**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

**“JNANA SANGAMA”, BELAGAVI - 570018, KARNATAKA**

****

**A Project Report**

**On**

**"Blood Donation Management System: Reducing the Gap Between Donors and Recipients with Compassion and Efficiency"**

***In the partial fulfilment of the requirement for the award of degree***

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

***Submitted by***

|  |  |
| --- | --- |
| **PAVAN CHANDRAPPA HOTTIGOUDRA** | **(4VZ22CS019)** |
| **SHIVANI VEERESH LINGADAHALLI** | **(4VZ22CS026)** |
| **SOURABHA SHRENIKRAJA HALLI** | **(4VZ22CS029)** |
| **VIKAS** | **(4VZ22CS031)** |

***Under the guidance of***

**Dr. G. F. Ali Ahammed**

**Program Coordinator,**

**Dept. of Computer Science and Engineering,**

**VTU Regional Center, Mysore.**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

**Centre for Post Graduate Studies, Sathagalli, Mysuru – 570029.**

**2024 – 2025**

**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

**“JNANA SANGAMA”, BELAGAVI - 570018, KARNATAKA**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**Centre for Post Graduate Studies, Sathagalli, Mysuru – 570029.**

**2024 – 2025**

**CERTIFICATE**

This is to certify that the Project work entitled **"Blood Donation Management System: Reducing the Gap Between Donors and Recipients with Compassion and Efficiency"** is a bonafied work carried out by Pavan Chandrappa Hottigoudra, Shivani Veeresh Lingadahalli, Sourabha Shrenikraj Halli, Vikas bearing USN 4VZ22CS019, 4VZ22CS026, 4VZ22CS029, 4VZ22CS031 at Department of Computer Science and Engineering, Visvesvaraya Technological University, Centre for Post Graduate Studies, Mysuru in partial fulfilment for the award of Bachelor of Technology in Computer Science and Engineering, Visvesvaraya Technological University, Belagavi during the academic year 2024-2025. It is certified that all the corrections/suggestions indicated during Internal Assessment have been incorporated in the report. The Mini Project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the Bachelor of Technology degree.

**Signature of the Guide**

Dr. G. F. ALI AHAMMED

Program Coordinator,

Dept. of CS&E,

VTU, CPGS Mysuru - 570029

**Signature of Program Coordinator**

Dr. G. F. ALI AHAMMED

Program Coordinator,

Dept. of CS&E,

VTU, CPGS Mysuru - 570029

**ACKNOWLEDGEMENT**

The satisfaction and euphoria that accompany the completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement ground our efforts with success.

We express our sincere thanks to **Dr. G F ALI AHAMMED, Program Coordinator, PG Studies, Department of Computer Science & Engineering, VTU Regional Centre, Mysuru** for his kind help and constant encouragement and for providing us necessary facilities for carrying out this work successfully.

In particular, We would like to take this opportunity to express our Honor, Respect, Deepest Gratitude and Genuine Regards to our guide, **Dr. G F ALI AHAMMED**, **Program Coordinator**, Department of Computer Science and Engineering, VTU Regional Centre, for giving us all guidance required for our project apart from being a constant source of inspiration and motivation.

We owe our special thanks to **Our Parents** for their moral support and warm wishes and finally We would like to express appreciation to all **Our Friends** for their unconditional support which helped us to complete this work successfully.

|  |  |
| --- | --- |
| Pavan Chandrappa Hottigoudra | 4VZ22CS019 |
| Shivani Veeresh Lingadahalli | 4VZ22CS026 |
| Sourabha Shrenikraja Halli | 4VZ22CS029 |
| Vikas | 4VZ22CS031 |

**DECLARATION**

We Pavan Chandrappa Hottigoudra, Shivani Veeresh Lingadahalli, Sourabha Shrenikraja Halli, Vikas bearing USN 4VZ22CS019, 4VZ22CS026, 4VZ22CS029, 4VZ22CS031 hereby declare that this project work entitled **"Blood Donation Management System: Reducing the Gap Between Donors and Recipients with Compassion and Efficiency"**, is a bonafide work carried out by me under the guidance and supervision of Dr. G. F. Ali Ahammed, Program Coordinator, Department of Computer Science and Engineering, VTU Center for PG Studies, Mysore. This project work is submitted to Visvesvaraya Technological University, Belagavi in partial fulfilment of the requirements for the award to degree of Bachelor of Technology in Computer Science and Engineering during the academic year 2024-2025.

|  |  |
| --- | --- |
| Pavan Chandrappa Hottigoudra | 4VZ22CS019 |
| Shivani Veeresh Lingadahalli | 4VZ22CS026 |
| Sourabha Shrenikraja Halli | 4VZ22CS029 |
| Vikas | 4VZ22CS031 |

**ABSTRACT**

In the evolving healthcare landscape, managing blood donation processes has been significantly enhanced through technologies such as web pages, computer applications, and database servers. This analysis led to the development of the Web-Based Blood Donation Management System, a computer application designed to streamline the management of donor information, donation schedules, and blood bank inventory. The system offers a user-friendly interface that enables blood donors to register, schedule, and track their donation appointments online. Donors can confirm their own donation appointments, view donation history, and check the availability of donation camps or blood banks.

Additionally, healthcare providers and blood bank staff can manage donation requests, monitor donor availability, and update donor status in real time. This online system reduces the administrative burden on healthcare staff, enhances coordination, and optimizes blood bank resources. A significant issue in blood donation operations is inefficient management, often leading to overcrowding, uncoordinated schedules, and delays in collection. These challenges can result in missed donation opportunities and dissatisfaction among donors. The proposed system addresses these issues by improving appointment scheduling, donor information management, and communication, thereby reducing waiting times, enhancing the donor experience, and increasing the overall efficiency of blood donation processes.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Title** | **Page No.** |
| 1 | Introduction | 1-3 |
| 2 | System Analysis   1. Existing System 2. Proposed System | 4-6 |
| 3 | System Requirements   1. Hardware Specification 2. Software Specification 3. Functionality Requirements 4. Non-Functionality Requirements | 7-8 |
| 4 | System Architecture   1. ER Diagram | 9-11 |
| 5 | System Design   1. Database Table Structure | 12-14 |
| 6 | Implementation   1. Packages 2. Functions | 15-18 |
| 7 | Testing | 19-21 |
| 8 | Results | 22-29 |
| 9 | Conclusion | 30 |
| 10 | References | 31 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Title** | **Page No.** |
| Fig.1 | Home page | 20 |
| Fig.2 | Blood bank registration | 21 |
| Fig.3 | Donor registration | 21 |
| Fig.4 | Reset password | 21 |
| Fig.5 | Donor dashboard | 22 |
| Fig.6 | Admin details | 23 |
| Fig.7 | Donars list | 23 |
| Fig.8 | Blood bank list | 23 |
| Fig.9 | Ongoing requests list | 24 |
| Fig.10 | Fulfilled requests list | 24 |
| Fig.11 | Blood bank dashboard | 25 |
| Fig.12 | Blood request form | 26 |
| Fig.13 | Request Message to donor | 27 |
| Fig.14 | Certificate | 27 |

**List of Tables**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Title** | **Page No.** |
| Table 5.1 | Donar registration table | 10 |
| Table 5.2 | Blood request table | 11 |
| Table 5.3 | Blood bank registration table | 12 |
| Table 5.4 | Admin table | 12 |
| Table 7.1 | User authentication test case | 17 |
| Table 7.2 | Donar dashboard test case | 17 |
| Table 7.3 | Blood bank dashboard test case | 18 |
| Table 7.4 | Blood request management test case | 18 |
| Table 7.5 | Donar registration test case | 19 |
| Table 7.6 | Blood bank registration test case | 19 |
| Table 7.7 | Password reset with SMS OTP test case | 19 |

**CHAPTER 1**

**INTRODUCTION**

Developing an efficient Blood System Management application significantly enhances the management and allocation of blood supplies in healthcare settings, ensuring timely access to blood for patients in need. This system facilitates better management of blood donations, tracking of blood inventory, and scheduling of blood transfusions, thereby improving overall healthcare outcomes. By automating key processes and streamlining administrative tasks, the system reduces human error and manual effort, enhancing operational efficiency for healthcare providers and ensuring that blood is available when needed.

A Database Management System (DBMS) is essential for the smooth operation of a blood system management application. It acts as a structured repository for vital data, such as donor information, blood inventory levels, donation histories, and patient requirements. The database ensures accurate, real-time data retrieval and manipulation, enabling quick decision-making and efficient resource management. This centralized data management system ensures the accuracy and up-to-date status of all records, contributing to the overall reliability and efficiency of blood supply management.

In this system, donors, hospitals, and administrative personnel interact through a secure and user-friendly interface. Donors can register their details, schedule appointments for blood donation, and track their donation history. Hospitals and blood banks can manage blood inventories, monitor the stock levels of different blood types, and allocate supplies according to patient requirements. Administrators oversee and manage all records and transactions to ensure smooth operations and compliance with medical standards.

The system's design aims to simplify the entire blood donation and management process by utilizing modern technologies such as the Express framework and MySQL for backend development, and React for frontend development. Express, a minimalist web application framework for Node.js, powers the server-side logic of the application, while MySQL, a relational database management system (RDBMS), handles the secure storage and management of data. On the frontend, React is used to create a dynamic and responsive user interface for donors, healthcare providers, and administrators. HTML and CSS are used in conjunction with React to structure and style the pages, ensuring a user-friendly and visually appealing experience.

This system offers a streamlined approach for managing blood donations and blood inventory while ensuring security, reducing errors, and enhancing overall time efficiency. By providing real-time updates on blood supply levels and facilitating efficient scheduling of blood donations, it helps prevent shortages, reduce delays, and optimize resource allocation. With these features, the system is designed to improve access to blood supplies, reduce waste, and ultimately save lives.

In addition to addressing the administrative challenges associated with blood supply management, the proposed system also focuses on overcoming the limitations of traditional offline methods. By moving to an online, database-driven solution, it offers greater security, convenience, and efficiency, contributing to better healthcare delivery.

**AIM:**

The aim of this web-based mini project is to design and develop a web application that utilizes a DBMS to streamline and manage the blood donation process. This application allows donors to register and maintain their profiles, manage blood donation schedules, and track donation history. It also enables healthcare providers to manage donor profiles, availability, and blood inventory efficiently. The system offers features such as viewing of upcoming donations, and cancellation of appointments. Additionally, it includes a search function for donors to find nearby donation centers or schedule donation appointments based on availability.

The use of a DBMS in this web-based application ensures the efficient storage, retrieval, and management of critical data related to donors, donations, and inventory. The web-based nature of the system ensures accessibility, ease of use, and real-time updates, thereby improving overall management efficiency. The ultimate goal is to enhance the blood donation process, reduce manual errors, and increase the convenience and satisfaction of donors and healthcare providers.

**MOTIVATION:**

The motivation behind developing a web-based blood donation management system is to address the common challenges of managing blood donation schedules, tracking donor information, and optimizing blood inventory. By leveraging a DBMS, the goal is to create a solution that enhances donor satisfaction, improves the management of blood supplies, and simplifies coordination between donors, healthcare providers, and blood banks. This system aims to reduce manual errors, streamline the donation process, and ensure timely availability of blood, ultimately improving efficiency and effectiveness in the blood donation process.

