Chapter 1

INTRODUCTION

A Database Management System (DBMS) refers to the technology for creating and managing databases. Basically, DBMS is a software tool to organize (create, retrieve, update or manage) data in a database. The main aim of a DBMS is to supply a way to store up and retrieve database information that is both convenient by efficient data, we mean known facts that can be recorded and that have the embedded meaning. Database systems are meant to handle a large collection of information.

1.1 Problem Statement

The project blood bank management system is known to be a pilot project that is designed to gather blood from various sources. The software is designed to handle the daily transaction of the blood bank and search the details when required. It also helps to register the details of donors, blood collection details as well as blood-issued reports. The software application is designed in such a manner that it can suit the needs of all blood bank requirements in the course of the future.

1.2 Objective

1. To help admins to create camp easily.

2. To help provide options for various users to contribute their donations.

3. Online storage of blood bank details.

1.3 SQL

SQL (Structured Query Language) is a standardized programming language that's used to manage relational databases and perform various operations on the data in them. SQL is regularly used not only by database administrators but also by developers writing data integration scripts and data analysts looking to set up and run analytical queries. The uses of SQL include modifying database table and index structures; adding, updating, and

deleting rows of data; and retrieving subsets of information from within a database for transaction processing and analytics applications. Queries and other SQL operations take the form of commands written as statements -- commonly used SQL statements include select, add, insert, update, delete, create, alter and truncate.

1.4 PHP

The PHP Hypertext Preprocessor (PHP) is a programming language that allows web developers to create dynamic content that interacts with databases. PHP is basically used for developing web-based software applications. Five important characteristics make PHP's practical nature possible −

- Simplicity
- Efficiency
- Security
- Flexibility
- Familiarity

PHP code is usually processed on a web server by a PHP interpreter implemented as a module, a daemon, or as a Common Gateway Interface (CGI) executable. On a web server, the result of the interpreted and executed PHP code – which may be any type of data, such as generated HTML or binary image data – would form the whole or part of an HTTP response. Various web template systems, web content management systems, and web frameworks exist which can be employed to orchestrate or facilitate the generation of that response. Additionally, PHP can be used for many programming tasks outside the web context, such as standalone graphical applications and robotic drone control. PHP code can also be directly executed from the command line.

1.5 HTML5

HTML5 is a markup language used for structuring and presenting content on the World Wide Web. It is the fifth and last major HTML version that is a World Wide Web Consortium (W3C) recommendation. HTML5 includes detailed processing models to encourage more interoperable implementations; it extends, improves, and rationalizes the markup available for documents and introduces markup and application programming

interfaces (APIs) for complex web applications. For the same reasons, HTML5 is also a candidate for cross-platform mobile applications because it includes features designed with low-powered devices in mind.

1.6 CSS3

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility; provide more flexibility and control in the specification of presentation characteristics; enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, which reduces complexity and repetition in the structural content; and enable the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.

CSS3 is a collaboration of CSS2 specifications and new specifications and this collaboration is called a module. Some of the modules are shown below –

- Selectors
- Box Model
- Backgrounds
- Image Values and Replaced Content
- Text Effects
- 2D Transformations
- 3D Transformations
- Animations
- Multiple Column Layout
- User Interface

1.7 JAVA SCRIPT

JavaScript is a text-based programming language used both on the client-side and server-side that allows you to make web pages interactive. Where HTML and CSS are languages that give structure and style to web pages, JavaScript gives web pages interactive elements that engage user. It is used to enhance HTML pages and in commonly found embedded in HTML code. JavaScript is an interpreted language. Thus, it doesn’t need to comply.

Chapter 2

REQUIREMENT ANALYSIS AND SPECIFICATION

2.1 Functional Requirements

These are statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations. In some cases, the functional requirements may also explicitly state what the system should not do. The functional requirements for a system describe what the system should do. These requirements depend on the type of software being developed, the expected users of the software, and the general approach taken by the organization when writing requirements. When expressed as user requirements, the requirements are usually described in an abstract way. However, functional system requirements describe the system function in detail, its inputs and outputs, exceptions, and so on. Functional requirements for a software system may be expressed in a number of ways.

Login: User can login to the account by filling in the information about them and click on the Login button. The user has to login to get more information about the blood bank.

Signup: User Signup includes the information of the donor who want to register. Donor can register the account by clicking on the new register. He/she can add the account for the further enquiry of the blood donation.

Registration: Registration page includes the information of the donor who wants to register. User clicks on the register button then data is added to register table

Deregistration: User clicks on the Deregistration button then the registration information is removed from register table.

Camp Creation: Admin enters the new camp details and the new camp is stored in the camp table.

Hardware requirements

- Processor : Intel i3/i5,1.8GHz machine or above

- Main memory : 4GB RAM or more.
- Hard disk drive : 1TB

Software requirements

- Operating System : Windows 7 and higher
- Front end : HTML
- Back end : PHP (XAMPP)
- Framework : Google Fonts , Font Awesome

2.2 Non -Functional Requirements

Non-functional requirements are requirements that are not directly concerned with the specific functions delivered by the system. They may relate to emergent system properties such as reliability, response time, and store occupancy. Alternatively, they may define constraints on the system such as the capabilities of I/O devices and the data representations used in system interfaces. The plan for implementing functional requirements is detailed in the system design. The plan for implementing non-functional requirements is detailed in the system architecture. Non-functional requirements are often called qualities of a system. Other terms for non-functional requirements are "constraints", "quality attributes", "quality goals", "quality of service requirements" and "non-behavioural requirements". Qualities, that are non-functional requirements, can be divided into two main categories: Execution qualities, such as security and usability, which are observable at run time.

The non-functional requirements of Red vault are as follows:

Reliability: Red Vault is a reliable interface as it provides data security and data safety. Data provided by the user is confidential and safe. User cannot use other user account without password and user id verification.

