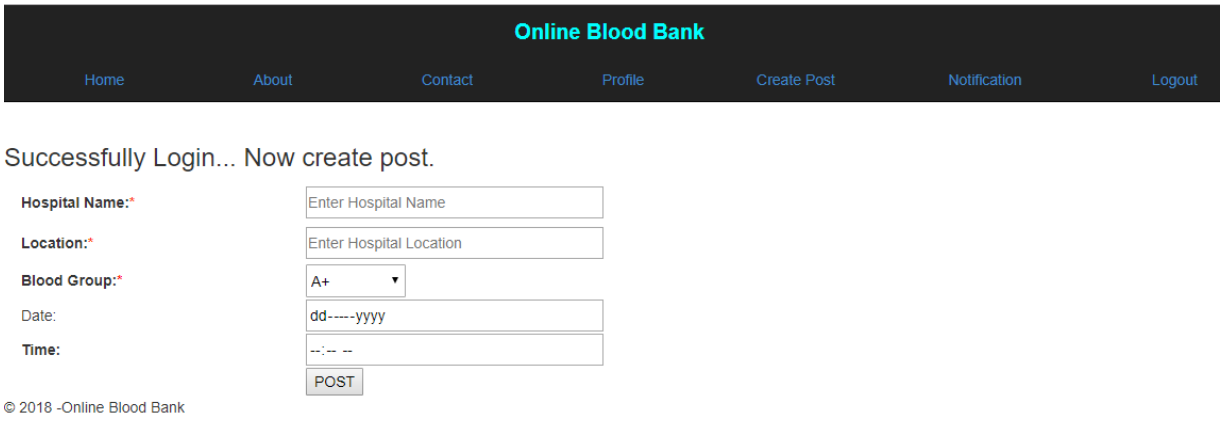
6.8.5 Test Case for Blood Seeker Login:

Test Case ID: TC 05	**Module Name: Login**
Sub Module: Seeker Login	**Test Designed by:** Md Mojadded
Test Priority (Low/Medium/High): High	**Test Designed date:** 05.08.2018
Test Title: Blood seeker Login with valid email/user name and password	**Test Executed by:** Md Mojadded
Description: Test the system's Login page	**Test Execution date: 1**0.08.2018

Pre-conditions: The user has valid email and password.
The current email is mojaddedkawsar@gmail.com and password 123@123.
The system navigates to Login page. And click on the Login button.

Step	Test Steps	Test Data	Code Module	Expected Result	Actual Result	Pass/Fail
1	**Navigate to Login Page**	Click on Login Tab	**Login.aspx**	seeker should be able to login successfully .	Seeker navigate or Home Page successfully .	Pass
2	**Input Email and it must be Unique**	Mojaddedkawsar@gmail.com				
3	**Input Password**	123@123				
4	**Click on Signup Button**					

Post-conditions: seeker is validated with database and successfully login to account. The account session details are logged in database.



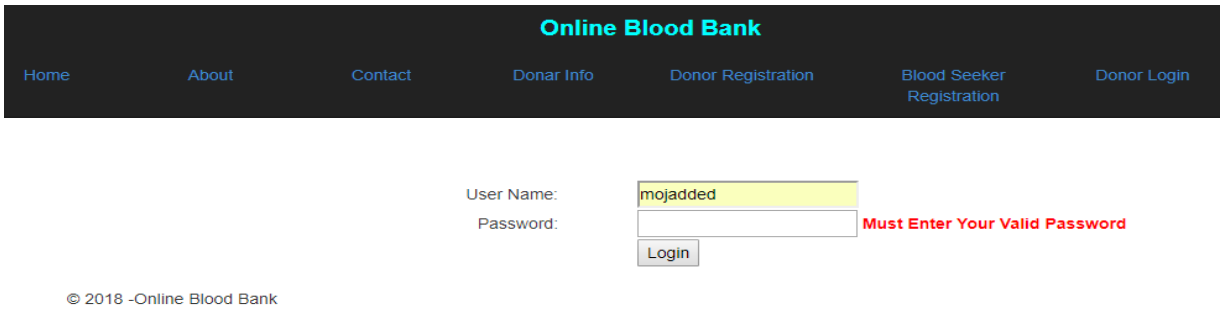
6.8.6 Test Case for Blood Seeker Login Failed:

Test Case ID: TC 06	**Module Name: Login**
Sub Module: Blood Seeker Login	**Test Designed by:** Md Mojadded
Test Priority (Low/Medium/High): High	**Test Designed date:** 05.08.2018
Test Title: Donor Login with valid email/user name and password	**Test Executed by:** Md Mojadded
Description: Test the system's Login page	**Test Execution date: 1**0.08.2018

Pre-conditions: The user has valid email and password.
The current email is **mojaddedkawsar@gmail.com** and password 123@123.
The system navigates to Login page. And click on the Login button.

