CHAPTER 1

INTRODUCTION

In this chapter, Database and its types, advantages, components etc. are discussed.

1. DBMS:

A database management system (DBMS) refers to the technology for creating and managing databases. DBMS is a software tool to organize, create, retrieve, update and manage data in a database. Databases and database technology have a major impact on the growing use of computers. It is fair to say that databases play a critical role in almost all areas where computers are used including business, electronic, commerce, engineering, medicine, genetics, law, education, and library science.

The main aim of a DBMS is to supply a way to store up and retrieve database information that is both convenient and efficient. By data, we mean known facts that can be recorded and that have embedded meaning. Normally people use software such as DBASE IV or V, Microsoft ACCESS, or EXCEL to store data in the form of database. A datum is a unit of data. Meaningful data combined to form information. Hence, information is interpreted data-data provided with semantics. MS ACCESS is one of the most common examples of database management software. Database Management Systems are now as indispensable tool for managing information, and a course on the principles and practice of database systems is now an integral part of computer science curricular. This book covers the fundamentals of modern database management systems, in particular relational database systems.

1.1.1 Causes to use DBMS

· To develop software applications in less time.
· Data independence and efficient use of data.
· For uniform data administration.
· For data integrity and security.
· For concurrent access to data, and data recovery from crashes.
· To use user-friendly declarative query language.

1.1.2 Uses of DBMS

· **Airlines**: reservations, schedules, etc.
· **Telecom**: calls made customer details, network usage, etc.
· **Universities**: registration, results, grades, etc.
· **Sales**: products, purchases, customers, etc.
· **Banking**: all transactions etc.

1.1.3 Advantage of DBMS

A DBMS manage data and has many advantages. These are:
· **Data independence:** Application programs should be as free or independent as possible from details of data representation and storage. DBMS can supply an abstract view of the data for insulating application code from such facts.

· **Efficient data access:** DBMS utilizes a mixture of sophisticated concepts and techniques for storing and retrieving data competently, and this feature becomes important in cases where the data is stored on external storage devices.
· **Data integrity and security:** If data is accessed through the DBMS, the DBMS can enforce integrity constraints on the data.
· **Data administration:** When several users share the data, integrating the administration of data can offer major improvements. Experienced professionals understand the nature of the data being managed and can be responsible for organizing the data representation to reduce redundancy and make the data to retrieve efficiently.

1.1.4 Components of DBMS

· **Users:** Users may be of any kind such as DB administrator, System developer or database users.
· **Database application:** Database application may be Departmental, Personal, organization's and / or Internal.
· **DBMS:** Software that allows users to create and manipulate database access.
· **Database**: Collection of logical data as a single unit.
· **PHP (Hypertext Pre-processor)** is an open-source HTML-embedded server-side scripting language which is used to develop dynamic and interactive web applications and also used as a general-purpose programming language.

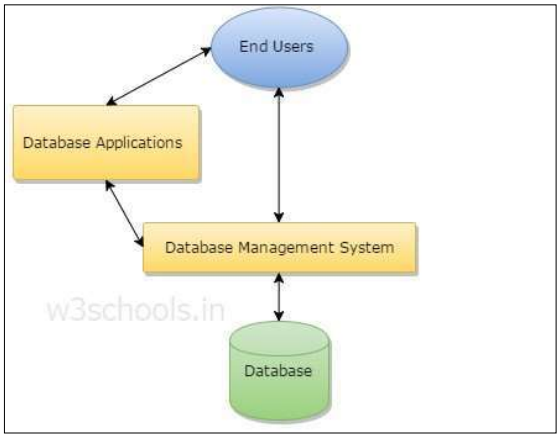

Fig 1.1: components of database management system

2. TYPES OF DBMS:

There are four main types of Database Management Systems (DBMS) and these are based upon their management of database structures. In other words, the types of DBMS are entirely dependent upon how the database is structured by that particular DBMS.

1.2.1 Hierarchical DBMS

A DBMS is said to be hierarchical if the relationships among data in the database are established in such a way that one data item is present as the subordinate of another one or a sub unit. The hierarchical data model was developed by IBM in 1968 and introduced in information management systems. This model is like a structure of a tree with the records forming the nodes.

1.2.2 Network DBMS

A DBMS is said to be a Network DBMS if the relationships among data in the database are of type many-to-many. The many-to-many communication paradigm is one of three major Internet computing paradigms, characterized by multiple users contributing and receiving information, with the information elements often interlinked across different websites.

1.2.3 Relational DBMS

A DBMS is said to be a Relational DBMS or RDBMS if the database relationships are treated in the form of a table. There are three keys on relational DBMS: relation, domain and attributes.

1.2.4 Object-oriented DBMS

Able to handle many new data types, including graphics, photographs, audio, and video, object oriented databases represent a significant advance over their other database cousins. Hierarchical and network databases are all designed to handle structured data; that is, data that fits nicely into fields, rows, and columns.

1.3 Existing System:

Despite the immense technological advancement, blood bank systems use manual date for storing valuable data. Even it is time consuming to retrieve any data if required. Consequently, one of the major issues in blood bank systems, as talked in many articles and research papers it has lack of data security. People doubt whether their personal information and medical records are safely secured or not. Therefore our project aims to develop blood bank management system along with database security and encryption.

4. Proposed System:

Admin/User has to login first. All the personal details of the person is recorded and stored in the database. Admin can search for any personal details. He can retrieve any donation/receive history. They can even have the complete information about the stock present in the blood bank. Apart from this, we will be using concepts of database encryption to make sure that the person's information is kept secure and confidential. This will help us keep their donation and receive records protected from any threats from individuals with potentially malicious intentions or unforeseen hazards to the security of the data.

❖

CHAPTER 2

REQUIREMENTS SPECIFICATION

In this chapter, hardware requirements have been discussed in section 2.1 and software requirements have been discussed in 2.2

System Specification can be divided into two-
2.1 Hardware specifications
2.2 Software specifications

Table 2.1: Specifications

Hardware specifications	Software specifications
32/64 bit operating system	Operating System: Ubuntu.
2 GB RAM or above	Languages: PHP,HTML,JS,CSS(Bootstrap)
40 GB hard disk or above	Database: MySQL
VGA COLOR Monitor	Server: XAMPP enabled with Apache and MySQL
Keyboard	Browser: Chrome etc.
Mouse	Text editor: Sublime Text.

❖

CHAPTER 3

MODULE DESCRIPTION

In this chapter the list of modules incorporated with "BLOOD MANGAMENT IN BLOOD BANK" are:

3.1 Login module:

This module is for employee of blood bank to login, so that only he can access the database.

3.2 Person module:

This module is used to store all the personal details of the donor or receiver.

3.3 Donor module:

This module is used to record the date and time along with quantity of blood donted by a person. Personal details is accessed using person ID.

3.4 Receiver module:

In this module date and time, quantity of blood received along with hospital details are recorded.

3.5 Stock module:

Here we can check availability of each blood group blood.