Consistency: Red Vault provide consistency of data. User can only to register to future camps. Admin can only confirm the donation of the users who have registered for that camp.

Performance: Red Vault interface performs smoothly for user to have a good and easy experience to register for a camp. It is easy to understand and can accessed anywhere through the internet.Security: Security is very important aspect which Red Vault provides. Data of user are maintained with confidentiality. No data can be accessed by any third party and data is only for user to access.

Chapter 3

SYSTEM DESIGN

System Design process partitions the system into subsystems based on the requirements. It establishes overall system architecture and is concerned with identifying various components, specifying relationships among components, specifying software structure, maintaining a record of design decisions and providing a blue print for the implementation phase [6]. Design consists of architecture design and detailed design is concerned with the details of how to package processing modules and how to implement the processing algorithms, data structures and interconnections among modules and data structures.

3.1 ER Diagram

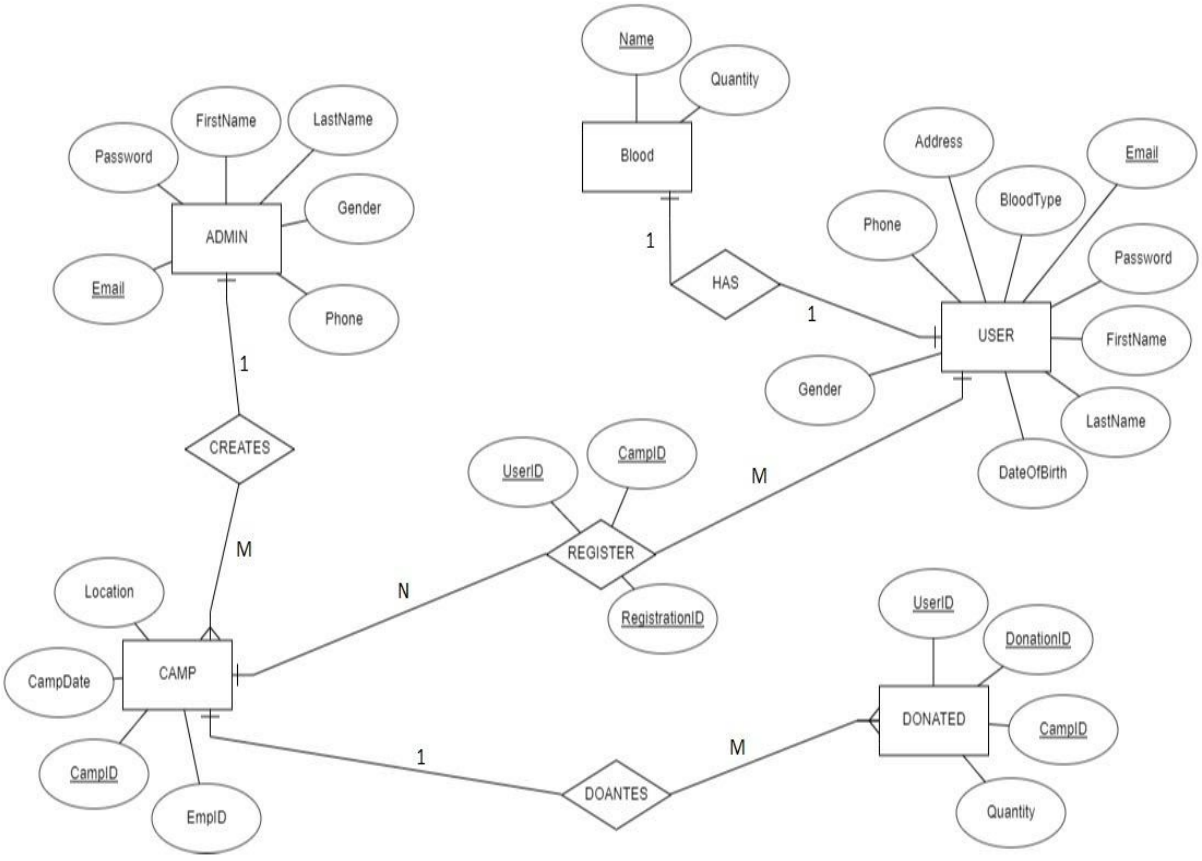

Figure 3.1

3.2 Schema Diagram

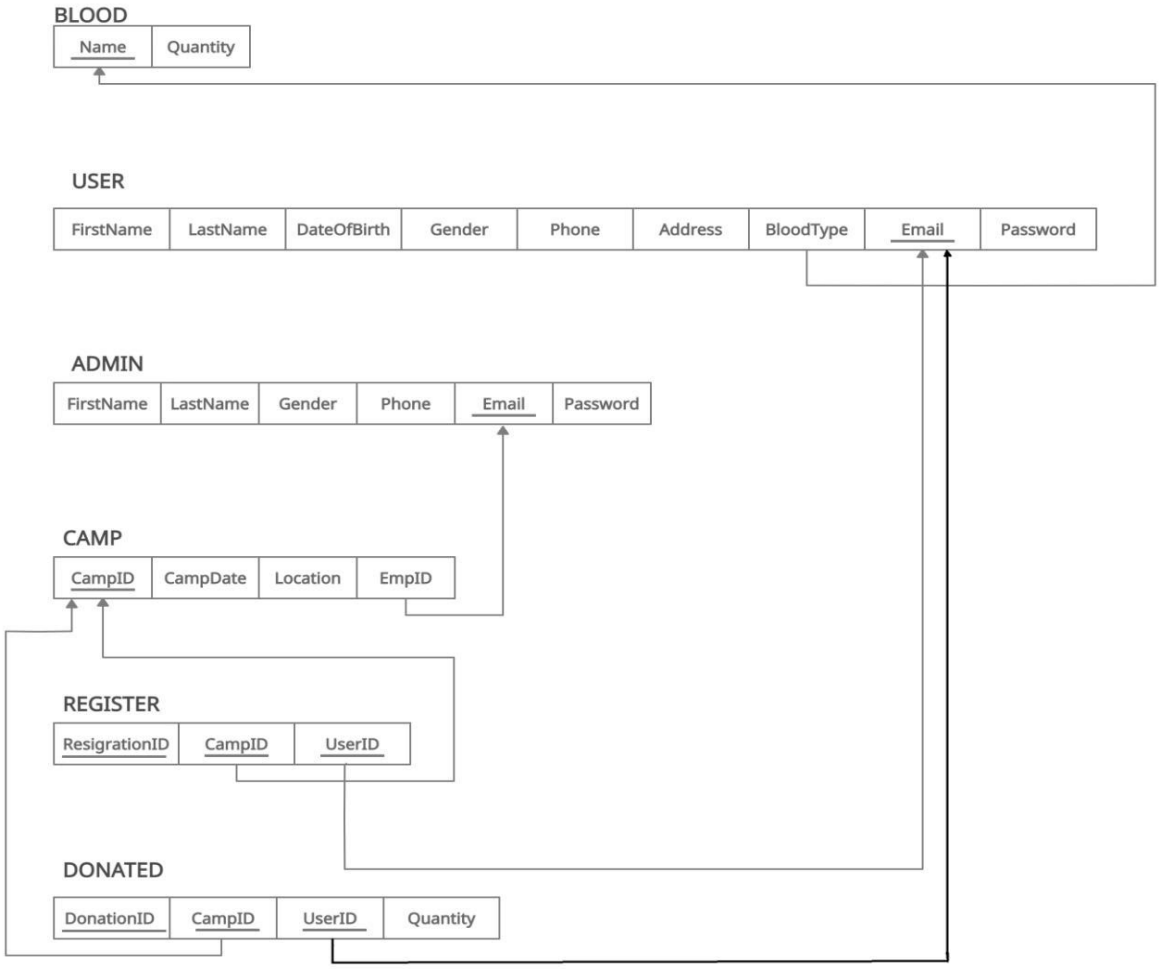

Figure 3.2

3.3 Description of the Tables

1.BLOOD: Stores the blood details