**PROBLEM STATEMENT:**

In modern healthcare settings, efficient management of blood donations is crucial for ensuring a constant and adequate supply of blood. However, traditional blood donation management systems often face significant challenges, including poor tracking of donor schedules, inefficient management of blood inventory, and delays in processing donations. These issues can lead to shortages of blood, delays in transfusions, and increased operational inefficiencies. The problem addressed by this mini-project is the inefficiency and potential delays in the blood donation process experienced by both donors and healthcare providers. Current systems may struggle with issues such as donor availability conflicts, inaccurate tracking of blood supply, poor communication between blood banks and healthcare providers, and lack of real-time updates on blood inventory levels.

**OBJECTIVES:**

Creating interfaces for donors, healthcare providers, and administrators: This refers to the design and development of user interfaces tailored to the needs of different stakeholders in the blood donation process, such as donors who wish to schedule donation appointments, healthcare providers who manage blood inventory and donor information, and administrators who oversee the system's operations.

Storing data from the administrator for the donors: This refers to the process of securely storing and managing donor information, including personal details, donation history, and availability. Administrators can input and manage this data, ensuring that records are up-to-date and accurate for effective management of blood donations.

Allow healthcare providers (blood banks) to approve or cancel donation requests: This refers to giving healthcare providers or blood banks the authority to manage and approve or cancel blood donation appointments. They can review donor requests, confirm donation times based on blood bank needs, or reschedule appointments as necessary to optimize donation schedules and blood inventory.

**CHAPTER 2**

**SYSTEM ANALYSIS**

1. **Existing System**

The current blood donation management system predominantly relies on manual or semi-automated methods, which face various challenges in managing donors, blood requests, and communication.

* + **Paper-Based Records :** Blood banks and hospitals depend on manual paperwork to maintain records of donors and blood requests, which is prone to errors and difficult to maintain.
  + **Inefficient Process :** Searching for suitable donors during emergencies becomes a time-consuming and labor-intensive task.
  + **No Real-Time Updates :** Donor availability and blood stock levels are not tracked or updated dynamically, leading to delays in meeting urgent requirements.
  + **Communication Challenges :** Reaching out to donors during emergencies involves manual phone calls, making it an inefficient process.
  + **Limited Accessibility :** Individuals seeking to register as donors or request blood must visit the blood bank in person, adding to the inconvenience.
  + **Use of Basic Tools :** Some organizations use tools like Excel for record-keeping, which is prone to data loss and lacks centralized access.
  + **No Proximity-Based Matching :** Existing systems fail to leverage location data to prioritize donors near the requester, resulting in inefficiencies.
  + **Minimal Integration of Technology :** SMS or email notifications for communication with donors are not integrated into the system.
  + **Outdated Records :** Donor data is often not updated, making it challenging to identify eligible donors promptly.

1. **Proposed System**

To address the shortcomings of the existing system, the proposed Blood Donation Management System leverages modern technology to enhance efficiency, accessibility, and scalability.

**Donor Registration and Management :**

* + - An intuitive online portal for blood donors to register and maintain their profiles.
    - Detailed profiles capture blood type, availability status, and location for better donor management.

**Comprehensive Blood Bank Dashboard :**

* + - Real-time display of donor availability and active blood requests.
    - Analytics and visualizations powered by Chart.js for inventory management and request tracking.

**Location-Based Donor Matching :**

* + - Identifies donors based on their proximity to the blood request location (city, taluk, district, or state).
    - Prioritizes donors within a 10km radius for urgent situations, reducing response time.

**Blood Request Management :**

* + - Blood banks can create and manage requests with details such as urgency level, blood group, and quantity needed.
    - Available donors are displayed based on matching criteria, streamlining the process.

**Integrated SMS Notifications :**

* + - Twilio integration ensures automated SMS notifications to eligible donors during emergencies.
    - Facilitates immediate communication, improving response times.

**Dynamic Updates and Availability Toggle :**

* + - Real-time updates for donor availability and blood stock levels.
    - Donors can toggle their availability to inform the system of their current status.

**Role-Based Dashboards :**

* + - Donor Dashboard : Displays donation history, upcoming events, and an availability toggle.
    - Blood Bank Dashboard : Provides a comprehensive view of blood requests, donor data, and inventory levels.

**Secure and Scalable System :**

* + - Developed with React (frontend), Node.js (backend), and MySQL (database) for robustness and scalability.
    - Implements secure login and session-based authentication for data protection.

**Advanced Filters and Sorting Options :**

* + - Filters donors by blood group, location, and availability for better search results.
    - Allows prioritization of donors based on the urgency of requests.

**Benefits of the Proposed System**

* + - Enhanced Efficiency : Automates donor matching and notification processes, reducing response times.
    - Improved Accessibility : Provides an online platform for donors and blood banks to interact seamlessly.
    - Real-Time Decision Making : Ensures accurate and up-to-date information for informed decision-making.
    - Scalability and Security : Designed to accommodate a growing user base while maintaining data security and reliability.

**CHAPTER 3**

**SYSTEM REQUIREMENT**

The Blood Donor Management System requires a centralized database, such as MySQL or SQLite, to store donor and recipient information securely. The system will run on a web-based or standalone platform, requiring a web server (e.g., Apache) for hosting and client-side technologies like HTML, CSS, and JavaScript for the user interface. A backend framework such as PHP, Python, or Java will handle data processing and system logic. The application must be compatible with desktop and mobile devices to ensure accessibility. It should include essential hardware, such as a server with sufficient storage and reliable internet connectivity. The system also requires secure authentication mechanisms to protect user data.

1. **Hardware Specification**

Processor: Intel core 5

RAM: 512MB or more

Operating system: Windows 11

1. **Software Specification**

Front end: HTML,CSS, Javascript, ReactJS.

Back end: ExpressJS, MySql.

Server: Xampp

Code editor: VS code

1. **Functionality Requirements**
2. **Data Management:**

* Allow users to create, update, retrieve, and delete (CRUD) records for agro-related data such as crops, farming techniques, and market prices.
* Enable efficient storage and retrieval of data using MySQL as the backend database.

1. **User Interface:**

* Provide a web-based user interface designed with HTML, CSS, and JavaScript for easy interaction with the system.
* Support user-friendly navigation for accessing different modules like crop data, reports, or analysis.

1. **Data Security:**

* Ensure secure data transactions through encrypted connections.
* Implement user authentication mechanisms to protect access to sensitive data.

1. **Compatibility:**

* Ensure the system works with standard input devices like RS/32 or USB keyboards and compatible mice.

1. **Server Functionality:**

* Use the XAMPP server to manage the web server and database connection seamlessly.
* Support concurrent user access without performance degradation.

1. **Code Editing and Maintenance:**

* Allow ease of development and code maintenance using VS Code as the primary code editor.

1. **Non-Functionality Requirements**
2. **Performance:**

* The system should operate efficiently with a minimum of 512MB RAM, ensuring responsiveness even with basic hardware configurations.
* Handle up to a specific number of simultaneous users without significant lag.

1. **Reliability:**

* The system must be reliable with minimal downtime during regular operations or maintenance.
* Ensure consistent data integrity, even in case of unexpected server crashes or restarts.

1. **Usability:**

* The interface must be intuitive and accessible to users with minimal training or technical knowledge.
* Provide responsive design for compatibility across different screen sizes.

1. **Scalability:**

* The system should support future upgrades to handle larger datasets or more complex queries.
* Ensure ease of scaling the database and server configuration as the user base grows.

1. **Security:**

* Protect against common vulnerabilities like SQL injection and cross-site scripting (XSS).
* Ensure regular updates to the server and codebase to address potential security risks.

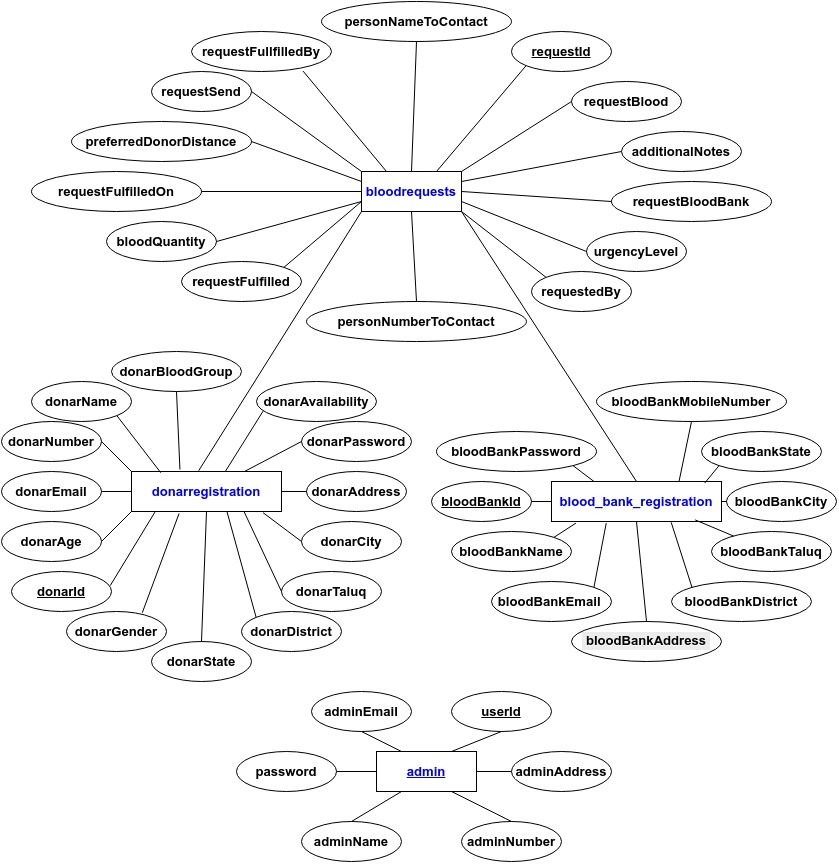
1. **Maintainability:**

* The codebase should be modular and well-documented for ease of debugging and future enhancements.
* Allow seamless integration of new features without disrupting existing functionality.