Step	Test Steps	Test Data	Code Module	Expected Result	Actual Result	Pass/Fail
1	**Navigate to Login Page**	Click on Login Tab	**Login.aspx**	Blood Seeker should not be able to login successfully .	Blood Seekernavigate or redirect to the login page with error message	Fail
2	**Input Email and it must be Unique**	Mojadded@diu.edu.bd				
3	**Input Password**	123@123				
4	**Click on Signup Button**					

Post-conditions: Seeker is validated with database and successfully login to account. The account session details are logged in database.



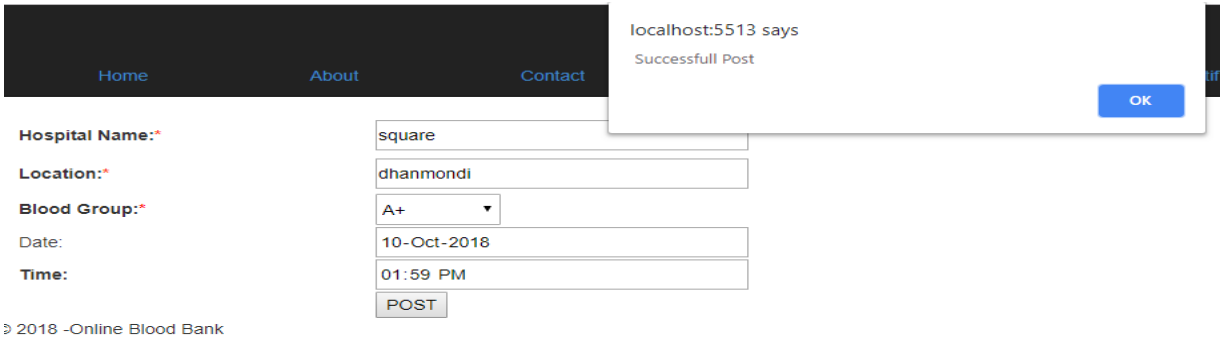
6.8.7 Test Case for Create Post:

Test Case ID: TC 07	**Module Name:** Create Post
Sub Module: Create Post	**Test Designed by:** Md Mojadded
Test Priority (Low/Medium/High): High	**Test Designed date:** 06.08.2018
Test Title: Seeker Login with valid email/user name and password. Then Full fil the required all field.	**Test Executed by:** Md Mojadded
Description: Test the system's Create post	**Test Execution date:** 10.08.2018

Pre-conditions: The user has valid email and password. The current email is **mojaddedkawsar@gmail.com** and password 123@123.
The system navigates to Login page. And click on the Login button.

Step	Test Steps	Test Data	Code Module	Expected Result	Actual Result	Pass/Fail
1	Navigate to Create Post	Click on Create post Tab	CreatePost .aspx	Blood Seeker successfully create post.	Blood Seeker see successful message & redirect to the profile page	Pass
2	Input Hospital Name	Squre Hospital				
3	Input Location	Dhanmondi				
4	Blood Group	AB+				
5	Date	15-08-2018				
6	Time	6.00 pm				
4	Click on Submit Button					

Post-conditions: Seeker is validated with database and successfully login to account. The account session details are logged in database.



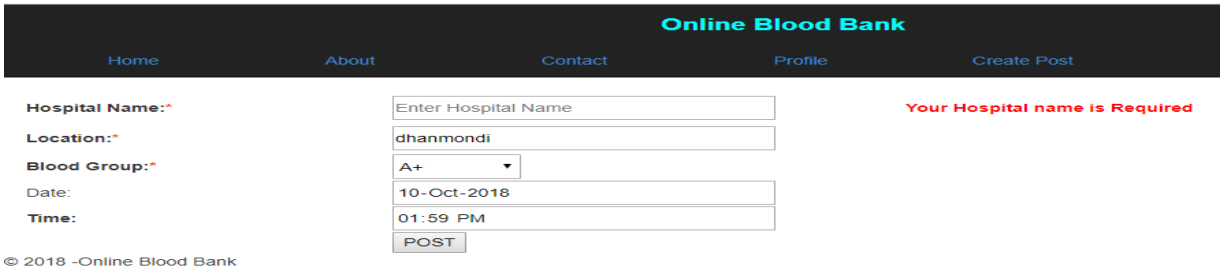
6.8.8 Test Case for Blood Seeker Create Post failed:

Test Case ID: TC 07	**Module Name:** Create Post
Sub Module: Create Post	**Test Designed by:** Md Mojadded
Test Priority (Low/Medium/High): High	**Test Designed date:** 06.08.2018
Test Title: Seeker Login with valid email/user name and password. Then Full fil the required all field.	**Test Executed by:** Md Mojadded
Description: Test the system's Create post	**Test Execution date:** 10.08.2018

Pre-conditions: The user has valid email and password.
The current email is **mojaddedkawsar@gmail.com** and password 123@123.
The system navigates to Login page. And click on the Login button.

Step	Test Steps	Test Data	Code Module	Expected Result	Actual Result	Pass/Fail
1	Navigate to Create Post	Click on Create post Tab	CreatePost .aspx	Blood Seeker can't successfully create post.	Blood Seeker see an error message	Fail
2	Input Hospital Name	Squre Hospital				
3	Input Location					
4	Blood Group	AB+				
5	Date	15-08-2018				
6	Time	6.00 pm				
4	Click on Submit Button					

Post-conditions: Seeker is validated with database and successfully login to account. The account session details are logged in database.