Name: Name of the blood type, Primary key

Quantity: Quantity of the blood in mL

2.USER: Stores the user details

FirstName: First name of the user

Last Name: Last name of the user

Date -Of-Birth: Date of Birth of the user

Gender: Gender of the user

Phone: Phone number of the user

Address: Address of the user

Blood Type: User's blood type, references Name in BLOOD

Email: User's email address, Primary key

Password: Login password of the user

3.ADMIN: Stores admin details

First Name: First name of the admin

Last Name: Last name of the admin

Gender: Gender of the admin

Phone: Phone number of the admin

Email: Admin's email address Primary key

Password: Login password of the admin

4.CAMP: Stores the camp details

Camp ID: ID of a particular camp, Primary key

Camp Date: Date of camp

Location: Location where the camp takes place

Emp ID: Email of the admin who created the camp, references ADMIN

5.REGISTER: Stores the registration details

Registration ID: Unique ID for each registration, Primary key

Camp ID: Camp ID referenced from CAMP, Primary key

User ID: Email of the registered user, referenced from USER, Primary key

6.DONATED: Stores the blood donation details

Donation ID : Unique ID for each donation, Primary key

Camp ID: Camp ID referenced from CAMP, Primary key

User I D: Email of the donated user, referenced from USER, Primary key

Quantity: Blood donated in Ml.