3.6 SCOPE OF PROJECT BLOOD BANK SYSTEM :

- Donor management (registration, history)
- Inventory tracking (type, quantity, expiry)
- Blood request management (priority, fulfillment)
- Donation camp scheduling
- Testing and screening record keeping
- User roles and permissions
- Reporting and analytics

3.7 FUNCTIONALITIES PROVIDED BY BLOOD BANK SYSTEM:

Login Page

The page where the admin user submits their system credentials to access the admin side of the system.

Home Page

The page where the admin user is being redirected by default after logging into the system. This page displays a summary of the data of the system.

Report Page The page where the printable list of students of the Dance workshop with the selected skills are listed.

3.8 REPORTS OF BLOOD BANK SYSTEM:

- Streamlined processes
- Reduced wastage
- Enhanced donor engagement
- Improved inventory management
- Better response to emergency requests
- Maintenance and Support:

❖

CHAPTER 4

BACK-END DESIGN

4.1 SQL SERVER (DATABASE):

A database management, or DBMS, gives the user access to their data and helps them transform the data into information. Such database management systems include dBase, paradox, IMS, SQL Server and SQL Server. These systems allow users to create, update and extract information from their database. A database is a structured collection of data. Data refers to the characteristics of people, things and events. SQL Server stores each data item in its own fields. In SQL Server, the fields relating to a particular person, thing or event are bundled together to form a single complete unit of data, called a record (it can also be referred to as raw or an occurrence). Each record is made up of a number of fields. No two fields in a record can have the same field name. During an SQL Server Database design project, the analysis of your business change over time, you define any additional fields or change the definition of existing fields.

4.2 SQL Server Tables:

SQL Server stores records relating to each other in a table. Different tables are created for the various groups of information. Related tables are grouped together to form a database.

4.2.1 Primary Key

Every table in SQL Server has a field or a combination of fields that uniquely identifies each record in the table. The Unique identifier is called the Primary Key, or simply the Key. The primary key provides the means to distinguish one record from all other in a table. It allows the user and the database system to identify, locate and refer to one particular record in the database.

4.2.2 Foreign Key

When a field is one table matches the primary key of another field is referred to as a foreign key. A foreign key is a field or a group of fields in one table whose values match those of the primary key of another table needs identifies all the fields or attributes of interest.

4.2.3 Apache

The Apache HTTP server software notable for playing a key role in the initial growth of the World Wide Web. Apache is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation. In 2009 it became the first web server software to surpass the 100 million website milestone.

4.2.4 XAMPP

It is a small and light Apache distribution containing the most common web development technologies in a single package. Its contents, small size and portability make it the ideal tool for students developing and testing applications in PHP and MySQL. XAMPP is available as a free download in two specific packages: full and lite.

4.3 Triggers:

In a DBMS, a trigger is a SQL procedure that initiates an action (i.e., fires an action) when an event (INSERT, DELETE or UPDATE) occurs. Since triggers are event-driven specialized procedures, they are stored in and managed by the DBMS.A trigger cannot be called or executed; the DBMS automatically fires the trigger as a result of a data modification to the associated table. Triggers are used to maintain the referential integrity of data by changing the data in a systematic fashion. Each trigger is attached to a single, specified table in the database.

Triggers can be viewed as similar to stored procedures in that both consist of procedural logic that is stored at the database level. Stored procedures, however, are not event-drive and are not attached to a specific table as triggers are. Stored procedures are explicitly executed by invoking a CALL to the procedure while triggers are implicitly executed. In addition, triggers can also execute stored procedures.

A trigger can also contain INSERT, UPDATE and DELETE logic within itself, so when the trigger is fired because of data modification it can also cause another data modification, thereby firing another trigger. A trigger that contains data modification logic within itself is called a nested trigger.

4.4 Stored Procedures:

A stored procedure is a subroutine available to applications that access a relational database management system (RDBMS). Such procedures are stored in the database data dictionary. Uses for stored procedures include data-validation (integrated into the database) or access-control mechanisms. Furthermore, stored procedures can consolidate and centralize logic that was originally implemented in applications. To save time and memory, extensive or complex processing that requires execution of several SQL statements can be saved into stored procedures, and all applications call the procedures. One can use nested stored procedures by executing one stored procedure from within another. Stored procedures may return result sets, i.e., the results of a SELECT statement. Such result sets can be processed using cursors, by other stored procedures, by associating a result set locator, or by applications. Stored procedures may also contain declared variables for processing data and cursors that allow it to loop through multiple rows in a table. Stored procedure flow-control statements typically include IF, WHILE, LOOP, REPEAT and CASE statements, and more. Stored procedures can receive variables, return results or modify variables and return them, depending on how and where the variable is declared.

Schema diagram:

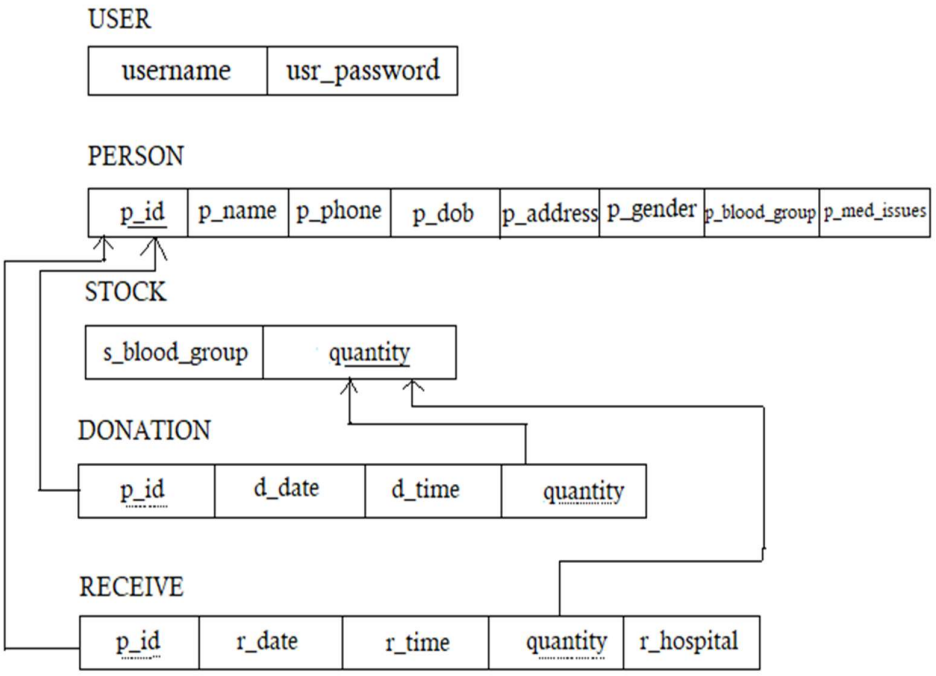

Fig. 4.1 Schema diagram of blood management system

Features:

- Donor management (registration, history)
- Inventory tracking (type, quantity, expiry)
- Blood request management (priority, fulfillment)
- Donation camp scheduling
- Testing and screening record keeping
- User roles and permissions
- Reporting and analytics

4.6 ER-Diagram:

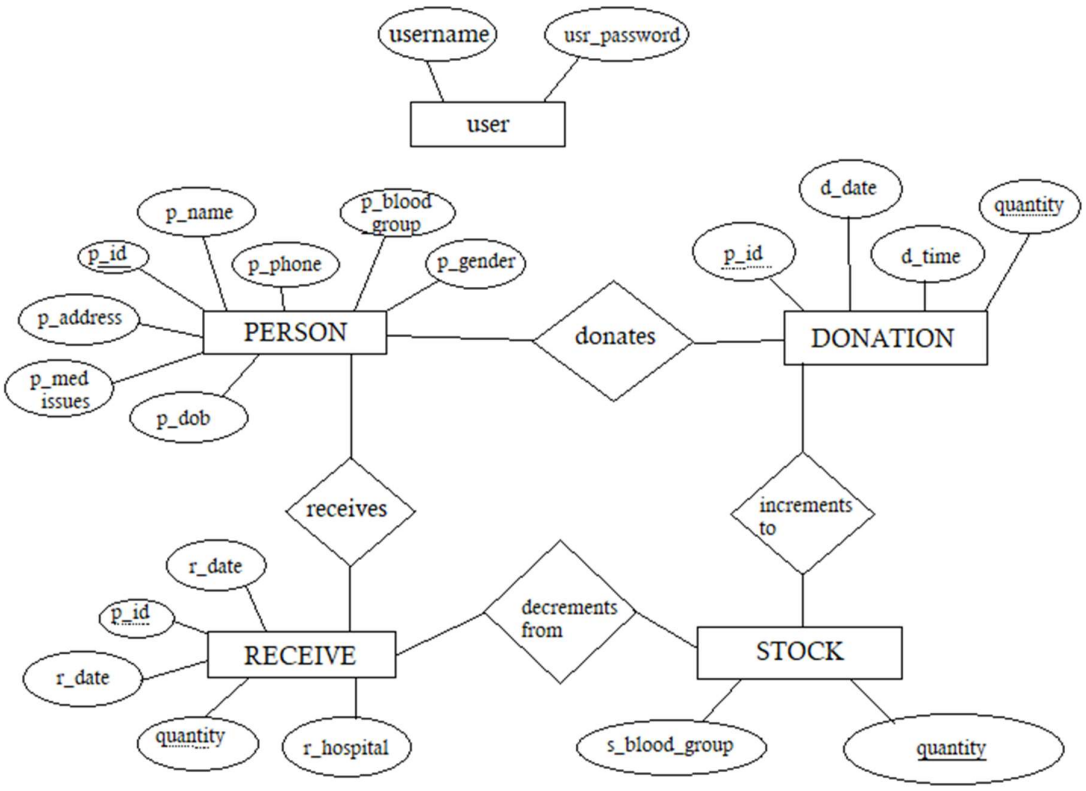

Fig. 4.2 E-R diagram of blood management system

4.7 DATABASE TABLES:

The database consists of one or more tables. Each table is made of rows and column. Each row in a relational uniquely by primary key. This can be by one or more sets of columns value in most of scenarios it is a single column.

TABLE 4.1: USER

NAME	TYPE	FIELD NAME	DESCRIPTION
username	varchar(10)	not null	username of employee
usr_password	varchar(16)	not null	password to login

TABLE 4.72: PERSON

NAME	TYPE	FIELD NAME	DESCRIPTION
p_id	int(10)	not null	unique person id (auto increment)
p_name	varchar(25)	not null	person name
p_phone	char(10)	not null	phone number
p_dob	date	not null	date of birth
p_address	varchar(100)	not null	address
p_gender	char(1)	not null	gender
p_blood_group	varchar(3)	not null	blood group
p_med_issues	varchar(100)	null	any medical issues to be mentioned

TABLE 4.73: DONATION

NAME	TYPE	FIELD NAME	DESCRIPTION
p_id	int(10)	not null	unique person id
d_date	date	not null	blood donation date
d_time	time	not null	blood donation time
quantity	int(1)	not null	units of blood donated (bottles)

TABLE 4.74: RECEIVE

NAME	TYPE	FIELD NAME	DESCRIPTION
p_id	int(10)	not null	unique person id
r_date	date	not null	date of blood received
r_time	time	not null	time of blood received
quantity	int(1)	not null	units of blood received (bottles)
r_hospital	varchar(50)	not null	name of the hospital

TABLE 4.75: STOCK

NAME	TYPE	FIELD NAME	DESCRIPTION
s_blood_group	varchar(3)	not null	blood group name
quantity	int(5)	null	units of blood available

❖

CHAPTER 5

FRONT END DESIGN

In this chapter we have explained about the front end design tools such as Sublime Text Editor a long with front end language PHP.

5.1 Sublime Text Editor:

Sublime Text is a proprietary cross-platform source code editor with a Python application programming interface (API).Inactively support many programming languages and markup languages, and functions can be added by users with plug-in, typically community-built and maintained under free-software licenses.

5.2 Java Script:

JavaScript, often abbreviated as JS, is a high-level interpreted scripting language. JavaScript has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first-class functions.

5.3 HTML:

Hyper Text Mark-up Language is the standard mark-up language for documents designed in a web browser. It can be assisted by technologies such as cascading style sheets and scripting languages such as JavaScript.

5.4 PHP (Hypertext Pre-processor):

Hypertext Pre-processor (or simply PHP) is a server-side scripting language designed for Web development, and also used as a general-purpose programming language. It was originally created by Rasmus Lerdorf in 1994.the PHP reference implementation is now produced by The PHP Group. PHP originally stood for Personal Home Page, but it now stands for the recursive initialise. PHP code may be embedded into HTML code, or it can be used in combination with various web template systems, web content management systems, and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and can be used to implement standalone graphical applications. The standard PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge. The PHP language evolved without a written formal specification or standard until 2014, with the original implementation acting as the de facto standard which other implementations aimed to follow. Since 2014 work has gone on to create a formal PHP specification.

5.5 Pseudo code

5.5.1 Login page:

Step1: Enter the USERNAME in the given field.
Step2: Enter the valid PASSWORD.
Step3: Click on LOGIN button

```
<?php
session_start();
include('connection.php');

if(isset($_POST['submit'])){

    $username = stripcslashes($_POST['user']);
  $username = mysqli_real_escape_string($con, $username);
  $password = stripcslashes($_POST['pass']);
  $password = mysqli_real_escape_string($con, $password);

    if($username === '' or $password === ''){
      $_SESSION["login_error"] = 'Username or Password cannot be empty';
      header('Location: index.php');
  }

  else{

      $sql = "select * from user where username = '$username' and password = '$password'";
      $result = $con->query($sql);


      if($result->num_rows > 0){
          $row = $result->fetch_assoc();
          $_SESSION["login"] = 1;
          $_SESSION["username"] = $row["username"];
          header('Location: Home.php');
      }


      else{
          $_SESSION["login"] = 0;
          $_SESSION["login_error"] = 'Invalid login credentials';
          header('Location: index.php');
      }
  }
}
```

```
else
   header('Location: index.php');
?>
```

5.5.2 Home page:

This page contains agenda of the database. Reason behind creating the database and what is the motive of this db.