6.9 TEST REPORT

Generally, this is a communication sent out to establish transparency to the QA team's activities of the day during the test cycle – includes both defect information and test case run information.

Total unit of test case sample are 7. Some test cases are succeeding in 1st iteration and some are in second. The succession percent are shown in following table:

Table 6.9.1: Test Report

Number of Unit Test Case	100% Success in first iteration	Less than 100%	Total Succession %
Total : 7	6	3	80%
Total : 9	9	0	100%

7 CONCLUSION

7.1 Project Summary:

This project has been started from June. From that beginning time I have to work hard to know the requirement clearly. After that I proposed a design to them by help of my supervisor. I started to develop the project. From then I gradually develop the project. I think storing the data in database neatly is very important. That's why I did this first and made a relationship with the tables. After that I design the UI. This project's UI is very simple and clean which is very help for the user's experience. Then I started coding and executing the project. If I did not test this project, there will stay some bug on this project which will ruin the full project. That why give importance to test this project and then I solved some bug which I got after testing this project.

7.2 Limitations:

- ✓ Password is to encrypted.
- ✓ Donor Current Location can't track.
- ✓ Not Fully Responsive

8 FUTURE IMPROVEMENT

This application avoids the manual work and the problems concern with it. Well I have worked hard in order to present the website. Still, I found out that the project can be done in a better way.

Due to time and technology concern I am unable to deploy the current location trace and notification send the seeker or donor phone and other small features. For that reason, I am going to develop new features like following:

- ✓ Real Time Location Trace.
- ✓ Send Notification to the Donor or Blood Seeker Phone.

9 IMPLEMENTATION CODE:

I explain my implementation code here. This code is design and logic also.

9.1SEEKER LOGIN ASPX CODE:

```
10      <table class="nav-justified">
11          <tr>
12              <td style="width: 383px" class="text-right">User Name:</td>
13              <td>
14                  <asp:TextBox ID="TextSUserName" runat="server"
    Width="179px"></asp:TextBox>
15                  <asp:RequiredFieldValidator ID="RequiredFieldValidator1"
    runat="server" ControlToValidate="TextSUserName" ErrorMessage="Must Enter Your User
    Name" Font-Bold="True" ForeColor="Red"></asp:RequiredFieldValidator>
16              </td>
17              <td> </td>
18          </tr>
19          <tr>
20              <td style="width: 383px" class="text-right">Password:</td>
21              <td>
22                  <asp:TextBox ID="TextPass" runat="server" Width="178px"
    TextMode="Password"></asp:TextBox>
23                  <asp:RequiredFieldValidator ID="RequiredFieldValidator2"
    runat="server" ControlToValidate="TextPass" ErrorMessage="Must Enter Your Valid
    Password" Font-Bold="True" ForeColor="Red"></asp:RequiredFieldValidator>
24              </td> <td> </td> </tr>
25          <tr>
26              <td style="width: 383px" class="text-right"> </td>
27              <td>
28    <asp:Button ID="BtnLogin" runat="server" Text="Login" OnClick="BtnLogin_Click" />
29              <td> </td>
30          </tr>
31      </table>
32      <br />
33  </asp:Content>
```

9.2 SEEKER LOGIN ASPX.CS CODE:

```
10  using System;
11  using System.Collections.Generic;
12  using System.Linq;
13  using System.Web;
14  using System.Web.UI;
15  using System.Web.UI.WebControls;
16  using System.Data;
17  using System.Configuration;
18  using System.Data.SqlClient;
19  using System.Data.Sql;
20  using BloodInfo1.Models;
21
22  namespace BloodInfo1.SeekerLogin
23  {
24      public partial class seekerLogin : System.Web.UI.Page
25      {
26          protected void Page_Load(object sender, EventArgs e)
27          {
28
29          }
30
31          protected void BtnLogin_Click(object sender, EventArgs e)
32          {
```

```
33              online_blood_bankEntities db = new online_blood_bankEntities();
34
35              tb_SeekerProfile info = db.tb_SeekerProfile.FirstOrDefault(x =>
    x.UserName == TextSUserName.Text);
36              if (info != null && info.UserName == TextSUserName.Text &&
    info.Password == TextPass.Text)
37              {
38                  Session["userid"] = info.id;
39                  Response.Redirect("~/CreatePost/CreatePost.aspx");
40              }
41              else
42
43                  Response.Write("User Name or Password is not correct!");
44
45          }
46      }
47  }
```