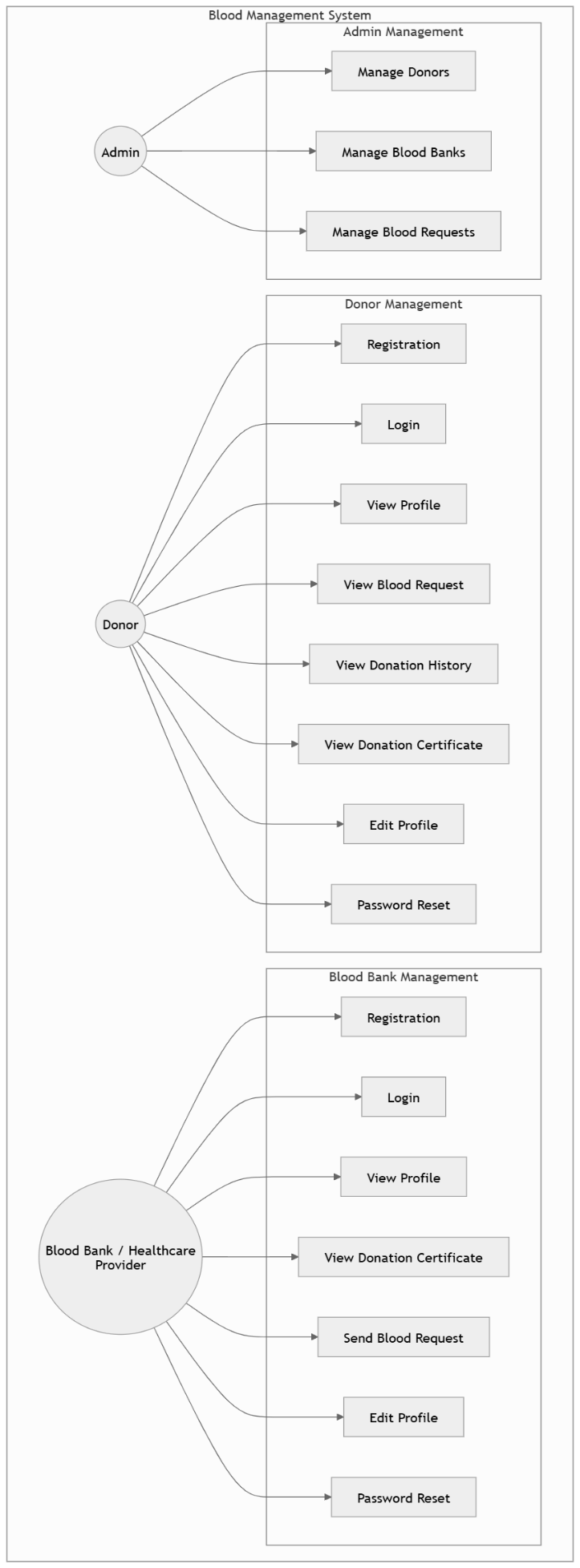
**CHAPTER 4**

**SYSTEM ARCHITECTURE**

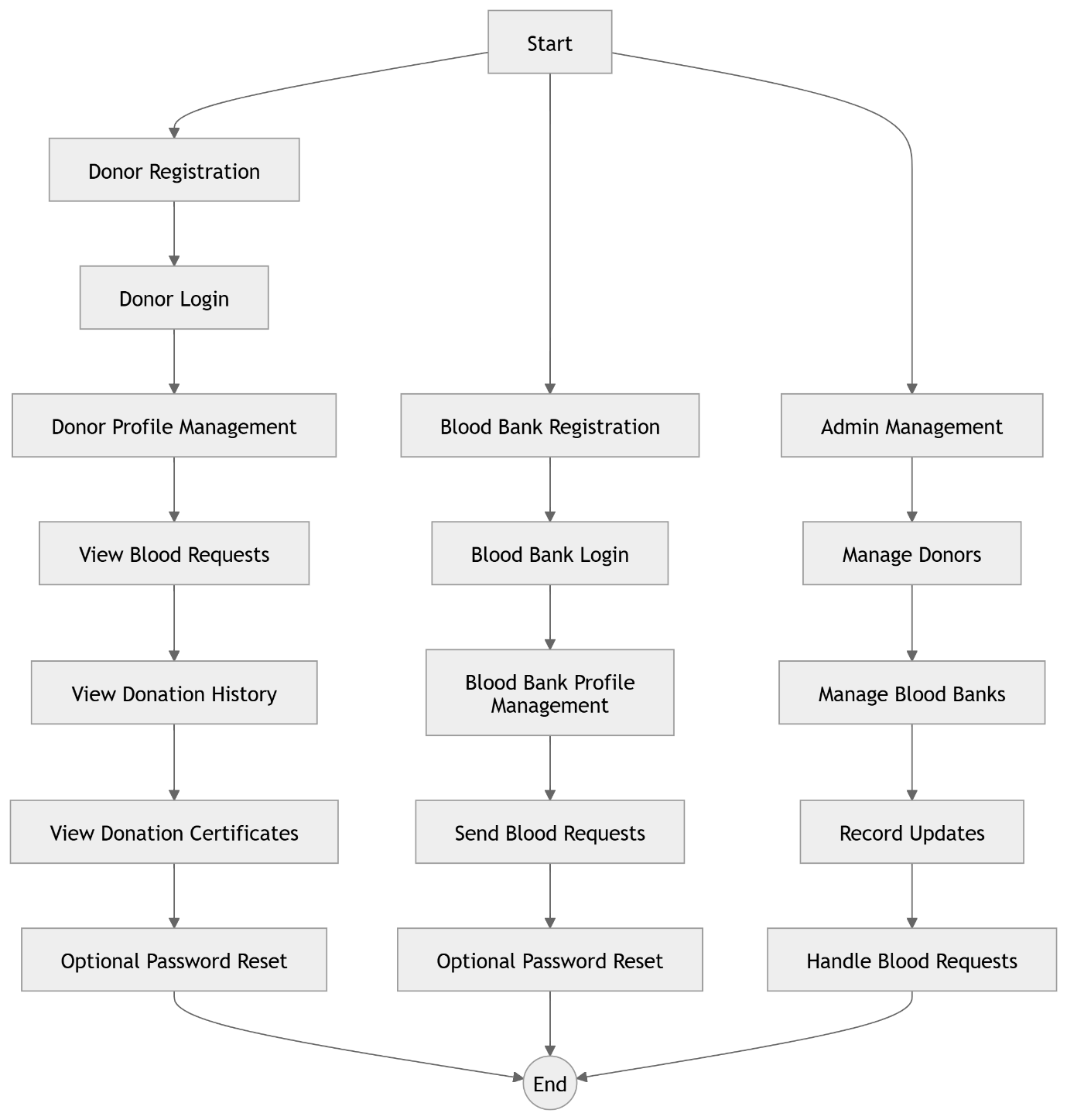
1. **ER Diagram**



1. **UseCase Diagram**



1. **Activity Diagram**



**CHAPTER 5**

**SYSTEM DESIGN**

**Table 5.1 : Donar registration table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **donarAvailability** | 1 | 1 | 0 | 1 | 1 |
| **donarPassword** | pavi | 4321 | labrocky@123 | ssh | 1234 |
| **donarAddress** | ashwini nagar at haveri | chikbasuru | chikbasuru | indi | kavital |
| **donarCity** | Haveri | Haveri | Chikkabasur | Chikkabasur | Devihosur |
| **donarTaluq** | Haveri | Haveri | Haveri | Haveri | Haveri |
| **donarDistrict** | Haveri | Haveri | Haveri | Haveri | Haveri |
| **donarState** | Karnataka | Karnataka | Karnataka | Karnataka | Karnataka |
| **donarGender** | Male | Female | Female | Male | Male |
| **donarBloodGroup** | A+ | O+ | AB+ | B+ | O+ |
| **donarAge** | 22 | 20 | 18 | 21 | 21 |
| **donarEmail** | pavi@gmail.com | shivani@gmail.com | shivanivllingadahalli@gmail.com | sourabhahalli@gmail.com | vpdinniasd@gmail.com |
| **donarNumber** | 7483022523 | 7483022523 | 9353854281 | 2147483647 | 9901185636 |
| **donarName** | Pavan | Shivani | Shivani | Sourabha shrenikraja halli | Vikas |
| **donarId** | 523AP749 | 523OP852 | 281ABP31 | 914BP189 | 523OP128 |

**Table 5.2 : Blood request table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **requestFulfilledOn** | 45649.11751 | 45649.11719 | NULL | NULL | NULL |
| **requestFullfilledBy** | 523AP749 | 523OP852 | 0 | 0 | 0 |
| **requestFulfilled** | 1 | 1 | 0 | 0 | 0 |
| **requestSend** | 45649.0891 | 45649.0918 | 45649.1123 | 45649.114 | 45649.5761 |
| **additionalNotes** | blood need urgently | very critical | need urgently | 0 | 0 |
| **preferredDonorDistance** | withinCity | withinCity | withinCity | withinCity | withinCity |
| **selectedCity** | Haveri | Devihosur | Chikkabasur | Chikkabasur | Haveri |
| **selectedTaluq** | Haveri | Haveri | Haveri | Haveri | Haveri |
| **selectedDistrict** | Haveri | Haveri | Haveri | Haveri | Haveri |
| **selectedState** | Karnataka | Karnataka | Karnataka | Karnataka | Karnataka |
| **urgencyLevel** | Critical | Urgent | Urgent | Critical | Routine |
| **personNumberToContact** | 9901185636 | 9901185636 | 9901185636 | 9901185636 | 7483022523 |
| **personNameToContact** | virat | pavan | pavan | virat | pavan |
| **bloodQuantity** | 2 | 3 | 6 | 3 | 3 |
| **requestBlood** | A+ | O+ | B+ | B+ | A+ |
| **requestBloodBank** | aarogya hospital haveri | aarogya hospital haveri | aarogya hospital haveri | aarogya hospital haveri | CMRIT HOSPITAL |
| **requestedBy** | 636AAR35 | 636AAR35 | 636AAR35 | 636AAR35 | 523CMR97 |
| **requestId** | N8W9MX | 20UIJ7 | SM87EW | 1IDZH1 | YRH2WJ |

**Table 5.3 : Blood bank registration table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **bloodBankPassword** | 80820 | cmr | 80883 | spa |
| **bloodBankAddress** | ashwini nagar at haveri | ashwini nagar at devihosur | ashwini nagar at haveri | chikbasuru |
| **bloodBankCity** | Haveri | Haveri | Haveri | Haveri |
| **bloodBankTaluq** | Haveri | Haveri | Haveri | Haveri |
| **bloodBankDistrict** | Haveri | Haveri | Haveri | Haveri |
| **bloodBankState** | Karnataka | Karnataka | Karnataka | Karnataka |
| **bloodBankEmail** | abc420@gmail.com | cmr@gmail.com | iamhere@gamil.com | pavandvh27@gmail.com |
| **bloodBankMobileNumber** | 9901185636 | 7483022523 | 9901185636 | 8722032914 |
| **bloodBankName** | manjunath medical kavtl | CMRIT HOSPITAL | aarogya hospital haveri | spandana hospital indi |
| **bloodBankId** | 636MAN59 | 523CMR97 | 636AAR35 | 914SPA80 |

**Table 5.4 : Admin table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| userId | password | adminName | adminNumber | adminEmail | adminAddress |
| admin | admin | Pavan C H | 7483022523 | bdms@gmail.com | Mysuru |

**CHAPTER 6**

**IMPLEMENTATION**

1. **Packages :**
   1. **Backend Packages**
      * **cors:**

Prevents unauthorized access: You can configure CORS to restrict access to your backend only from specific origins (e.g., your frontend), preventing potential security vulnerabilities.

Handles preflight requests: Ensures that the browser sends the correct preflight request to the server before the actual request, making sure that complex requests like POST with non-standard headers work seamlessly.

* + - **dotenv:**

Keeps sensitive data safe: Allows you to store sensitive information such as database passwords or Twilio API credentials in a `.env` file, which is not committed to version control (Git).

Environment-specific configuration: Easily configure different settings for development, testing, and production environments by creating separate `.env` files.

* + - **express:**

Middleware support: Easily integrate middleware like `body-parser` to handle request bodies and `express-validator` for input validation, making your backend more modular and maintainable.

Routing: Enables you to define clean, structured routes for API endpoints, helping with organizing functionality, such as donor management or blood request handling.

* + - **mysql:**

Parameterized queries: Helps prevent SQL injection attacks by allowing parameterized queries, where input data is not directly embedded in SQL statements.

Connection pooling: Supports connection pooling to manage multiple database connections efficiently, improving performance in a production environment.

* + - **nodemon:**

Improves development speed: Automatically detects file changes and restarts the server without the need for manual intervention, allowing for a faster development cycle.

