BLOOD DONATION

A Project Report

Submitted in partial fulfillment of the
Requirements for the award of the Degree of

BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY) BY

Name: JIYA NANDU

UID: 23BSD022

Under the esteemed guidance of

Prof Mr. Wilson Rao

Co-ordinator

and

Ms. Sunita Jena Assistant Professor

Ms. Tejashree Parab Assistant Professor

Ms. Fatima Shaikh Assistant Professor

DEPARTMENT OF INFORMATION AND TECHNOLOGY JAI HIND COLLEGE

(*Autonomous*)

MUMBAI, 400020

MAHARASHTRA

2024-25

JAI HIND COLLEGE
(Autonomous)
MUMBAI, 400020 MAHARASHTRA
DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the project entitled, "BLOOD DONATION" is bonafied work of Student Name bearing UID (23BSD022) submitted in partial fulfillment of the requirements for the award of degree of BACHELOR OF VOCATION IN SOFTWARE DEVELOPMENT from Jai Hind College Autonomous (University of Mumbai).

Internal Guide

Coordinator

Date

External Examiner

College Seal

DECLARATION

I hereby declare that the project entitled, "Blood Donation" done at Jai Hind College, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfillment of the requirements for the award of degree of BACHELOR BVOC SOFTWARE DEVELOPMENT to be submitted as final semester project as part of our curriculum.

Name and Signature of the Student

ACKNOWLEDGEMENT

I would like to express my thanks to the people who have helped me most throughout my project. I am grateful to my guide Ms. Sunita Jena Ms. Tejashree Parab Ms. Fatima Shaikh for nonstop support for the project. I can't say thank you enough for her tremendous support and help. I owe my deep gratitude to our HOD of Information Technology Department Mr. Wilson Rao who took keen interest in our project work and guided us all along, till the completion of our project work by providing all the necessary information for developing a good system.

At last but not the least I want to thank all of my friends who helped/treasured me out in completing the project, where they all exchanged their own interesting ideas, thoughts and made this possible to complete my project with all accurate information. I wish to thank my parents for their personal support or attention who inspired/encouraged me to go my own way.

ABSTRACT

In today's fast-paced world, the need for timely blood donation and management has become a matter of life and death. Traditional methods of arranging blood often result in delays, mismanagement, and lack of availability when urgently required. To overcome these issues, the **Blood Donation Management System** is developed as a centralized digital platform to connect donors, recipients, and administrators.

The system provides a structured way to handle **user registration, donor eligibility checks, blood requests, and stock management**, ensuring that every request is processed efficiently and on time. Donors can register and update their details, recipients can submit requests and track their status, while administrators can manage stock levels, organize blood donation camps, and publish health awareness content.

By offering **real-time tracking, easy donor search, and organized camp management**, the project creates a transparent and reliable platform that bridges the gap between donors and patients. Ultimately, the system ensures that the right blood unit reaches the right person at the right time, saving countless lives and promoting a healthier society.

INDEX

Sr No			PARTICULARS	PAGE NO.	DATE
1			**Introduction**	9-14	05-07-2025
	1.1		Background	9	05-07-2025
	1.2		Objectives	10	05-07-2025
	1.3		Purpose, Scope, and Applicability	11	05-07-2025
		1.3.1	Purpose	11	05-07-2025
		1.3.2	Scope	11	05-07-2025
		1.3.3	Applicability	11	05-07-2025
	1.4		Achievements	12	05-07-2025
	1.5		Organization of Report	13-14	05-07-2025
2			**Survey Of Technologies**	15	05-07-2025
3			**Requirement And Analysis**	16-34	05-07-2025
	3.1		Problem Definition	16	05-07-2025
	3.2		Requirement Specification	17	05-07-2025
	3.3		Planning and Scheduling	18-19	26-07-2025
	3.4		Software and Hardware Requirements	20	26-07-2025
	3.5		Preliminary Product Description	21	26-07-2025
	3.6		Conceptual Models	22	26-07-2025

		3.6.1	Event Table	22	26-07-2025
		3.6.2	Use Case Diagram	23	26-07-2025
		3.6.3	Entity Relationship Diagram	24	03-08-2025
		3.6.4	Class Diagram	25	03-08-2025
		3.6.5	Object Diagram	26	03-08-2025
		3.6.6	Activity Diagram	27	03-08-2025
		3.6.7	Sequence Diagram	28	03-08-2025
		3.6.8	State Flow Diagram	29	13-08-2025
		3.6.9	Context Diagram	30	13-08-2025
		3.6.10	Data Flow Diagram	31	13-08-2025
		3.6.11	Component Diagram	32	13-08-2025
		3.6.12	Package Diagram	33	13-08-2025
		3.6.13	Deployment Diagram	34	13-08-2025
4			**System Coding**	35-66	06-08-2025
	4.1		Code	35-54	06-08-2025
	4.2		Data Dictionary	55-56	17-08-2025
	4.3		Program Description	57-59	17-08-2025
	4.4		Naming Conventions	60-63	17-08-2025
	4.5		Validations	64-66	17-08-2025
5			**Program Listing**	67-78	06-08-2025

	5.1		Cost Estimation	67-69	24-08-2025
	5.2		Schema Design	70-73	24-08-2025
	5.3		User Manual With Screenshots	74-76	24-08-2025
	5.4		Test Cases Design	77-78	24-08-2025
6			**Conclusion**	79-80	26-08-2025
	6.1		Conclusion	79	26-08-2025
	6.2		Limitations of the System	79	26-08-2025
	6.3		Future Scope of the Project	80	26-08-2025
7			**Bibliography**	131	26-08-2025

1. INTRODUCTION

1.1 Background

In India, the shortage of blood during medical emergencies continues to be a serious issue, especially in cases of accidents, surgeries, and critical illnesses. Many patients and their families struggle to arrange blood on time due to lack of coordination and awareness. Traditional methods such as contacting hospitals, blood banks, or personal networks are often slow and unreliable.

The **Blood Donation Website** aims to overcome these challenges by providing a centralized digital platform where donors, recipients, and hospitals can easily connect. It improves accessibility, transparency, and efficiency in blood donation and management. The system also promotes awareness by educating people about the importance of donating blood and ensures timely organization of blood donation camps.

1.2 Objectives

The objectives of the Blood Donation Website are:

1. Build a user-friendly web application for donor registration, blood requests, and camp organization.
2. Enable quick communication between donors, recipients, and hospitals.
3. Provide a transparent system for tracking donations, requests, and availability.
4. Maintain a real-time digital database of donors and blood stock.
5. Share health guidelines and eligibility criteria for safe blood donation.
6. Organize blood donation camps through online scheduling and promotion.
7. Support better healthcare services by ensuring timely availability of blood.

1.3 Purpose, Scope and Applicability

1.3.1 Purpose

The purpose of the Blood Donation Website is to provide a centralized digital platform that connects voluntary donors with recipients and hospitals. It simplifies the process of donating and requesting blood, promotes awareness, and ensures that blood is available when needed most.

1.3.2 Scope

The system allows:

- **Donors**: Register, update availability, and participate in camps.
- **Recipients/Hospitals**: Request and search for specific blood types.
- **Admins**: Manage donor records, monitor requests, update stock, and organize camps.
- Additionally, health-related content is included to ensure donor safety and awareness.

1.3.3 Applicability

- **For Users/Donors**: They can register, search/request blood, and access health guidelines.
- **For Admins**: Manage donor details, update blood stock, approve/reject requests, and organize donation camps.
- **For Hospitals/Organizations**: Collaborate with admins to arrange camps and ensure sufficient stock.

1.4 Achievements

With features like real-time donor search, camp management, and health education, the Blood Donation Website simplifies blood donation. It creates transparency in the system, ensures quick communication, and encourages voluntary donations. The digital records make it easier to track availability and maintain reliability, directly contributing to saving lives.

1.5 Organization of Report

This report is structured as follows:

1. **Requirement and Analysis** – Identifies the problem, requirements, and planning.
2. **System Design** – UML diagrams to explain data flow and relationships.
3. **Implementation and Testing** – Coding, database integration, and validations.
4. **Results and Discussion** – Explains system performance and effectiveness.
5. **Conclusion** – Provides summary, limitations, and future scope of the project.

2. SURVEY OF TECHNOLOGIES

The Blood Donation Website is built using modern web technologies that ensure efficiency, scalability, and user-friendliness. The technologies are categorized into **frontend, backend, database, and tools/utilities**.

2.1 Frontend Technologies

1. **HTML5**
   - Provides the structure and layout of the website.
   - Ensures that the web pages are lightweight, accessible, and browser-compatible.
2. **CSS3**
   - Used for styling and designing the user interface.
   - Provides responsive layouts, animations, and an attractive design for better user experience.
3. **Bootstrap**
   - A frontend framework for creating responsive and mobile-first designs.
   - Helps ensure that the website looks and works seamlessly across devices (desktop, tablet, and mobile).
4. **JavaScript**
   - Adds interactivity and client-side validations to the website.
   - Used for form validation (e.g., donor registration, blood requests) and enhancing user engagement.

2.2 Backend Technologies

1. **PHP**
    - A widely used server-side scripting language.
    - Handles donor registration, login authentication, blood request processing, and camp management.
2. **Apache Server (XAMPP/WAMP)**
    - Provides a local server environment to run and test the application.
    - Ensures smooth communication between the frontend, backend, and database.

2.3 Database Technology

1. **MySQL**
    - A relational database management system (RDBMS).
    - Stores donor information, blood stock, camp schedules, and user details.
    - Ensures structured data management, fast retrieval, and secure storage of records.

2.4 Additional Tools & Utilities

1. **phpMyAdmin**
    - A web-based interface to manage MySQL databases easily.
    - Used to run queries, create tables, and monitor data.
2. **Visual Studio Code / Sublime Text**
    - Code editors used for writing and debugging PHP, HTML, CSS, and JavaScript.
3. **GitHub**
    - Provides version control and collaboration tools for developers.

- Helps in tracking changes, maintaining backups, and working in teams.

3. REQUIREMENT AND ANALYSIS

3.1 Problem Definition

In India, arranging blood during emergencies is often slow and unreliable. Patients' families face challenges in finding suitable donors due to:

- Lack of coordination between hospitals, blood banks, and NGOs.
- Absence of a centralized donor database.
- Manual methods like phone calls, notices, or social media appeals, which are time-consuming.
- No transparency or proper tracking of donations and requests.

The **Blood Donation Website** addresses these problems by providing an online platform where donors, patients, and hospitals can connect instantly. It ensures efficiency, transparency, and timely availability of blood.

3.2 Requirement Specification

Functional Requirements

1. **User Registration & Verification** – Allows donors and recipients to register with valid details.
2. **Donor Module** – Stores donor details such as blood group, location, and availability.
3. **Search Blood** – Users can search for donors or available blood units by blood group and location.
4. **Blood Request** – Patients or hospitals can request specific blood groups.
5. **Donation History** – Maintains records of past donations and requests.
6. **Health Content** – Provides eligibility criteria, donor safety guidelines, and FAQs.
7. **Organizing Camps** – Admin can schedule, promote, and manage blood donation camps.
8. **Admin Dashboard** – Central control panel to manage donors, requests, blood stock, and camps.

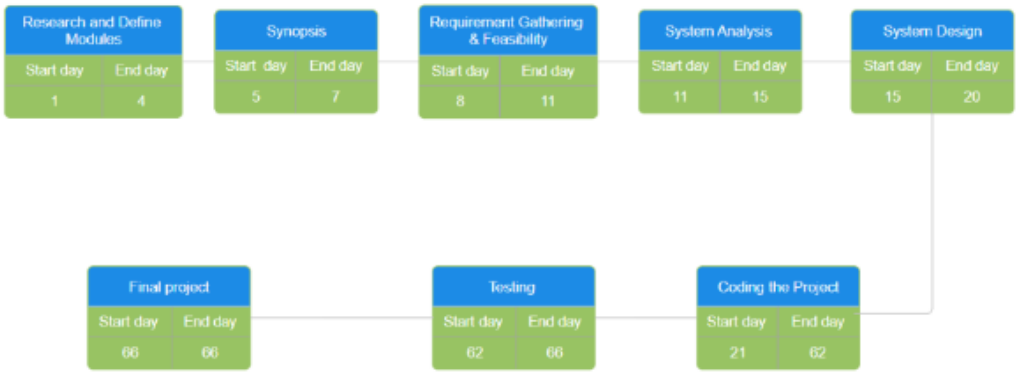
Non-Functional Requirements

1. **Performance** – The system should handle multiple donor searches and requests simultaneously.
2. **Scalability** – Must support increasing numbers of donors, hospitals, and requests.
3. **Security** – Protects sensitive user information through secure authentication.
4. **Usability** – The interface must be simple, mobile-friendly, and intuitive.
5. **Reliability** – Provides real-time and accurate updates.
6. **Compatibility** – Works across different devices and browsers.
7. **Maintainability** – The system should be easily upgradable with future features.

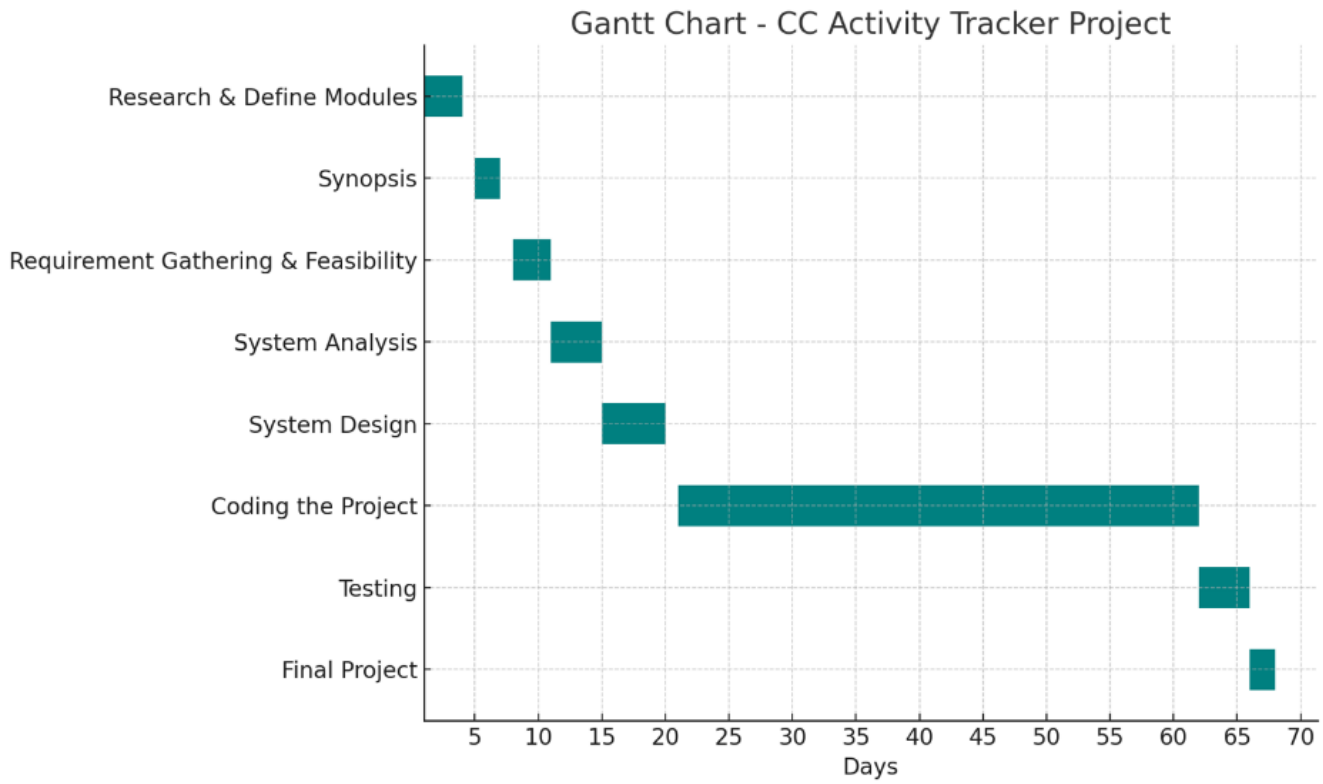
3.3 Planning and Scheduling

Pert Chart

Pert chart



Gantt Chart


Gantt Chart - CC Activity Tracker Project
Research & Define Modules
Synopsis
Requirement Gathering & Feasibility
System Analysis
System Design
Coding the Project
Testing
Final Project
5
10
15
20
25
30
35
40
45
50
55
60
65
70
Days

3.4 Software and Hardware Requirements

Software Requirements

- **Frontend:** HTML5, CSS3, Bootstrap, JavaScript
- **Backend:** PHP
- **Database:** MySQL
- **Server:** Apache (via XAMPP/WAMP)
- **IDE:** Visual Studio Code / Sublime Text

Hardware Requirements

Development Workstation:

- Processor: Intel i3/i5 or higher
- RAM: Minimum 4 GB (8 GB recommended)
- Storage: 250 GB HDD/SSD

Server Requirements:

- Processor: Multi-core
- RAM: Minimum 8 GB
- Storage: 512 GB SSD
- Internet: High-speed connectivity

3.5 Preliminary Product Description

1. **User Registration & Verification** – Donors and recipients register with valid details for authenticity.

2. **Donor Module** – Stores donor details including blood group, contact, and last donation date.

3. **Search Blood** – Users search for donors by blood group and location.

4. **Blood Request** – Patients submit requests, and matching donors are notified.

5. **Donation History** – Tracks previous donations and requests.

6. **Health Content** – Provides donor eligibility rules, safe donation tips, and awareness material.

7. **Organizing Camps** – Admin schedules and promotes blood donation camps. Donors can register online.