9.3 DONOR LOGIN ASPX CODE:

```
10  <asp:Content ID="Content1" ContentPlaceHolderID="MainContent" runat="server">
11      <table class="nav-justified">
12          <tr>
13              <td style="width: 383px" class="text-right">User Name:</td>
14              <td>
15                  <asp:TextBox ID="TextUserName" runat="server"
    Width="179px"></asp:TextBox>
16                  <asp:RequiredFieldValidator ID="RequiredFieldValidator1"
    runat="server" ControlToValidate="TextUserName" ErrorMessage="Must Enter Your User
    Name" Font-Bold="True" ForeColor="Red"></asp:RequiredFieldValidator>
17              </td>
18              <td> </td>
19          </tr>
20          <tr>
21              <td style="width: 383px" class="text-right">Password:</td>
22              <td>
23                  <asp:TextBox ID="TextPassword" runat="server" Width="178px"
    TextMode="Password"></asp:TextBox>
24                  <asp:RequiredFieldValidator ID="RequiredFieldValidator2"
    runat="server" ControlToValidate="TextPassword" ErrorMessage="Must Enter Your Valid
    Password" Font-Bold="True" ForeColor="Red"></asp:RequiredFieldValidator>
25              </td>
26              <td> </td>
27          </tr>
28          <tr>
29              <td style="width: 383px" class="text-right"> </td>
30              <td>
31                  <asp:Button ID="Btnlogin" runat="server" Text="Login"
    OnClick="Btnlogin_Click" />
32              </td>
33              <td> </td>
34          </tr>
35      </table>
36      <br />
37
38  </asp:Content>
```

9.4 DONOR LOGIN ASPX .CS CODE:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Configuration;
using System.Data.SqlClient;
using System.Data.Sql;
using BloodInfo1.Models;

namespace BloodInfo1.Login
{
    public partial class Login : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        { }

        protected void Btnlogin_Click(object sender, EventArgs e)
        {
            online_blood_bankEntities db = new online_blood_bankEntities();

            tb_DonorInfo info = db.tb_DonorInfo.FirstOrDefault(x => x.UserName == TextUserName.Text);
            if (info != null && info.UserName == TextUserName.Text && info.Password == TextPassword.Text)
            {
                Session["NID"] = info.NID;
                Session["userid"] = info.DID;
                Response.Redirect("~/Profile/Profile.aspx");
            }
            else

                Response.Write("User Name or Password is not correct!");

        }
    }
}
```

9.5 PROFILE ASPX CODE:

```
    <div>
        <h3 class="center"> Successfully Login</h3>
     <table>
 <tr>  <td> Name: </td>
  <td><asp:TextBox ID="textName" runat="server" Enabled="False"></asp:TextBox>
</td> </tr>

<tr> <td> Blood Group: </td>
<td><asp:TextBox ID="textBg" runat="server"  Enabled="False"></asp:TextBox></td>
   </tr>  <tr>
  <td>Address: </td>
  <td><asp:TextBox ID="textAddress" runat="server"  Enabled="False"></asp:TextBox>
</td></tr>     </table>    </div>
```

9.6 PROFILE ASPX.CS CODE:

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

using System.Data;
using System.Configuration;
using System.Data.SqlClient;
using System.Data.Sql;
using BloodInfo1.Models;

namespace BloodInfo1.Profile
{
    public partial class profile : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (Session["NID"] != null)
            {
                string nid = (string)(Session["NID"]);
                online_blood_bankEntities db = new online_blood_bankEntities();
                tb_UserInfo  userinfo = db.tb_UserInfo.FirstOrDefault(x => x.NID == nid);
                if (userinfo != null)
                {
                    textName.Text = userinfo.Name;
                    textBg.Text = userinfo.BloodGroup;
                    textAddress.Text = userinfo.Address;
                }

            }
            else
            {
                Response.Redirect("~/Login/Login.aspx");
            }
        }
    }
}
```