Log file monitoring: Monitors changes in code and logs them in real-time, helping developers quickly detect and debug issues without restarting the server manually.

* + - **otp-generator:**

Enhances security: Generates unique OTPs for verification of donor registration or blood request submissions, ensuring that only legitimate users can interact with the system.

Customizable OTP length and complexity: Allows you to adjust the OTP's length and character set (numeric, alphanumeric, etc.) to fit the security requirements of your project.

* + - **twilio:**

User notifications: Sends SMS notifications to users upon successful donor registration, blood request approval, or donation reminders.

Two-factor authentication (2FA): Provides an extra layer of security by using OTPs sent via SMS to authenticate users during critical actions like registration or login.

* 1. **Frontend Packages**
     + **axios:**

Error handling: Allows handling of API errors through `.catch()` or `try/catch`, enabling proper error messages to be shown to users in case something goes wrong (e.g., failed submission or network issues).

Request cancellation: Supports canceling requests (e.g., when a user navigates away from a page), improving performance and avoiding unnecessary backend calls.

* + - **jspdf:**

Custom PDF formatting: Customize the design of generated PDFs, adding logos, tables, or images to enhance the document's appearance, like creating blood donation certificates.

Exporting data: Allows exporting user-generated data (such as donor information or blood requests) into a downloadable PDF file for offline access or records.

* + - **react & react-dom:**

Component-based architecture: Promotes the use of reusable components to create modular and maintainable code, allowing for easier updates and testing.

State management: Integrates with state management libraries (e.g., Redux or Context API) to manage application-wide state, such as tracking donor status or blood request progress.

* + - **react-router-dom:**

Dynamic URL parameters: Enables dynamic routing with parameters (e.g., viewing a specific donor or blood request based on an ID), making the UI more interactive and detailed.

Navigation guards: Allows the implementation of navigation guards to protect certain routes (like the admin dashboard), requiring authentication or specific user roles.

* + - **concurrently:**

Efficiency in development: Runs both the backend and frontend simultaneously, helping developers test the full-stack functionality without having to manually switch between two terminal windows.

Streamlined deployment: Simplifies the deployment process by enabling both parts of the app to be launched together in production, saving time and resources during setup.

* 1. **Database:**

**Database: MySQL**

MySQL is chosen for this project due to its reliability, ease of use, and robust features for relational database management. Below are the key database features utilized in the project:

**Structured Data Storage:** MySQL allows for organizing data in a relational structure, making it easy to query and manage donor details, blood requests, and user information.

**Data Integrity:** Enforces constraints like primary keys, foreign keys, and unique constraints to maintain data accuracy and consistency.

**Scalability:** MySQL supports horizontal and vertical scaling, ensuring the system can handle increased donor registrations and blood requests as the user base grows.

**Query Optimization:** MySQL’s query execution engine and indexing capabilities enable fast retrieval of data, which is crucial for features like location-based donor searches.

**Security Features:** Role-based access control, encrypted connections, and support for secure authentication ensure data security and compliance with privacy requirements.

**Integration with Node.js:** The MySQL Node.js package facilitates seamless communication between the backend and the database, leveraging parameterized queries and connection pooling for efficiency.

**SQL Features Utilized:**

**Joins:** Used to fetch related data across multiple tables, such as finding donors within a certain distance for a blood request.

**Stored Procedures:** Encapsulate complex database logic for reusability, such as calculating donor counts by location.

**Triggers:** Automatically log activities like blood requests or donor registration updates.

**Views:** Provide summarized information for dashboards, like total donors and pending blood requests.

1. **Functions**
   * 1. **Use :** Purpose: Attaches middleware to the Express app. Middleware functions are executed sequentially to process requests and responses.
     2. **Set :** Purpose: Used to set application settings or configurations in an Express app.
     3. **Get :** Purpose: Defines a route to handle HTTP GET requests.
     4. **Post :** Purpose: Defines a route to handle HTTP POST requests, often used for form submissions or API calls.
     5. **Listen :** Purpose: Starts the server and listens for incoming connections on a specified port.
     6. **Require Purpose:** Imports external modules or files into your Node.js application.

**CHAPTER 7**

**TESTING**

**7.1 User Authentication**

This module tests the login, registration, and session management functionalities for both donors and blood banks.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Test Description** | **Test Steps** | **Expected Result** |
| UA\_01 | Verify user can log in with valid credentials. | Enter valid username and password. Click "Login." | User successfully logs in and is redirected to the dashboard. |
| UA\_02 | Verify login fails with invalid credentials. | Enter invalid username or password. Click "Login." | Error message: "Invalid username or password." |
| UA\_03 | Verify password reset link functionality. | Click "Forgot Password." Enter email and submit. | Reset link sent to email. |
| UA\_04 | Verify session timeout after inactivity. | Log in and remain inactive for 15 minutes. | User session ends, and they are redirected to the login page. |

**7.2 Donor Dashboard**

This module ensures that donors can manage their profiles, update their availability, and view donation history.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Test Description** | **Test Steps** | **Expected Result** |
| DD\_01 | Verify donor can view their profile details. | Log in as a donor. Navigate to "Profile" section. | Profile details are displayed correctly. |
| DD\_02 | Verify donor can update their availability. | Toggle availability status. Click "Save." | Status updated successfully. |
| DD\_03 | Verify donation history is displayed. | Log in as a donor. Navigate to "Donation History." | Past donations are displayed correctly. |

**7.3 Blood Bank Dashboard**

This module allows blood banks to view and manage blood requests, search for donors, and track request statuses.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Test Description** | **Test Steps** | **Expected Result** |
| BB\_01 | Verify blood bank can view requests. | Log in as a blood bank. Navigate to "Requests" section. | List of all blood requests is displayed. |
| BB\_02 | Verify blood bank can create a new request. | Fill in request form with valid details and submit. | Request is created successfully. |
| BB\_03 | Verify request status updates after fulfillment. | Mark request as fulfilled. | Request status changes to "Fulfilled." |

**7.4 Blood Request Management**

This module focuses on finding eligible donors based on location and notifying them via SMS.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Test Description** | **Test Steps** | **Expected Result** |
| BR\_01 | Verify donors are displayed based on location. | Create a request and set a preferred location. | Donors near the specified location are displayed. |
| BR\_02 | Verify SMS notifications are sent to donors. | Create a blood request. Click "Notify Donors." | SMS notifications sent to eligible donors. |

**7.5 Donor Registration**

This module ensures that the donor registration process validates critical fields like age, blood type, and contact information.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Test Description** | **Test Steps** | **Expected Result** |
| DR\_01 | Verify donor registration with valid age. | Enter valid age (>= 18). Submit form. | Donor successfully registered. |
| DR\_02 | Verify registration fails for underage donors. | Enter age below 18 and submit. | Error: "Donor must be 18 years or older." |

**7.6 Blood Bank Registration**

This module ensures that blood bank registration validates essential fields like contact information and address.

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case ID | Test Description | Test Steps | Expected Result |
| BB\_04 | Verify blood bank registration with valid details. | Enter all required fields with valid data. Submit form. | Blood bank successfully registered. |
| BB\_05 | Verify registration fails with missing mandatory fields. | Leave mandatory fields empty and submit the form. | Error: "Please fill in all required fields." |

**7.7 Password Reset with SMS OTP**

This module tests the password reset functionality using SMS OTP for both donors and blood banks.

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case ID | Test Description | Test Steps | Expected Result |
| PR\_01 | Verify SMS OTP is sent for password reset. | Click "Forgot Password." Enter mobile number and submit. | OTP sent to the provided mobile number. |
| PR\_02 | Verify password reset with valid OTP. | Enter valid OTP. Submit new password. | Password reset successfully. |
| PR\_03 | Verify password reset fails with invalid OTP. | Enter invalid OTP. Submit new password. | Error: "Invalid OTP." |

**CHAPTER 8**

**RESULTS**

****

Figure 1 Home page

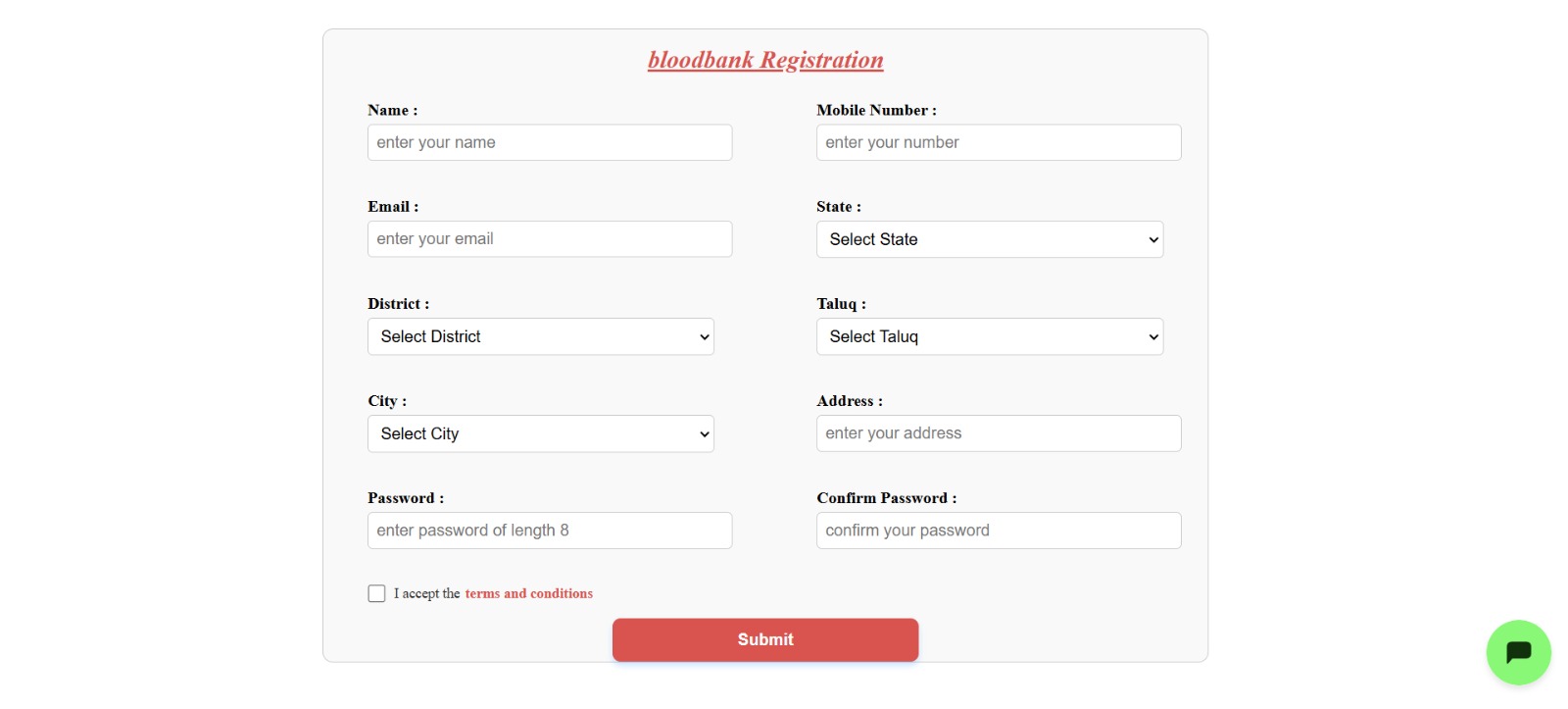


Figure 2 Blood bank registration.