It also gives the brief information about number of registrations, donations and receives taken place in the blood management.

```
<?php
session_start();
$_SESSION["tab"] = "Home";

if ($_SESSION["login"] != 1) {
   echo '<h2 style="color: red; text-align: center;">Authentication Error!</h2>';
} else {
   include_once('header.php');

   echo '<div style="text-align: center;">';

   echo "<h2>Welcome to our Blood Bank - Where Hope Flows Through Generosity.</h2><br>";

echo "<p>Step into a sanctuary of compassion and care, where the beating heart of humanity resides. Here, amidst the hustle and bustle of life, we stand as a beacon of hope, a haven for the weary and the wounded.</p>";

echo "<p>In our halls, the echo of countless lifesaving miracles reverberates, each drop of blood a testament to the indomitable spirit of giving. It's not merely about blood, but about the profound connection between donor and recipient, the unspoken bond that transcends barriers and brings communities together.</p>";

echo "<p>With every step you take through our doors, you become a part of something greater than yourself - a network of heroes, united by the singular purpose of saving lives. Here, in this sacred space, every act of generosity is a symphony of humanity, each donor a conductor orchestrating the melody of hope.</p>";

echo "<p>Welcome to our Blood Bank, where we believe that through the simple act of giving, we can transform despair into hope, and darkness into light. Together, let us embark on a journey of compassion and kindness, where every drop of blood becomes a beacon of hope for those in need.</p>";

   echo '</div>';
```

```
echo '<div style="float: left; margin-right: 20px;">';

include_once('checkstock.php');

echo '</div>';

echo '<div style="float: right;">';

echo '<img src="res\images\Bloodtable.jpg" width="400" height="400" title="Compatibility of
Blood groups" alt="Blood Table" />';
echo '</div>';


   include_once('footer.php');
}
?>
```

5.5.3 Person page:

Here the user must enter the personal details of the new donor/receiver.
Step1: Open the ADD PERSON page.
Step2: The PERSON ID is auto incremented because its unique for all.
Step3: Enter the PERSON NAME.
Step 4: Enter the PHONE NUMBER.
Step5: Enter the GENDER.
Step6: Select the DATE OF BIRTH.
Step7: Select the BLOOD GROUP.
Step 8: Enter the ADDRESS.
Step9: Enter the MEDICAL ISSUES if any.
Step10: Click on REGISTER button.

```
<?php
session_start();
$_SESSION["tab"] = "Add Person";

if($_SESSION["login"] != 1)
   echo '<h2 txtcolor="red">Authentication Error!</h2>';
else{
   include_once('header.php');
   $name = $_POST['name'];
   $phone = $_POST['phone'];
   $gender = $_POST['gender'];
   $dob = $_POST['dob'];
   $blood_group = $_POST['blood_group'];
   $address = $_POST['address'];
   $med_issues = $_POST['med_issues'];
```

```
    $sql = "insert into Person (p_name,p_phone, p_dob, p_address, p_gender, p_blood_group, p_med_issues) values('$name', '$phone', '$dob', '$address', '$gender', '$blood_group', '$med_issues')";
    if ($con->query($sql) === TRUE) {
        $sql = "select p_id from Person where p_name ='$name' and p_phone = '$phone' and p_gender = '$gender' and p_dob = '$dob' and p_blood_group = '$blood_group' and p_address = '$address' and p_med_issues = '$med_issues'";
        $result = mysqli_query($con, $sql);
        $row = mysqli_fetch_array($result);
        $count = mysqli_num_rows($result);
        $pid = $row['p_id'];
    }
    else
        echo "Error: " . $sql . "<br>" . $con->error;

    if($count == 1){
        echo'<h2>' .$name .'</h2><br><br>';
        echo 'Your registration is succesful!<br><br>';
        echo'Personal ID : '.$pid .'<br><br>';
        echo'Name : '.$name .'<br><br>';
        echo 'Phone Number: ' .$phone .'<br><br>';
        echo 'DOB : ' .$dob .'<br><br>';
        echo 'Blood Group : ' .$blood_group .'<br><br>';

        echo 'Gender : ';
        if ($gender === 'm')
            echo 'Male<br><br>';
        if ($gender === 'f')
            echo 'Female<br><br>';
        if ($gender === 'o')
            echo 'Others<br><br>';
        echo 'Address : ' .$address .'<br><br>';
        echo 'Medical Issues : ';
        if ($med_issues === "")
            echo 'None<br><br>';
        echo '<div style ="color:red;">NB:Please keep your Personal ID for future reference!';
    }
    include_once('footer.php');
}
?>
```

5.5.4 Search Person Page:

In this page one can retrieve the details of the person using PERSON ID.

Step1: Open the SEARCH PERSON page

Step2: Enter the PERSON ID.

Step3: Click on SUBMIT button.

```
<?php
session_start();
$_SESSION["tab"] = "Search Person";

if($_SESSION["login"] != 1)
   echo '<h2 txtcolor="red">Authentication Error!</h2>';
else{
   include_once('header.php');
   echo '
   <form name="searchPerson" action = "searchPerson.php"  method = "POST">
   <h2>SEARCH PERSON</h2>
   <br>

   <p>
   <label>Personal ID: </label>
   <br>
   <input type = "text" name  = "pid" class="input" required/>
   </p>

   <p>
   <button class="btn">SUBMIT</button>
   </p>
   </form>';
   include_once('footer.php');
}
?>
```

5.5.5 Donation page:

If any person is willing to donate the blood.

Step1: Open the NEW DONATION page.

Step2: Enter the unique PERSON ID of the person.

Step3: Enter the Units of blood donated.

Step4: Click on SUBMIT button.

(Note: Person details are accessed using the PERSON ID, date and time is automatically accessed using computers default time.)

```
<?php
session_start();
$_SESSION["tab"] = "New Donation";

if($_SESSION["login"] != 1)
```

```
    echo '<h2 txtcolor="red">Authentication Error!</h2>';
else{
    include_once('header.php');
    echo '
    <form name="newDonation" action = "newDonation.php"  method = "POST">
    <h2>NEW DONATION</h2>
    <br>

    <p>
    <label>Personal ID:</label>
    <br>
    <input type = "Number" name  = "pid" class="input" required/>
    </p>

    <p>
    <label>Units of blood donated:</label>
    <br>
    <input type = "Number" maxlength="1" name  = "units" class="input" required/>
    </p>

    <p>
    <button class="btn">SUBMIT</button>
    </p>
    </form>';
    include_once('footer.php');
}
?>
```

5.5.6 Receive page:

If any person require blood for emergency purpose.