9.7 CREATE POST ASPX CODE:

```
    <div>
        <h3 class="centre">Successfully Login... Now create post.</h3>
        <table>
            <tr>
                <td class="text-left" style="width: 206px; font-weight: bold; text-align: left; height: 38px;">Hospital Name:<span style="color: #FF3300">*</span></td>
                <td style="width: 437px; height: 38px;">
                    <asp:TextBox ID="TextHName" placeholder="Enter Hospital Name" runat="server" Height="30px" Width="280px"></asp:TextBox>
                </td>
                <td style="height: 38px">
                    <asp:RequiredFieldValidator ID="RequiredFieldValidator3" runat="server" ErrorMessage="Your Hospital name is Required" ControlToValidate="TextHName" Font-Bold="True" ForeColor="Red"></asp:RequiredFieldValidator>
                </td>
            </tr>
            <tr>
```

```
23                  <td class="text-left" style="width: 206px; font-weight: bold; text-
    align: left; height: 38px;">Location:<span style="color: #FF3300">*</span></td>
24                  <td style="width: 437px; height: 38px;">
25                      <asp:TextBox ID="TextHlocation" placeholder="Enter Hospital
    Location" runat="server" Height="30px" Width="280px"></asp:TextBox>
26                  </td>
27                  <td style="height: 38px">
28                      <asp:RequiredFieldValidator ID="RequiredFieldValidator1"
    runat="server" ErrorMessage="Your Location is Required"
    ControlToValidate="TextHlocation" Font-Bold="True"
    ForeColor="Red"></asp:RequiredFieldValidator>
29                  </td>
30              </tr>
31              <tr>
32                  <td class="text-left" style="width: 206px; font-weight: bold; text-
    align: left;">Blood Group:<span style="color: #FF0000">*</span></td>
33                <td>
34                    <asp:DropDownList ID="BloodDropDown" runat="server" Height="31px"
    ValidateRequestMode="Disabled" Width="94px">
35                      <asp:ListItem Value="A+">A+</asp:ListItem>
36                      <asp:ListItem Value="A-">A-</asp:ListItem>
37                      <asp:ListItem Value="B+">B+</asp:ListItem>
38                      <asp:ListItem Value="B-">B-</asp:ListItem>
39                      <asp:ListItem Value="O+">O+</asp:ListItem>
40                      <asp:ListItem Value="O-">O-</asp:ListItem>
41                      <asp:ListItem Value="AB+">AB+</asp:ListItem>
42                      <asp:ListItem Value="AB-">AB-</asp:ListItem>
43                  </asp:DropDownList>
44                  </td>
45
46              </tr>
47              <tr>
48                  <td>Date:</td>
49                  <td class="text-left" style="width: 206px; font-weight: bold; text-
    align: left;">
50                      <asp:TextBox ID="TextDate"  runat="server" Height="30px"
    Width="280px" TextMode="Date"></asp:TextBox>
51                  </td>
52                  <td style="width: 437px">
53                       </td>
54                  <td> </td>
55              </tr>
56              <tr>
57                  <td class="text-left" style="width: 206px; font-weight: bold; text-
    align: left;">Time:</td>
58                  <td style="width: 437px">
59                      <asp:TextBox ID="TextTime"  runat="server" Height="30px"
    Width="280px" TextMode="Time"></asp:TextBox>
60                  </td>
61                  <td> </td>
62              </tr>
63             <tr>
64                 <td></td>
65                 <td>
66                     <asp:Button ID="BtnPost" runat="server" Text="POST"
    OnClick="BtnPost_Click" />
67                  </td>
68              </tr>
69          </table>
70      </div>
```

9.8 CREATE POST ASPX.CS CODE:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Configuration;
using System.Data.SqlClient;
using System.Data.Sql;
using BloodInfo1.Models;
namespace BloodInfo1.CreatePost
{
    public partial class CreatePost : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void BtnPost_Click(object sender, EventArgs e)
        {

            online_blood_bankEntities db = new online_blood_bankEntities();
            tb_PostInfo model = new tb_PostInfo();


            model.BloodGroup = BloodDropDown.SelectedValue;
            model.Date = TextDate.Text;
            model.Time =
TimeSpan.FromTicks(Convert.ToDateTime(TextTime.Text).Hour);
            model.Location = TextHlocation.Text;
            model.HospitalName = TextHName.Text;

            db.tb_PostInfo.Add(model);

            var result = from user in db.tb_UserInfo
                         join donar in db.tb_DonorInfo on
                         user.NID equals donar.NID
                         where user.BloodGroup == model.BloodGroup
                         select new { NID =  donar.NID,
                                      donarId = donar.DID   };

            foreach (var item in  result.ToList())
            {
                tb_Notification nmodel = new tb_Notification();
                 if(Session["userid"] != null) {
                    int sid = (int)(Session["userid"]);
                    nmodel.SID = sid;
                    nmodel.DID = item.donarId;
                    nmodel.Location = model.HospitalName + " "+ model.Location + "
"  + model.BloodGroup + model.Date.ToString() + model.Time.ToString()  ;
                    nmodel.BloodGroup = model.BloodGroup;
                     db.tb_Notification.Add(nmodel);
                     db.SaveChanges();
                }
            }

             Page.ClientScript.RegisterStartupScript(this.GetType(), "Scripts",
"<script>alert(' Successfull Post')</script>");
        }
    }
}
```

9.9 DONOR REGISTRATION ASPX CODE:

```
10        <script type="text/javascript"
    src="http://ajax.googleapis.com/ajax/libs/jquery/1.3.2/jquery.min.js"></script>
11  <script type="text/javascript">
12      $(function () {
13          $('input:text#<%= TextNID.ClientID  %>').keyup(function () {
14
15              var nid=this.value;
16              $.ajax({
17                  type: "GET",
18                  url: "DonorReg.aspx/check",
19                  data: dataValue,
20                  error: function (XMLHttpRequest, textStatus, errorThrown) {
21                      alert("Request: " + XMLHttpRequest.toString() + "\n\nStatus: "
    + textStatus + "\n\nError: " + errorThrown);
22                  },
23                  complete: function (jqXHR, status) {
24                      alert("complete: " + status + "\n\nResponse: " +
    jqXHR.responseText);
25                  }
26              });
27          });
28      });
29  </script>
30
31      <div class="row">
32
33          <div>
34              <table>
35  <tr>
36   <td class="text-left" style="width: 206px; font-weight: bold; text-align:
    left;">NId:<span style="color: #FF0000">*</span></td>
37                  <td style="width: 437px"><asp:TextBox ID="TextNID"
    placeholder="Enter your NID" runat="server" Height="30px"
    Width="280px"></asp:TextBox>
38  </td>
39    <td> <asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server"
    ErrorMessage="Enter Your NID " ControlToValidate="TextPresentAddress" Font-
    Bold="True" ForeColor="Red"></asp:RequiredFieldValidator>
40  </td>
41  </tr><tr>
42  <td class="text-left" style="width: 206px; font-weight: bold; text-align:
    left;">Present Address:<span style="color: #FF0000">*</span></td>
43   <td style="width: 437px">
44       <asp:TextBox ID="TextPresentAddress" placeholder="Enter your present address"
    runat="server" TextMode="MultiLine" Height="70px" Width="280px"></asp:TextBox>
45    </td>
46     <td>
47    <asp:RequiredFieldValidator ID="RequiredFieldValidator6" runat="server"
    ErrorMessage="Enter Your Present Address" ControlToValidate="TextPresentAddress"
    Font-Bold="True" ForeColor="Red"></asp:RequiredFieldValidator>
48                  </td>
49              </tr>
50              <tr>
51                  <td class="text-left" style="width: 206px; font-weight: bold; text-
    align: left;">Phone Number:<span style="color: #FF0000">*</span></td>
52                  <td style="width: 437px">
53                      <asp:TextBox ID="TextPhone" placeholder="Enter your
    phone/mobile number"  runat="server" Height="30px" Width="280px"
    TextMode="Phone"></asp:TextBox>
54                  </td>
55                  <td>
```

```
56                      <asp:RequiredFieldValidator ID="RequiredFieldValidator7"
    runat="server" ErrorMessage="Enter your phone number" ControlToValidate="TextPhone"
    Font-Bold="True" ForeColor="Red"></asp:RequiredFieldValidator>
57                  </td>
58              </tr>
59              <tr>
60                  <td class="text-left" style="width: 206px; font-weight: bold; text-
    align: left;">Email:<span style="color: #FF0000">*</span></td>
61                  <td style="width: 437px">
62                      <asp:TextBox ID="TextEmail" placeholder="Enter your email
    address"  runat="server" TextMode="Email" Height="30px"
    Width="280px"></asp:TextBox>
63                  </td>
64                  <td>
65                      <asp:RequiredFieldValidator ID="RequiredFieldValidator8"
    runat="server" ErrorMessage="Enter Your Mail Address" ControlToValidate="TextEmail"
    Font-Bold="True" ForeColor="Red"></asp:RequiredFieldValidator>
66                      <br />
67                      <asp:RegularExpressionValidator
    ID="RegularExpressionValidator1" runat="server" ControlToValidate="TextEmail"
    ErrorMessage="You must enter the valid email" Font-Bold="True" ForeColor="Red"
    ValidationExpression="\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-
    .]\w+)*"></asp:RegularExpressionValidator>
68                  </td>
69              </tr>
70              <tr>
71                  <td class="text-left" style="width: 206px; height: 22px; font-
    weight: bold; text-align: left;">User Name:<span style="color:
    #FF0000">*</span></td>
72                  <td style="width: 437px; height: 22px">
73                      <asp:TextBox ID="TextUserName" placeholder="Enter user name"
    runat="server" Height="30px" Width="280px"></asp:TextBox>
74                  </td>
75                  <td style="height: 22px">
76                      <asp:RequiredFieldValidator ID="RequiredFieldValidator9"
    runat="server" ErrorMessage="Username is Required" ControlToValidate="TextUserName"
    Font-Bold="True" ForeColor="Red"></asp:RequiredFieldValidator>
77                  </td>
78              </tr>
79              <tr>
80                  <td class="text-left" style="width: 206px; font-weight: bold; text-
    align: left;">Password:<span style="color: #FF0000">*</span></td>
81                  <td style="width: 437px">
82                      <asp:TextBox ID="TextPassword" placeholder="Choice a password"
    runat="server" TextMode="Password" Height="30px" Width="280px"></asp:TextBox>
83                  </td>
84                  <td>
85                      <asp:RequiredFieldValidator ID="RequiredFieldValidator10"
    runat="server" ErrorMessage="You mast have to fill with pass"
    ControlToValidate="TextPassword" Font-Bold="True"
    ForeColor="Red"></asp:RequiredFieldValidator>
86                  </td>
87              </tr>
88              <tr>
89                  <td class="text-left" style="width: 206px; height: 24px; font-
    weight: bold; text-align: left;">Confirm Password:<span style="color:
    #FF0000">*</span></td>
90                  <td style="width: 437px; height: 24px;">
91                      <asp:TextBox ID="TextRpass" placeholder="Re-enter your
    password"  runat="server" TextMode="Password" Height="30px"
    Width="280px"></asp:TextBox>
92                  </td>
93                  <td style="height: 24px">
```