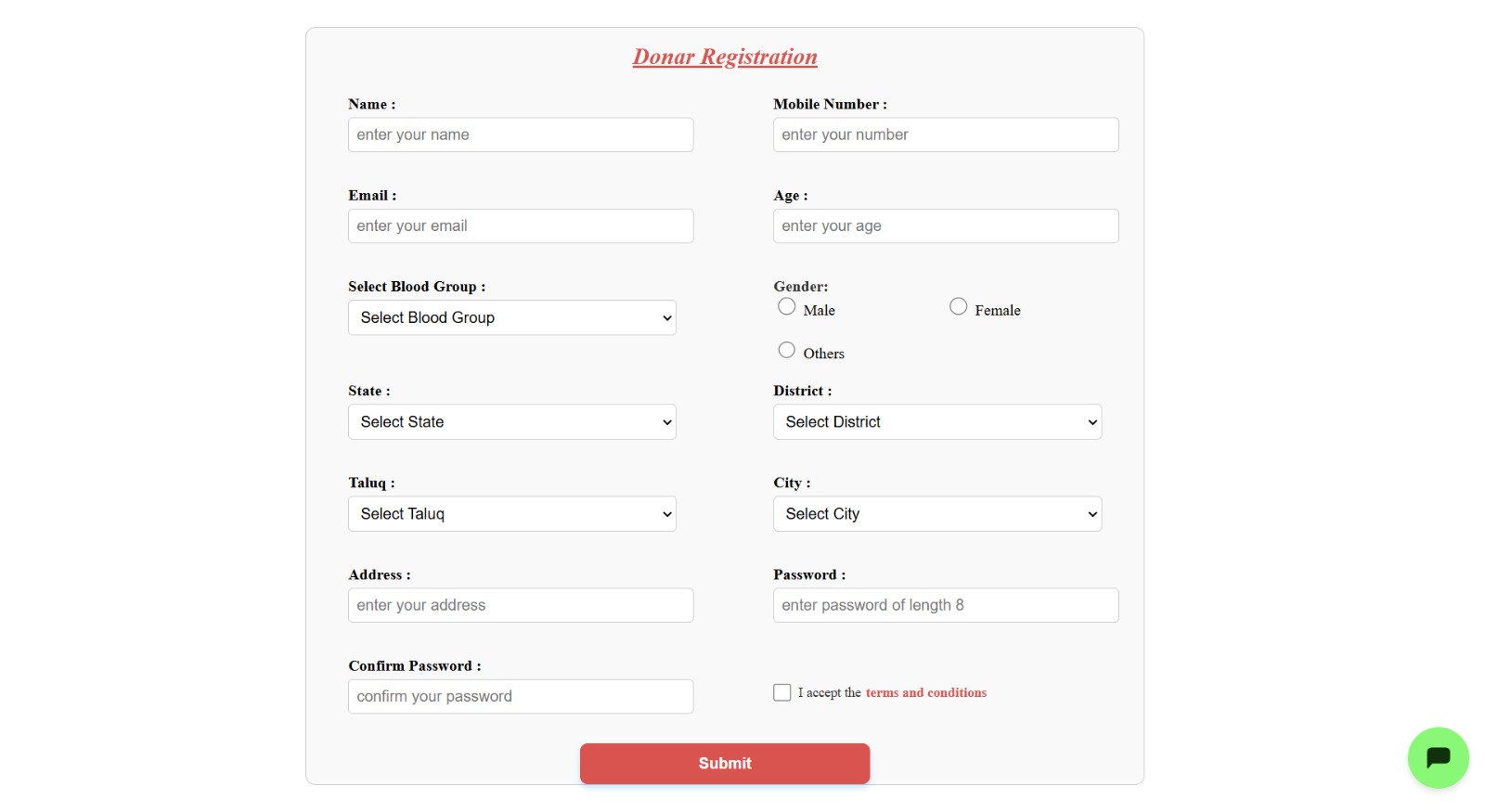


Figure.3 Donor registration.

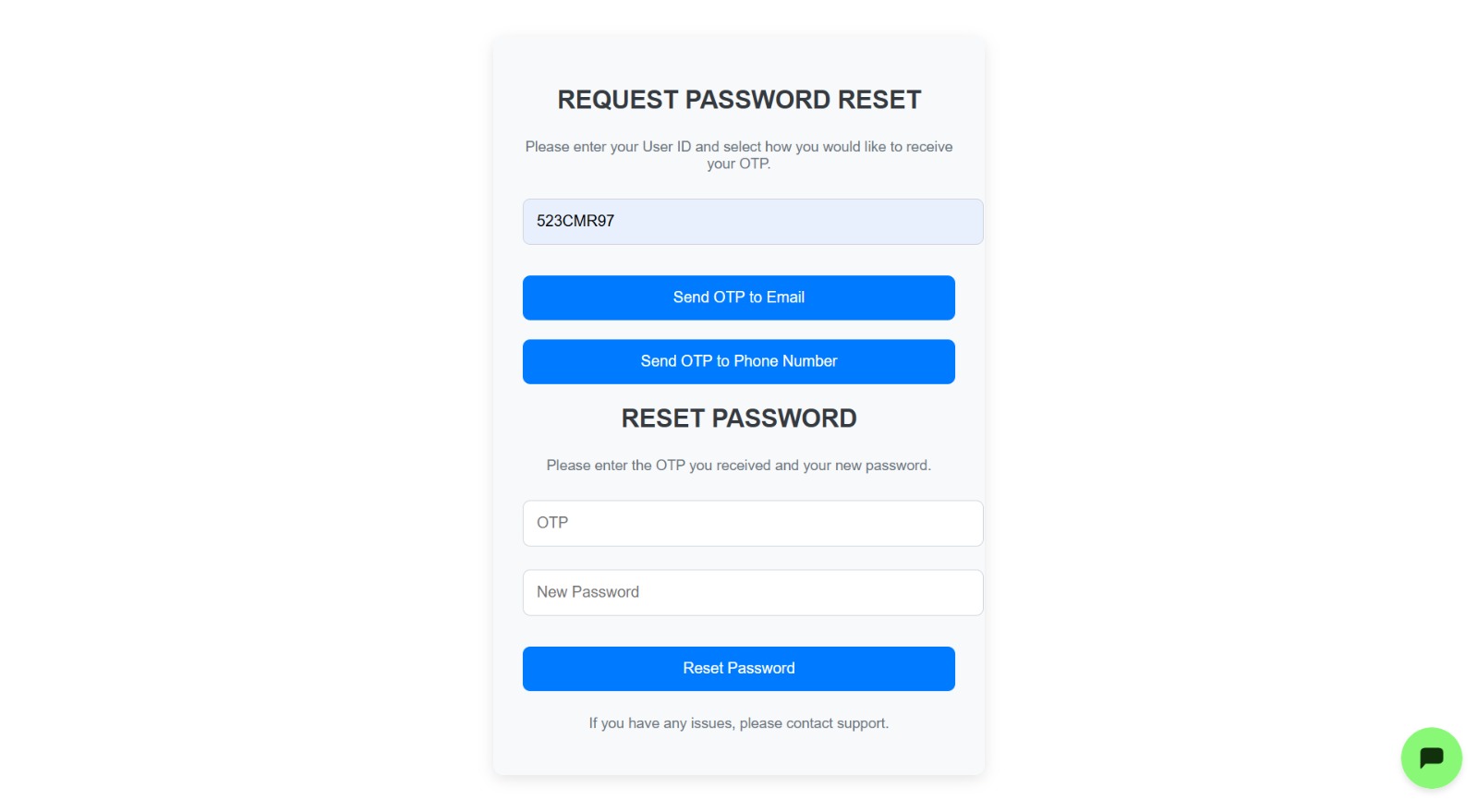


Figure 4 Reset password.