8. **Feedback & Improvement** – Collects user suggestions to improve the platform.
9. **Admin Dashboard** – A control panel for managing requests, donors, stock, and camps efficiently.8

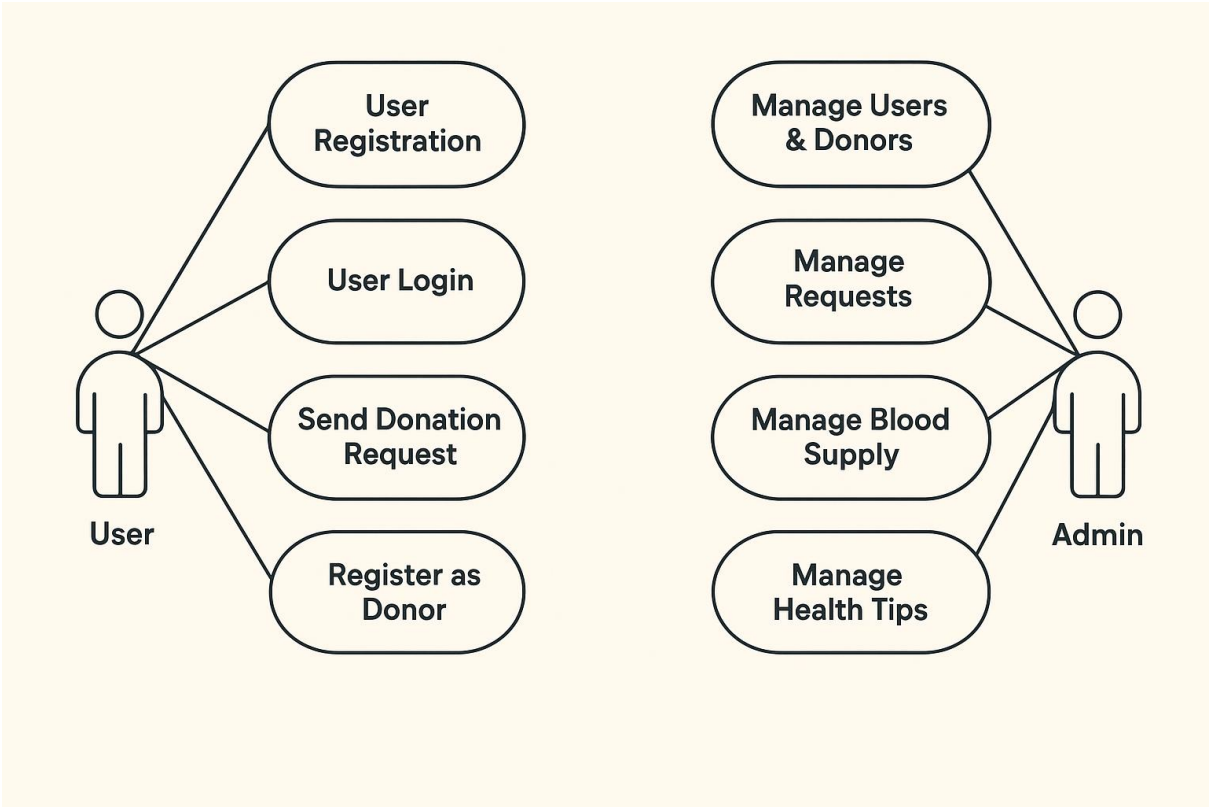
3.6 Conceptual Models

3.6.1 Event Table

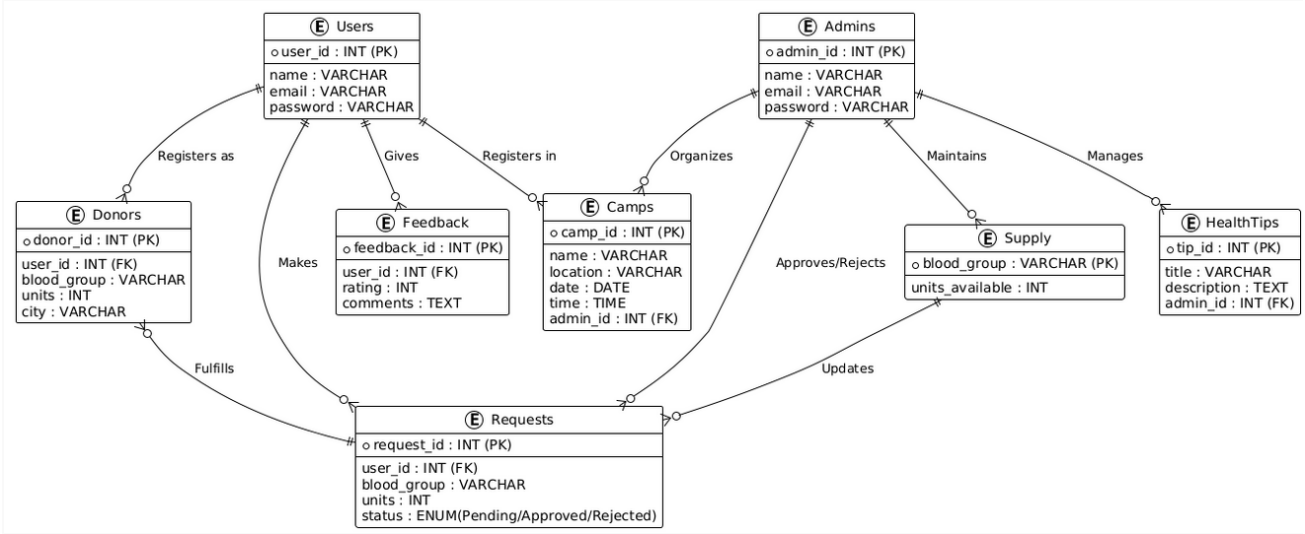
Sr No	Event	Trigger	Source	Use Case	Response	Destination
1	User Registration	User submits registration form	User (Donor/Seeker)	Register User	Store user details in database, show success message	User Dashboard
2	User Login	User enters username & password	User	Login	Authenticate and provide access	User/Admin Home Page
3	Donor Registration	Donor submits blood details	Donor	Register Donor	Save donor info, mark availability	Donor Database
4	Search for Blood	User entes blood group/location	User (Seeker)	Search Blood	Display list of matching donors	Search Results Page
5	Send Donation Request	User submits blood request	Website	Request blood of specific type & units	Request stored and notification sent	Requests Table

6	Give Feedback	User submits rating & comment	Website	Provide feedback on site/service	Feedback stored and confirmation shown	Feedback Table
7	Add/Update Health Tip	Admin adds or edits health tip	Admin Dashboard	Manage donation awareness tips	Health tip updated confirmation	Health Tips Table
8	Manage Blood Supply	Admin updates stock after donation/usage	Admin Dashboard	Track blood units per blood type	Blood supply updated	Supply Table

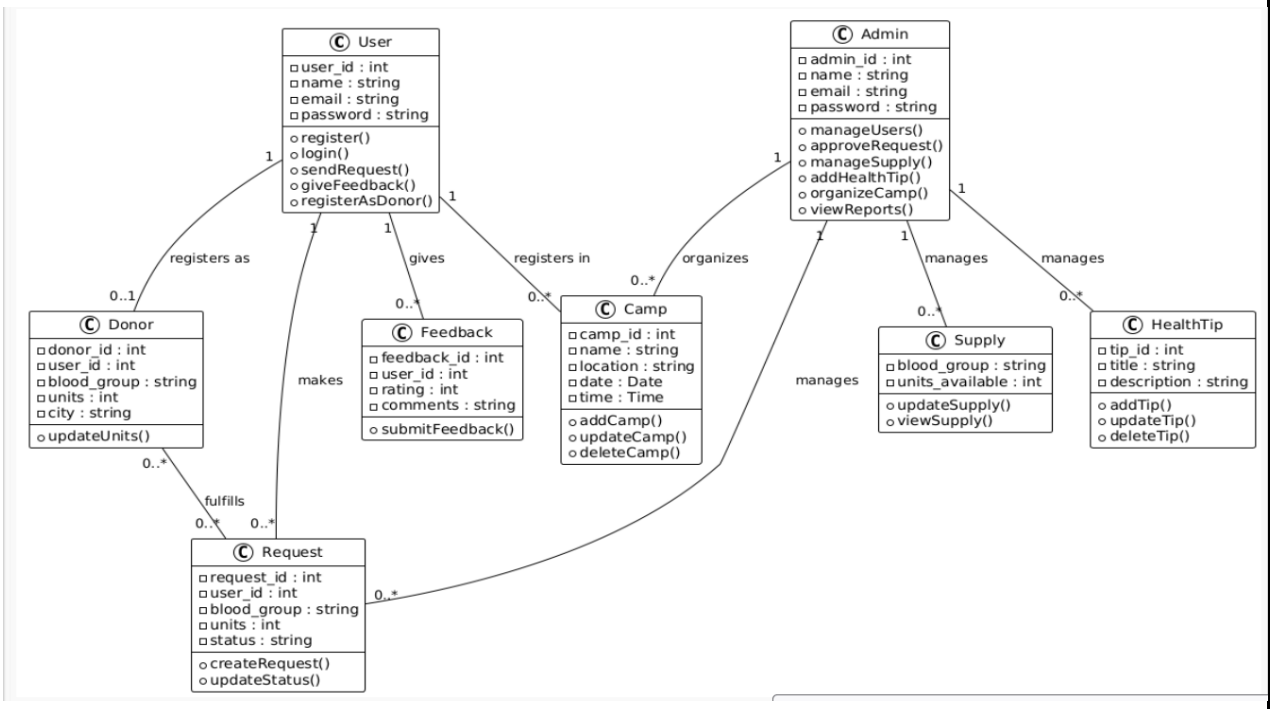
3.6.2 Use Case Diagram



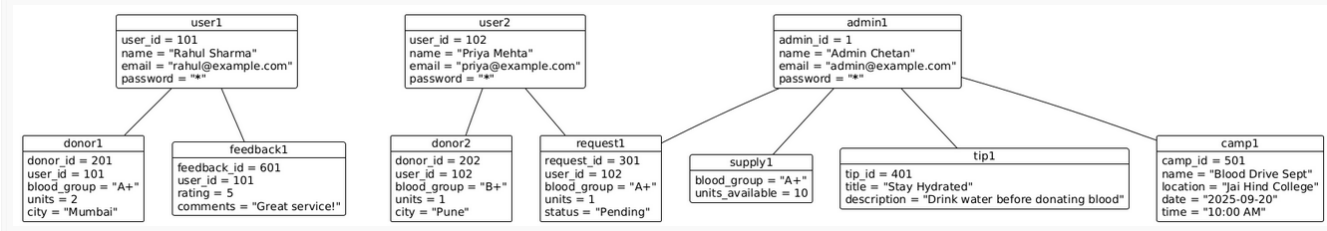
3.6.3 Entity Relationship Diagram


Users
user_id : INT (PK)
name : VARCHAR
email : VARCHAR
password : VARCHAR
Admins
admin_id : INT (PK)
name : VARCHAR
email : VARCHAR
password : VARCHAR
Registers as
Gives
Registers in
Organizes
Maintains
Manages
Donors
donor_id : INT (PK)
user_id : INT (FK)
blood_group : VARCHAR
units : INT
city : VARCHAR
Feedback
feedback_id : INT (PK)
user_id : INT (FK)
rating : INT
comments : TEXT
Camps
camp_id : INT (PK)
name : VARCHAR
location : VARCHAR
date : DATE
time : TIME
admin_id : INT (FK)
Supply
blood_group : VARCHAR (PK)
units_available : INT
HealthTips
tip_id : INT (PK)
title : VARCHAR
description : TEXT
admin_id : INT (FK)
Makes
Approves/Rejects
Fulfills
Updates
Requests
request_id : INT (PK)
user_id : INT (FK)
blood_group : VARCHAR
units : INT
status : ENUM(Pending/Approved/Rejected)