```
94                      <asp:RequiredFieldValidator ID="RequiredFieldValidator11"
    runat="server" ErrorMessage="re-enter your password" ControlToValidate="TextRpass"
    Font-Bold="True" ForeColor="Red"></asp:RequiredFieldValidator>
95                      <br />
96                      <asp:CompareValidator ID="CompareValidator1" runat="server"
    ControlToCompare="TextPassword" ControlToValidate="TextRpass"
    ErrorMessage="Password Not match " Font-Bold="True"
    ForeColor="Red"></asp:CompareValidator>
97                  </td>
98              </tr>
99              <tr>
100                 <td class="text-right" style="width: 206px; font-weight: bold;
    text-align: left;"> </td>
101                 <td style="width: 437px">
102                     <asp:Button ID="BtnSubmit" runat="server" Height="37px"
    Text="Submit" Width="80px" OnClick="BtnSubmit_Click" />
103                     <input id="Reset1"  style="padding: 2px; font-size: large;
    color: #000000; background-color: #00FFCC; font-weight: bold;" type="reset"
    value="Reset" /></td>
104                 <td> </td>
105             </tr>
106             <tr>
107                 <td class="text-right" style="width: 206px; font-weight:
    bold;"> </td>
108                 <td style="width: 437px">
109                     <asp:Label ID="Label1" runat="server" Text="Registration
    Successfull....." Font-Bold="True" ForeColor="#009933"
    Visible="False"></asp:Label>
110                 </td>
111                 <td> </td>
112             </tr>
113         </table>
114     </div>
115         </div>
```

9.10 DONOR REGISTRATION ASPX.CS CODE:

```
10  using System;
11  using System.Collections.Generic;
12  using System.Linq;
13  using System.Web;
14  using System.Web.UI;
15  using System.Web.UI.WebControls;
16
17  using System.Data;
18  using System.Configuration;
19  using System.Data.SqlClient;
20  using System.Data.Sql;
21
22  using BloodInfo1.Models;
23
24  namespace BloodInfo1.RegInfo
25  {
26      public partial class DonorReg : System.Web.UI.Page
27      {
28          protected void Page_Load(object sender, EventArgs e)
29          {
30
31
32          }
33
34          public void check()
35          {
36
```

```
        }

        protected void BtnSubmit_Click(object sender, EventArgs e)
        {
            online_blood_bankEntities db = new online_blood_bankEntities();

            tb_DonorInfo model = new tb_DonorInfo();
            model.PresentAddress = TextPresentAddress.Text;
            model.Phone = TextPhone.Text;
            model.mail = TextEmail.Text;
            model.UserName = TextUserName.Text;
            model.Password = TextPassword.Text;
            model.NID = TextNID.Text;

            db.tb_DonorInfo.Add(model);
            db.SaveChanges();

            Page.ClientScript.RegisterStartupScript(this.GetType(), "Scripts",
"<script>alert('Registration Successfull')</script>");
            Response.Redirect("~/Login/Login.aspx");
        }
    }
}
```

9.11 SEEKER REGISTRATION ASPX CODE:

```
    <div>
        <label class="centre"> Blood Seeker Registration</label>
        <table>
            <tr>
                <td class="text-left" style="width: 206px; font-weight: bold;
text-align: left; height: 38px;">Name:<span style="color: #FF3300">*</span></td>
                <td style="width: 437px; height: 38px;">
                    <asp:TextBox ID="TextName" placeholder="Enter your name"
runat="server" Height="30px" Width="280px"></asp:TextBox>
                </td>
                <td style="height: 38px">
                    <asp:RequiredFieldValidator ID="RequiredFieldValidator1"
runat="server" ErrorMessage="Your name is Required" ControlToValidate="TextName"
Font-Bold="True" ForeColor="Red"></asp:RequiredFieldValidator>
                </td>
            </tr>
            <tr>
                <td class="text-left" style="width: 206px; font-weight: bold;
text-align: left; height: 38px;">Phone Number:<span style="color:
#FF3300">*</span></td>
                <td style="width: 437px; height: 38px;">
                    <asp:TextBox ID="TextPhone" placeholder="Enter your phone
number" runat="server" Height="30px" Width="280px" TextMode="Phone"></asp:TextBox>
                </td>
                <td style="height: 38px">
                    <asp:RequiredFieldValidator ID="RequiredFieldValidator2"
runat="server" ErrorMessage="Your name is Required" ControlToValidate="TextName"
Font-Bold="True" ForeColor="Red"></asp:RequiredFieldValidator>
                </td>
            </tr>
            <tr>
                <td class="text-left" style="width: 206px; font-weight: bold;
text-align: left; height: 38px;">E-Mail:<span style="color: #FF3300">*</span></td>
                <td style="width: 437px; height: 38px;">
                    <asp:TextBox ID="Textmail" placeholder="Enter your mail"
runat="server" Height="30px" Width="280px" TextMode="Email"></asp:TextBox>
                </td>
                <td style="height: 38px">
```