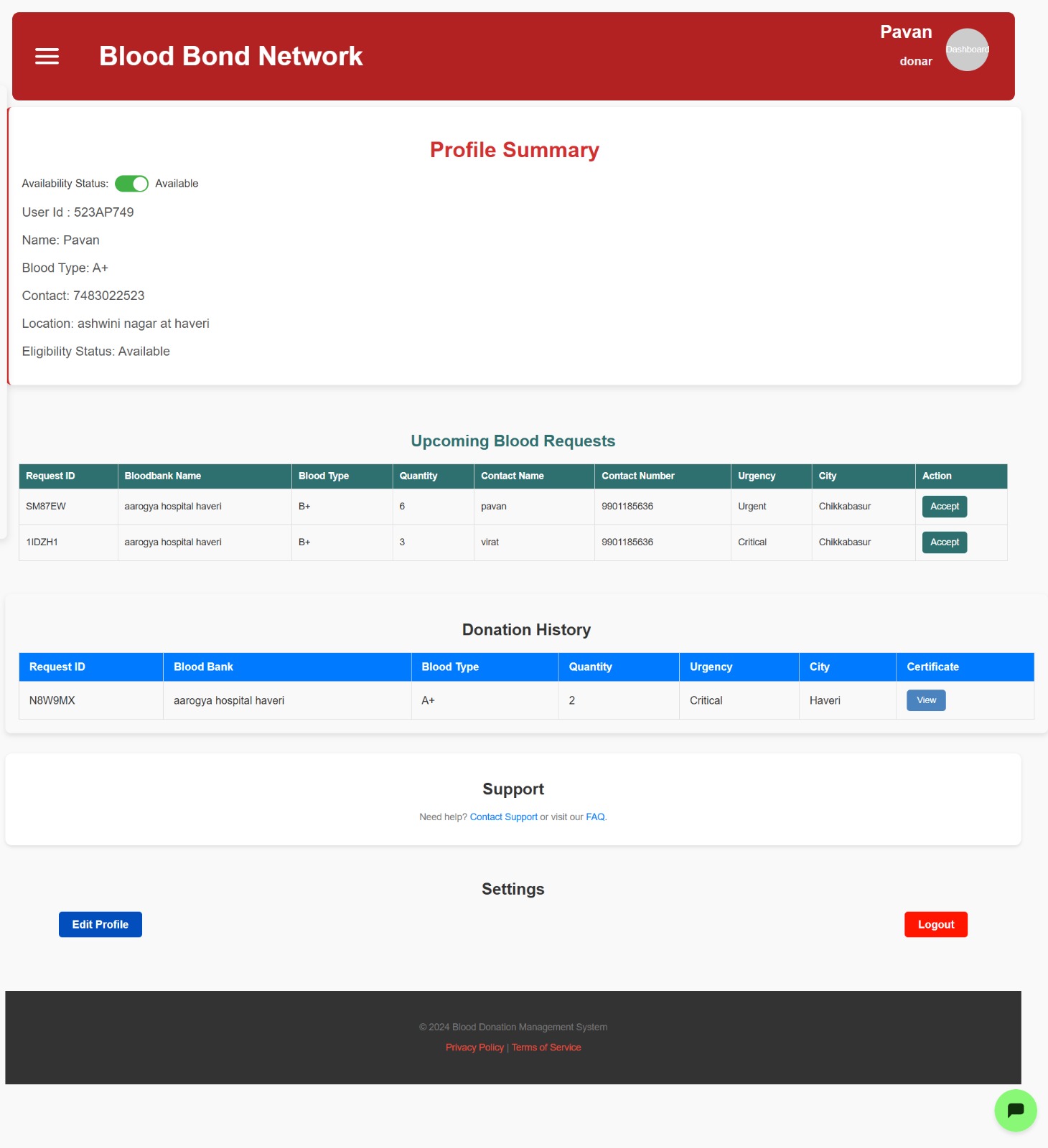
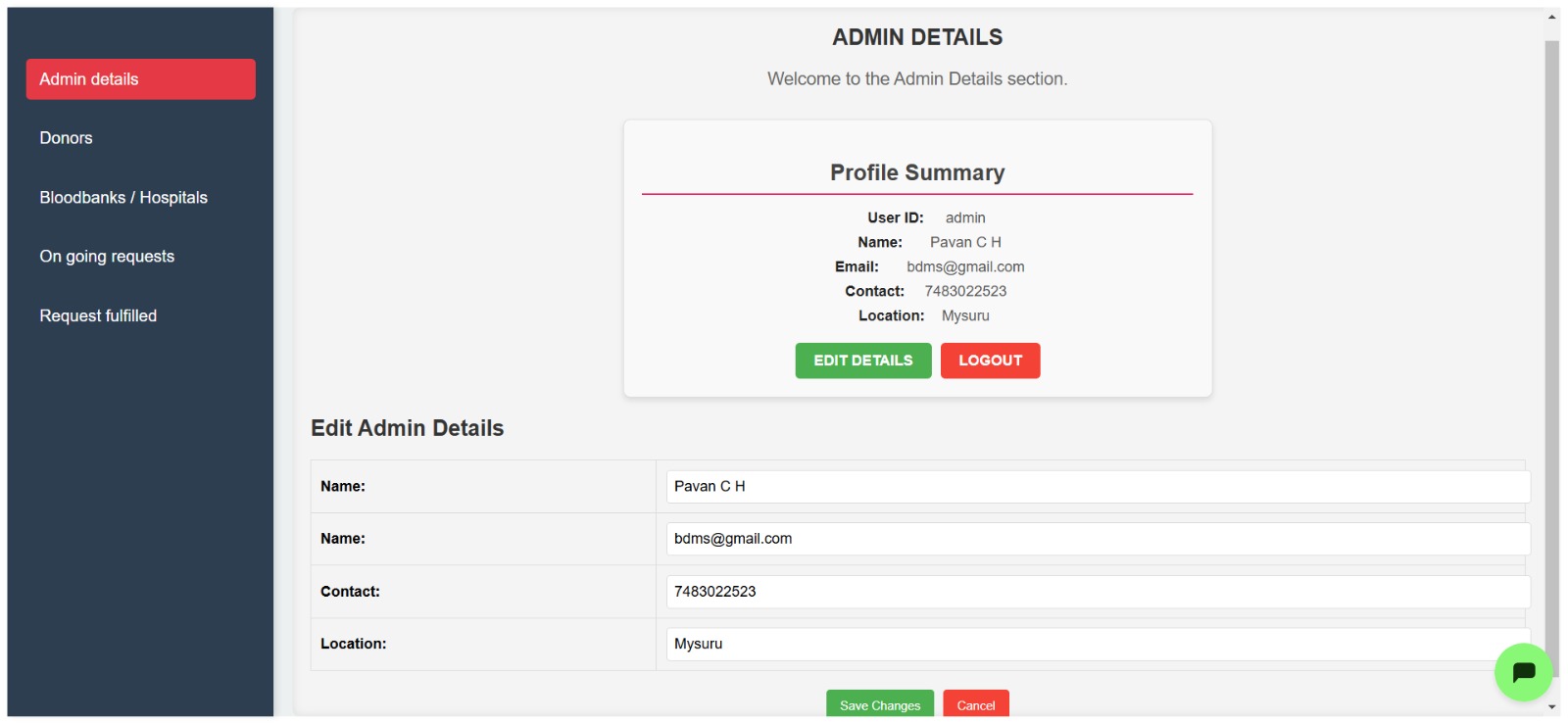


Figure 5 Donor dashboard.

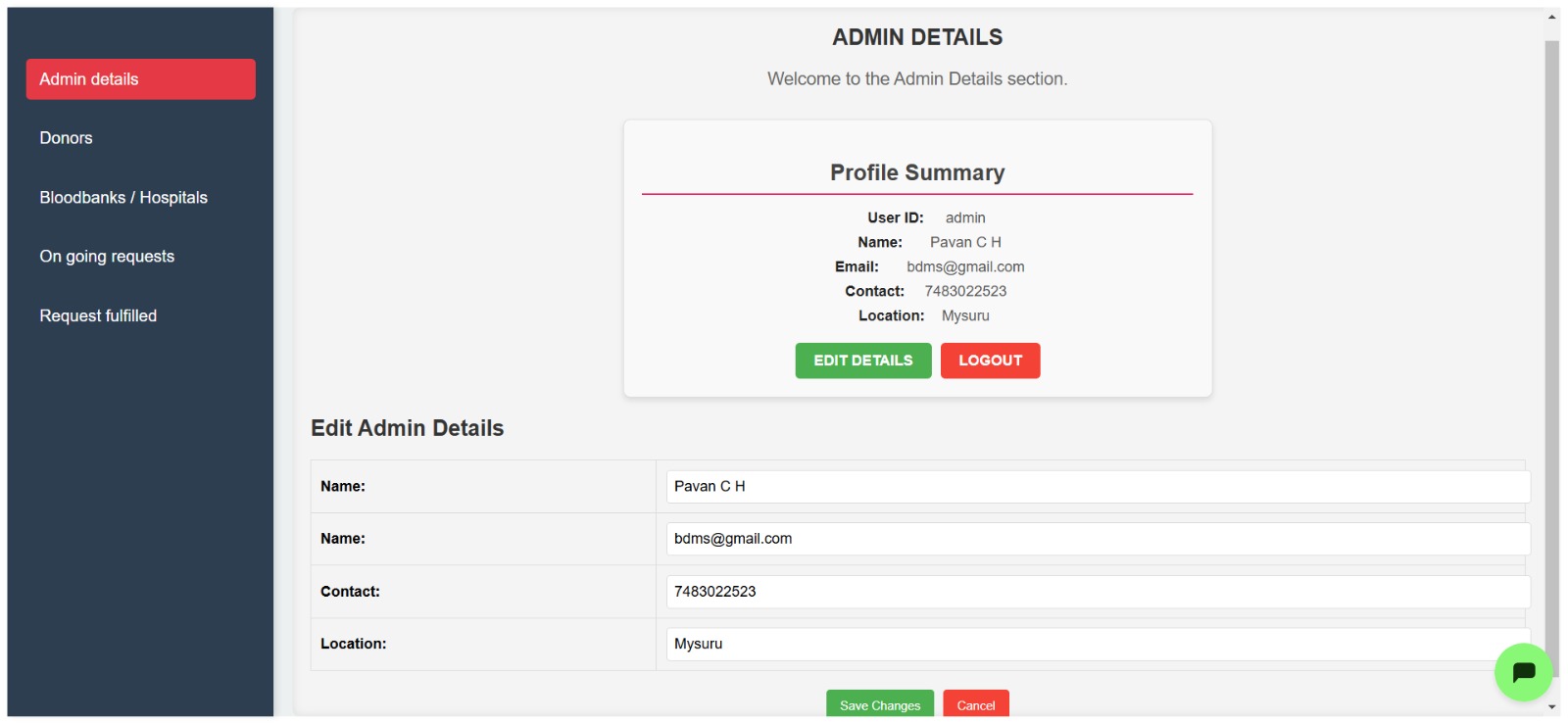


Figure 6 Admin details.

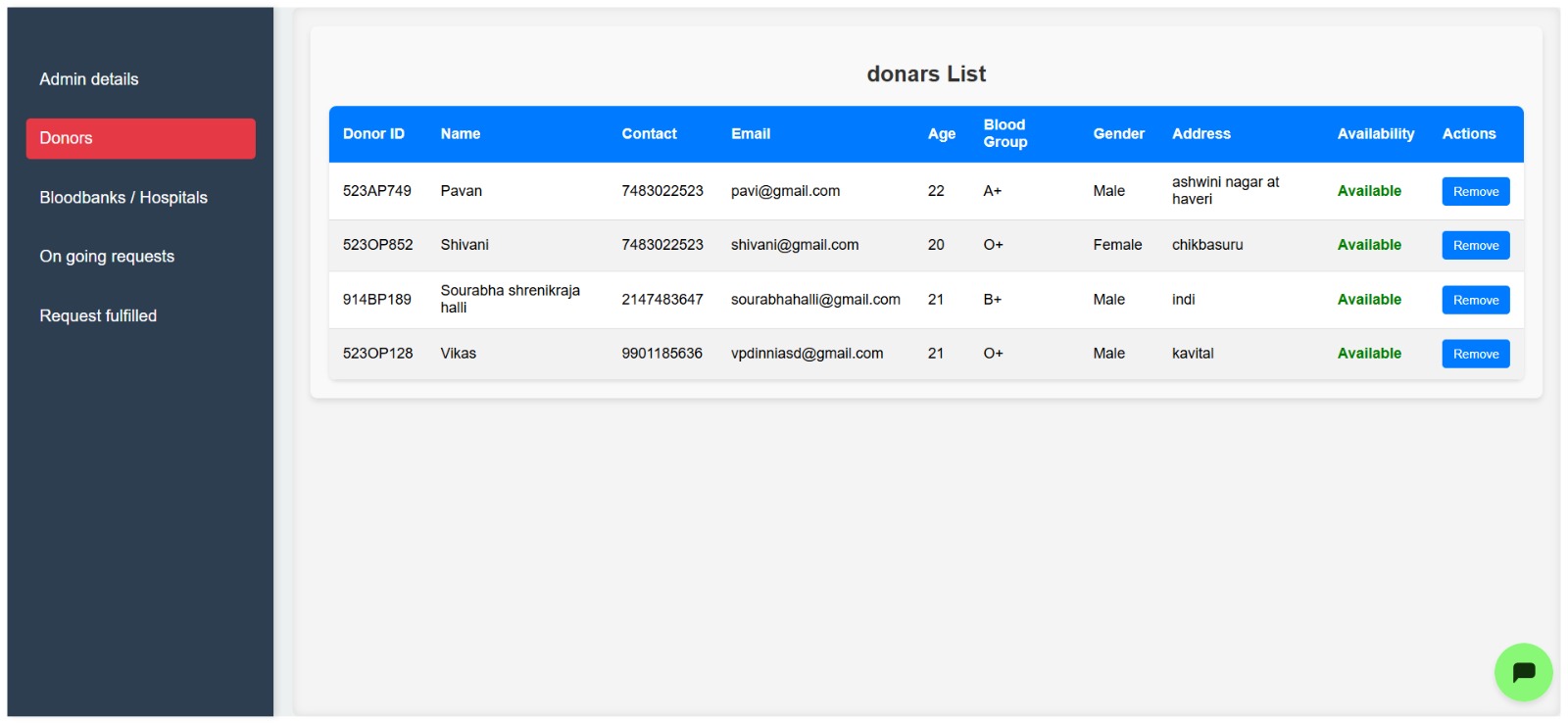


Figure 7 Donars list.

