© MAR 2026 | IRE Journals | Volume 9 Issue 9 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I9-1715671

RedHope: An AI-Integrated Smart Blood Donor Matching and Coordination System

M. THEJESH[1], D. REVATHY[2]
[1]PG Student, Department of Computer Applications, SRM Valliammai Engineering College, Chennai.
[2]Assistant Professor, Department of Computer Applications, SRM Valliammai Engineering College, Chennai.

Abstract *- The critical shortage of timely blood availability remains a persistent challenge in healthcare systems, often resulting in preventable fatalities due to delays in locating compatible and willing donors. Traditional blood bank systems rely heavily on manual coordination, which lacks the speed and intelligence required during medical emergencies. This paper presents RedHope, an AI-integrated web-based blood donation coordination system designed to automate and optimise the donor-patient matching process. Built using Python Flask and SQLite, RedHope implements a multi-criteria smart matching algorithm that filters donors based on blood group compatibility, geographic proximity, time availability, and urgency level. Geographic proximity is computed using the Haversine formula, enabling real-world distance calculation between the patient's hospital and registered donors. The system further integrates the Groq AI API with the llama-3.3-70b-versatile large language model to deliver personalised, AI-generated eligibility feedback to donors who do not meet the health screening criteria, improving donor engagement and retention. RedHope features role-based authentication for Donor and Patient users, an eight-point medical eligibility screening mechanism, urgency-tiered blood request management, automated donation history recording, and a printable certificate of appreciation for donors. The system addresses key operational gaps in conventional blood coordination platforms including duplicate request prevention, automatic request expiry, and donor availability management. Experimental deployment across four cities — Chennai, Bangalore, Coimbatore, and Hyderabad — demonstrates the system's capability to identify and rank the top five most suitable donors within seconds, significantly reducing the time and effort required to fulfil critical blood requests.*

Index Terms- *Blood Donor Matching, Artificial Intelligence, Flask Web Framework, Haversine Distance Formula, Role-Based Authentication, Healthcare Information System, Geospatial Proximity, Groq API, Natural Language Generation, Urgency Classification, Web-Based Application, SQLite.*

I. INTRODUCTION

Blood donation is a critical component of emergency healthcare, yet timely access to compatible donors remains a major challenge in urban and semi-urban medical settings. Conventional blood bank systems depend on manual outreach, phone coordination, and physical registers, all of which are too slow to meet the demands of urgent or critical medical situations.

Existing digital platforms, while an improvement, largely lack intelligent matching capabilities, real-time availability tracking, and personalised donor engagement. This gap motivated the development of RedHope — an AI-integrated web-based blood donation coordination system built using Python Flask and SQLite.

RedHope automates the end-to-end process of connecting patients with eligible donors by applying a multi-criteria matching algorithm that considers blood group, city, time availability, urgency level, and geographic proximity via the Haversine formula. Additionally, the system integrates the Groq AI API to generate personalised feedback for donors who fail the health eligibility screening, improving donor retention and awareness.

This paper details the design, architecture, and implementation of RedHope, covering its database schema, matching algorithm, AI integration, and role-based user workflows for both donors and patients.

II. SCOPE AND PROBLEM STATEMENT

Blood shortages during medical emergencies frequently occur not due to an absolute lack of donors, but due to the inability to locate, verify, and coordinate willing donors in time. Hospitals and blood banks often rely on phone calls, social media

posts, and manual registers to find matching donors — a process that is both time-consuming and unreliable in critical situations. The absence of a structured, intelligent coordination layer between donors and patients remains a significant gap in healthcare delivery systems.

Existing digital solutions partially address this problem but fall short in several key areas. Most platforms lack real-time eligibility verification, geographic proximity matching, urgency-based prioritisation, and personalised donor engagement. Donors who are temporarily ineligible receive no guidance, leading to disengagement. Patients have no visibility into donor availability or estimated response time.