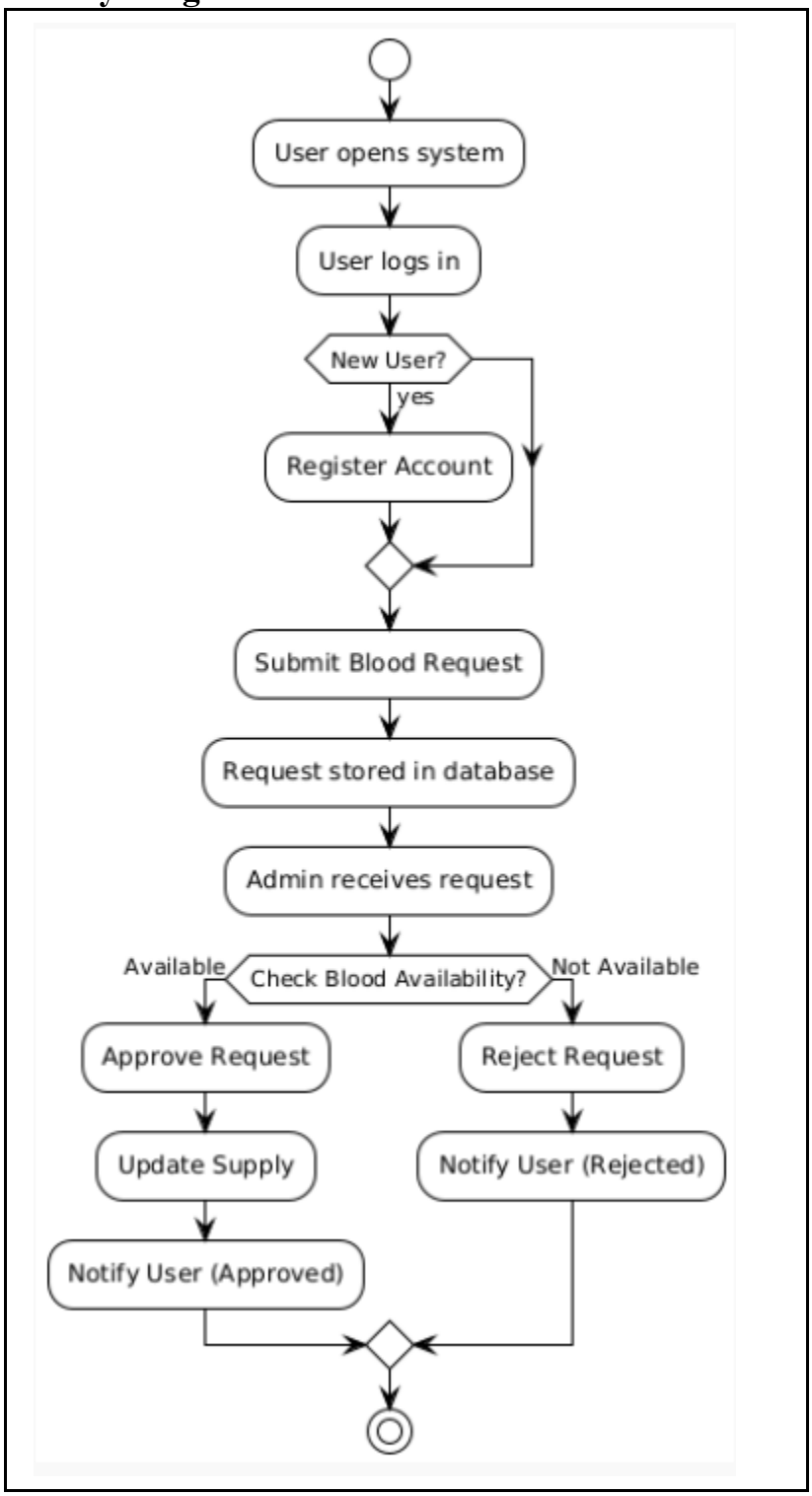
3.6.4 Class Diagram



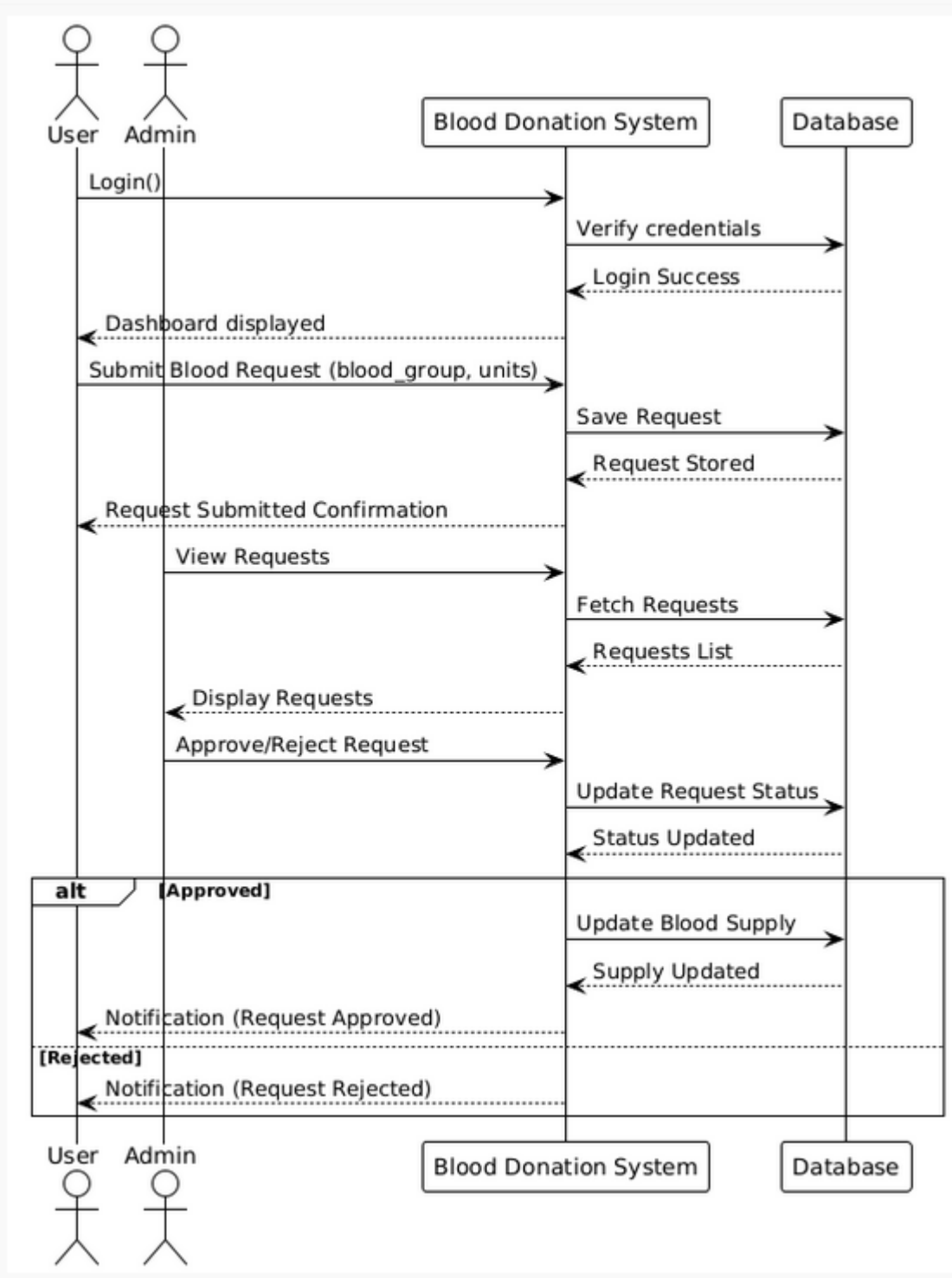
3.6.5 Object Diagram



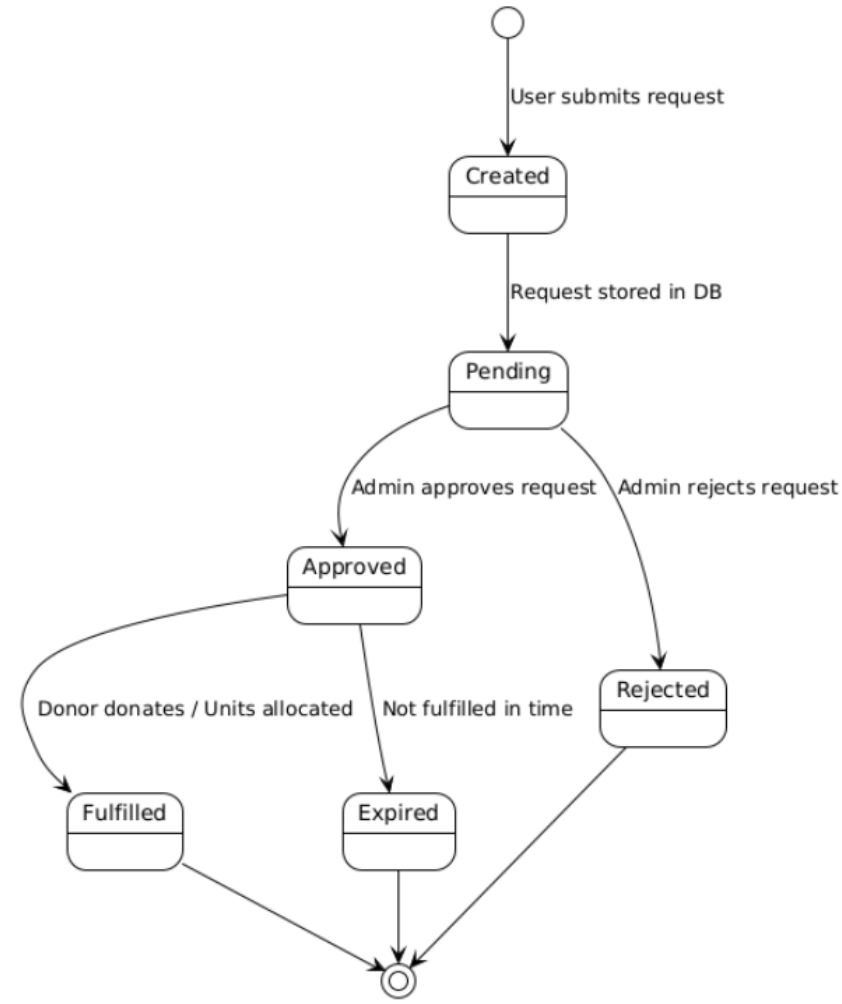
3.6.6 Activity Diagram



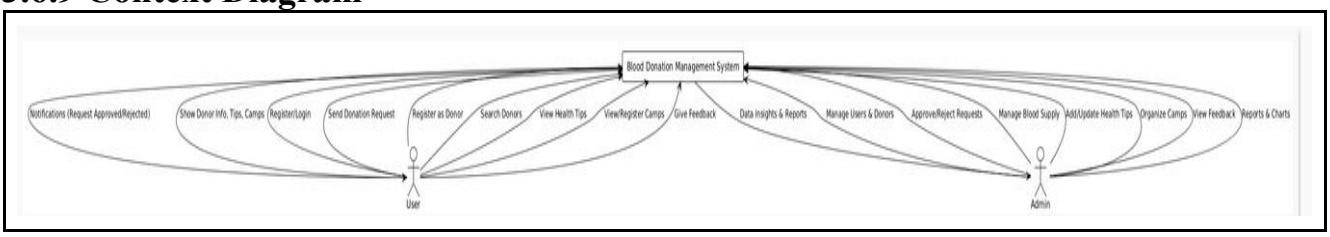
3.6.7 Sequence Diagram



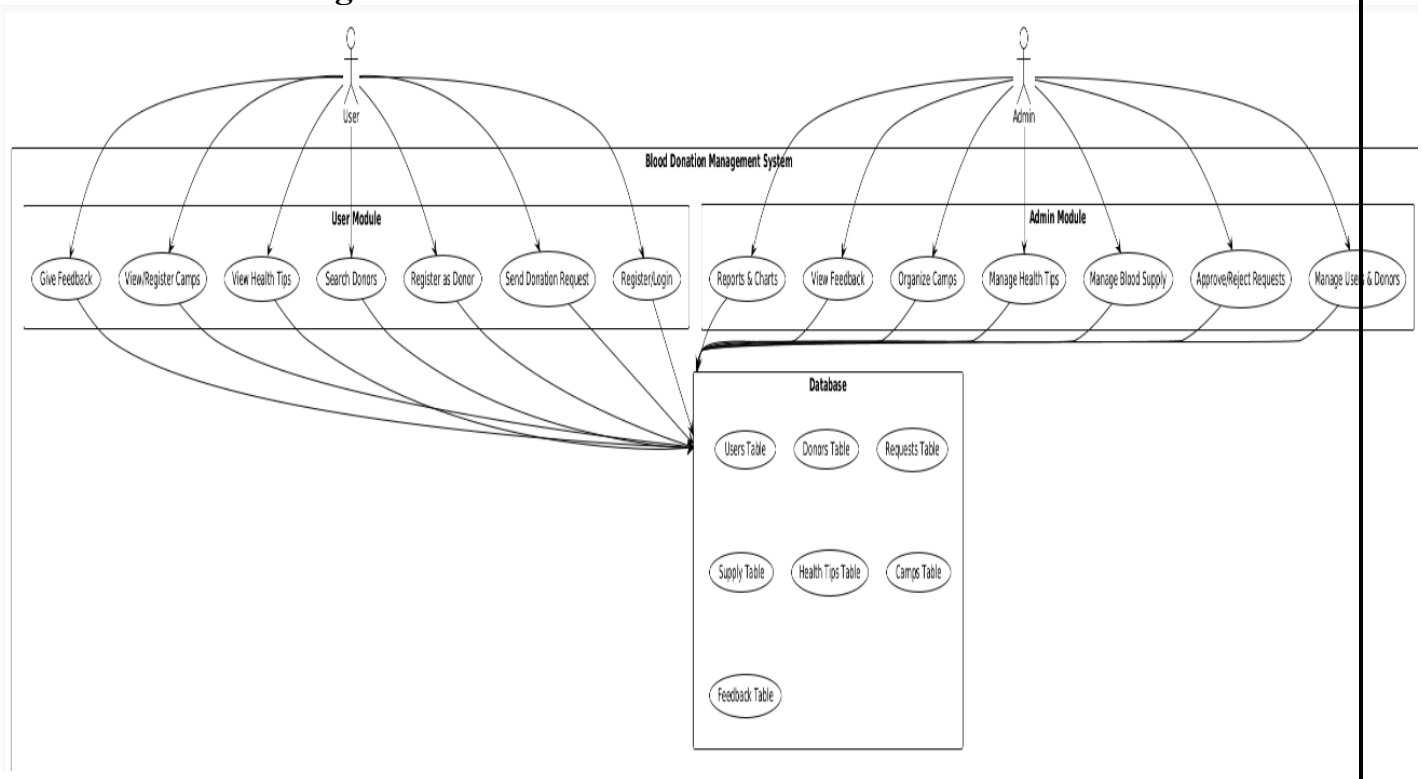
3.6.8 State Flow Diagram



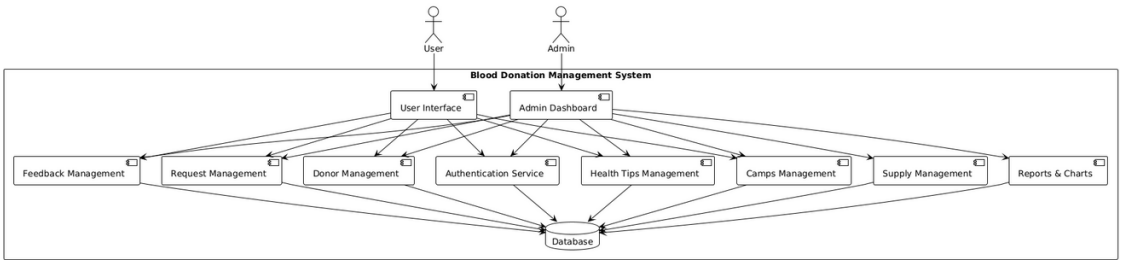
3.6.9 Context Diagram



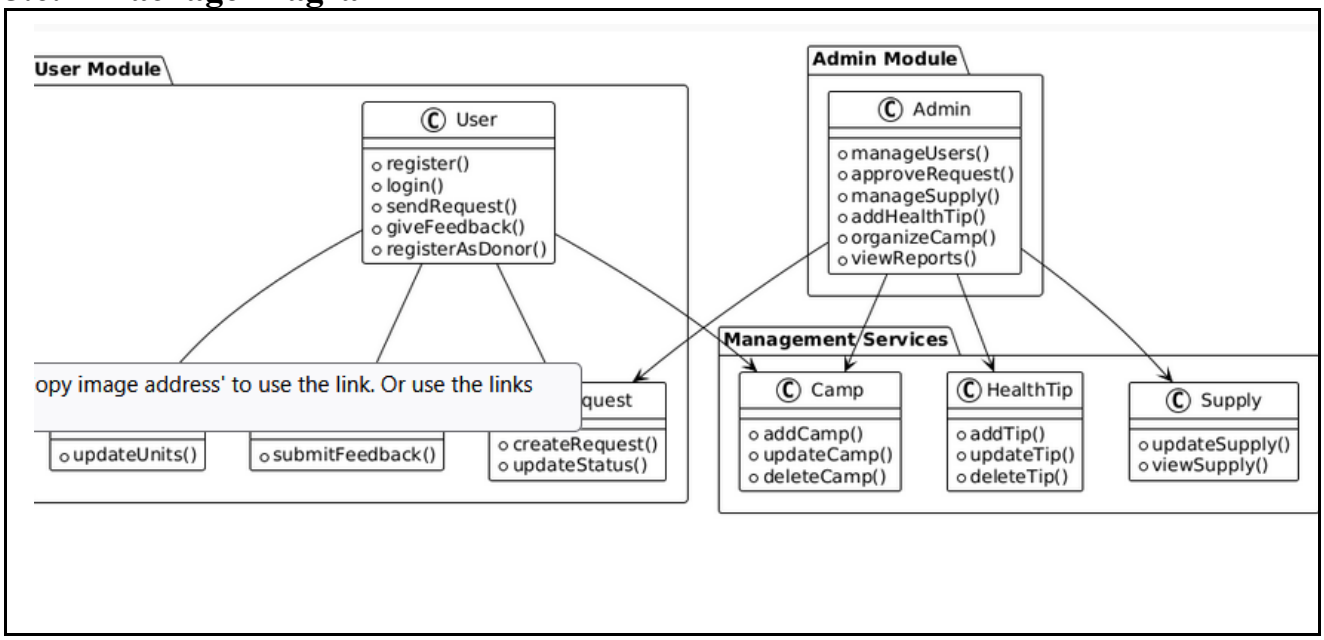
3.6.10 Data Flow Diagram



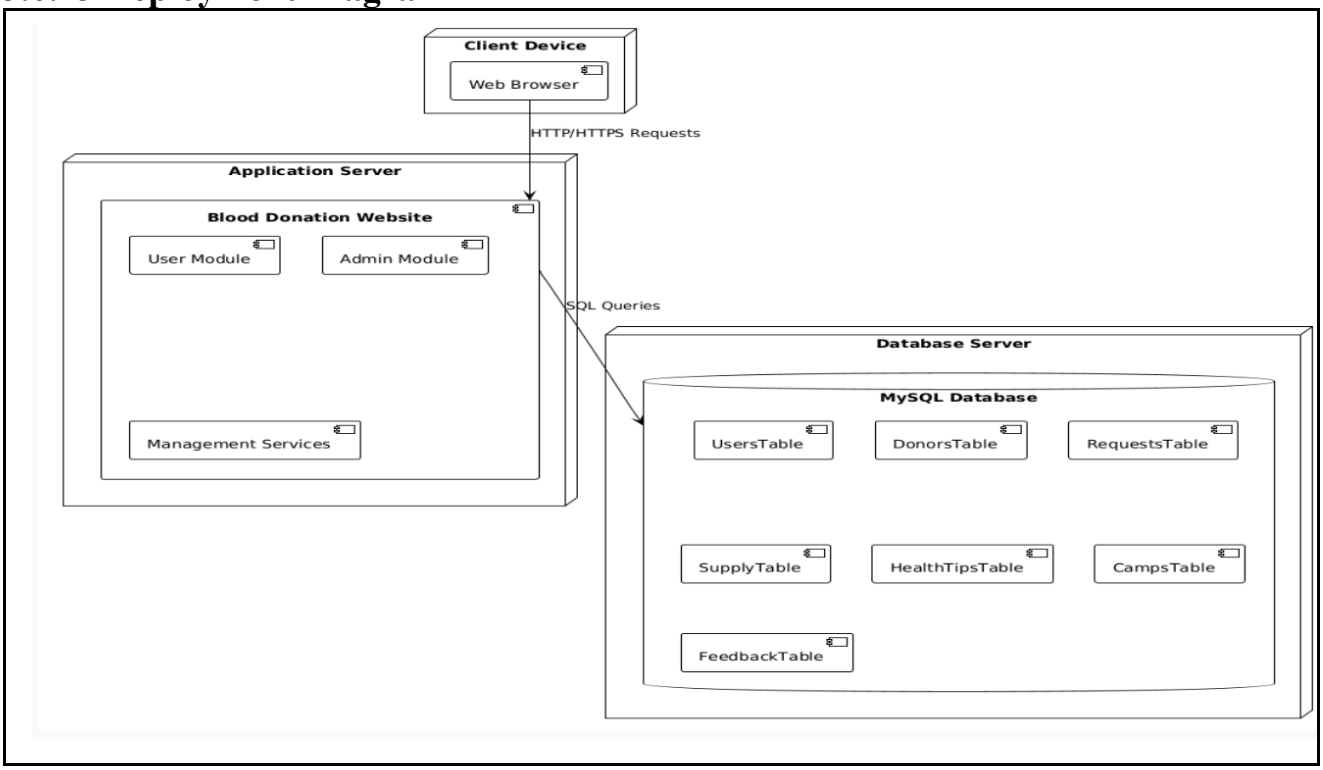
3.6.11 Component Diagram



3.6.12 Package Diagram



3.6.13 Deployment Diagram



4. System Coding

4.1 Code

Admin_camp

```
<?php

require 'config.php';

require_role('admin');



// Handle new camp creation

if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['name'])) {

      $stmt = $pdo->prepare('INSERT INTO blood_camps (name, location, camp_date,
camp_time, notes) VALUES (?,?,?,?,?)');

      $stmt->execute([

      $_POST['name'],

      $_POST['location'],

      $_POST['camp_date'],

      $_POST['camp_time'],

      $_POST['notes']

      ]);

}



// fetch all camps

$camps = $pdo->query('SELECT * FROM blood_camps ORDER BY camp_date DESC')-
>fetchAll();

?>
```

```html
<!doctype html>

<html>

<head>

  <title>Admin — Manage Blood Camps</title>

  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">

</head>

<body>

<?php include 'navbar.php'; ?>

<div class="container py-4">

  <h3>Manage Blood Donation Camps</h3>



  <div class="card p-3 mb-4">

      <h5>Create New Camp</h5>

      <form method="post">

      <div class="mb-2"><input name="name" placeholder="Camp Name" class="form-
control" required></div>

      <div class="mb-2"><input name="location" placeholder="Location" class="form-
control" required></div>

      <div class="mb-2"><input type="date" name="camp_date" class="form-control"
required></div>

      <div class="mb-2"><input type="time" name="camp_time" class="form-control"
required></div>

      <div class="mb-2"><textarea name="notes" class="form-control"
placeholder="Notes"></textarea></div>

      <button class="btn btn-primary">Create Camp</button>
```

```
      </form>
  </div>


  <h5>Existing Camps</h5>
  <table class="table table-striped">
      <thead><tr><th>Name</th><th>Location</th><th>Date</th><th>Time</th><th>Registr
ations</th></tr></thead>
      <tbody>
      <?php foreach($camps as $c):
      $regCount = $pdo->prepare('SELECT COUNT(*) FROM camp_registrations WHERE
camp_id=?');
      $regCount->execute([$c['id']]);
      $count = $regCount->fetchColumn();
      ?>
      <tr>
      <td><?=htmlspecialchars($c['name'])?></td>
      <td><?=htmlspecialchars($c['location'])?></td>
      <td><?=htmlspecialchars($c['camp_date'])?></td>
      <td><?=htmlspecialchars($c['camp_time'])?></td>
      <td><?=$count?></td>
      </tr>
      <?php endforeach; ?>
      </tbody>
  </table>
```

```
</div>

</body>

</html>
```