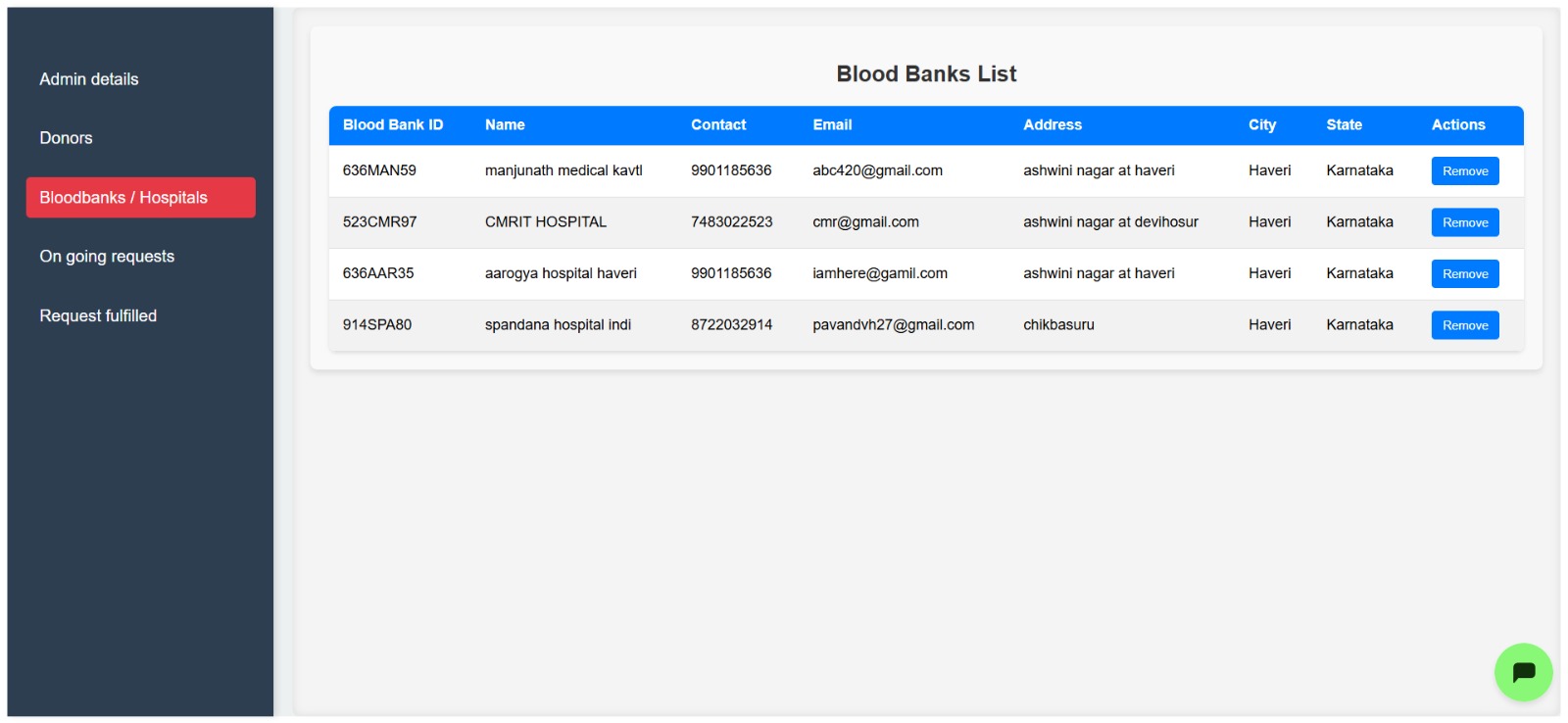


Figure 8 Blood bank list.

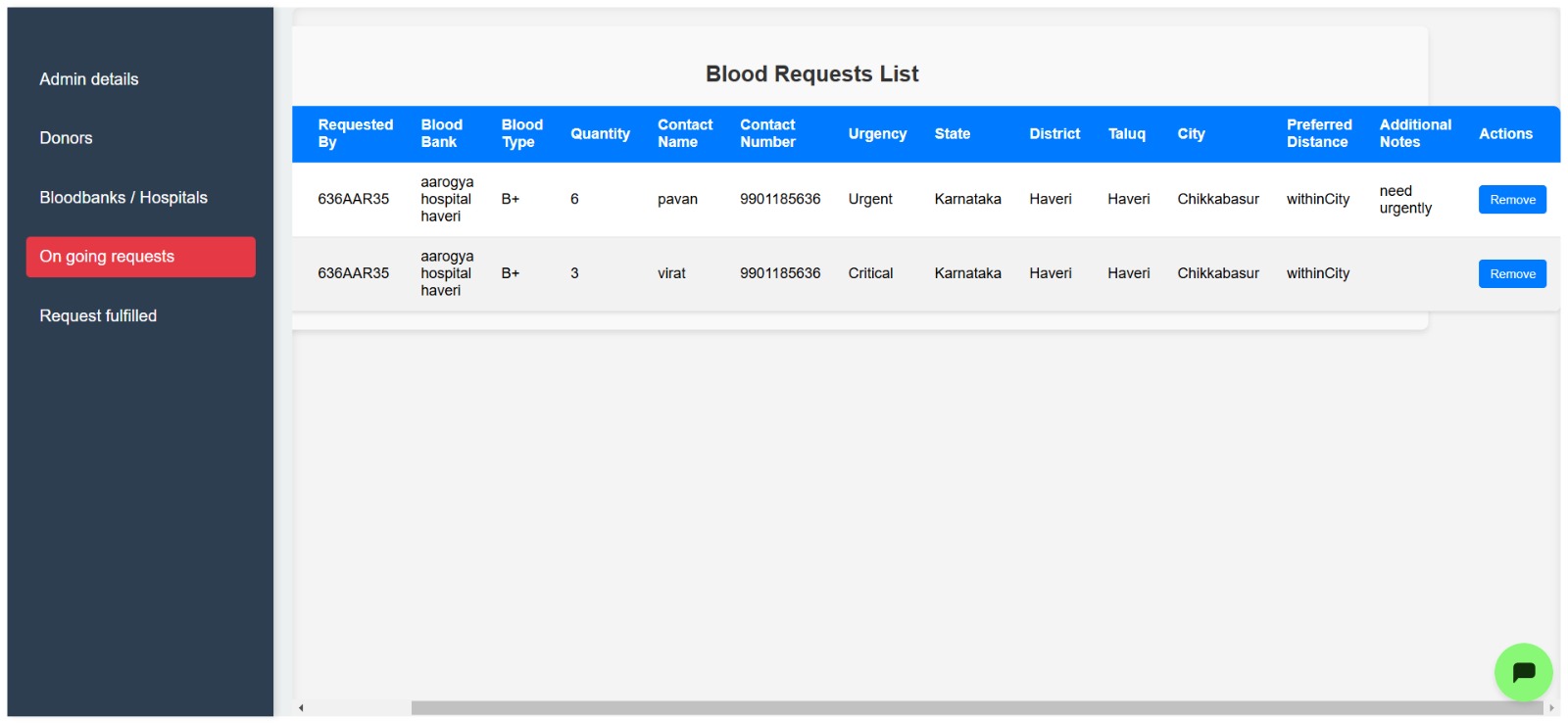


Figure 9 Ongoing requests list.