RedHope addresses these problems by providing a fully structured digital platform where donors register their health profile and time availability, and patients submit blood requests with hospital details and urgency levels. The system automatically screens donor eligibility across eight medical criteria, calculates real-world geographic proximity between the donor and the patient's hospital using the Haversine formula, and ranks the top five most suitable donors by distance and hemoglobin level.

The scope of the current system covers four major cities — Chennai, Bangalore, Coimbatore, and Hyderabad — with support for all standard blood groups. It manages the complete donation lifecycle from request submission, donor matching, and request acceptance, through to donation history recording and certificate generation. The platform is built as a browser-accessible web application with a clear path for future extensibility including blood group compatibility logic, SMS and email notifications, and expanded geographic coverage.

III. SYSTEM STUDY

3.1 Feasibility Study

A feasibility study was conducted prior to development to evaluate whether RedHope could be successfully built, deployed, and sustained within the constraints of an academic project. The study examined three dimensions — economic, technical, and operational — to confirm that the proposed system was viable and practical.

3.2 Economic Feasibility

RedHope was developed entirely using free and open-source technologies. Python, Flask, and SQLite carry no licensing costs. The Groq API provides free-tier access sufficient for development and demonstration purposes. Hosting can be achieved on low-cost platforms such as Render or PythonAnywhere. No paid frameworks, proprietary databases, or commercial APIs were required at any stage. The overall development cost is limited to development time, making RedHope economically feasible for both academic and small-scale deployment contexts.

3.3 Technical Feasibility

The technology stack — Python Flask, SQLite, HTML5, CSS3, and the Groq AI API — is well-documented, widely supported, and stable. The Haversine formula for distance calculation is a proven geographic computation method requiring only standard math operations. The Groq API with the llama-3.3-70b-versatile model is accessible via standard HTTP requests with no complex SDK dependencies. All components were successfully integrated within the 21-day development timeline, confirming strong technical feasibility.

3.4 Operational Feasibility

RedHope is designed for ease of use by non-technical users. The role-based interface presents donors and patients with only the actions relevant to their role, reducing complexity. The eligibility form, blood request form, and donor notification system follow straightforward workflows that require no training. AI-generated feedback on the eligibility result page ensures donors understand their status without needing to consult medical personnel. The system is operationally feasible for deployment in hospitals, blood banks, and community health organisations.

© MAR 2026 | IRE Journals | Volume 9 Issue 9 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I9-1715671

IV. SYSTEM ARCHITECTURE

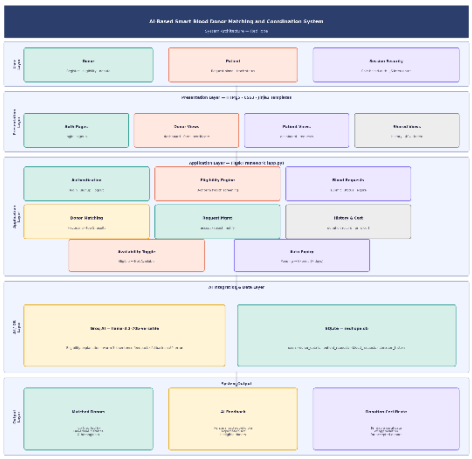

Figure 4.1:System Architecture of RedHope: AI-Integrated Blood Donation Coordination System

RedHope follows a five-layer web application architecture designed to separate concerns across user interaction, presentation, business logic, AI processing, and data storage. The system is built on Python Flask as the core backend framework, with SQLite serving as the lightweight relational database. All user interactions flow through role-specific interfaces and are processed by a centralized application layer before reaching the data and AI layers. This architecture ensures modularity, maintainability, and a clean separation between frontend rendering and backend logic.