admin _dashbord

```
<?php

require 'config.php';

require_role('admin');


// counts and supplies

$donorCount = intval($pdo->query('SELECT COUNT(*) FROM donors')->fetchColumn());

$pendingCount = intval($pdo->query("SELECT COUNT(*) FROM donations WHERE
status='pending'")->fetchColumn());

$totalUsers = intval($pdo->query('SELECT COUNT(*) FROM users')->fetchColumn());

$supplies = $pdo->query('SELECT * FROM supplies ORDER BY blood_type')->fetchAll();


// Prepare data for Chart.js

$bloodTypes = [];

$units = [];

foreach ($supplies as $s) {

      $bloodTypes[] = $s['blood_type'];
```

```
        $units[] = intval($s['units']);
    }
?>
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Admin — BloodSave</title>
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap-icons/font/bootstrap-icons.css"
rel="stylesheet">
  <link href="styles.css" rel="stylesheet">
  <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
  <style>
      .card:hover { transform: scale(1.02); transition: 0.3s; }
      .progress { height: 12px; }
  </style>
</head>
<body class="bg-light">
<?php include 'navbar.php'; ?>
<div class="container py-4">
  <div class="d-flex justify-content-between align-items-center mb-3">
```

```
    <h3 class="mb-0">Admin Dashboard</h3>
    <a href="manage_requests.php" class="btn btn-danger">
    Manage Requests (<?= $pendingCount ?>)
    </a>
</div>

<!-- Stats Cards -->
<div class="row g-3 mb-4">
    <div class="col-md-3">
    <div class="card text-white bg-primary shadow-sm p-3">
    <div class="d-flex align-items-center">
          <i class="bi bi-people-fill fs-1 me-3"></i>
          <div>
          <small>Total Users</small>
          <h4><?= $totalUsers ?></h4>
          </div>
    </div>
    </div>
    </div>
    <div class="col-md-3">
    <div class="card text-white bg-success shadow-sm p-3">
    <div class="d-flex align-items-center">
          <i class="bi bi-droplet-fill fs-1 me-3"></i>
          <div>
```

```
        <small>Total Donors</small>
        <h4><?= $donorCount ?></h4>
        </div>
    </div>
    </div>
    </div>
    <div class="col-md-3">
    <div class="card text-white bg-warning shadow-sm p-3">
    <div class="d-flex align-items-center">
        <i class="bi bi-exclamation-triangle-fill fs-1 me-3"></i>
        <div>
        <small>Pending Requests</small>
        <h4><?= $pendingCount ?></h4>
        </div>
    </div>
    </div>
    </div>
    <div class="col-md-3">
    <div class="card text-white bg-danger shadow-sm p-3">
    <div class="d-flex align-items-center">
        <i class="bi bi-box-seam fs-1 me-3"></i>
        <div>
        <small>Manage Supply</small>
        <a href="supply.php" class="btn btn-light btn-sm mt-1">Edit Supply</a>
```

```
        </div>
    </div>
    </div>
    </div>

    <!-- Module Buttons -->
    <div class="col-md-3">
    <div class="card p-3 shadow-sm">
    <h5>Manage Health Tips</h5>
    <a class="btn btn-warning mt-2" href="admin_health_tips.php">
        <i class="bi bi-heart-pulse"></i> Go
    </a>
    </div>
    </div>

    <div class="col-md-3">
    <div class="card p-3 shadow-sm">
    <h5>Manage Blood Donation Camps</h5>
    <a class="btn btn-info mt-2" href="admin_camps.php">
        <i class="bi bi-calendar-event"></i> Go
    </a>
    </div>
    </div>
```

```
  </div>


  <!-- Blood Supply Table -->

  <h5 class="mb-2">Current Supply Levels</h5>

  <input type="text" id="searchBlood" onkeyup="searchTable()" placeholder="Search
blood type..." class="form-control mb-3 w-50 shadow-sm">


  <div class="table-responsive shadow-sm mb-4">

      <table class="table table-hover align-middle" id="supplyTable">

      <thead class="table-dark">

      <tr><th>Blood Type</th><th>Units</th><th>Progress</th><th>Last
Updated</th></tr>

      </thead>

      <tbody>

      <?php foreach ($supplies as $s): ?>

            <tr class="<?= $s['units'] < 5 ? 'table-danger' : '' ?>">

            <td><?=htmlspecialchars($s['blood_type'])?></td>

            <td><span class="badge bg-<?= $s['units'] < 5 ? 'danger' : 'success'
?>"><?=intval($s['units'])?></span></td>

            <td>

            <div class="progress">

                  <div class="progress-bar bg-<?= $s['units'] < 5 ? 'danger' :
'success' ?>" role="progressbar" style="width: <?= min(100, $s['units']*5)
?>%"></div>

            </div>

            </td>
```

```
            <td><?=htmlspecialchars($s['last_updated'])?></td>
            </tr>
        <?php endforeach; ?>
        </tbody>
        </table>
    </div>

    <!-- Charts Section -->
    <div class="row">
        <div class="col-md-6">
        <div class="card shadow-sm p-3">
        <h6 class="text-center">Blood Supply (Bar Chart)</h6>
        <canvas id="barChart"></canvas>
        </div>
        </div>
        <div class="col-md-6">
        <div class="card shadow-sm p-3">
        <h6 class="text-center">Blood Supply Distribution (Doughnut)</h6>
        <canvas id="pieChart"></canvas>
        </div>
        </div>
    </div>
</div>
```

```
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"><
/script>

<script>

// Chart.js Data

const bloodTypes = <?= json_encode($bloodTypes) ?>;

const units = <?= json_encode($units) ?>;



// Bar Chart

new Chart(document.getElementById('barChart'), {

  type: 'bar',

  data: {

      labels: bloodTypes,

      datasets: [{

      label: 'Units Available',

      data: units,

      backgroundColor: 'rgba(220, 53, 69, 0.7)',

      borderColor: 'rgba(220, 53, 69, 1)',

      borderWidth: 1

      }]

  },

  options: { responsive: true, scales: { y: { beginAtZero: true } } }

});
```

```
// Doughnut Chart
new Chart(document.getElementById('pieChart'), {
  type: 'doughnut',
  data: {
      labels: bloodTypes,
      datasets: [{
      label: 'Blood Distribution',
      data: units,
      backgroundColor: [
      '#dc3545','#fd7e14','#ffc107','#198754',
      '#0dcaf0','#6f42c1','#20c997','#6f42c1'
      ]
      }]
  },
  options: { responsive: true, plugins: { legend: { position: 'bottom' } } }
});


// Search Filter
function searchTable() {
  let input = document.getElementById("searchBlood").value.toUpperCase();
  let rows = document.getElementById("supplyTable").getElementsByTagName("tr");
  for (let i = 1; i < rows.length; i++) {
      let td = rows[i].getElementsByTagName("td")[0];
      if (td) {
```

```
        rows[i].style.display = td.textContent.toUpperCase().indexOf(input) > -1 ? ""
: "none";

        }

    }

}

</script>

</body>

</html>
```

Admin_health_tips

```
<?php

require 'config.php';

if (empty($_SESSION['user_id']) || $_SESSION['role'] !== 'admin') {
header('Location: login.php'); exit; }


// Handle Add/Edit/Delete

if ($_SERVER['REQUEST_METHOD'] === 'POST') {

        $action = $_POST['action'];

        if ($action === 'add') {

        $title = $_POST['title'];

        $content = $_POST['content'];

        $stmt = $pdo->prepare('INSERT INTO health_tips (title, content) VALUES (?,
?)');

        $stmt->execute([$title, $content]);

        } elseif ($action === 'delete') {
```

```
        $id = intval($_POST['id']);

        $pdo->prepare('DELETE FROM health_tips WHERE id=?')->execute([$id]);

        } elseif ($action === 'edit') {

        $id = intval($_POST['id']);

        $title = $_POST['title'];

        $content = $_POST['content'];

        $stmt = $pdo->prepare('UPDATE health_tips SET title=?, content=? WHERE id=?');

        $stmt->execute([$title, $content, $id]);

        }

        header('Location: admin_health_tips.php'); exit;

}



// fetch tips

$tips = $pdo->query('SELECT * FROM health_tips ORDER BY created_at DESC')-
>fetchAll();

?>

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Manage Health Tips</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css'
rel="stylesheet">
```

```
</head>

<body>

<div class="container py-4">

<h3>Manage Health Tips</h3>


<!-- Add New Tip Form -->

<div class="card mb-4 p-3">

<form method="post">

<input type="hidden" name="action" value="add">

<div class="mb-2"><input name="title" class="form-control" placeholder="Tip Title"
required></div>

<div class="mb-2"><textarea name="content" class="form-control" placeholder="Tip
Content" required></textarea></div>

<button class="btn btn-success">Add Tip</button>

</form>

</div>


<!-- Existing Tips -->

<?php foreach($tips as $tip): ?>

<div class="card mb-2">

<div class="card-body">

<form method="post" class="d-flex gap-2 align-items-start">

<input type="hidden" name="id" value="<?=intval($tip['id'])?>">

<input type="hidden" name="action" value="edit">
```

```
<div class="flex-grow-1">

<input name="title" class="form-control mb-1"
value="<?=htmlspecialchars($tip['title'])?>">

<textarea name="content" class="form-
control"><?=htmlspecialchars($tip['content'])?></textarea>

</div>

<button class="btn btn-primary">Update</button>

</form>

<form method="post" class="mt-1">

<input type="hidden" name="id" value="<?=intval($tip['id'])?>">

<input type="hidden" name="action" value="delete">

<button class="btn btn-danger">Delete</button>

</form>

</div>

</div>

<?php endforeach; ?>

<a href="admin_dashboard.php" class="btn btn-secondary mt-3">Back</a>

</div>

</body>

</html>
```

Config.php

```
<?php

// config.php
```

```php
// Start session for flash messages & auth
session_start();

// Simple flash helper
function flash_set($key, $msg) {
      $_SESSION['flash'][$key] = $msg;
}
function flash_get($key) {
      if (!empty($_SESSION['flash'][$key])) {
      $m = $_SESSION['flash'][$key];
      unset($_SESSION['flash'][$key]);
      return $m;
      }
      return null;
}

// DB connection (PDO)
$host = 'localhost';
$db   = 'blood_donation';
$user = 'root';
$pass = '';
$charset = 'utf8mb4';
$dsn = "mysql:host=$host;dbname=$db;charset=$charset";
$options = [
```

```
        PDO::ATTR_ERRMODE          => PDO::ERRMODE_EXCEPTION,
        PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
        PDO::ATTR_EMULATE_PREPARES   => false,
];
try {
        $pdo = new PDO($dsn, $user, $pass, $options);
} catch (PDOException $e) {
        http_response_code(500);
        echo "Database connection failed: " . htmlspecialchars($e->getMessage());
        exit;
}


// Helper: require login / role
function require_role($role = 'user') {
        if (empty($_SESSION['user_id']) || ($_SESSION['role'] ?? '') !== $role) {
        header('Location: login.php');
        exit;
        }
}
function require_logged_in() {
        if (empty($_SESSION['user_id'])) {
        header('Location: login.php');
        exit;
```

```
        }
}
?>

dontent
<?php
require 'config.php';
require_logged_in();

// Handle create donor / remove donor and donation requests in one endpoint
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        $uid = $_SESSION['user_id'];

        // Create donor record (user wants to be listed as donor)
        if (!empty($_POST['create_donor'])) {
        $b = $_POST['donor_blood_type'] ?? '';
        $units = max(1, intval($_POST['donor_units'] ?? 1));
        $name = $_SESSION['name'];
        $phone = ''; // optional: fetch from users table or form
        // Upsert donor for this user
        $stmt = $pdo->prepare('INSERT INTO donors
(user_id,name,blood_type,units,phone) VALUES (?,?,?,?,?) ON DUPLICATE KEY UPDATE
name=VALUES(name), blood_type=VALUES(blood_type), units=VALUES(units)');
        $stmt->execute([$uid, $name, $b, $units, $phone]);
```

```
        flash_set('success', 'You are now listed as a donor.');

        }


        // Remove donor record

        if (!empty($_POST['remove_donor'])) {

        $del = $pdo->prepare('DELETE FROM donors WHERE user_id = ?');

        $del->execute([$uid]);

        flash_set('success', 'Your donor record was removed.');

        }


        // Standard donation request (user requests to donate / offer)

        if (!empty($_POST['blood_type']) && !empty($_POST['units']) &&
empty($_POST['create_donor'])) {

        $blood_type = $_POST['blood_type'];

        $units = max(1, intval($_POST['units']));

        $notes = trim($_POST['notes'] ?? '');

        $stmt = $pdo->prepare('INSERT INTO donations (user_id,blood_type,units,notes)
VALUES (?,?,?,?)');

        $stmt->execute([$uid, $blood_type, $units, $notes]);

        flash_set('success', 'Donation request submitted.');

        }

}

header('Location: user_dashboard.php');

exit
```

Donor_search

```php
<?php
require 'config.php';
$term = trim($_GET['q'] ?? '');
$params = [];
$sql = 'SELECT * FROM donors';
if ($term !== '') {
      $sql .= ' WHERE name LIKE ? OR blood_type LIKE ? OR city LIKE ?';
      $like = "%$term%";
      $params = [$like, $like, $like];
}
$sql .= ' ORDER BY created_at DESC LIMIT 200';
$stmt = $pdo->prepare($sql);
$stmt->execute($params);
$donors = $stmt->fetchAll();
?>
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Find Donors — BloodSave</title>
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
```

```
rel="stylesheet">
  <link href="styles.css" rel="stylesheet">
</head>
<body class="bg-light">
<?php include 'navbar.php'; ?>
<div class="container py-4">
  <div class="row mb-3">
      <div class="col-md-8">
      <form class="input-group" method="get">
      <input name="q" value="<?=htmlspecialchars($term)?>" class="form-control"
placeholder="Search by name, blood type or city">
      <button class="btn btn-danger">Search</button>
      </form>
      </div>
      <div class="col-md-4 text-end">
      <a class="btn btn-outline-secondary" href="donor_search.php">Reset</a>
      </div>
  </div>


  <?php if (!empty($donors)): ?>
      <div class="row g-3">
      <?php foreach ($donors as $d): ?>
      <div class="col-md-6">
            <div class="card shadow-sm">
```

```
            <div class="card-body d-flex justify-content-between align-items-start">

            <div>

                  <h6 class="mb-1 fw-bold"><?=htmlspecialchars($d['name'])?> <small
class="text-muted">• <?=htmlspecialchars($d['blood_type'])?></small></h6>

                  <div class="small text-muted">City: <?=htmlspecialchars($d['city']
?? '—')?></div>

                  <div class="mt-2"><strong>Units:</strong> <?=intval($d['units'])?>
  <strong>Phone:</strong> <?=htmlspecialchars($d['phone'] ?? '—')?></div>

            </div>

            <div class="text-end">

                  <div class="small text-
muted"><?=htmlspecialchars($d['created_at'])?></div>

                  <?php if (!empty($_SESSION['user_id'])): ?>

                  <a href="tel:<?=htmlspecialchars($d['phone'])?>" class="btn btn-sm
btn-outline-danger mt-2">Call</a>

                  <?php endif; ?>

            </div>

            </div>

            </div>

      </div>

      <?php endforeach; ?>

      </div>

  <?php else: ?>

      <div class="alert alert-info">No donors found. Try a different query or
register as a donor.</div>

  <?php endif; ?>
```

```
</div>


<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"><
/script>

</body>

</html>


Health_tip

<?php

require 'config.php';

if (empty($_SESSION['user_id'])) { header('Location: login.php'); exit; }


// fetch tips

$tips = $pdo->query('SELECT * FROM health_tips ORDER BY created_at DESC')-
>fetchAll();

?>

<!doctype html>

<html>

<head>

      <meta charset="utf-8">

      <meta name="viewport" content="width=device-width, initial-scale=1">

      <title>Health Tips & Awareness</title>

      <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
```

```
</head>

<body>

<div class="container py-4">

      <h3>Health Tips & Awareness</h3>

      <?php foreach($tips as $tip): ?>

      <div class="card mb-3">

            <div class="card-body">

                  <h5 class="card-title"><?=htmlspecialchars($tip['title'])?></h5>

                  <p class="card-
text"><?=nl2br(htmlspecialchars($tip['content']))?></p>

                  <small class="text-muted">Added on
<?=htmlspecialchars($tip['created_at'])?></small>

            </div>

      </div>

      <?php endforeach; ?>

      <?php if(empty($tips)): ?>

      <div class="alert alert-info">No health tips available yet.</div>

      <?php endif; ?>

      <a href="user_dashboard.php" class="btn btn-secondary">Back</a>

</div>

</body>

</html>
```

Index

```
<?php require 'config.php'; ?>

<!doctype html>

<html lang="en">

<head>

  <meta charset="utf-8">

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <title>BloodSave — Home</title>

  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">

  <link href="styles.css" rel="stylesheet">

</head>

<body class="bg-light">

<?php include 'navbar.php'; ?>

<header class="py-5 bg-danger text-white">

  <div class="container text-center">

      <h1 class="display-5 fw-bold">BloodSave</h1>

      <p class="lead">Connect donors, recipients and admins — save lives
together.</p>

      <div class="mt-4">

      <?php if (empty($_SESSION['user_id'])): ?>

      <a class="btn btn-light btn-lg me-2" href="register.php">Get Started</a>

      <a class="btn btn-outline-light btn-lg" href="donor_search.php">Find
Donors</a>

      <?php else: ?>
```

```
      <a class="btn btn-light btn-lg" href="<?= $_SESSION['role'] === 'admin' ?
'admin_dashboard.php' : 'user_dashboard.php' ?>">Go to Dashboard</a>

      <?php endif; ?>

      </div>

  </div>

</header>


<main class="container my-5">

  <div class="row gy-4">

      <div class="col-md-6">

      <div class="card h-100 shadow-sm">

      <div class="card-body">

            <h5 class="card-title">Why donate?</h5>

            <p class="card-text">A single blood donation can save multiple lives.
Join our community of donors and help hospitals maintain critical supply.</p>

            <a href="register.php" class="btn btn-danger">Become a Donor</a>

      </div>

      </div>

      </div>

      <div class="col-md-6">

      <div class="card h-100 shadow-sm">

      <div class="card-body">

            <h5 class="card-title">Search donors nearby</h5>

            <p class="card-text">Quickly find donors by blood type and city. Admins
can manage supplies and requests.</p>
```

```
            <a href="donor_search.php" class="btn btn-outline-danger">Search
Donors</a>

      </div>

      </div>

      </div>

  </div>

</main>


<footer class="py-4 bg-white border-top">

  <div class="container text-center">

      <small class="text-muted">© <?=date('Y')?> BloodSave • Built with ❤️</small>

  </div>

</footer>


<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"><
/script>

</body>

</html

Login

<?php

require 'config.php';

if ($_SERVER['REQUEST_METHOD'] === 'POST') {

      $email = strtolower(trim($_POST['email'] ?? ''));

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