3.4 Block Diagram

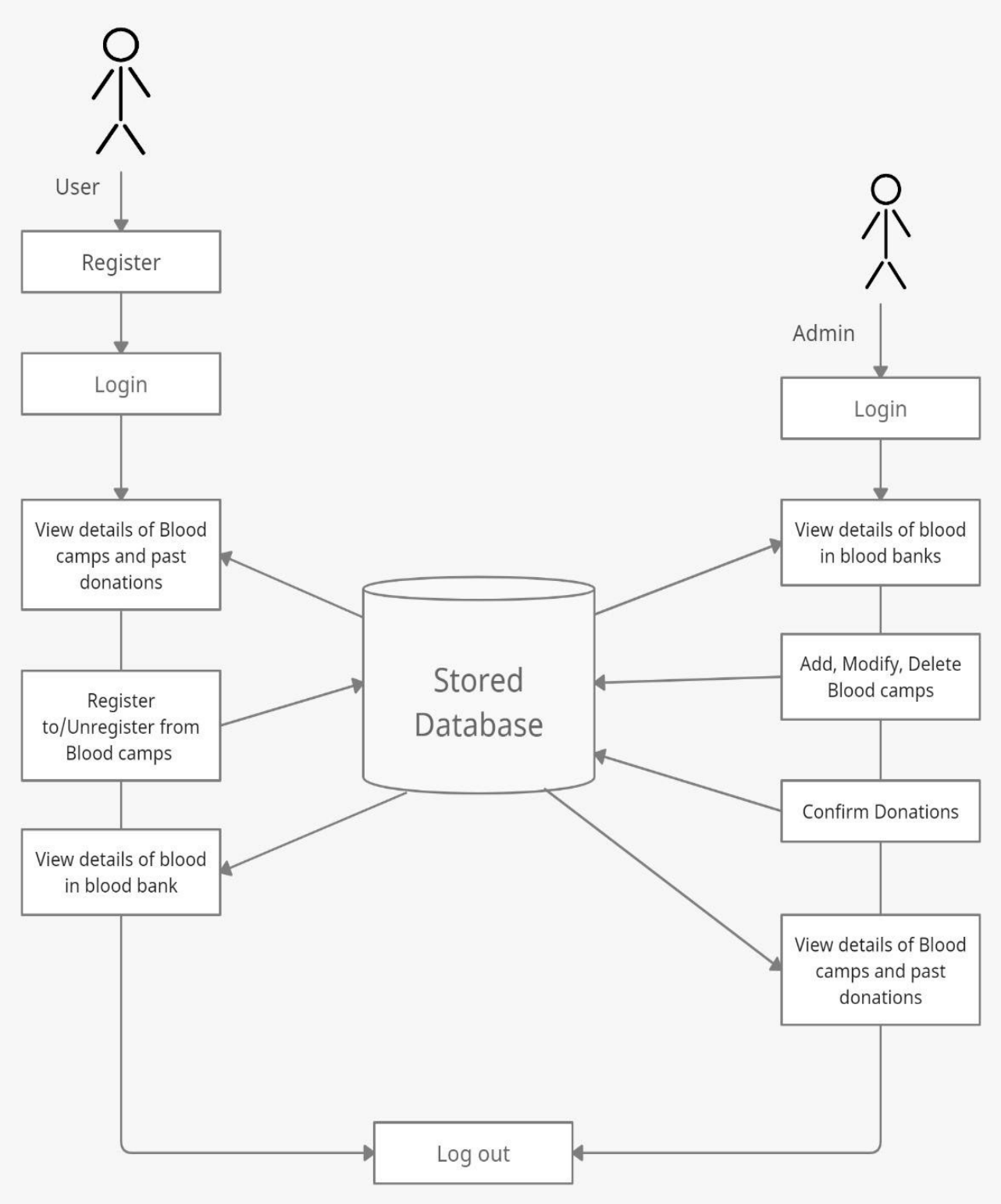

Figure 3.3

3.5 Flowchart

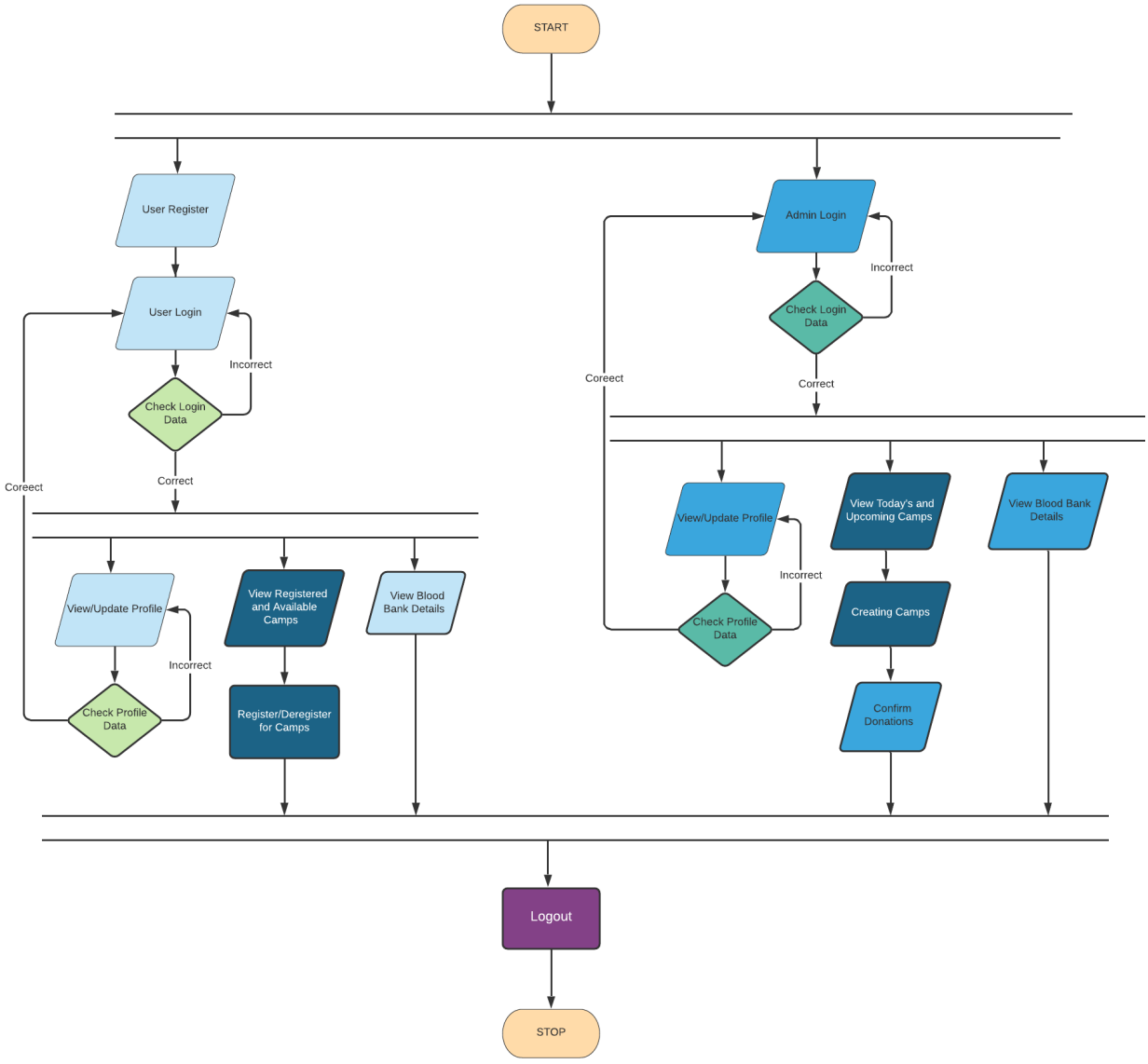

Figure 3.4

Chapter 4

IMPLEMENTATION

PHP: 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. 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.

This project uses HTML as front-end tool. Hypertext Mark-up Language (HTML) is the standard mark-up language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript, it forms a triad of cornerstone technologies for the world wide web. Web browser receive HTML documents from a web server or from local storage and render the documents into multimedia web pages.HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page.HTML provides a means to create structured documents by structural semantics for text such as headings, paragraphs, lists, links, quotes and other items.HTML elements are delineated by tags, written using angle brackets. Browsers do not display the HTML tags, but use them to interrupt the content of the page.