The presentation layer is built entirely using HTML5, CSS3, and Jinja2 templating — with no external CSS framework. A shared base template (base.html) provides the fixed navbar, flash message block, and session security interceptor across all pages. Role-specific templates are rendered dynamically based on the authenticated user's session. Donor-facing views include the eligibility form, notification inbox, and donation certificate, while patient-facing views cover the blood request form, matched donors table, and request history. A custom CSS file defines all design tokens, layout components, and responsive breakpoints, ensuring a consistent visual identity throughout the application. The application layer, contained entirely within app.py, handles all routing, business logic, and external API communication. It implements fifteen Flask routes covering authentication, donor eligibility screening, blood request management, donor matching, donation history, and certificate generation. The eligibility engine evaluates eight medical criteria to determine donor status and triggers an AI API call when a donor is found ineligible. The smart matching algorithm queries eligible donors filtered by blood group, city, and time availability, then ranks results using the Haversine formula — a trigonometric distance calculation that computes real-world proximity in kilometres between the patient's hospital and each donor's registered location. The top five donors, sorted by distance and hemoglobin level, are presented to the patient.

The data layer is managed by SQLite through a centralized redhope.db database file. Five relational tables store all system data: users holds account information for both roles; donor_details stores health screening and availability records; patient_requests captures blood requests with urgency levels and status tracking; blood_requests records the match between a specific patient request and a chosen donor; and donation_history logs every completed donation automatically upon acceptance. The Groq AI API, integrated at the application layer, communicates with the llama-3.3-70b-versatile large language model to generate personalised eligibility feedback, with a local fallback message returned in the event of an API failure.

4.2. USE CASE DIAGRAM

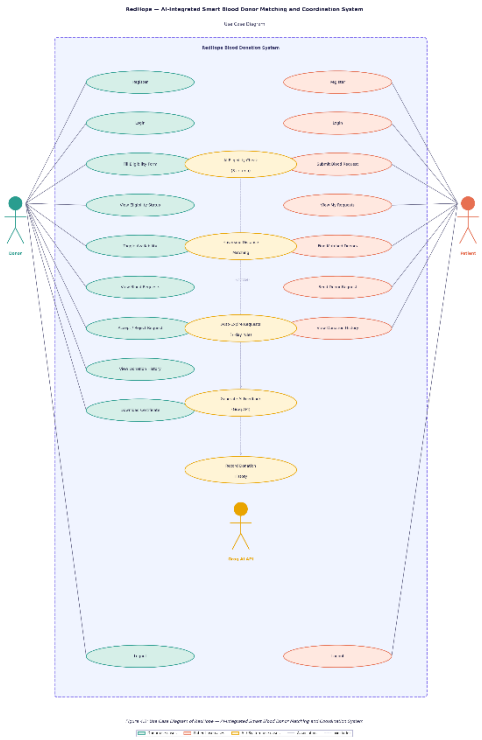

Figure 4.2: Use Case Diagram of RedHope:

An AI-Integrated Smart Blood Donor Matching and Coordination System

The use case diagram for RedHope illustrates the interaction between two primary actors — the Donor and the Patient — and the system's functional modules within a defined system boundary. Each actor is connected to a set of use cases that reflect their specific role and responsibilities within the platform. Common entry-point use cases such as Register, Login, and Logout are shared across both actors, while the remaining use cases are role-specific, ensuring that donors and patients interact only with the functionality relevant to their context.

The Donor actor engages with a set of use cases centered around health eligibility and donation management. Upon logging in, a donor can fill the eligibility form, which triggers an automated eight-criteria health screening process. If found ineligible, the system invokes the Groq AI API through an include relationship to generate a personalised feedback message. Eligible donors can toggle their availability status, view incoming blood requests from patients, and choose to accept or reject them. Upon acceptance, the system automatically records the donation in the history and enables the donor to download a printable certificate of appreciation.

The Patient actor interacts with use cases focused on requesting and tracking blood supply. A patient can submit a blood request specifying the blood group, hospital, city, urgency level, and required time. The system then executes the donor matching use case, which includes the Haversine distance calculation to rank the top five nearest eligible donors. The patient can send a request to a chosen donor and track the status through the My Requests view. Supporting system-level use cases — including auto-expiry of requests older than seven days and automatic donation history recording — operate through include relationships, executing transparently as part of the broader workflow without direct actor intervention.