```
        $stmt = $pdo->prepare('SELECT * FROM users WHERE email = ? LIMIT 1');

        $stmt->execute([$email]);

        $user = $stmt->fetch();

        if ($user && password_verify($password, $user['password'])) {

        // store session data

        session_regenerate_id(true);

        $_SESSION['user_id'] = $user['id'];

        $_SESSION['name'] = $user['name'];

        $_SESSION['role'] = $user['role'];

        // redirect

        if ($user['role'] === 'admin') header('Location: admin_dashboard.php'); else
header('Location: user_dashboard.php');

        exit;

        } else {

        $error = 'Invalid credentials.';

        }

}

?>

<!doctype html>

<html lang="en">

<head>

  <meta charset="utf-8">

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <title>Login — BloodSave</title>
```

```
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">

  <link href="styles.css" rel="stylesheet">

</head>

<body class="bg-light">

<?php include 'navbar.php'; ?>

<div class="container py-5">

  <div class="row justify-content-center">

      <div class="col-md-5">

      <div class="card shadow-sm">

      <div class="card-body p-4">

            <h4 class="mb-3">Welcome back</h4>

            <?php if ($m = flash_get('success')): ?><div class="alert alert-
success"><?=htmlspecialchars($m)?></div><?php endif; ?>

            <?php if (!empty($error)): ?><div class="alert alert-
danger"><?=htmlspecialchars($error)?></div><?php endif; ?>

            <form method="post">

            <div class="mb-3">

            <label class="form-label">Email</label>

            <input name="email" type="email" class="form-control" required>

            </div>

            <div class="mb-3">

            <label class="form-label">Password</label>

            <input name="password" type="password" class="form-control" required>
```

```
            </div>

            <div class="d-flex justify-content-between align-items-center">

            <button class="btn btn-danger">Login</button>

            <a href="register.php">Create account</a>

            </div>

            </form>

      </div>

      </div>

      <div class="text-center mt-3 text-muted small">No spam — only blood donation
related messages</div>

      </div>

  </div>

</div>



<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"><
/script>

</body>

</html

Logout

<?php

require 'config.php';

session_unset();

session_destroy();

header('Location: login.php');
```

```
exit;
```

Mange request

```
<?php
require 'config.php';
require_role('admin');

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
      $id = intval($_POST['id'] ?? 0);
      $action = $_POST['action'] ?? '';
      if ($id > 0 && $action) {
      if ($action === 'approve') {
            try {
                  $pdo->beginTransaction();
                  $stmt = $pdo->prepare('SELECT * FROM donations WHERE id = ? FOR
UPDATE');
                  $stmt->execute([$id]);
                  $don = $stmt->fetch();
                  if ($don && $don['status'] === 'pending') {
                  $pdo->prepare('UPDATE donations SET status = ? WHERE id = ?')-
>execute(['approved', $id]);
                  $up = $pdo->prepare('INSERT INTO supplies (blood_type,units) VALUES
(?,?) ON DUPLICATE KEY UPDATE units = units + VALUES(units)');
                  $up->execute([$don['blood_type'], $don['units']]);
                  }
```

```
                $pdo->commit();

                flash_set('success', 'Request approved.');

            } catch (Exception $e) {

                $pdo->rollBack();

                flash_set('error', 'Error approving request: ' . $e->getMessage());

            }

        } elseif ($action === 'reject') {

            $pdo->prepare('UPDATE donations SET status = ? WHERE id = ?')-
>execute(['rejected', $id]);

            flash_set('success', 'Request rejected.');

        }

        }

        header('Location: manage_requests.php');

        exit;

}


$rows = $pdo->query('SELECT d.*, u.name AS username FROM donations d JOIN users u ON
u.id = d.user_id ORDER BY d.created_at DESC')->fetchAll();

?>

<!doctype html>

<html lang="en">

<head>

  <meta charset="utf-8">

  <meta name="viewport" content="width=device-width, initial-scale=1">
```

```
  <title>Manage Requests — BloodSave</title>

  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">

  <link href="styles.css" rel="stylesheet">

</head>

<body class="bg-light">

<?php include 'navbar.php'; ?>

<div class="container py-4">

  <h4 class="mb-3">Manage Donation Requests</h4>

  <?php if ($m = flash_get('success')): ?><div class="alert alert-
success"><?=htmlspecialchars($m)?></div><?php endif; ?>

  <?php if ($e = flash_get('error')): ?><div class="alert alert-
danger"><?=htmlspecialchars($e)?></div><?php endif; ?>



  <div class="table-responsive shadow-sm">

      <table class="table table-striped align-middle">

      <thead class="table-
light"><tr><th>User</th><th>Type</th><th>Units</th><th>Notes</th><th>Status</th><th>
When</th><th>Action</th></tr></thead>

      <tbody>

      <?php foreach ($rows as $r): ?>

            <tr>

            <td><?=htmlspecialchars($r['username'])?></td>

            <td><?=htmlspecialchars($r['blood_type'])?></td>

            <td><?=intval($r['units'])?></td>
```

```
            <td><?=htmlspecialchars($r['notes'])?></td>
            <td>
            <?php if ($r['status'] === 'approved'): ?>
                  <span class="badge bg-success">Approved</span>
            <?php elseif ($r['status'] === 'rejected'): ?>
                  <span class="badge bg-danger">Rejected</span>
            <?php else: ?>
                  <span class="badge bg-warning text-dark">Pending</span>
            <?php endif; ?>
            </td>
            <td><?=htmlspecialchars($r['created_at'])?></td>
            <td>
            <?php if ($r['status'] === 'pending'): ?>
                  <form method="post" style="display:inline-block">
                  <input type="hidden" name="id" value="<?=intval($r['id'])?>">
                  <button name="action" value="approve" class="btn btn-sm btn-
success">Approve</button>
                  </form>
                  <form method="post" style="display:inline-block">
                  <input type="hidden" name="id" value="<?=intval($r['id'])?>">
                  <button name="action" value="reject" class="btn btn-sm btn-
danger">Reject</button>
                  </form>
            <?php else: ?>
```

```
                <span class="text-muted small">No actions</span>
            <?php endif; ?>
            </td>
            </tr>
        <?php endforeach; ?>
        </tbody>
        </table>
    </div>

    <a href="admin_dashboard.php" class="btn btn-secondary mt-3">Back to Dashboard</a>
  </div>

<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"><
/script>
</body>
</html>
```

Register

```
<?php
require 'config.php';
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $name = trim($_POST['name'] ?? '');
    $email = strtolower(trim($_POST['email'] ?? ''));
```

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

        $phone = trim($_POST['phone'] ?? '');

        $role = ($_POST['role'] === 'admin') ? 'admin' : 'user';


        if (!$name || !$email || !$password) {

        $error = 'Please fill all required fields.';

        } elseif (!filter_var($email, FILTER_VALIDATE_EMAIL)) {

        $error = 'Provide a valid email.';

        } else {

        $hash = password_hash($password, PASSWORD_DEFAULT);

        $stmt = $pdo->prepare('INSERT INTO users (name,email,password,role,phone)
VALUES (?,?,?,?,?)');

        try {

                $stmt->execute([$name,$email,$hash,$role,$phone]);

                flash_set('success', 'Registration successful. Please login.');

                header('Location: login.php');

                exit;

        } catch (PDOException $e) {

                if ($e->getCode() == 23000) $error = 'Email already registered.';

                else $error = 'Error: ' . $e->getMessage();

        }

        }

}

?>
```

```
<!doctype html>

<html lang="en">

<head>

  <meta charset="utf-8">

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <title>Register — BloodSave</title>

  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">

  <link href="styles.css" rel="stylesheet">

</head>

<body class="bg-light">

<?php include 'navbar.php'; ?>

<div class="container py-5">

  <div class="row justify-content-center">

      <div class="col-md-7 col-lg-6">

      <div class="card shadow-sm">

      <div class="card-body p-4">

            <h4 class="mb-3">Create an account</h4>

            <?php if (!empty($error)): ?><div class="alert alert-
danger"><?=htmlspecialchars($error)?></div><?php endif; ?>

            <form method="post" novalidate>

            <div class="mb-3">

            <label class="form-label">Full name</label>

            <input name="name" class="form-control" value="<?=htmlspecialchars($name
```

```
?? '')?>" required>

            </div>

            <div class="mb-3">

            <label class="form-label">Email</label>

            <input name="email" type="email" class="form-control"
value="<?=htmlspecialchars($email ?? '')?>" required>

            </div>

            <div class="mb-3 row">

            <div class="col">

                  <label class="form-label">Phone</label>

                  <input name="phone" class="form-control"
value="<?=htmlspecialchars($phone ?? '')?>">

            </div>

            <div class="col">

                  <label class="form-label">Role</label>

                  <select name="role" class="form-select">

                  <option value="user" <?= (($_POST['role'] ?? '')===='user') ?
'selected' : '' ?>>User</option>

                  <option value="admin" <?= (($_POST['role'] ?? '')===='admin') ?
'selected' : '' ?>>Admin</option>

                  </select>

            </div>

            </div>

            <div class="mb-3">

            <label class="form-label">Password</label>
```

```
            <input name="password" type="password" class="form-control" required>
            </div>
            <div class="d-flex justify-content-between align-items-center">
            <button class="btn btn-danger">Register</button>
            <a href="login.php">Already have an account?</a>
            </div>
            </form>
      </div>
      </div>
      <?php if ($msg = flash_get('success')): ?><div class="alert alert-success mt-
3"><?=htmlspecialchars($msg)?></div><?php endif; ?>
      </div>
  </div>
</div>


<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"><
/script>
</body>
</html>
```

Rewards

```
<?php
require 'config.php';
if (!isset($_SESSION['user_id'])) {
```

```
        header("Location: login.php");
        exit;
    }
    $uid = $_SESSION['user_id'];

    // Fetch user points
    $stmt = $pdo->prepare("SELECT points FROM reward WHERE user_id=?");
    $stmt->execute([$uid]);
    $user = $stmt->fetch();
    $points = $user ? $user['points'] : 0;

    // Fetch available rewards
    $rewards = $pdo->query("SELECT * FROM donation_rewards")->fetchAll();
    ?>
    <!DOCTYPE html>
    <html lang="en">
    <head>
      <meta charset="UTF-8">
      <title>Rewards</title>
      <link rel="stylesheet"
    href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css">
      <style>
          body {
              background: linear-gradient(135deg, #fce4ec, #f8bbd0);
```

```
        font-family: 'Segoe UI', sans-serif;
    }
    .points-box {
        background: #fff;
        border-radius: 15px;
        padding: 20px;
        text-align: center;
        box-shadow: 0 6px 18px rgba(0,0,0,0.1);
        margin-bottom: 30px;
    }
    .points-box h2 {
        font-weight: bold;
        color: #d81b60;
    }
    .reward-card {
        border-radius: 15px;
        transition: transform 0.3s ease, box-shadow 0.3s ease;
    }
    .reward-card:hover {
        transform: translateY(-8px);
        box-shadow: 0 10px 25px rgba(0,0,0,0.15);
    }
    .reward-card h5 {
        color: #d81b60;
```

```
            font-weight: bold;
        }
        .btn-success {
            border-radius: 30px;
            padding: 8px 18px;
        }
        .btn-secondary {
            border-radius: 30px;
            padding: 8px 18px;
        }
    </style>
</head>
<body class="container py-5">

    <!-- User Points Display -->
    <div class="points-box">
        <h2>🌟 Your Points: <?= $points ?></h2>
        <p class="text-muted">Keep donating to earn more rewards 🎁</p>
    </div>

    <!-- Rewards Grid -->
    <h3 class="mb-4 text-center text-dark fw-bold">Available Rewards</h3>
    <div class="row">
```

```
      <?php foreach ($rewards as $r): ?>
      <div class="col-md-4 mb-4">
      <div class="card reward-card h-100 p-3">
            <div class="card-body text-center">
            <h5 class="card-title"><?= htmlspecialchars($r['name']) ?></h5>
            <p class="card-text">✨ Cost: <b><?= $r['points'] ?></b> Points</p>
            <?php if ($points >= $r['points']): ?>
            <a href="redeem.php?id=<?= $r['reward_id'] ?>" class="btn btn-
success">Redeem</a>
            <?php else: ?>
            <button class="btn btn-secondary" disabled>Not Enough Points</button>
            <?php endif; ?>
            </div>
      </div>
      </div>
      <?php endforeach; ?>
  </div>


</body>
</html>


Supply
<?php
require 'config.php';
```

```
require_role('admin');


if ($_SERVER['REQUEST_METHOD'] === 'POST') {
      $type = trim($_POST['blood_type'] ?? '');
      $units = intval($_POST['units'] ?? 0);
      if ($type !== '' && $units >= 0) {
      $stmt = $pdo->prepare('INSERT INTO supplies (blood_type,units) VALUES (?,?) ON
DUPLICATE KEY UPDATE units = ?');
      $stmt->execute([$type, $units, $units]);
      flash_set('success', 'Supply updated.');
      } else {
      flash_set('error', 'Invalid input.');
      }
      header('Location: supply.php');
      exit;
}


$supplies = $pdo->query('SELECT * FROM supplies ORDER BY blood_type')->fetchAll();
?>
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
```

```
  <title>Supply Management — BloodSave</title>

  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">

  <link href="styles.css" rel="stylesheet">

</head>

<body class="bg-light">

<?php include 'navbar.php'; ?>

<div class="container py-4">

  <h4>Supply Management</h4>

  <?php if ($m = flash_get('success')): ?><div class="alert alert-
success"><?=htmlspecialchars($m)?></div><?php endif; ?>

  <?php if ($e = flash_get('error')): ?><div class="alert alert-
danger"><?=htmlspecialchars($e)?></div><?php endif; ?>



  <div class="table-responsive shadow-sm">

      <table class="table table-striped">

      <thead class="table-light"><tr><th>Blood
Type</th><th>Units</th><th>Update</th></tr></thead>

      <tbody>

      <?php foreach ($supplies as $s): ?>

            <tr>

            <td><?=htmlspecialchars($s['blood_type'])?></td>

            <td><?=intval($s['units'])?></td>

            <td style="width:320px;">

            <form method="post" class="d-flex">
```

```
                <input type="hidden" name="blood_type"
value="<?=htmlspecialchars($s['blood_type'])?>">

                <input type="number" name="units" class="form-control form-control-
sm me-2" value="<?=intval($s['units'])?>" min="0">

                <button class="btn btn-sm btn-danger">Save</button>

            </form>

            </td>

            </tr>

      <?php endforeach; ?>

      <!-- quick add new blood type -->

      <tr>

            <form method="post">

            <td><input name="blood_type" class="form-control form-control-sm"
placeholder="New type (e.g. X+)"></td>

            <td><input name="units" type="number" class="form-control form-control-
sm" value="0" min="0"></td>

            <td><button class="btn btn-sm btn-outline-danger">Add</button></td>

            </form>

      </tr>

      </tbody>

      </table>

  </div>


  <a href="admin_dashboard.php" class="btn btn-secondary mt-3">Back</a>

</div>
```

```
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"><
/script>

</body>

</html>


User_camp

<?php

require 'config.php';

require_role('user');


$uid = $_SESSION['user_id'];


// Handle registration

if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['camp_id'])) {

      $camp_id = intval($_POST['camp_id']);

      $units = max(1, intval($_POST['units']));


      // check if already registered

      $check = $pdo->prepare('SELECT * FROM camp_registrations WHERE camp_id=? AND
user_id=?');

      $check->execute([$camp_id, $uid]);

      if (!$check->fetch()) {

      $stmt = $pdo->prepare('INSERT INTO camp_registrations (camp_id, user_id,
```

```
units) VALUES (?,?,?)');

      $stmt->execute([$camp_id, $uid, $units]);

      }

}



// fetch upcoming camps

$camps = $pdo->query('SELECT * FROM blood_camps WHERE camp_date >= CURDATE() ORDER
BY camp_date ASC')->fetchAll();



// fetch user registrations

$myRegs = $pdo->prepare('SELECT cr.*, bc.name, bc.location, bc.camp_date,
bc.camp_time

                        FROM camp_registrations cr

                        JOIN blood_camps bc ON bc.id = cr.camp_id

                        WHERE cr.user_id=? ORDER BY bc.camp_date ASC');

$myRegs->execute([$uid]);

$myRegistrations = $myRegs->fetchAll();

?>

<!doctype html>

<html>

<head>

  <title>Blood Donation Camps</title>

  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">
```

```
</head>

<body>

<?php include 'navbar.php'; ?>

<div class="container py-4">

  <h3>Upcoming Blood Donation Camps</h3>

  <div class="row g-3">

      <?php foreach($camps as $c): ?>

      <div class="col-md-6">

      <div class="card p-3 shadow-sm">

      <h5><?=htmlspecialchars($c['name'])?></h5>

      <p>Location: <?=htmlspecialchars($c['location'])?></p>

      <p>Date: <?=htmlspecialchars($c['camp_date'])?> | Time:
<?=htmlspecialchars($c['camp_time'])?></p>

      <p><?=htmlspecialchars($c['notes'])?></p>

      <form method="post">

            <input type="hidden" name="camp_id" value="<?=$c['id']?>">

            <div class="input-group mb-2">

            <input type="number" name="units" class="form-control" value="1"
min="1">

            <button class="btn btn-success">Register</button>

            </div>

      </form>

      </div>

      </div>
```

```
      <?php endforeach; ?>
  </div>


  <h4 class="mt-4">My Camp Registrations</h4>
  <ul class="list-group mb-4">
      <?php foreach($myRegistrations as $r): ?>
      <li class="list-group-item">
      <?=htmlspecialchars($r['name'])?> at <?=htmlspecialchars($r['location'])?> on
<?=htmlspecialchars($r['camp_date'])?>
      (<?=intval($r['units'])?> unit(s))
      </li>
      <?php endforeach; ?>
      <?php if(empty($myRegistrations)) echo "<li class='list-group-item'>No
registrations yet.</li>"; ?>
  </ul>
</div>
</body>
</html>
```