Step1: Open the NEW RECEIVE page.

Step2: Enter the unique PERSON ID of person.

Step3: Enter the Units of blood received by a person.

Step4: Enter the Hospital Name in which the person is being treated.

Step5: Click on SUBMIT button.

(Note: Person details are accessed using the PERSON ID, date and time is automatically accessed using computers default time.)

```
<?php
session_start();
$_SESSION["tab"] = "New Receive";

if($_SESSION["login"] != 1)
    echo '<h2 txtcolor="red">Authentication Error!</h2>';
else{
    include_once('header.php');
```

```
    echo '<form name="addPerson" action = "newReceive.php"  method = "POST">
    <h2>NEW RECEIVE</h2>
    <br>
    <p>
    <label>Personal ID:</label>
    <br>
    <input type = "Number" name  = "pid" class="input" required/>
    </p>
    <p>
    <label>Units of blood Received:</label>
    <br>
    <input type = "Number" maxlength="1" name  = "units" class="input" required/>
    </p>
    <p>
    <label>Admitted Hospital:</label>
    <br>
    <input type = "text" name  = "hospital" class="input" required/>
    </p>
     <p>
    <button class="btn">SUBMIT</button>
    </p>
    </form>';
    include_once('footer.php');
}
?>
```

5.5.7 Stock Page:

If the passenger forgets password then he can recover it.

Step1: Go to HOME page

Step2: Press MOVIE button

Step3: select FORGOT PASSWORD

Step4: Enter the following fields

```
<?php
session_start();
$_SESSION["tab"] = "Check Stock";
if($_SESSION["login"] != 1)
    echo '<h2 txtcolor="red">Authentication Error!</h2>';
else{
    include_once('header.php');
    include_once('checkStock.php');

    include_once('footer.php');
}
?>
```

5.5.8 Donation History Page:

Specify the time interval so that you can get details from that time interval only.
Step1: Enter the time interval in after and before date.
Step2: Click on SEARCH button.
It displays all the donation history of that time interval.
If no donation history is present then it responds with “No record found on this time interval”.

```
<?php
session_start();
$_SESSION["tab"] = "Receiving History";

if($_SESSION["login"] != 1)
   echo '<h2 txtcolor="red">Authentication Error!</h2>';
else{
   include_once('header.php');
   echo '          <form name="ReceivingHistory" action = "receivingHistory.php"  method = "POST">
   <h2>HISTORY</h2>
   <br>
   <p>
   Specify time interval to view the Receiving History
   </p>
   <p>
   <label>After: </label>
   <br>
   <input type = "date" name  = "sdate" class="input" required/>
   </p>
   <p>
   <label>Before: </label>
   <br>
   <input type = "date" name  = "edate" class="input" required/>
   </p>
   <p>
   <button class="btn">SEARCH</button>
   </p>

   </form>';
   include_once('footer.php');
}
?>
```

5.5.9 Receive History Page:

Specify the time interval so that you can get details from that time interval only.
Step1: Enter the time interval in after and before date.
Step2: Click on SEARCH button.
It displays all the receive history of that time interval.
If no receive history is present then it responds with “No record found on this time interval”.

```
<?php
session_start();
$_SESSION["tab"] = "Receiving History";

if($_SESSION["login"] != 1)
   echo '<h2>Authentication Error!</h2>';
else{
   include_once('header.php');
   $sdate = $_POST['sdate'];
   $edate = $_POST['edate'];

   $sql = "select r.r_date, r.r_time, r.r_hospital, r.r_quantity, p.p_id, p.p_name, p_phone, p.p_blood_group from Person p,Receive r where p.p_id = r.p_id and r.r_date >= '$sdate' and r.r_date <= '$edate'";
   $result = mysqli_query($con, $sql);

   echo "<h2>Receiving History</h2><br>";

   if ($result->num_rows > 0) {
      echo "<table>
      <tr>
      <th>Personal ID</th>
      <th>Name</th>
      <th>Phone</th>
      <th>Blood Group</th>
      <th>Date</th>
      <th>Time</th>
      <th>Units of Blood</th>
      <th>Hospital</th>
      </tr>";
      while($row = $result->fetch_assoc()) {
         echo "
         <tr>
         <td>" . $row["p_id"]. "</td>
         <td>" . $row["p_name"]."</td>
         <td>" .$row["p_phone"] ."</td>
         <td>" . $row["p_blood_group"]. "</td>
         <td>" . $row["r_date"]. "</td>
```

```
            <td>" . $row["r_time"]. "</td>
            <td>" . $row["r_quantity"]. "</td>
            <td>" . $row["r_hospital"]. "</td>
            </tr>";
        }
        echo "</table> <br><br>";
    }

    else
        echo "No record found in the specified intervel";
    include_once('footer.php');
}

?>
```

5.5.10 Add User Page:

Adding new user.
Step1: Enter the SUPER ADMIN PASSWORD.
Step2: Enter the new USERNAME.
Step3: Enter the PASSWORD.
Step4: CONFIRM PASSWORD.
Step5: Click on CREATE USER button.
New user is created.

```
<?php
session_start();
$_SESSION["tab"] = "Add User";

if($_SESSION["login"] != 1)
  echo '<h2 txtcolor="red">Authentication Error!</h2>';
else{
  include_once('header.php');
  echo '
  <form name="addUser" action = "addUser.php"  method = "POST">
  <h2>ADD USER</h2>
  <br>

  <p>
  <label>Admin Password: </label>
  <br>
  <input type = "password" name  = "super_pwd" class="input" required/>
  </p>
  <p>
  <label>Username: </label>
  <br>
```

```
 <input type = "text" name  = "usr_name" class="input" required/>
 </p>
 <p>
 <label>Password: </label>
 <br>
 <input type = "password" name  = "usr_pwd" class="input" required/>
 </p>
 <p>
 <label>Confirm Password: </label>
 <br>
 <input type = "password" name  = "usr_cnfrm_pwd" class="input" required/>
 </p>

 <p>
 <button class="btn">CREATE USER</button>
 </p>
 </form>';
 include_once('footer.php');
}
?>
```

5.5.11 Logout:

Once you are done with the work. You can logout using “LOGOUT” button.

```
<?php
session_start();
$_SESSION["tab"] = "Home";
$_SESSION["login"] = 0;
session_destroy();
header('Location: index.php');
?>
```

❖

CHAPTER 6

RESULTS

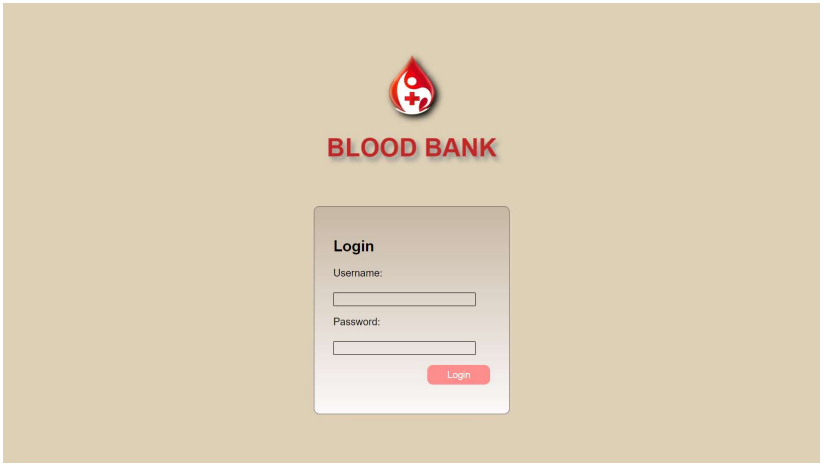

Fig 6.1: Login Page

In the above fig 6.1 shows the login page, if the user wants to login, they must enter valid/registered username and password and click on login button.