4.1 Code Snippet

```
$servername = "localhost";
$username = "root";
$password = "";
$database_name = "redvault";

$conn = mysqli_connect($servername,$username,$password,$database_name);

//"Now check the connection";

if(!$conn)
{
die("connection failed:" . mysqli_connect_error());
}
```

Figure 4.1

The above code snippet in Figure 4.1 is used for connecting purpose. This is used to connect back end and front end. If the connection fails then a message occurs saying could not connect.

```
if(isset($_POST['save']))
echo " Recieved";
   {
      $email=$_POST['emailID'];
      $password=$_POST['password'];
      $sql="select * from user where Email='".$email."'AND Password='".$password
            ."'limit 1";
      session_start();
      $_SESSION['email']=$email;
      $result=mysqli_query($conn,$sql);
      if(mysqli_num_rows($result)==1)
      {
         ?>
         <script type="text/javascript">
   alert("You have successfully logged in");
   window.location.href="homepage.php";
</script>
         <?php
      }
      else
      {
         ?>
         <script type="text/javascript">
                  alert("You have entered incorrect email or password");
           window.location.href="login.html";
          </script>
         <?php
      }
   }
```

Figure 4.2

The above code snippet in Figure 4.2 is used to verify the login information if the details are correct the user in redirected to the homepage or else error message occurs saying that the password or email is invalid.

```
$password = $_POST['password'];

$password_repeat =$_POST['password-repeat'];
$sql_query ="INSERT INTO `user`(`FirstName`,
`LastName`,`DateOfBirth`,`Gender`,`Phone`,`Address`,`BloodType`,`Email`,`Password`)
 VALUES ('$FName','$LName','$bDate','$gender','$phone','$address','$bloodType',
         '$email','$password')";

$exists = mysqli_query($conn,"SELECT Email FROM user WHERE email = '$email'");

if(mysqli_num_rows($exists)==0){
if($password===$password_repeat){
if(mysqli_query($conn,$sql_query))
{
         ?>
         <script type="text/javascript">
                  alert( "New Details entry inserted successfully !");
```

```
            window.location.href="login.html";
</script>
<?php
}
else
{
echo "Error: " . $sql_query . "" . mysqli_error($conn);
}
mysqli_close($conn);
}
else{
?>
<script type="text/javascript">
            alert("Password didn't match");
            window.location.href="signup.html";
</script>
<?php
}
}
else{
?>
<script type="text/javascript">
            alert("Email already exists. ");
            window.location.href="login.html";
</script>
<?php
```

Figure 4.3

The above code snippet in Figure 4.3 It takes signup data of the new user and adds it to the database .If the email is already in used or if the password and the confirm password are not same then it will show error message.

```
<?php
session_start();
$campID=$_SESSION['CampID'];
$userID=$_SESSION['UserID'];
$conn = mysqli_connect("localhost","root","","redvault");
$query="INSERT INTO register(CampID,UserID) VALUES('$campID','$userID')";
$connect=mysqli_query($conn,$query);
if($connect)
{
?>
   <script type="text/javascript">
alert("Registration Successful");
window.location.href="homepage.php";
</script>
<?php
}else{
   ?>
   <script type="text/javascript">
alert("Cannot Register");
window.location.href="homepage.php";
</script>
<?php }?>
```

Figure 4.4

The above code snippet in Figure 4.4 Adds the new Registration information into the database.

```
<?php
session_start();
$campID=$_SESSION['CampID'];
$userID=$_SESSION['UserID'];
$camp=$_GET['Camp'];
$conn = mysqli_connect("localhost","root","","redvault");
$query="DELETE FROM register WHERE CampID='$camp' AND UserID='$userID'";
$connect=mysqli_query($conn,$query);
if($connect)
{
?>
   <script type="text/javascript">
          alert("Deregistration Successful");
          window.location.href="login.html";
    </script>
<?php
}
else
{
echo "Error: " . $sql_query . "" . mysqli_error($conn);
}
mysqli_close($conn);
}
```

Figure 4.5

The above code snippet in Figure 4.5 Removes the registration information from the database.

```
if(isset($_POST['submit']))
echo "In Submit";
{
        $FName=$_POST['FName'];

        $LName=$_POST['LName'];

        $bDate=$_POST['bDate'];

        $gender=$_POST['gender'];
        echo $gender;

        $phone=$_POST['phone'];

        $address=$_POST['Address'];

        $bloodType=$_POST['bloodType'];

   //$email = $_POST['email'];

        $password = $_POST['password'];
        $password_repeat =$_POST['password-repeat'];
```

```
$sql_query ="UPDATE user set FirstName='$FName', LastName='$LName',
                DateOfBirth='$bDate',Gender='$gender', Phone='$phone', Address='$address',
                BloodType='$bloodType',Password='$password' WHERE Email='$emailID'";

//$exists = mysqli_query($conn,"SELECT Email FROM user WHERE email = '$email'");

//if(mysqli_num_rows($exists)==0){
        if($password===$password_repeat){
                if(mysqli_query($conn,$sql_query))
                {
                ?>
                        <script type="text/javascript">
                                alert( "New Details entry inserted successfully !");
                                window.location.href="login.html";
                        </script>
                <?php
                }
                else
                {
                        echo "Error: " . $sql_query . "" . mysqli_error($conn);
                }
                mysqli_close($conn);
        }
        else{
        ?>
                <script type="text/javascript">
                        alert("Password didn't match");
                        window.location.href="profile.php";
                </script>
        <?php
        }
```

Figure 4.6

The above code snippet in Figure 4.6 Updates the user details in the database .