User_dashboard

```
<?php
require 'config.php';
require_role('user');
```

```php
// ✅ Start session safely
if (session_status() === PHP_SESSION_NONE) {
    session_start();
}

if (!isset($_SESSION['user_id'])) {
    header("Location: login.php");
    exit;
}

$uid = $_SESSION['user_id'];

// ✅ fetch user's requests
$stmt = $pdo->prepare('SELECT * FROM donations WHERE user_id = ? ORDER BY created_at
DESC LIMIT 5');
$stmt->execute([$uid]);
$requests = $stmt->fetchAll();

// ✅ donor record
$donorStmt = $pdo->prepare('SELECT * FROM donors WHERE user_id = ? LIMIT 1');
$donorStmt->execute([$uid]);
$myDonor = $donorStmt->fetch();

?>
```

```html
<!doctype html>

<html lang="en">

<head>

  <meta charset="utf-8">

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <title>User Dashboard — BloodSave</title>

  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">

  <link href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-
icons.css" rel="stylesheet">

  <style>

      body { background-color: #f8f9fa; }

      .card { border-radius: 15px; }

      .dashboard-title { font-weight: 600; }

      .dashboard-actions a { margin-left: 10px; }

  </style>

</head>

<body>

<?php include 'navbar.php'; ?>



<div class="container py-4">

  <!-- Greeting -->

  <div class="d-flex justify-content-between align-items-center mb-4 flex-wrap">

      <h3 class="dashboard-title">👋 Welcome,
```

```
<?=htmlspecialchars($_SESSION['name'])?></h3>

      <div class="dashboard-actions">

      <a href="donor_search.php" class="btn btn-danger mb-2">

      <i class="bi bi-search-heart"></i> Find Donors

      </a>

      <a href="health_tips.php" class="btn btn-info mb-2">

      <i class="bi bi-heart-pulse"></i> Health Tips & Awareness

      </a>

      <a href="user_camps.php" class="btn btn-warning mb-2">

      <i class="bi bi-calendar-event"></i> Blood Donation Camps

      </a>

      </div>

  </div>


  <div class="row g-4">

      <!-- Left Column -->

      <div class="col-lg-6">


      <!-- Send Donation Request -->

      <div class="card shadow-sm mb-4">

      <div class="card-body">

            <h5 class="card-title"><i class="bi bi-droplet"></i> Send Donation
Request</h5>

            <form method="post" action="donate.php">
```

```
            <div class="mb-2">
            <label class="form-label">Blood Type</label>
            <select name="blood_type" class="form-select" required>
                  <option>A+</option><option>A-</option><option>B+</option><option>B-
</option>
                  <option>O+</option><option>O-
</option><option>AB+</option><option>AB-</option>
            </select>
            </div>
            <div class="mb-2">
            <label class="form-label">Units</label>
            <input name="units" type="number" class="form-control" value="1" min="1"
required>
            </div>
            <div class="mb-2">
            <label class="form-label">Notes (optional)</label>
            <textarea name="notes" class="form-control" rows="2"></textarea>
            </div>
            <div class="d-flex justify-content-between">
            <button class="btn btn-success"><i class="bi bi-send"></i>
Submit</button>
            <a href="donor_search.php" class="btn btn-outline-secondary"><i
class="bi bi-search"></i> Search Donors</a>
            </div>
            </form>
      </div>
```

```
        </div>


        <!-- Register as Donor -->
        <div class="card shadow-sm">
        <div class="card-body">
                <h5 class="card-title"><i class="bi bi-person-plus"></i> Register as
Donor</h5>
                <?php if ($myDonor): ?>
                <p class="mb-2">✅ You are registered as donor:
                <strong><?=htmlspecialchars($myDonor['blood_type'] ?? '')?></strong>
                (<?=intval($myDonor['units'] ?? 0)?> unit(s))
                </p>
                <form method="post" action="donate.php" onsubmit="return confirm('Remove
donor?');">
                <input type="hidden" name="remove_donor" value="1">
                <button class="btn btn-danger"><i class="bi bi-trash"></i> Remove Donor
Record</button>
                </form>
                <?php else: ?>
                <form method="post" action="donate.php" class="row g-2">
                <input type="hidden" name="create_donor" value="1">
                <div class="col-6">
                        <select name="donor_blood_type" class="form-select" required>
                        <option>A+</option><option>A-</option><option>B+</option><option>B-
</option>
```

```
                <option>O+</option><option>O-
</option><option>AB+</option><option>AB-</option>

                </select>

            </div>

            <div class="col-3">

                <input name="donor_units" type="number" class="form-control"
min="1" value="1" required>

            </div>

            <div class="col-3">

                <button class="btn btn-primary w-100">Register</button>

            </div>

            </form>

            <?php endif; ?>

        </div>

        </div>

        </div>


        <!-- Right Column -->

        <div class="col-lg-6">


        <!-- Donation Requests -->

        <div class="card shadow-sm mb-3">

        <div class="card-body">

            <h5 class="card-title"><i class="bi bi-clipboard2-pulse"></i> Your
Donation Requests</h5>
```

```
            <ul class="list-group">

            <?php if (!empty($requests)): foreach ($requests as $r): ?>

            <li class="list-group-item d-flex justify-content-between align-items-
start">

                  <div>

                  <div class="fw-bold"><?=htmlspecialchars($r['blood_type'])?> •
<?=intval($r['units'])?> unit(s)</div>

                  <div class="small text-
muted"><?=htmlspecialchars($r['notes'])?></div>

                  </div>

                  <div class="text-end">

                  <?php if ($r['status'] === 'approved'): ?>

                  <span class="badge bg-success"><i class="bi bi-check-circle"></i>
Approved</span>

                  <?php elseif ($r['status'] === 'rejected'): ?>

                  <span class="badge bg-danger"><i class="bi bi-x-circle"></i>
Rejected</span>

                  <?php else: ?>

                  <span class="badge bg-warning text-dark"><i class="bi bi-hourglass-
split"></i> Pending</span>

                  <?php endif; ?>

                  <div class="small text-muted mt-
1"><?=htmlspecialchars($r['created_at'])?></div>

                  </div>

            </li>

            <?php endforeach; else: ?>

            <li class="list-group-item">No requests yet. Use the form to send a
```

```
donation request.</li>
            <?php endif; ?>
            </ul>
      </div>
      </div>
      </div>
  </div>
</div>


<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"><
/script>
</body>
</html>
```

4.2 Data Dictionary

Field Name	Data Type	Description
request_id	INT (PK)	Unique request ID
user_id	INT (FK)	Requestor ID
blood_group	VARCHAR(5)	Required blood group
units	INT	Units of blood required
hospital	VARCHAR(150)	Hospital name
status	ENUM(pending, approved, rejected, fulfilled)	Request status

4.2.4 Camps Table

Field Name	Data Type	Description
camp_id	INT (PK)	Unique ID for camp
name	VARCHAR(100)	Camp name
location	VARCHAR(150)	Camp location
date	DATE	Camp date
time	TIME	Camp time

4.2.5 Health Content Table

Field Name	Data Type	Description
content_id	INT (PK)	Unique content ID
title	VARCHAR(200)	Title of health guideline
description	TEXT	Detailed information

4.2.6 Feedback Table

Field Name	Data Type	Description
feedback_id	INT (PK)	Unique feedback ID
user_id	INT (FK)	Feedback giver
message	TEXT	User feedback message
rating	INT (1–5)	User rating

4.4.3 Program Description

1] INTRODUCTION

→ Purpose:

The purpose of the Blood Donation Management System is to create a web-based platform that connects donors, recipients, hospitals, and administrators. The system enables citizens to register as blood donors, request blood in emergencies, and organize blood donation camps efficiently. It promotes awareness, transparency, and timely availability of blood for those in need.

→ Key Features:

Role Selection: Users can act as donors, recipients, or admins depending on their role.

Register: Citizens can create an account to either donate blood or request blood.

Login: Registered users can securely log in with their credentials.

Donor Registration: Donors can enter their blood group, health details, and availability.

Blood Request Submission: Recipients or hospitals can submit blood requests**.**

Request Tracking: Users can track the status of their requests (Pending, Approved, Fulfilled, Rejected).

Admin Dashboard: Admins can manage donors, approve/reject requests, and monitor stock.

Camp Management: Admins can schedule and organize blood donation camps.

Health Awareness: Provides guidelines, FAQs, and donor eligibility information.

Feedback System: Users can submit feedback to improve services.

2] OVERVIEW

→ **Project Purpose:**
The system is designed to streamline the process of blood donation and request management. It reduces manual effort, improves record-keeping, and ensures that blood is available during emergencies. The project also encourages voluntary donation and builds a digital network between hospitals, donors, and recipients.

→ **Target Audience:**
This platform is intended for:

- **Donors** – willing to contribute blood.
- **Recipients / Patients** – who require blood in emergencies.
- **Hospitals / Healthcare Providers** – who handle blood distribution.
- **Administrators** – who manage requests, donors, and stock.

3] FUNCTIONALITY

→ **Key Features and Their Working:**

1. **Role Selection**
    - **Working:** Users choose their role (Donor, Recipient, or Admin) at login.
    - **Key UI Components:** Dropdown or radio buttons for role selection.
2. **Register and Login**
    - **Working:**
        - *Register:* Allows users to sign up with details like name, email, phone, and password.
        - *Login:* Enables registered users to access their accounts securely.
    - **Validations:**
        - Email must be unique and valid.
        - Password must be at least 8 characters.
        - Mandatory fields must not be blank.
3. **Donor Registration**

- **Working:** Donors enter health details (age, weight, blood group, last donation date).
- **Key UI Components:** Input fields for health details, availability toggle.
- **Validations:** Donor must be 18–60 years, weight ≥ 50 kg, and 3 months gap after last donation

4 Blood Request Submission

- **Working:** Recipients enter details like blood group, number of units, hospital name, and urgency.
- **Key UI Components:** Input fields for blood group, hospital, units, and urgency.
- **Validations:** Blood group format must be valid (A+, B-, O+, etc.). Units must not exceed 5.

4. **Request Tracking**
   - **Working:** Recipients can track status of requests (Pending, Approved, Fulfilled, Rejected).
   - **Key UI Components:** Status indicators and request history list.
5. **Admin Dashboard**
   - **Working:** Admin can view requests, approve/reject them, update blood stock, and manage donors.
   - **Key UI Components:** List of requests, donor database, camp scheduling options.
6. **Camp Management**
   - **Working:** Admins schedule blood donation camps by entering date, location, and time.
   - **Key UI Components:** Input fields for camp details, notifications for donors.
7. **Health Awareness & Feedback**
   - **Working:** Provides donation guidelines and FAQs. Collects user feedback and ratings.
   - **Key UI Components:** Text display for awareness messages, form for feedback submission.

4] USER INTERFACE (UI) COMPONENTS

Main UI Elements Across Pages:

1. **Sign-Up and Login Pages**
    - **Components:** Input fields for name, email, password, and login button.
    - **Functionality:** Allows users to register or log in securely.
2. **Donor Registration Page**
    - **Components:** Input fields for age, weight, blood group, last donation date.
    - **Functionality:** Collects health details to verify donor eligibility.
3. **Blood Request Page**
    - **Components:** Input fields for hospital name, units, blood group, urgency level.
    - **Functionality:** Allows patients/hospitals to request blood.
4. **Request Tracking Page**
    - **Components:** List of user requests with colored status indicators.
    - **Functionality:** Displays the current status of blood requests.
5. **Admin Dashboard**
    - **Components:** Request list, approval buttons, camp scheduling options, stock update forms.
    - **Functionality:** Centralized control for managing the system.

5] ANALYSIS BASED ON UI AND CODE

1. **Login and Sign-Up Pages**
    - **UI Analysis:** Simple design with fields for credentials, easy navigation.
    - **Code Analysis:** Validates email and password, encrypts passwords, and stores user details in MySQL.
2. **Donor Registration Page**
    - **UI Analysis:** Collects donor-specific data with proper validations.
    - **Code Analysis:** Ensures eligibility by checking age, weight, and donation gap. Updates donor database.
3. **Blood Request Page**
    - **UI Analysis:** Input fields for hospital and blood details, with clear layout.
    - **Code Analysis:** Validates request details, checks stock, and stores request in database.
4. **Admin Dashboard**
    - **UI Analysis:** Provides a list of requests with status options.
    - **Code Analysis:** Handles approval/rejection of requests, manages blood stock, and schedules camps

4.4 NAMING CONVENTIONS

To ensure readability, consistency, and maintainability, proper naming conventions are used for functions, UI components, and variables in the Blood Donation Management System.

1) User Management

- **Function Name:** `handleUserSignUp()`
    - **Purpose:** Registers a new user by collecting and storing details like name, email, and password.
    - **Reason for Creation:** Ensures only verified users can access the platform.
- **Function Name:** `handleUserLogin()`
    - **Purpose:** Authenticates user credentials and grants access to the platform.
    - **Reason for Creation:** Provides secure login functionality.
- **Function Name:** `updateUserProfile()`
    - **Purpose:** Allows users to update their personal details such as name, phone, or email.
    - **Reason for Creation:** Enables users to keep their information accurate and up to date.

2) Donor Management

- **Function Name:** `registerDonor()`
    - **Purpose:** Registers a donor with details such as blood group, age, weight, and last donation date.
    - **Reason for Creation:** Maintains donor eligibility records.
- **Function Name:** `updateDonorAvailability()`

- **Purpose:** Updates donor status to available or unavailable after each donation.
- **Reason for Creation:** Ensures real-time donor availability tracking.

- **Function Name:** `validateDonorEligibility()`
  - **Purpose:** Validates donor based on age, weight, and donation gap.
  - **Reason for Creation:** Prevents ineligible donations.

3) Blood Request Management

- **Function Name:** `submitRequest()`
  - **Purpose:** Allows recipients or hospitals to request blood units.
  - **Reason for Creation:** Collects structured and valid request details.
- **Function Name:** `validateRequestForm()`
  - **Purpose:** Ensures that all required fields such as blood group, units, and hospital are filled correctly.
  - **Reason for Creation:** Prevents incomplete or invalid requests.
- **Function Name:** `updateRequestStatus()`
  - **Purpose:** Updates request status (Pending, Approved, Rejected, Fulfilled).
  - **Reason for Creation:** Enables admins to process and track requests efficiently.

4) Admin Dashboard & Management

- **Function Name:** `fetchAllRequests()`
  - **Purpose:** Retrieves all blood requests for admin review.
  - **Reason for Creation:** Provides centralized request monitoring.
- **Function Name:** `approveRequest()`
  - **Purpose:** Marks a blood request as approved.
  - **Reason for Creation:** Ensures timely fulfillment of requests.
- **Function Name:** `rejectRequest()`
  - **Purpose:** Marks a request as rejected with a reason.
  - **Reason for Creation:** Prevents invalid or duplicate requests.
- **Function Name:** `scheduleCamp()`
  - **Purpose:** Allows admin to create a new blood donation camp with location and date.
  - **Reason for Creation:** Organizes blood drives to increase stock.

5) Feedback Management

- **Function Name:** `submitFeedback()`
  - **Purpose:** Allows users to provide suggestions or ratings.
  - **Reason for Creation:** Improves system quality based on user experience.
- **Function Name:** `viewFeedback()`
  - **Purpose:** Allows admin to read and analyze feedback entries.
  - **Reason for Creation:** Helps in decision-making and improvements.

→ UI COMPONENTS

1. **Component:** `UserRegisterScreen`
    - **Naming Convention:** PascalCase
    - **Purpose:** UI for user registration with fields for name, email, password.
2. **Component:** `UserLoginScreen`
    - **Naming Convention:** PascalCase
    - **Purpose:** UI for user login with email and password.
3. **Component:** `DonorDashboardScreen`
    - **Naming Convention:** PascalCase
    - **Purpose:** Displays donor details and availability status.
4. **Component:** `BloodRequestScreen`
    - **Naming Convention:** PascalCase
    - **Purpose:** UI for submitting blood requests.
5. **Component:** `RequestTrackingScreen`
    - **Naming Convention:** PascalCase
    - **Purpose:** Shows request history with statuses.
6. **Component:** `AdminDashboardScreen`
    - **Naming Convention:** PascalCase
    - **Purpose:** Displays all requests and options to approve/reject.
7. **Component:** `CampManagementScreen`
    - **Naming Convention:** PascalCase

- **Purpose:** Allows admins to schedule and manage blood donation camps.

8. **Component:** `FeedbackScreen`
    - **Naming Convention:** PascalCase
    - **Purpose:** UI where users submit system feedback.

→ VARIABLES USED IN THE SYSTEM

User Variables

1. Variable: `userId`
    - **Purpose:** Stores the unique identifier for each user.
2. Variable: `userName`
    - **Purpose:** Stores the full name of the user.
3. Variable: `userEmail`
    - **Purpose:** Stores the user's email ID for login and communication.
4. Variable: `userPassword`
    - **Purpose:** Stores the encrypted password.
5. Variable: `userRole`
    - **Purpose:** Stores whether the user is a donor, recipient, or admin.

Donor Variables

1. Variable: `donorId`
    - **Purpose:** Stores the unique ID for each donor.
2. Variable: `bloodGroup`
    - **Purpose:** Stores the donor's blood group (e.g., A+, O-).
3. Variable: `availability`
    - **Purpose:** Tracks whether donor is available for donation.
4. Variable: `lastDonationDate`
    - **Purpose:** Records the date of the donor's last blood donation.