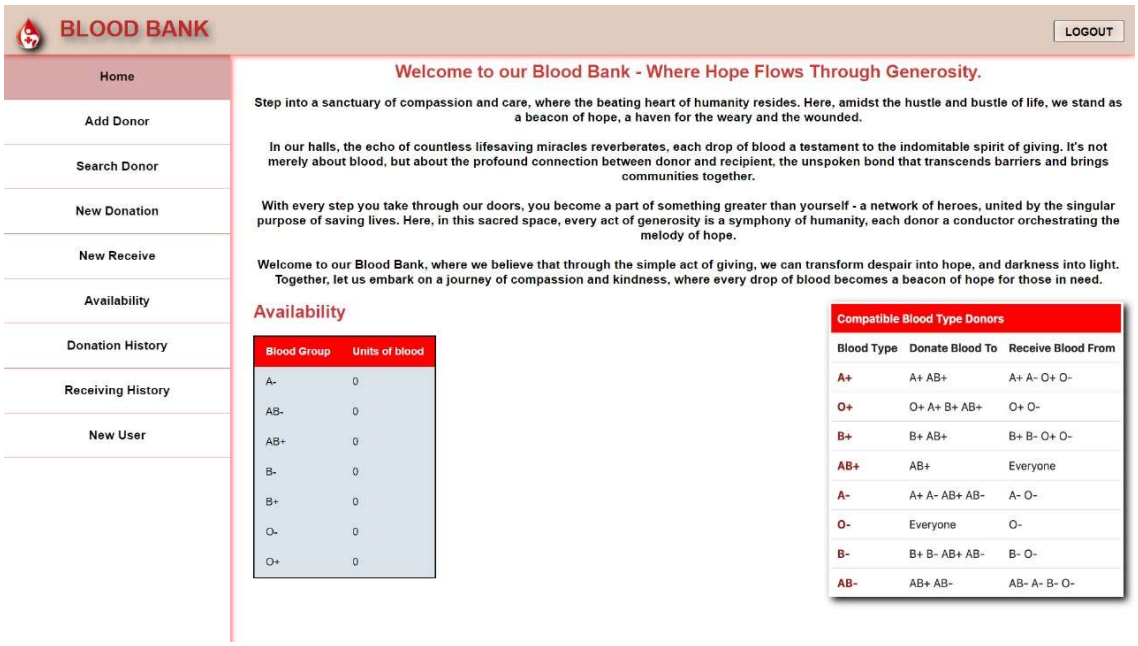

Fig 6.2: Home page

In the above fig 6.2 shows the information about the blood bank management system.

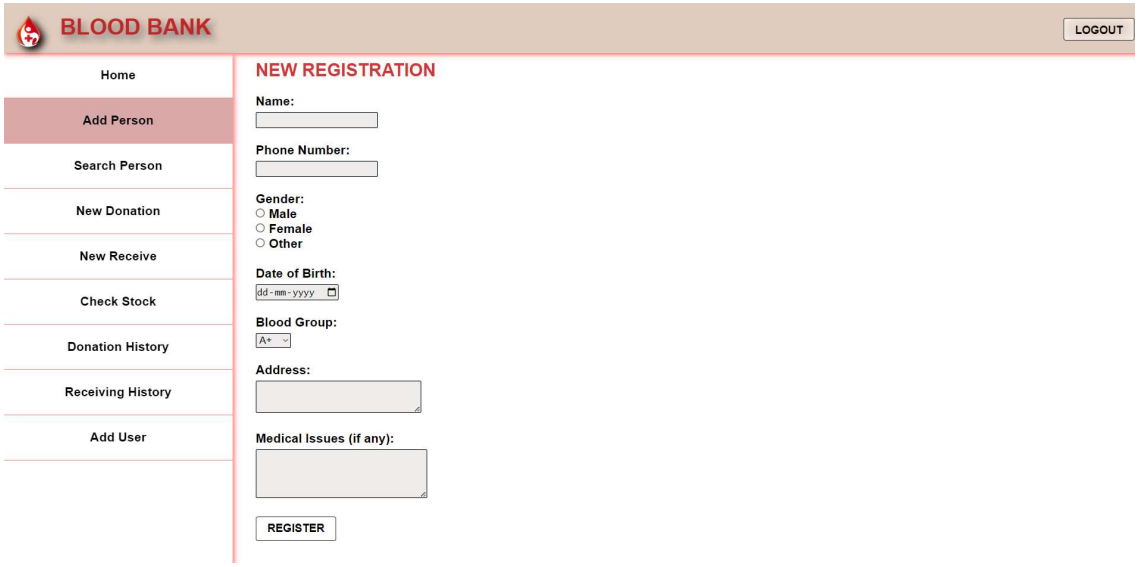

Fig 6.3: Add person page

In the above fig 6.3 shows the Add person page, here details of the donor/receiver is registered.

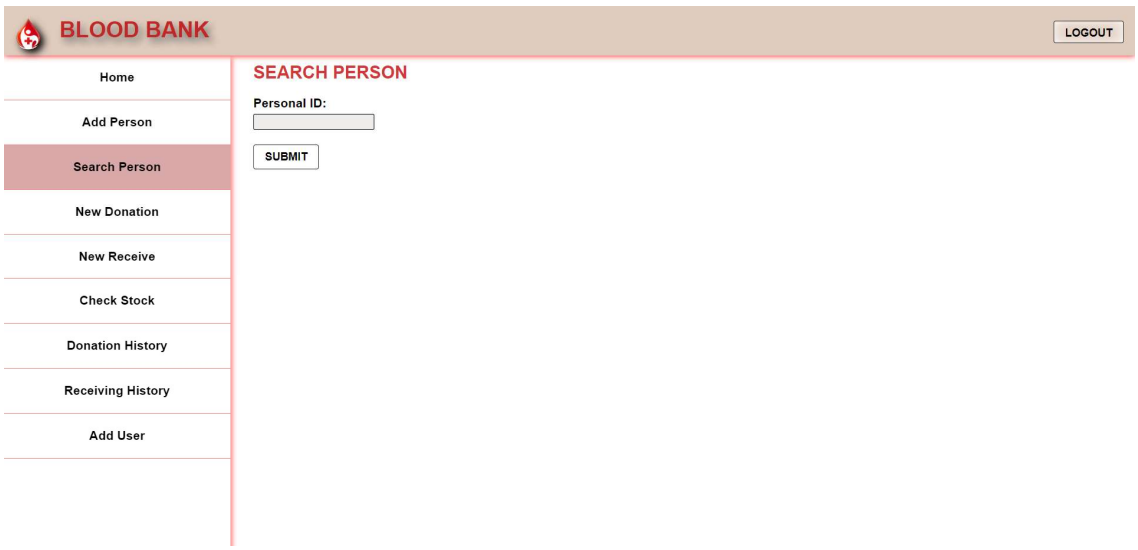

Fig 6.4: Search person page

In the fig 6.4 shows the result for search we done. If we enter the person id it will return with all the personal record present in the database.