Chapter 5

TESTING

Software testing is the process of used to identify the correctness, security, completeness and quality of developed computer software. This includes the process of executing the program or applications with the intent of finding errors. An individual unit, functions or procedures of developed project is verified and validated and these units are fit for use.

5.1 Testing process

Best testing process is to test each subsystem separately, as we have done in project. Best done during implementation. Best done after small sub-steps of the implementation rather than large chunks. Once each lowest level unit has been tested, units are combined with related units and retested in combination. This proceeds hierarchically bottom-up until the entire system is tested as a whole. Typical levels of testing:

- Module- package, abstract data type, class
- Sub-system- collection of related modules, cluster of classes, method-message paths
- Acceptance testing- whole system with real data (involve customer, user, etc)

Alpha testing is acceptance testing with a single client (common for bespoke systems).

Beta testing involves distributing system to potential customers to use and provide feedback. In this project, beta testing has been followed. This exposes system to situations and errors that might not be anticipated by us.

5.1.1 Unit testing

Unit testing is the process of testing individual software components unit or modules. Since it needs the detailed knowledge of the internal program design and code this task is done by the programmer and not by testers.

5.1.2 Integration Testing

Integration testing is another aspect of testing that is generally done in order to uncover errors associated with the flow of data across interfaces. The unit-tested modules are grouped together and tested in small segment, which makes it easier to isolate and correct errors. This approach is continued until we have integrated all modules to form the system as a whole. After the completion of each module it has been combined with the remaining module to ensure that the project is working properly as expected.

5.1.3 System Testing

System testing tests a completely integrated system to verify that it meets its requirements. After the completion of the entire module they are combined together to test whether the entire project is working properly.

5.2 Test Cases

A Test Case is a software testing document, which consists of events, action, input, output, expected result and actual result. Technically a test case includes test description, procedure, expected result and remarks. Test cases should be based primarily on the software requirements and developed to verify correct functionality and to establish conditions that reveal potential errors.

Test cases no	Test Case	Expected results	Status
1	Logging into website	Email and password provided correct	Successful
2	Logging into website	Email incorrect	Unsuccessful
3	Logging into website	Password Incorrect	Unsuccessful
4	Logging into website	Any field left empty	Unsuccessful

Table 5.1 Test Case for Login

Table 5.1 represents the test case for login module. It shows both successful and unsuccessful results for the test cases.

Test cases no	Test Case	Expected results	Status
1	Registration for new user	All details provided correctly	Successful
2	Registration for new user	Any one field is incorrect	Unsuccessful
3	Registration for new user	Any field left empty	Unsuccessful

Table 5.2 Test Case for Signup

Table 5.2 represents the test case for sign up module. It shows both successful and unsuccessful results for the test cases.

Chapter 6

SCREENSHOT

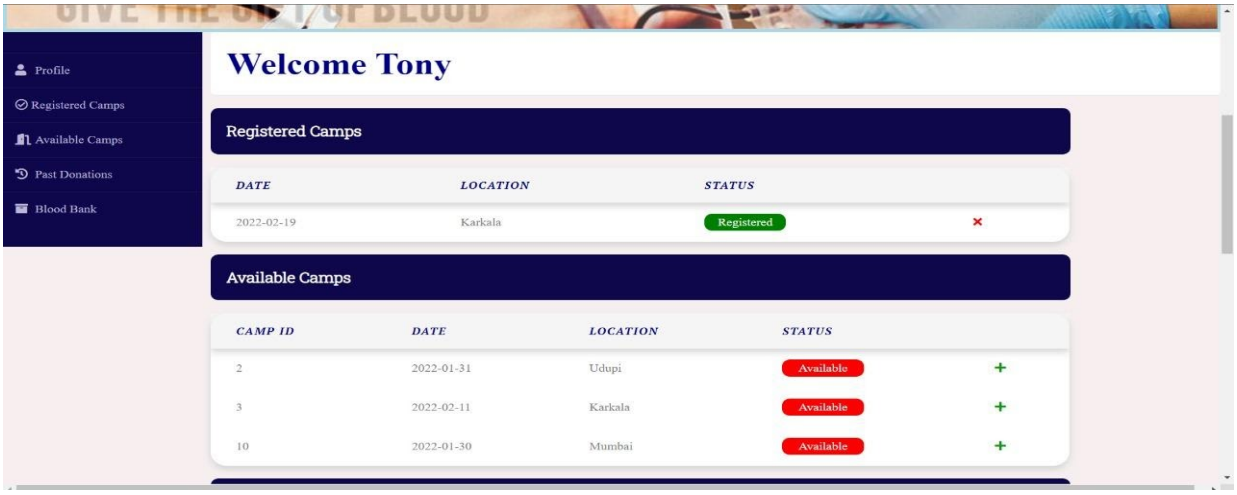

Figure 6.1 Screenshot of User Homepage

Figure 6.1 Includes user homepage view which shows the registered camps and available camps and past donations of the user and the blood details in the blood bank and navigation bar for other pages .