Request Variables

1. Variable: `requestId`
    - **Purpose:** Unique ID assigned to each request.
2. Variable: `requestStatus`
    - **Purpose:** Tracks request status (Pending, Approved, Fulfilled, Rejected).
3. Variable: `hospitalName`
    - **Purpose:** Stores the name of the hospital requesting blood.
4. Variable: `requestedUnits`
    - **Purpose:** Stores number of blood units required.

Admin Variables

1. Variable: `adminId`
    - **Purpose:** Stores unique identifier for each admin.
2. Variable: `adminName`
    - **Purpose:** Stores the full name of the admin.
3. Variable: `adminEmail`
    - **Purpose:** Stores admin's email for login.

Feedback Variables

1. Variable: `feedbackId`
    - **Purpose:** Unique identifier for each feedback entry.
2. Variable: `feedbackText`
    - **Purpose:** Stores the user's feedback message.
3. Variable: `feedbackRating`
    - **Purpose:** Stores user rating (1–5).

4.5 VALIDATIONS

Validations are implemented to ensure that only **accurate, secure, and meaningful data** is entered into the system. This prevents errors, enhances security, and maintains data integrity across all modules.

1. Registration Validation: validateUsername()

- **Purpose:** Ensures that the username is valid and formatted correctly.
- **Reason:** To prevent invalid or duplicate user registrations.
- **Validation Criteria:**
    - Only alphabets, numbers, and underscores allowed.
    - Length must be between **5–15 characters**.

2. Registration Validation: validateEmail()

- **Purpose:** Ensures that the provided email address follows a proper format.
- **Reason:** To prevent invalid email registrations and ensure reliable communication.
- **Validation Criteria:**
    - Must follow the format: `username@domain.com`.
    - Email must be unique in the system.

3. Registration Validation: validatePassword()

- **Purpose:** Ensures the password is strong and secure.
- **Reason:** To protect user accounts from unauthorized access.
- **Validation Criteria:**

- Minimum length of **8 characters**.
- Must include **letters and numbers**.
- Should not contain spaces.

4. Donor Validation: validateDonorEligibility()

- **Purpose:** Ensures that a donor is medically eligible to donate blood.
- **Reason:** To protect donor health and ensure safe donations.
- **Validation Criteria:**
  - Age must be between **18–60 years**.
  - Weight must be **≥ 50 kg**.
  - Minimum gap of **3 months** between donations.

5. Blood Request Validation: validateBloodRequest()

- **Purpose:** Ensures that blood requests are complete and accurate.
- **Reason:** To provide valid and meaningful requests for faster processing.
- **Validation Criteria:**
  - Blood group must be valid (A+, A-, B+, B-, AB+, AB-, O+, O-).
  - Requested units must be between **1–5**.
  - Hospital name must not be empty.

6. Login Validation: validateUserLogin()

- **Purpose:** Ensures that the user credentials match the records in the system.
- **Reason:** To allow access only to authorized users.
- **Validation Criteria:**
    - Both **email and password** must be entered.
    - Email must exist in the database.
    - Password must match the stored encrypted password.

7. Admin Validation: validateAdminLogin()

- **Purpose:** Ensures admin credentials are valid before accessing the dashboard.
- **Reason:** To protect sensitive operations like approving requests and managing camps.
- **Validation Criteria:**
    - Admin email and password must match records.
    - Access restricted only to authorized admin accounts.

8. Feedback Validation: validateFeedback()

- **Purpose:** Ensures user feedback entries are meaningful.
- **Reason:** To avoid spam or incomplete submissions.
- **Validation Criteria:**
    - Feedback message cannot be empty.
    - Rating must be between **1–5**.

5. PROGRAM LISTING

5.1 Cost Estimation

The cost estimation of the Blood Donation Management System is done using the **Function Point (FP) Method**. This method estimates the size of the software based on inputs, outputs, inquiries, files, and interfaces.

5.1.1 Domain Characteristics Table

MEASUREMENT PARAMETER	COUNT (value >= 0)	WEIGHTING FACTOR		
		Simple	Average	Complex
Number of User Input	7	○	●	○
Number of User Outputs	6	○	●	○
Number of User Inquiries	3	●	○	○
Number of Files	6	○	●	○
Number of External Interfaces	1	●	○	○

5.1.2 Complexity Adjustment Table

Complexity Adjustment Table

ITEM	COMPLEXITY ADJUSTMENT QUESTIONS	SCALE No Influence 0	1	2	3	4	Essential 5
1	Does the system require reliable backup and recovery?	○	○	○	●	○	○
2	Are data communications required?	○	○	●	○	○	○
3	Are there distributed processing functions?	●	○	○	○	○	○
4	Is performance critical?	○	○	○	●	○	○
5	Will the system run in an existing, heavily utilized operational environment?	○	●	○	○	○	○
6	Does the system require on-line data entry?	○	○	○	○	●	○
7	Does the on-line data entry require the input transaction to be built over multiple screens or operations?	○	●	○	○	○	○
8	Are the master files updated on-line?	○	○	○	○	●	○
9	Are the inputs, outputs, files or inquiries complex?	○	○	●	○	○	○
10	Is the internal processing complex?	○	○	●	○	○	○
11	Is the code to be designed reusable?	○	●	○	○	○	○
12	Are conversion and installation included in the design?	●	○	○	○	○	○
13	Is the system designed for multiple installations in different organizations?	○	●	○	○	○	○
14	Is the application designed to facilitate change and ease of use by the user?	○	○	○	●	○	○

Function Point (FP) Calculation

RESULT	
PROJECT FUNCTION POINTS	121.44000000000001

5.2 Schema Design

Table	Action	Rows	Type	Collation	Size	Overhead
blood_camps	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	16.0 KiB	-
camp_registrations	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	48.0 KiB	-
donations	Browse Structure Search Insert Empty Drop	4	InnoDB	utf8mb4_general_ci	32.0 KiB	-
donors	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	32.0 KiB	-
feedback	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	32.0 KiB	-
health_tips	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	16.0 KiB	-
supplies	Browse Structure Search Insert Empty Drop	8	InnoDB	utf8mb4_general_ci	32.0 KiB	-
users	Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	32.0 KiB	-

Blood_comp

	id	name	location	camp_date	camp_time	notes	created_at
Edit Copy Delete	1	Jiya Nandu	grant road	2025-10-02	10:00:00	to donate the blood to cancer people	2025-08-30 11:06:26

Camp_registration

id	camp_id	user_id	units	registered_at
1	1	3	4	2025-08-30 11:07:01

Donation

id	user_id	blood_type	units	status	notes	created_at
1	1	O+	10	approved		2025-08-16 14:15:19
2	3	O+	20	approved		2025-08-23 14:08:01
3	3	A+	1	approved		2025-08-25 17:32:17
4	3	O+	22	approved		2025-08-26 14:23:49

Donors

id	user_id	name	blood_type	units	phone	city	created_at
1	1	JIYA CHETAN NANDU	O+	10		NULL	2025-08-16 14:15:40
2	3	veer nandu	O+	20		NULL	2025-08-23 14:07:33

Health_tips

id	title	content	created_at
1	Stay Hydrated Before Donation	Drink plenty of water before donating blood. Prope...	2025-08-30 10:46:17

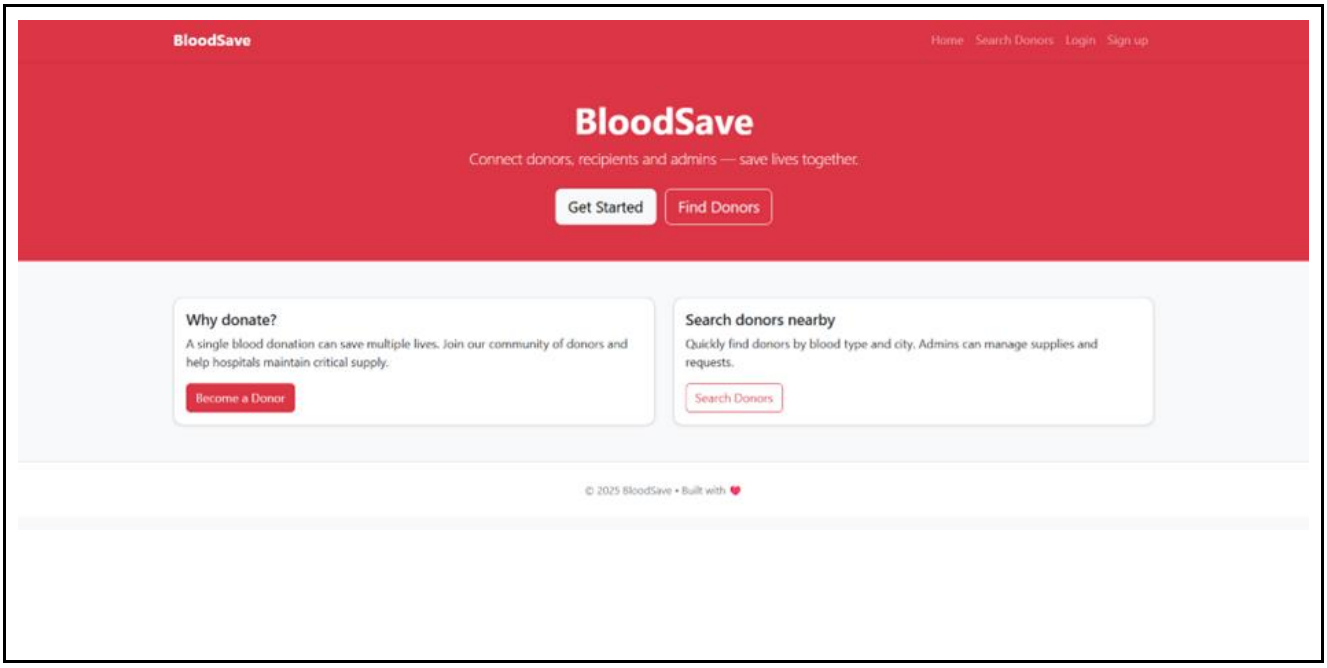
Supplies

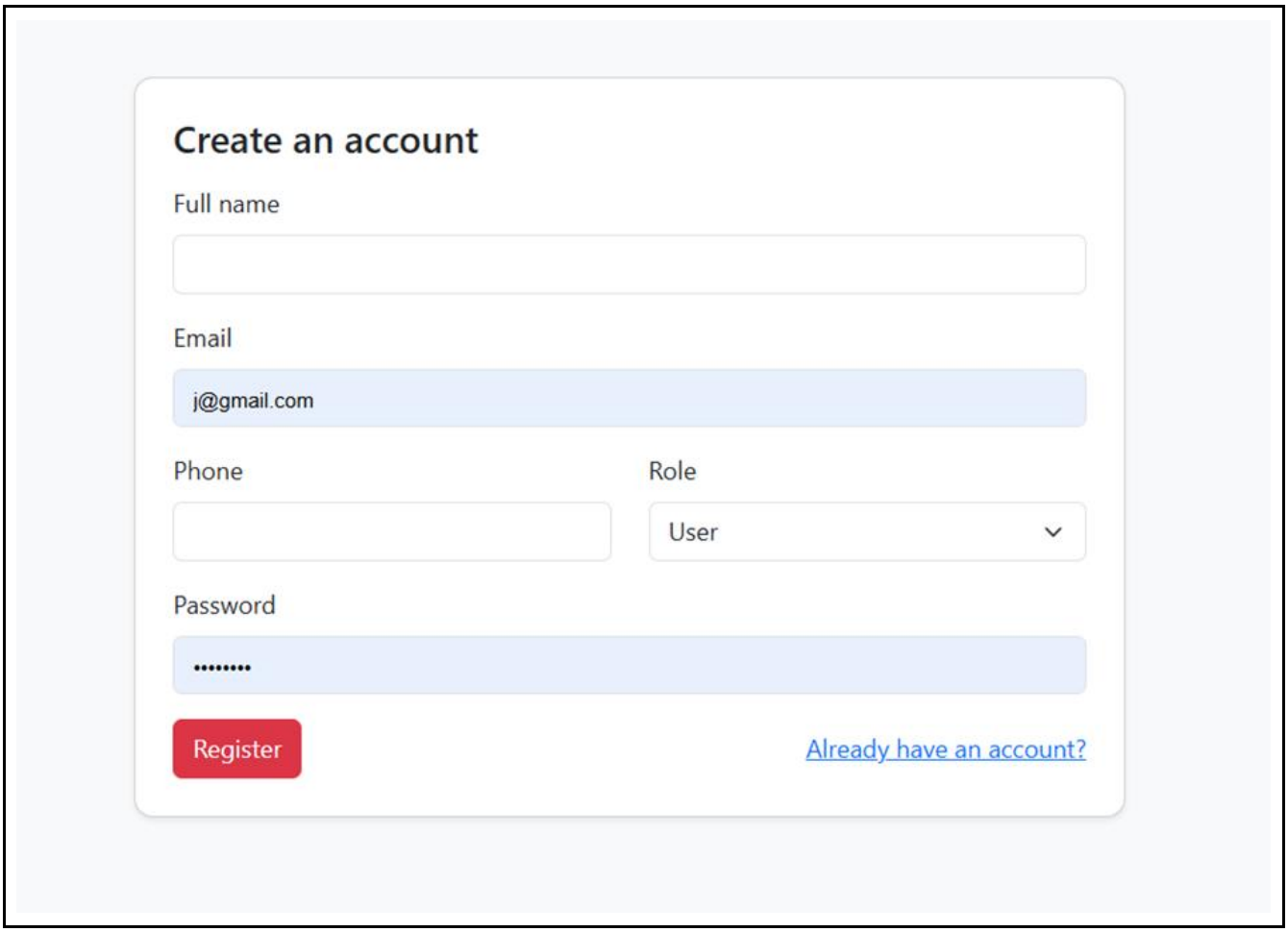
id	blood_type	units	last_updated
1	A+	21	2025-08-28 11:09:14
2	A-	5	2025-08-16 14:03:40
3	B+	12	2025-08-25 15:45:13
4	B-	4	2025-08-16 14:03:40
5	O+	4	2025-08-26 16:51:50
6	O-	7	2025-08-16 14:03:40
7	AB+	6	2025-08-16 14:03:40
8	AB-	1	2025-08-17 17:44:53