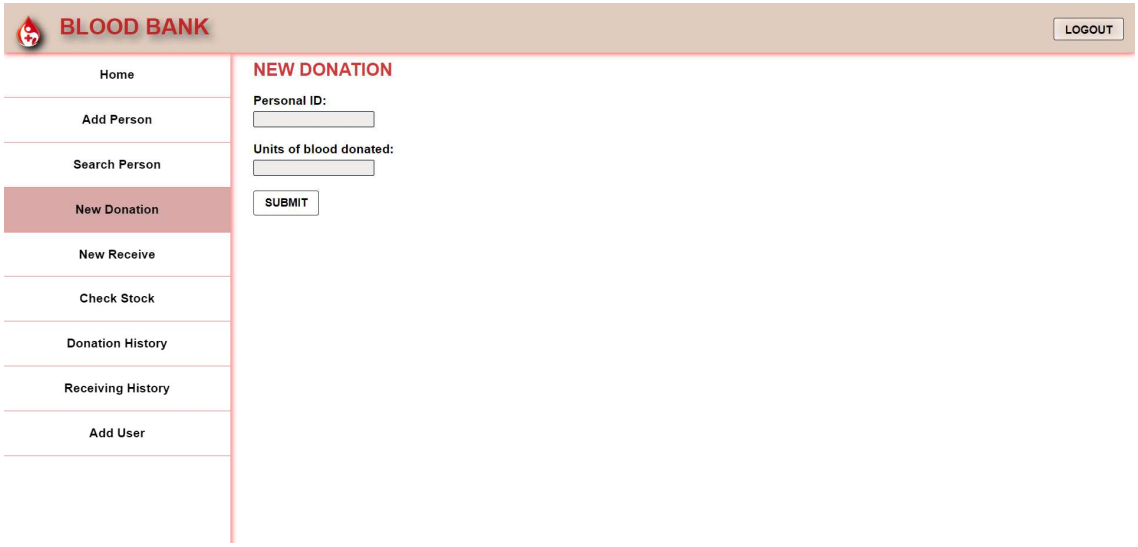

Fig 6.5: Donation page

In the above fig 6.5 shows the New donation page, here person id and units of blood donated is recorded and submit button is clicked. On successful completion of entering it displays "Your donation is successful".

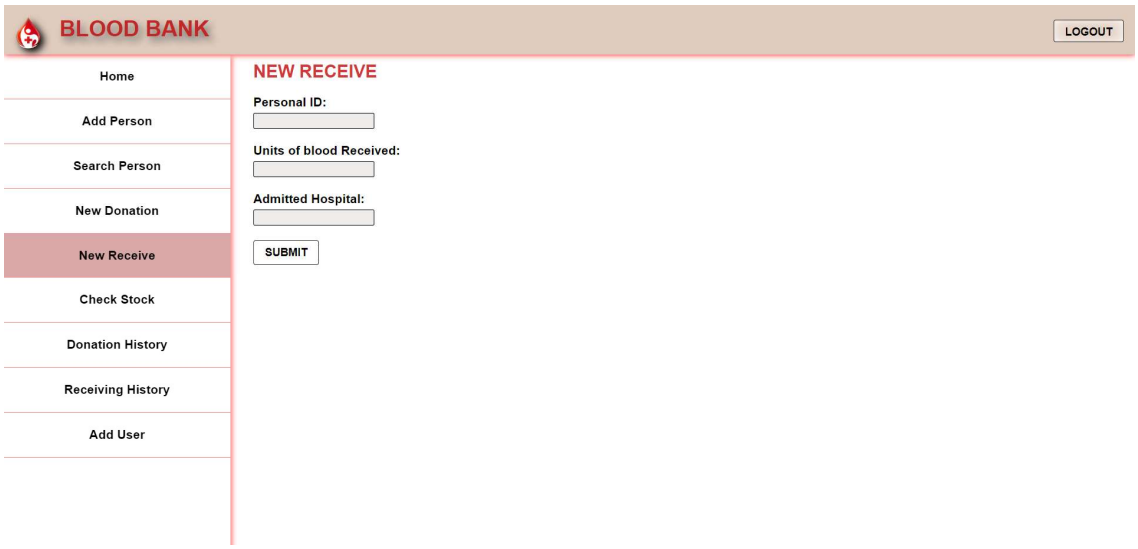

Fig 6.6: Receive page

In the above fig 6.6 shows the New Receive page, here person id, units of blood donated and hospital details is recorded and submit button is clicked. On successful completion of entering it displays "Your receiving is successful".

Fig 6.7: Check Stock page

In the above fig 6.7 shows the stock details i.e units of blood present in blood bank of each blood group.

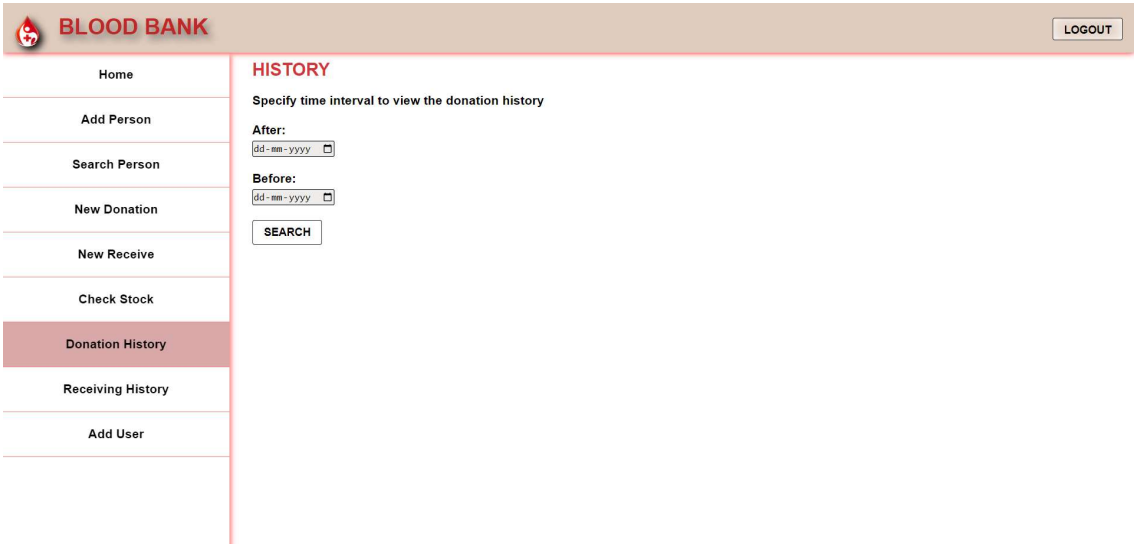

Fig 6.8: Donation History page

In the above fig 6.8 shows the Donation history page, the user has to give the time interval from when to when he wants to see the donation history details. Upon clicking submit he shows the donation history. If there is donation took place at that time interval is displays “No record found in the specified time interval”.