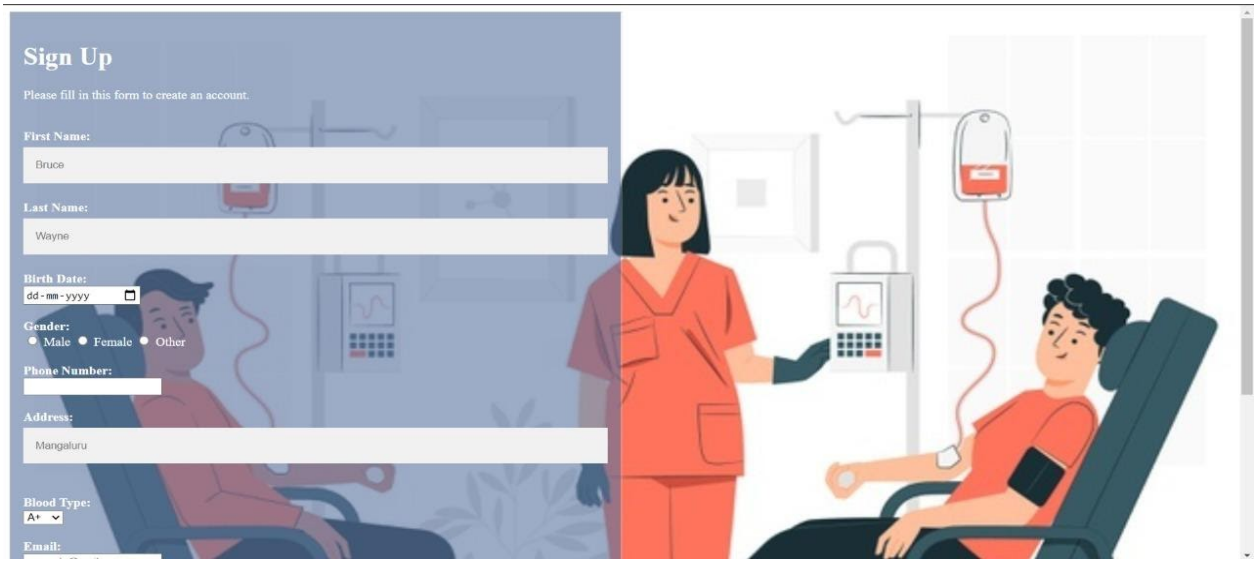

Figure 6.2 Screenshot of User Signup

Figure 6.2 User Signup includes the information of the donor who want to register. Donor can Register the account by clicking on new register. He/she can add the account for the further enquiry of the blood donation.

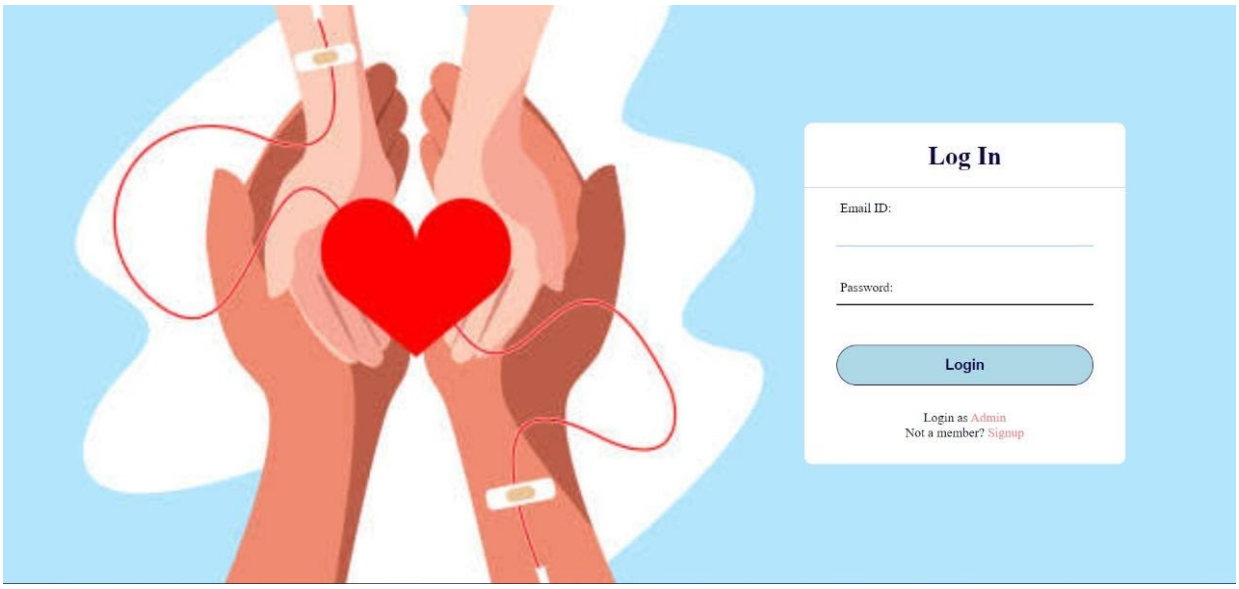

Figure 6.3 Screenshot of User Login

Figure 6.3 User can register the account by filling the information about them and click on Login button. He/she can add the account for the further enquiry of the blood donation. The user has to login to get more information about the blood bank