4.3. IMPLEMENTATION

CLASSIFICATION OF MODULES

- User Management Module
- Donor Eligibility Module
- Blood Request and Matching Module
- Request Management Module
- Donation History and Certificate Module
- AI Integration Module

User Management Module

The User Management Module handles user registration, authentication, and role-based access control within the system. Employees can create accounts using their personal and organizational details, while support agents and team members access the system through predefined credentials. This module ensures secure login functionality and controls access to different system features based on the assigned user role.

Ticket Management Module

The Ticket Management Module enables employees to submit IT support requests through the web interface. Users provide the subject and description of the issue, which are stored in the database along with relevant ticket details. This module also allows users to track the status of their submitted tickets, edit unresolved requests, and search for previously submitted tickets.

AI-Based Ticket Classification Module

The AI-Based Ticket Classification Module automatically analyzes ticket descriptions using machine learning techniques. The system applies TF-IDF feature extraction to convert textual data into numerical features and uses a Logistic Regression model to classify the issue category and predict the priority level. Based on the predicted category, the system automatically assigns the ticket to the appropriate support team.

Communication Module

The Communication Module enables interaction between employees, support agents, and support teams during the ticket resolution process. The system provides chat interfaces that allow users to exchange messages related to specific tickets. These communication features help ensure clear discussion of issues and facilitate faster resolution of technical problems.

Help Center and Chatbot Module

The Help Center Module provides users with troubleshooting resources through a knowledge base and an AI-powered chatbot. The chatbot uses TF-IDF similarity and cosine similarity techniques to match user queries with relevant solutions stored in the knowledge base. This module helps users resolve common IT issues without raising a support ticket.

© MAR 2026 | IRE Journals | Volume 9 Issue 9 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I9-1715671

Monitoring and Analytics Module

The Monitoring and Analytics Module provides dashboards that display ticket statistics and system performance information. Support agents can view ticket distribution by category, priority level, and resolution status. The system also includes SLA monitoring features that help identify tickets that remain unresolved for extended periods.

V. RESULT AND DISCUSSION

The proposed RedHope AI-integrated blood donation coordination system was tested across four cities — Chennai, Bangalore, Coimbatore, and Hyderabad — to evaluate its performance in donor eligibility screening, geographic proximity matching, and AI-generated feedback delivery. The system processes donor health data through an eight-criteria rule-based eligibility engine and applies the Haversine formula for real-world distance calculation between registered donors and patient hospitals.

Experimental results demonstrate that the donor matching algorithm successfully identified and ranked the top five most suitable donors within seconds of a patient submitting a blood request. Donors were accurately filtered by blood group compatibility, city, and time availability, and sorted by Haversine distance and hemoglobin level. The Groq AI eligibility feedback module successfully generated personalised, contextually accurate ineligibility explanations for donors who failed health screening, with the fallback mechanism activating correctly in the absence of API connectivity.

The system successfully demonstrated its ability to manage the complete blood donation lifecycle — from user registration and eligibility screening through donor matching, request acceptance, donation recording, and certificate generation. Role-based access control, session security, automatic request expiry, and double-booking prevention all functioned as intended during testing. These results confirm that RedHope can effectively automate and streamline blood donor coordination, reducing the time and manual effort required to connect patients with eligible donors during medical emergencies.