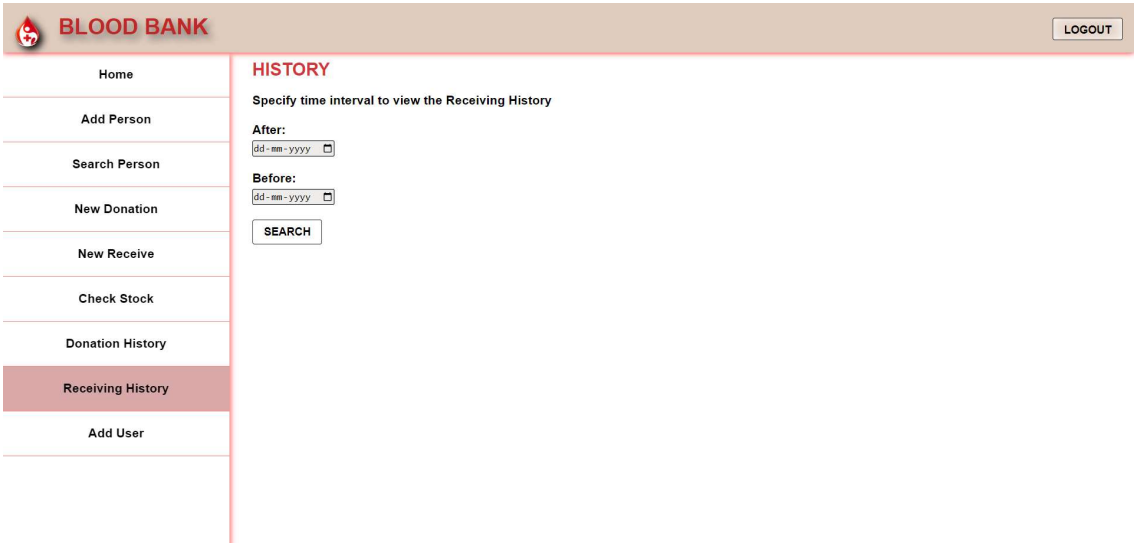

Fig 6.9: Receive History page

In the above fig 6.9 shows the Receiving history page, the user has to give the time interval from when to when he wants to see the receive history details. Upon clicking submit he shows the receive history. If there is receiving of blood took place at that time interval is displays "No record found in the specified time interval".

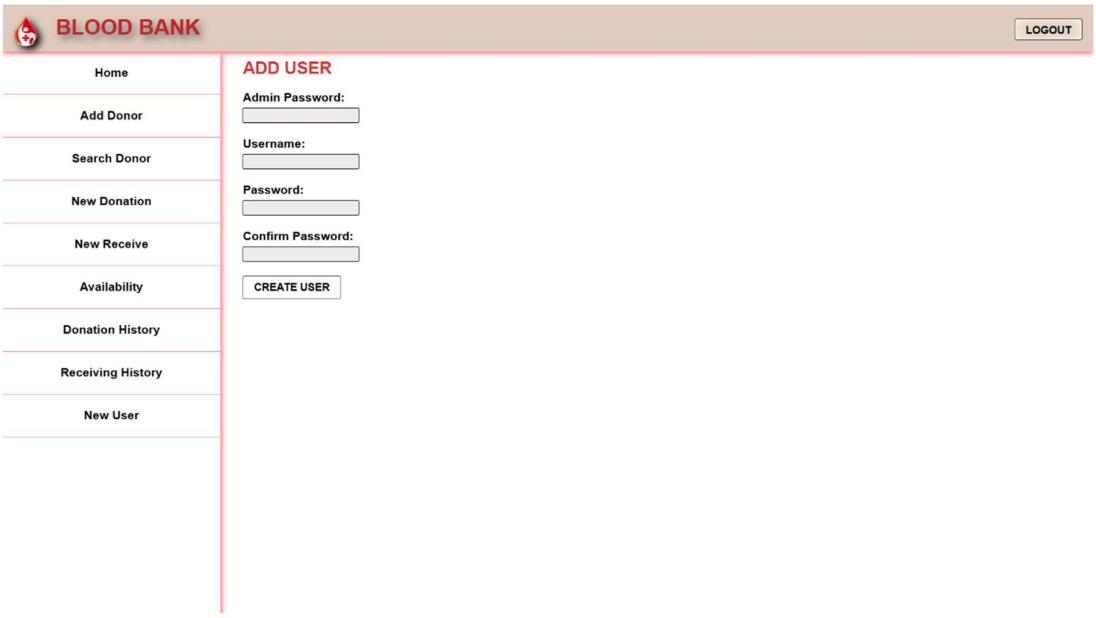

Fig 6.10: Add user page

In the above fig 6.10 shows the add user page, here it asks for super admin password. Next you have to enter the new username, password and confirm the password. Click on create user button. If the super admin password is wrong or password and confirm password don't match. It will pop invalid password. On successfully creating user it will display "New user created successfully".

CHAPTER 7

CONCLUSION

The main purpose of our blood management system is to provide blood bank with easier way to store and retrieve data and keep record of the availability of blood in blood bank.

After inserting the data to database, staff need not register of the same person again. They can simply search for recorded data and retrieve them for future blood donation or receiving purpose of that person.

In the nutshell, it can be summarized that the future scope of the project circles around maintaining information regarding:

- The person can fix their donation schedule using online reservation for donation of blood.
- The person can search for availability of required blood in the local blood bank in the case of emergency.
- The blood bank to store the details of the blood donated by person, like RBC, WBC, platelet count etc.

The above mentioned points are the enhancements which can be done to increase the applicability and usage of this project.

❖

REFERENCES

1) Silberschatz Korth and Sudharshan, Database System Concepts, 6th Edition, McGraw Hill, 2013.
2) Fundamentals of Database Systems, Ramez Elmasri and Shamkant B. Navathe, 7th Edition, 2017, Pearson.
3) Database management systems, Ramakrishna, and Gehrke, 3rd Edition, 2014, McGraw Hill, 2013.
4) Google
5) https://www.w3schools.com

❖