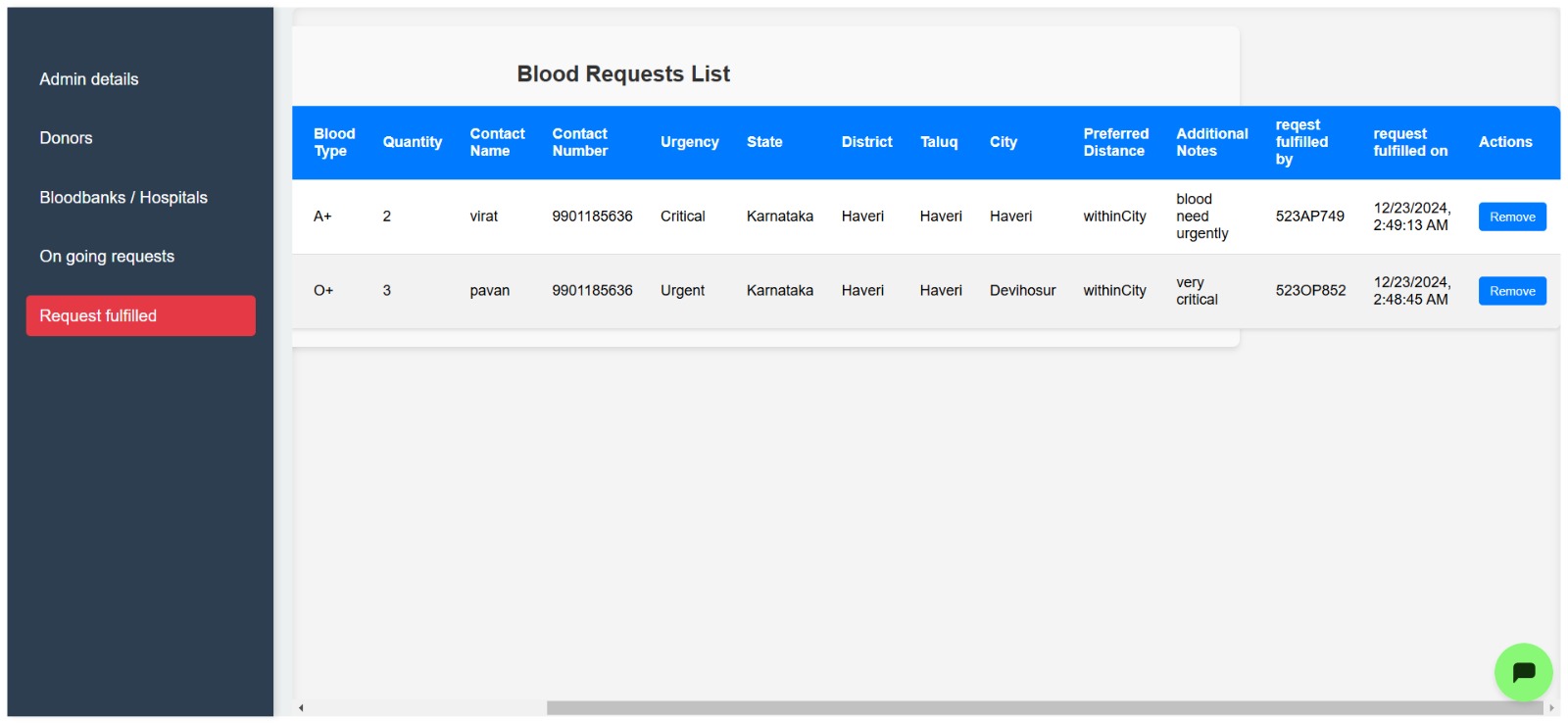


Figure 10 Fulfilled requests list

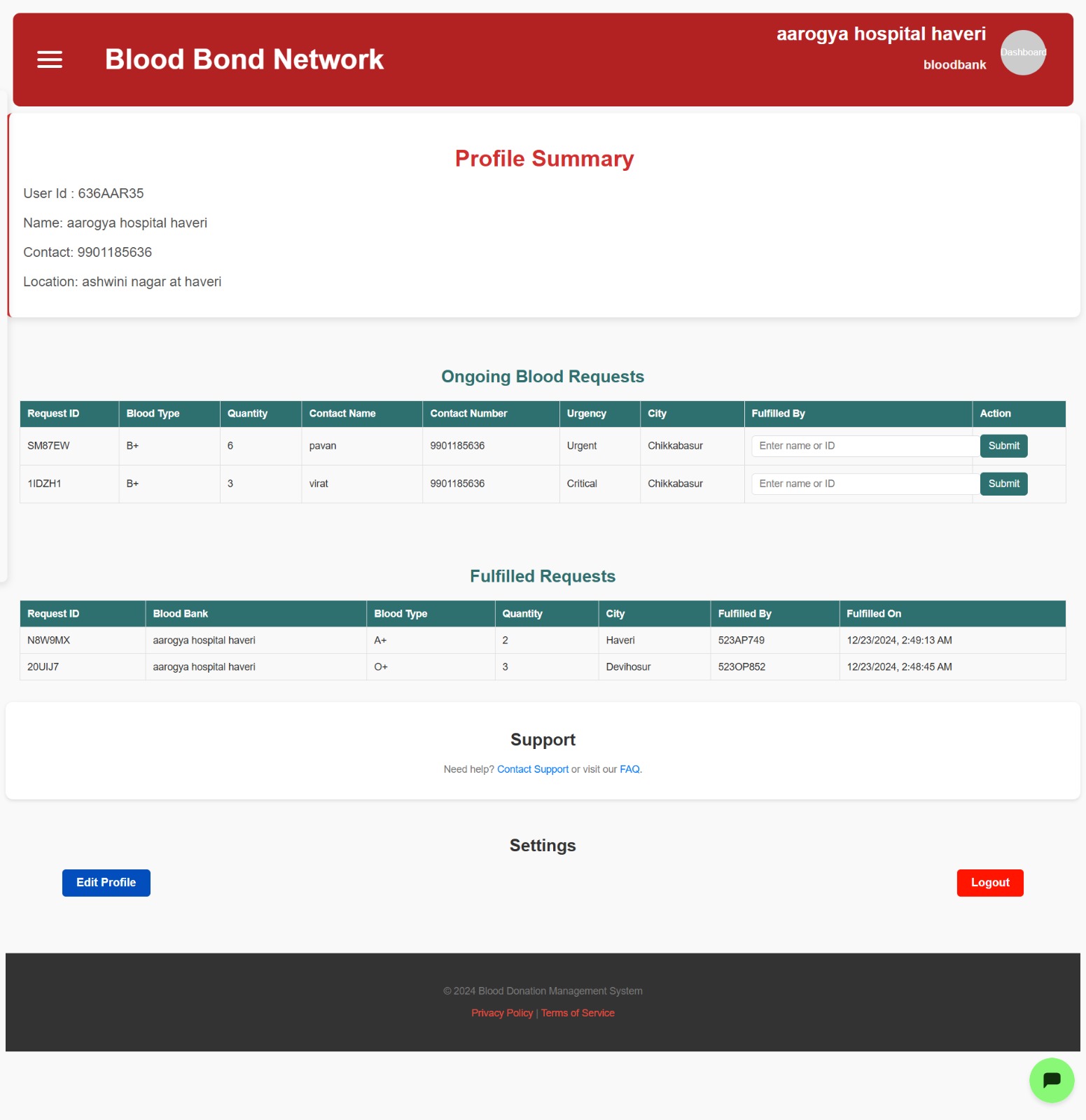


Figure 11 Blood bank dashboard.

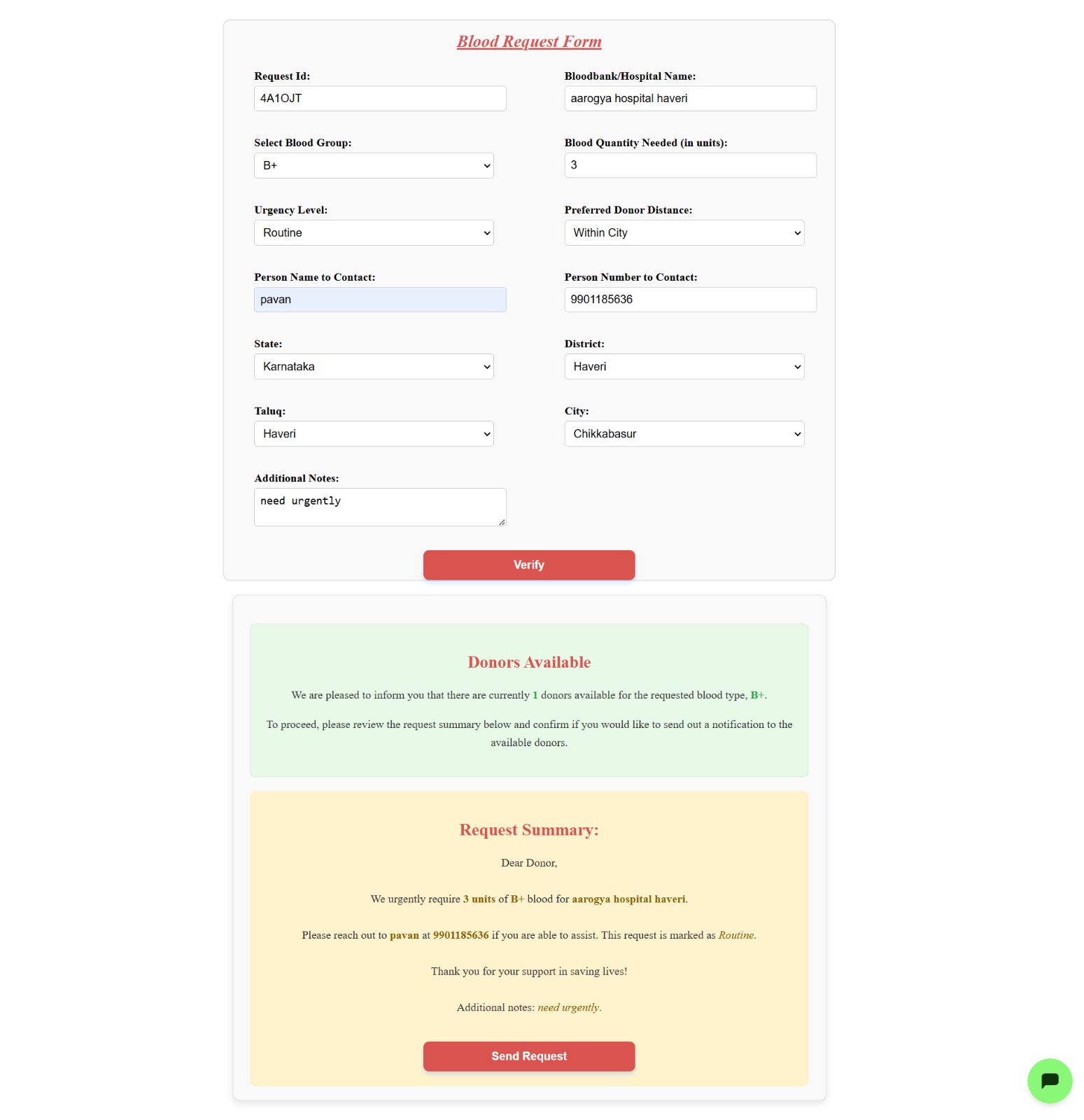


Figure 12 Blood request form.

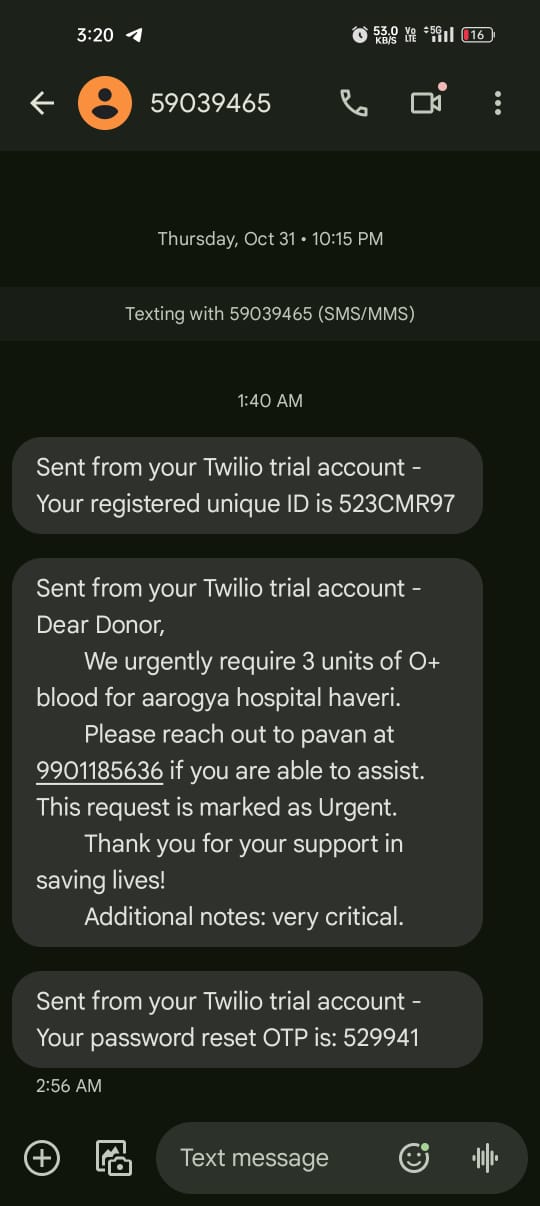


Figure 13 Request Message to donor.



Figure 14 Certificate.

**CONCLUSION**

In conclusion, the proposed web-based blood donation management system offers a significant improvement over traditional methods by providing a more user-friendly, secure, and efficient solution for managing blood donation processes. Express framework and MySQL for backend operations and react for the frontend, the system streamlines the process of scheduling blood donations, tracking donor information, and managing donation events. This system reduces the need for manual coordination, enhances data security, and improves the overall efficiency of blood donation management.

The platform allows donors to easily view donation center information, register as donors, addressing key challenges in blood donation management. Additionally, it offers administrators the tools needed to oversee and manage donor information, track donations, and organize blood donation drives more effectively.

This system aligns with the increasing demand for organized and efficient blood donation management, offering a robust solution to improve the donor experience, increase donation rates, and ensure timely blood availability. By integrating healthcare and blood donation information, the system contributes to the optimization of blood supply management, benefitting both donors and recipients.

**References**

1. **ChatGPT**: [https://chat.openai.com](https://chat.openai.com/)
2. **MDN Web Docs**: [https://developer.mozilla.org](https://developer.mozilla.org/)
3. **React**: [https://reactjs.org](https://reactjs.org/)
4. **React Router DOM**: [https://reactrouter.com](https://reactrouter.com/)
5. **Axios**: [https://axios-http.com](https://axios-http.com/)
6. **html2pdf.js**: <https://github.com/eKoopmans/html2pdf.js>
7. **jsPDF**: <https://github.com/parallax/jsPDF>
8. **Node.js**: [https://nodejs.org](https://nodejs.org/)
9. **Express.js**: [https://expressjs.com](https://expressjs.com/)
10. **dotenv**: <https://github.com/motdotla/dotenv>
11. **mysql**: <https://github.com/mysqljs/mysql>
12. **Nodemailer**: [https://nodemailer.com](https://nodemailer.com/)
13. **Nodemon**: [https://nodemon.io](https://nodemon.io/)
14. **Twilio**: [https://www.twilio.com](https://www.twilio.com/)
15. **otp-generator**: <https://www.npmjs.com/package/otp-generator>