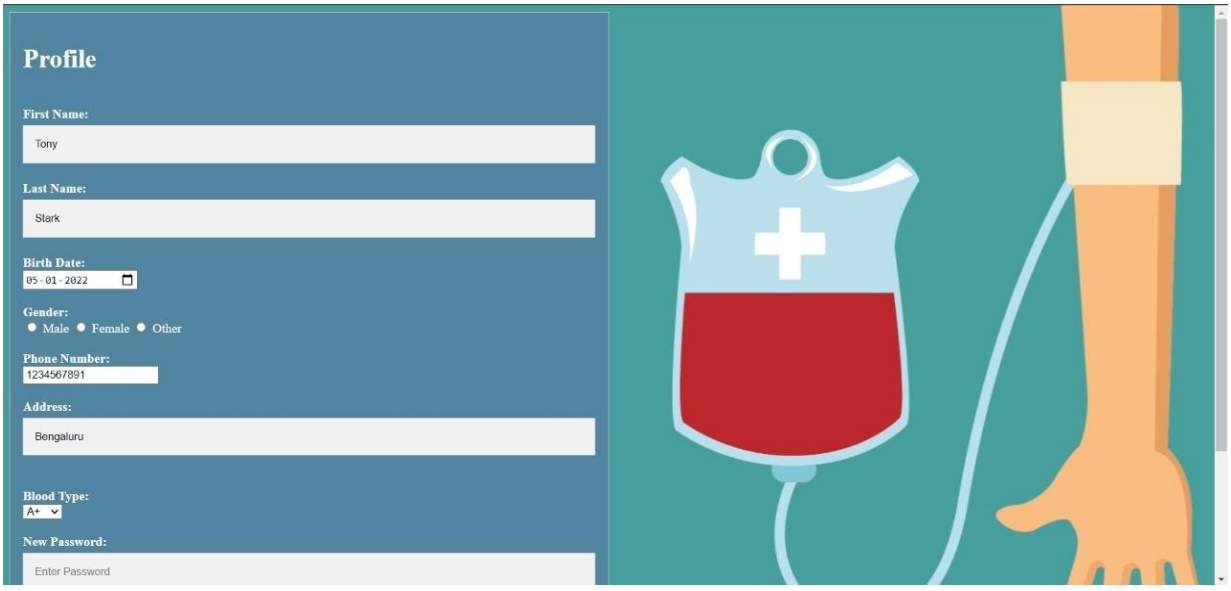

Figure 6.4 Screenshot of User Profile Update

Figure 6.4 Includes the details of the user which can be updated.

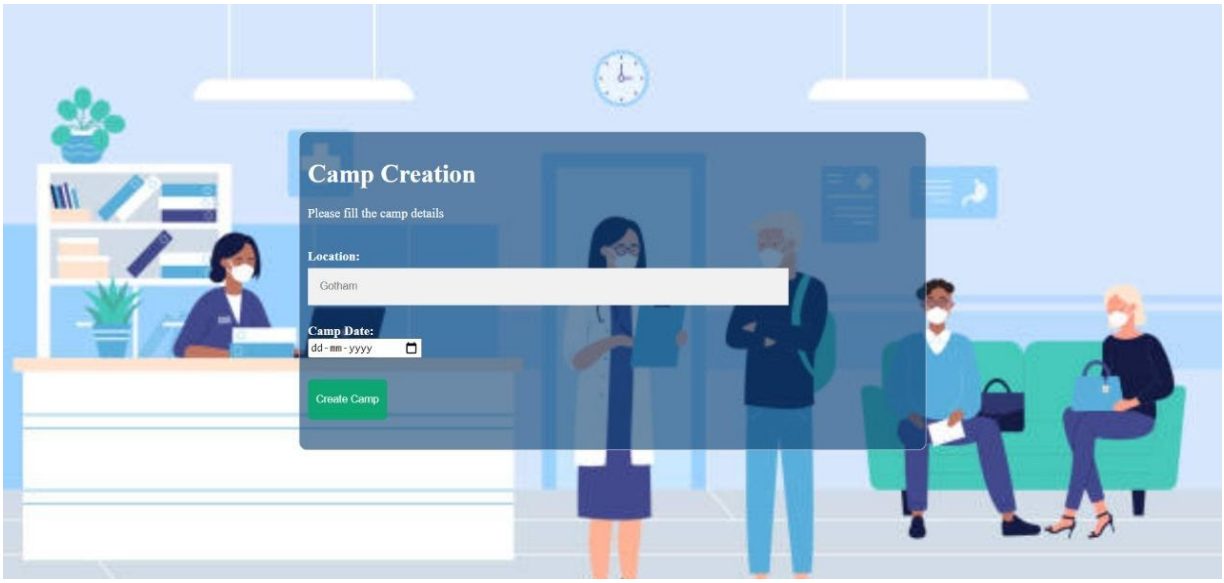

Figure 6.5 Screenshot of Camp Creation

Figure 6.5 Includes the page where Admin can create a new camp by entering the camp details.

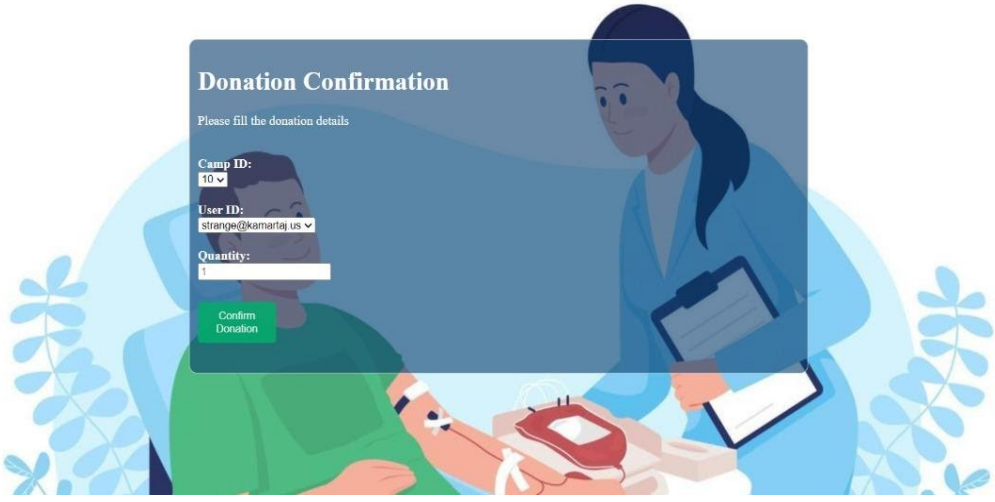

Figure 6.6 Screenshot of Confirm Donation

Figure 6.6 Includes the page where Admin can confirm the donation of the user by entering the donation information.

Chapter 7

CONCLUSION

The system provides a much-needed online web-based application to maintain a Blood Bank Database. It helps in automating the database maintenance, providing a paperless, hassle-free, easy way of creating, modifying, and deleting Donation camp events for Admins. It provides a quick and user-friendly way for Donors to register themselves for the Blood camps organized by the Admins. It also provides details about past donations of the user and the amount of each blood type preserved in the Blood Bank.

REFERENCES

[1] Fundamentals of database system, Ramez Elmarsri Shamkanth B 7th edition,2017, Pearson.

[2]www.YouTube.com

[3]www.w3school.com