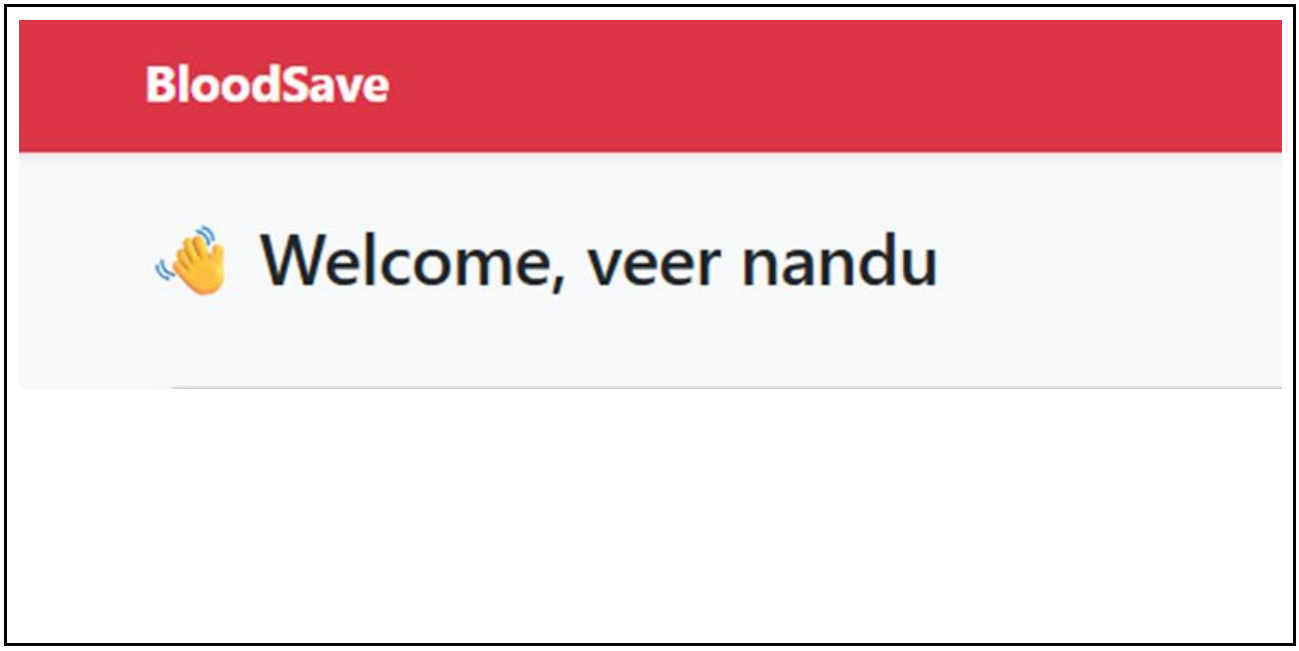
Users

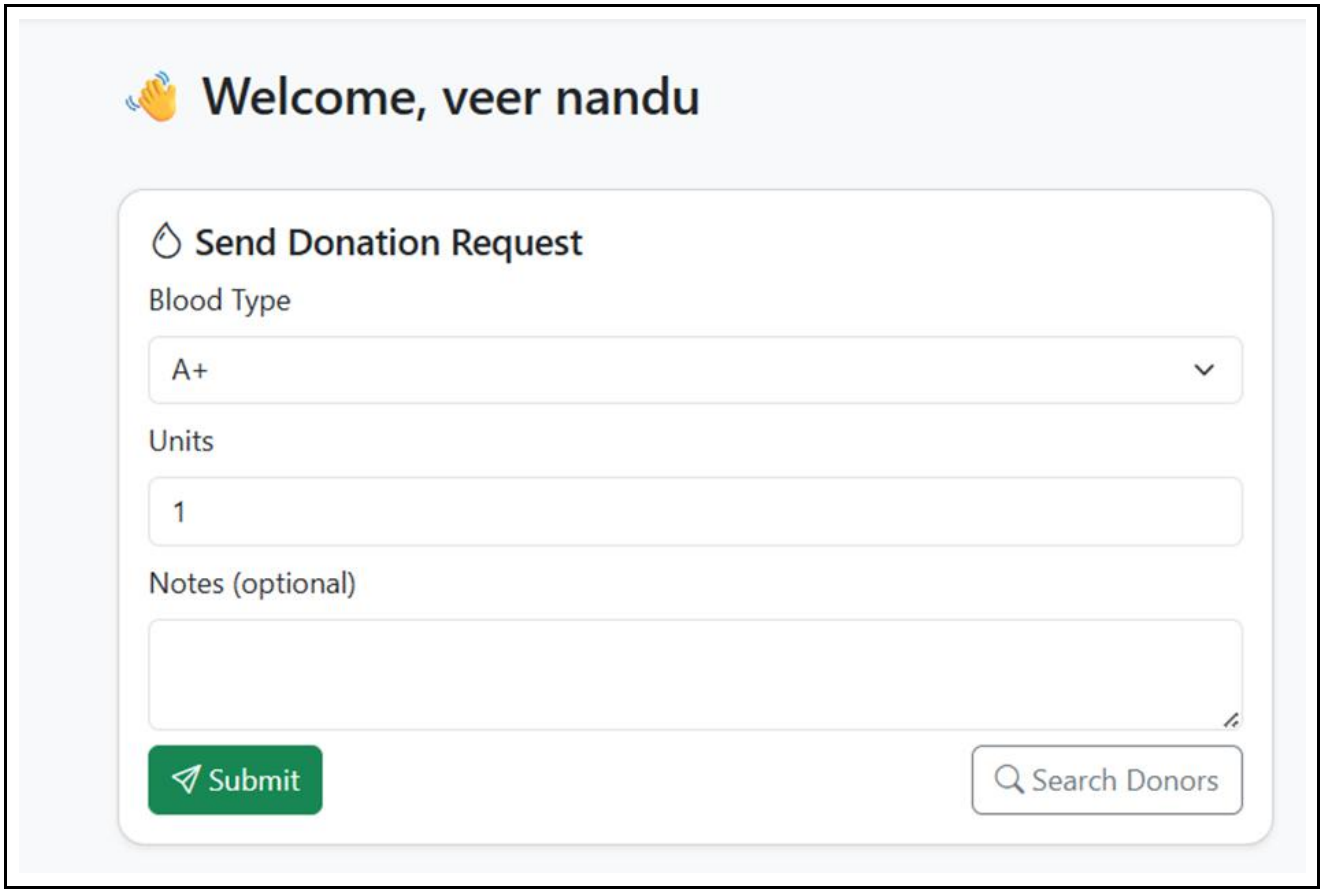
id	name	email	password	role	phone	created_at
1	JIYA CHETAN NANDU	jiyanandu28@gmail.com	$2y$10$J/6OX1w54cmXN1EWteJc8uqNzPNKdJRS4aktR/bETVE...	user	08291882380	2025-08-16 14:14:52
2	blod	blood@gmail.com	$2y$10$25tSL4rHFwSFNLqxLgVd6uKc96fhKCcDkQES04NGcGs...	admin	08291882380	2025-08-16 14:17:20
3	veer nandu	v@gmail.com	$2y$10$qdllk3KOgi75oVl1CQLRteEe78BaFJF6i8iLqhNs02D...	user	1234568910	2025-08-23 14:05:46
4	blood1	b1@gmail.com	$2y$10$iMZ3MMbudbX2nLUasNkqfeUAPIAKdOlAefOafx1NiST...	admin	1234568910	2025-08-25 15:37:22
5	blood2	b2@gmail.com	$2y$10$OHtW1dmt6OclaK0W9I.8Xe0uRJOQcuHR8iwCs18GDCR...	admin	9833459125	2025-08-26 13:55:26

5.3 User Manual With Screenshots






BloodSave
Welcome, veer nandu


Welcome, veer nandu
Send Donation Request
Blood Type
A+
Units
1
Notes (optional)
Submit
Search Donors

Register as Donor

You are registered as donor: **O+** (20 unit(s))

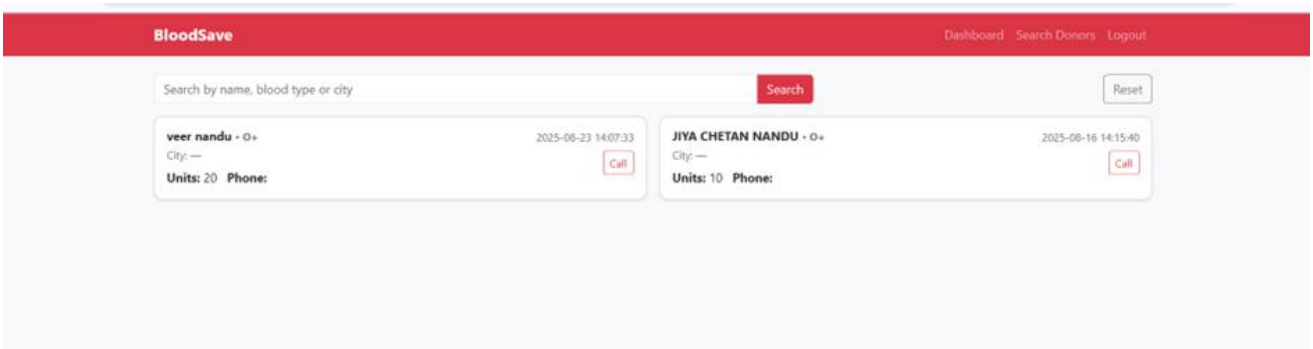
Remove Donor Record

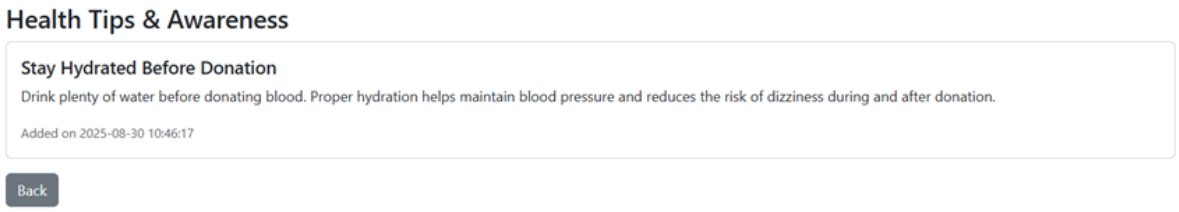
Your Donation Requests

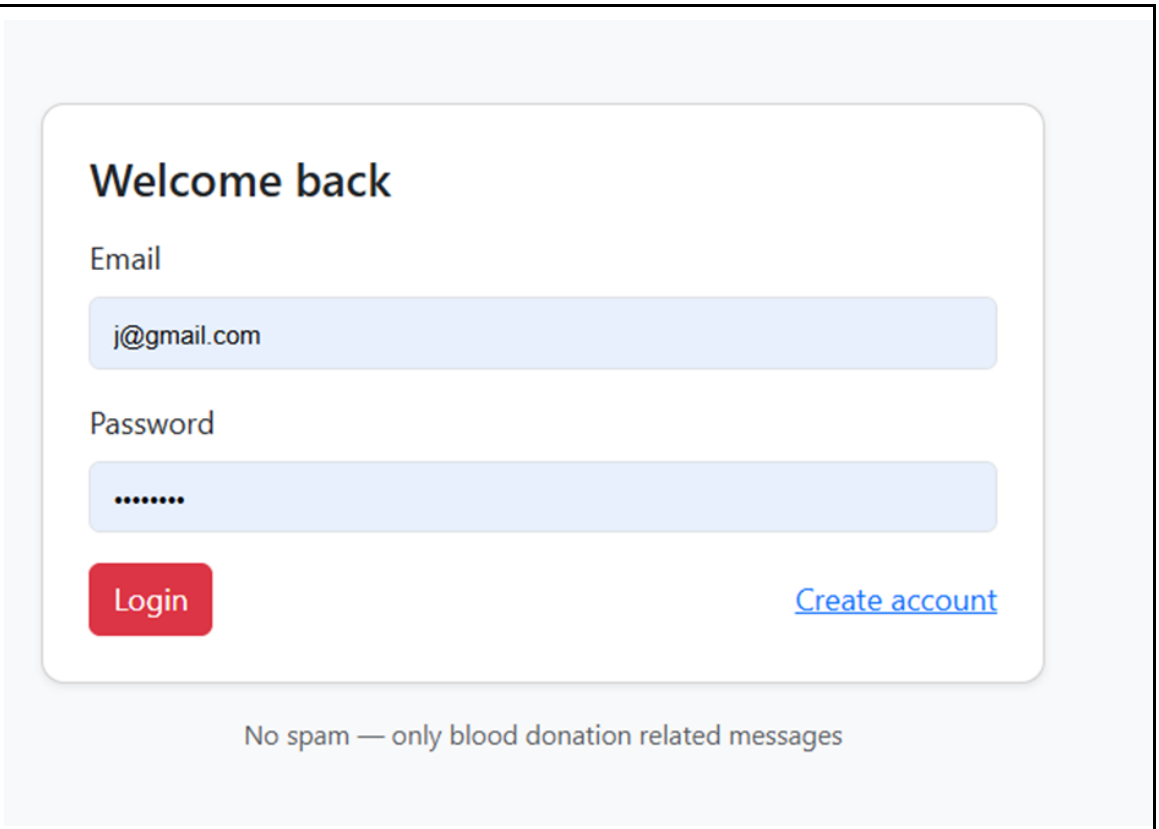
O+ • 22 unit(s) Approved 2025-08-26 14:23:49

A+ • 1 unit(s) Approved 2025-08-25 17:32:17

O+ • 20 unit(s) Approved 2025-08-23 14:08:01


BloodSave
Dashboard
Search Donors
Logout
Search by name, blood type or city
Search
Reset
veer nandu • O+
2025-08-23 14:07:33
City: —
Units: 20 Phone:
Call
JIYA CHETAN NANDU • O+
2025-08-16 14:15:40
City: —
Units: 10 Phone:
Call


Health Tips & Awareness
Stay Hydrated Before Donation
Drink plenty of water before donating blood. Proper hydration helps maintain blood pressure and reduces the risk of dizziness during and after donation.
Added on 2025-08-30 10:46:17
Back


Welcome back
Email
j@gmail.com
Password
••••••••
Login
Create account
No spam — only blood donation related messages

BloodSave

Admin Search Donors Logout

Admin Dashboard

Manage Requests (0)

Total Users 5

Total Donors 2

Pending Requests 0

Manage Supply Edit Supply

Manage Health Tips

Go

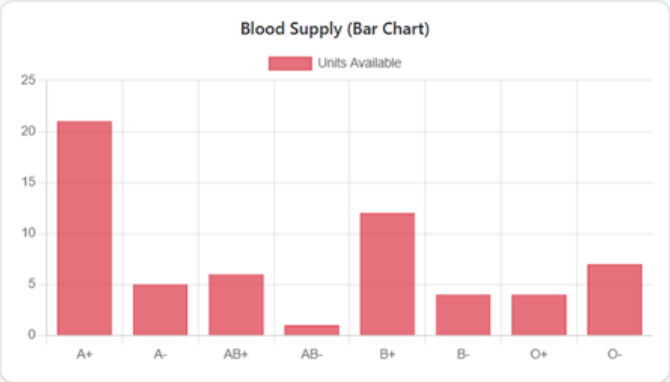
Manage Blood Donation Camps

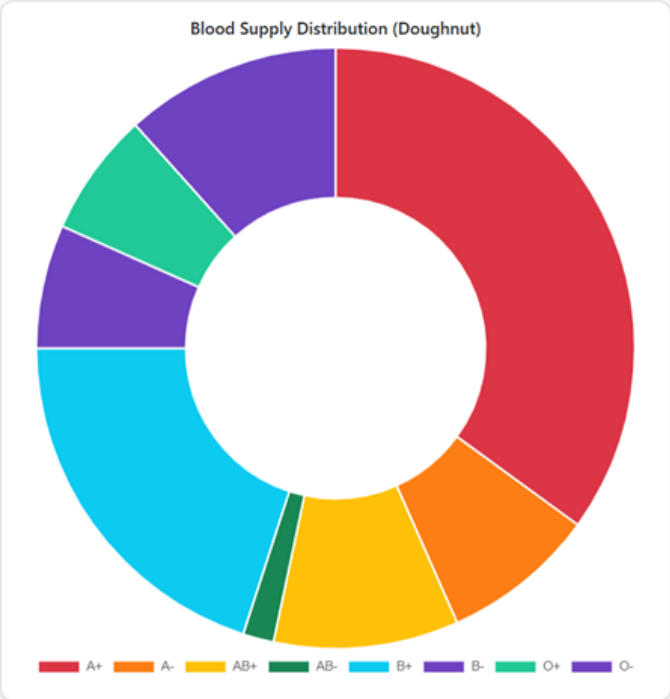
Go

Current Supply Levels

Search blood type...

Blood Type	Units	Progress	Last Updated
A+	21		2025-08-28 11:09:14
A-	5		2025-08-16 14:03:40
AB+	6		2025-08-16 14:03:40
AB-	1		2025-08-17 17:44:53
B+	12		2025-08-25 15:45:13
B-	4		2025-08-16 14:03:40
O+	4		2025-08-26 16:51:50
O-	7		2025-08-16 14:03:40





Supply Management

Blood Type	Units	Update
A+	21	21 Save
A-	5	5 Save
AB+	6	6 Save
AB-	1	1 Save
B+	12	12 Save
B-	4	4 Save
O+	4	4 Save
O-	7	7 Save
New type (e.g. X+)	0	Add

Back

Manage Donation Requests

User	Type	Units	Notes	Status	When	Action
veer nandu	O+	22		Approved	2025-08-26 14:23:49	No actions
veer nandu	A+	1		Approved	2025-08-25 17:32:17	No actions
veer nandu	O+	20		Approved	2025-08-23 14:08:01	No actions
JIYA CHETAN NANDU	O+	10		Approved	2025-08-16 14:15:19	No actions

Back to Dashboard

Manage Health Tips

Tip Title

Tip Content

Add Tip

Stay Hydrated Before Donation

Update

Drink plenty of water before donating blood. Proper hydration helps maintain blood pressure and reduces the risk of dizziness during and after donation.

Delete

Back

Manage Blood Donation Camps

Create New Camp

Camp Name

Location

dd-mm-yyyy

--:--

Notes

Create Camp

Existing Camps

Name	Location	Date	Time	Registrations
Jiya Nandu	grant road	2025-10-02	10:00:00	1

5.4 Test Cases Design

Test Case ID	Test Description	Test Input	Expected Result	Actual Result	Pass/Fail
TC01	User Registration with valid details	Name, Email, Password, City, Blood Group	User account created, confirmation message shown	As Expected	Pass
TC02	User Login with wrong password	Valid Email + Wrong Password	Error message “Invalid Credentials”	As Expected	Pass
TC03	Send Blood Donation Request	Blood Group = O+, Units = 2	Request stored in DB, admin notified	Not Working	Fail
TC04	Register as Donor	User provides Blood Group, Units available	Donor registered successfully and added to donor list	As Expected	Pass
TC05	Search Donors by City	City = "Pune"	System displays all donors from Pune	As Expected	Pass
TC06	View Health Tips	User clicks Health Tips module	List of health tips is displayed	As Expected	Pass
TC07	Admin approves a request	Pending Request ID	Status updated to Approved, notification sent to user	As Expected	Pass
TC08	Manage Blood Supply (Update Units)	Admin updates A+ units from 10 → 15	Database updated, supply shows A+ = 15 units	Not Updated	Fail
TC09	Submit Feedback with rating	Rating = 4, Comment = "Good service"	Feedback stored successfully, confirmation message shown	As Expected	Pass

6. CONCLUSION

6.1 Conclusion

The Blood Donation Management System provides a digital platform to streamline the process of blood donation and request management. It connects donors, recipients, hospitals, and administrators in a transparent and efficient way. Donors can easily register and track their donations, while recipients and hospitals can raise requests in emergencies without delays. The system ensures proper management of blood stock, organizes donation camps, and encourages voluntary blood donation. Overall, the project improves coordination, reduces manual errors, and promotes a reliable and accessible healthcare support system.

6.2 Limitations

1. Internet Dependency – Users in rural or remote areas may face difficulties accessing the system due to limited internet connectivity.

2. Verification Process – Donor verification (medical fitness, eligibility) still requires manual confirmation by healthcare staff.

3. Hospital Dependency – The actual availability and delivery of blood depend on hospitals and blood banks, which may cause delays in emergencies.

4. Limited Awareness – Some users may not be aware of the system, which can reduce its reach and effectiveness.

6.3 Future Scope of the Project

1. Mobile Application – A dedicated Android and iOS app for easy accessibility and quick emergency requests.

2. Multilingual Support – Adding multiple languages to make the system user-friendly for a wider audience.

3. Integration with Hospitals & Blood Banks – Direct integration with hospital systems to update blood stock in real time.

4. SMS / Email Notifications – Automatic alerts to donors and recipients about requests, approvals, and camp schedules.

5. Donation Camp Expansion – Organizing and promoting blood donation drives at schools, colleges, and public events.

7. BIBLIOGRAPHY

1. National Blood Transfusion Council (NBTC), Ministry of Health & Family Welfare, Government of India
   https://nbtc.naco.gov.in

2. National Blood Transfusion Services – World Health Organization (WHO)
   https://www.who.int/health-topics/blood-products

3. Indian Red Cross Society – Blood Donation
   https://indianredcross.org/ircs/program/blood-banks

4. NACO – National AIDS Control Organization, Blood Safety Division
   https://www.naco.gov.in

5. e-RaktKosh – Centralized Blood Bank Management System (MoHFW, Govt. of India)
   https://www.eraktkosh.in

6. Books / Reference Materials:

   - Rajkamal, *Software Engineering*, McGraw Hill Education.

   - Pressman, R.S., *Software Engineering: A Practitioner's Approach*, McGraw Hill.

7. Research Papers / Articles:

   - "Improving Blood Bank Management Systems using Digital Platforms" – International Journal of Computer Applications.

   - "ICT-enabled Blood Donation Systems in India" – Journal of Health Informatics in Developing Countries.