```
37                      <asp:RequiredFieldValidator ID="RequiredFieldValidator3"
    runat="server" ErrorMessage="Your name is Required" ControlToValidate="TextName"
    Font-Bold="True" ForeColor="Red"></asp:RequiredFieldValidator>
38                  </td>
39              </tr>
40              <tr>
41                  <td class="text-left" style="width: 206px; height: 22px; font-
    weight: bold; text-align: left;">User Name:<span style="color:
    #FF0000">*</span></td>
42                  <td style="width: 437px; height: 22px">
43                      <asp:TextBox ID="TextUserName" placeholder="Enter user name"
    runat="server" Height="30px" Width="280px"></asp:TextBox>
44                  </td>
45                  <td style="height: 22px">
46                      <asp:RequiredFieldValidator ID="RequiredFieldValidator9"
    runat="server" ErrorMessage="Username is Required"
    ControlToValidate="TextUserName" Font-Bold="True"
    ForeColor="Red"></asp:RequiredFieldValidator>
47                  </td>
48              </tr>
49              <tr>
50                  <td class="text-left" style="width: 206px; font-weight: bold;
    text-align: left;">Password:<span style="color: #FF0000">*</span></td>
51                  <td style="width: 437px">
52                      <asp:TextBox ID="TextPassword" placeholder="Choice a password"
    runat="server" TextMode="Password" Height="30px" Width="280px"></asp:TextBox>
53                  </td>
54                  <td>
55                      <asp:RequiredFieldValidator ID="RequiredFieldValidator10"
    runat="server" ErrorMessage="You mast have to fill with pass"
    ControlToValidate="TextPassword" Font-Bold="True"
    ForeColor="Red"></asp:RequiredFieldValidator>
56                  </td>
57              </tr>
58              <tr>
59                  <td class="text-left" style="width: 206px; height: 24px; font-
    weight: bold; text-align: left;">Confirm Password:<span style="color:
    #FF0000">*</span></td>
60                  <td style="width: 437px; height: 24px;">
61                      <asp:TextBox ID="TextRpass" placeholder="Re-enter your
    password"  runat="server" TextMode="Password" Height="30px"
    Width="280px"></asp:TextBox>
62                  </td>
63                  <td style="height: 24px">
64                      <asp:RequiredFieldValidator ID="RequiredFieldValidator11"
    runat="server" ErrorMessage="re-enter your password" ControlToValidate="TextRpass"
    Font-Bold="True" ForeColor="Red"></asp:RequiredFieldValidator>
65                      <br />
66                      <asp:CompareValidator ID="CompareValidator1" runat="server"
    ControlToCompare="TextPassword" ControlToValidate="TextRpass"
    ErrorMessage="Password Not match " Font-Bold="True"
    ForeColor="Red"></asp:CompareValidator>
67                  </td>
68              </tr>
69              <tr>
70                  <td class="text-right" style="width: 206px; font-weight: bold;
    text-align: left;"> </td>
71                  <td style="width: 437px">
72                      <asp:Button ID="BtnSubmit" runat="server" Height="37px"
    Text="Submit" Width="80px" OnClick="BtnSubmit_Click" />
73                    </td>
74                  <td> </td>
75              </tr>
76          </table>
77      </div>
```

9.12 SEEKER REGISTRATION CS CODE:

```
using System;
using System.Collections.Generic;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace BloodInfo1.Blood_Seeker_Reg
{
    public partial class Blood_seeker_Reg : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void BtnSubmit_Click(object sender, EventArgs e)
        {
            String Reg = ConfigurationManager.ConnectionStrings["ConnStringDb1"].ConnectionString;

            SqlConnection set = new SqlConnection(Reg);


            if (set.State == ConnectionState.Closed)
            {
                set.Open();

            }

            string query = @"
            INSERT INTO [dbo].[tb_SeekerProfile]
           ([Name]
           ,[Phone]
           ,[UserName]
           ,[Password])
     VALUES
           ('" + TextName.Text + "','" + TextPhone.Text + "','" + Textmail.Text + "','" + TextUserName.Text + "','" + TextPassword.Text + "','" + "')";

            SqlCommand tr = new SqlCommand(query, set);
            tr.BeginExecuteNonQuery();

            TextName.Text = string.Empty;
            TextPhone.Text = string.Empty;
            Textmail.Text = string.Empty;
            TextUserName.Text = string.Empty;
            TextPassword.Text = string.Empty;
            TextRpass.Text = string.Empty;

            Page.ClientScript.RegisterStartupScript(this.GetType(), "Scripts", "<script>alert('Registration Successfull')</script>");
            Response.Redirect("~/SeekerLogin/seekerLogin.aspx");
        }
    }
}
```

9 APPENDIX

10 REFERENCES

[1]. http://stackoverflow.com/questions/16475979/what-is-functional-and-non-functional-requirement

[2].http://www.tutorialspoint.com/uml/uml_use_case_diagram.htm

[3]. http://en.wikipedia.org/wiki/Data_flow_diagram

[4]. http://www.tutorialspoint.com/software_testing_dictionary/test_case.htm

[5]. http://www.software-tester.ch/PDF-Files/ISTQB%20Glossary%20of%20Testing%20Terms%202.4.pdf

[6]. http://www.softwaretestinghelp.com/test-execution-report/

[7] https://docs.microsoft.com/en-us/aspnet/core/fundamentals/servers/?view=aspnetcore-2.1