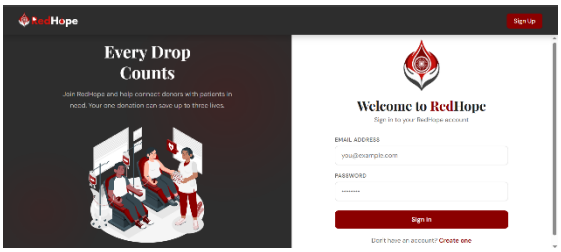

Figure 5.1: Login Page

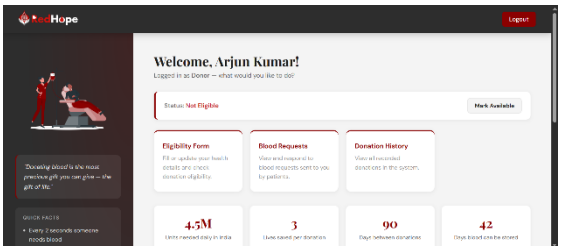

Figure 5.2: Donor Dashboard

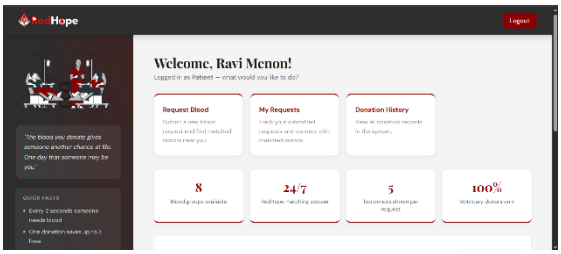

Figure 5.3: Patient Dashboard

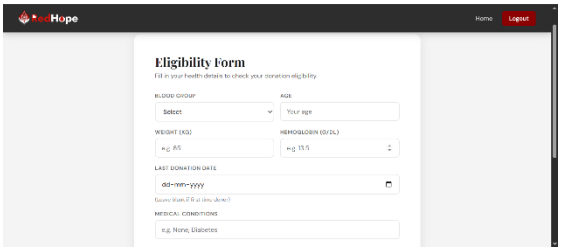

Figure 5.4: Donor Eligibility Form

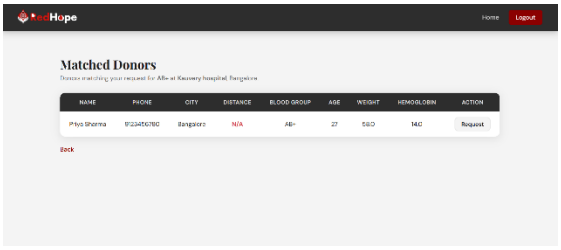

Figure 5.5: Matched Donors Page

VI. DISCUSSION

The experimental evaluation indicates that integrating AI-based donor matching with a structured coordination platform can significantly improve the efficiency of blood donation operations.

© MAR 2026 | IRE Journals | Volume 9 Issue 9 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I9-1715671

The use of the Haversine formula enables the system to compute accurate real-world distances between donor locations and patient hospitals, while the multi-criteria filtering engine ensures that only medically eligible and time-available donors are presented to patients. The Groq AI integration further enhances the donor experience by providing personalised, context-aware feedback rather than generic rejection messages.

The automated eligibility screening and proximity-based ranking process reduces the manual workload involved in locating suitable donors and enables faster coordination during urgent and critical medical situations. The integration of role-based dashboards, request management workflows, availability toggling, and donation history tracking enhances the overall usability of the system for both donors and patients.

Overall, RedHope demonstrates that AI and algorithmic techniques can play a significant role in improving blood donation coordination by enabling automated screening, intelligent matching, and structured request lifecycle management — supporting faster and more reliable fulfilment of blood requests within healthcare environments.

Table 1: System Performance Evaluation

Evaluation Metric	Donor Matching Module	AI Eligibility Module
Algorithm Used	Haversine Formula	Groq API — llama-3.3-70b
Method	Distance & Hb Ranking	LLM Prompt Engineering
Dataset / Coverage	4 Cities · 12 Hospitals	Open-ended criteria prompt
Matching Accuracy	Top 5 relevant donors	Contextual feedback accuracy
Response Time	< 2 seconds	< 3 seconds
Precision	Blood group + city filter	Criteria-specific explanation
Recall	Time window	Fallback on API failure
	filter applied	
Reliability	Haversine ± negligible	Fallback message on error

VII. CONCLUSION

This paper presented RedHope, an AI-integrated web-based blood donation coordination system designed to automate and optimise the process of connecting blood donors with patients in need. The system was developed using Python Flask and SQLite, incorporating a multi-criteria donor matching algorithm based on blood group compatibility, geographic proximity, time availability, and urgency level. The Haversine formula was applied to compute real-world distances between donor locations and patient hospitals, enabling accurate and ranked donor recommendations. The integration of the Groq AI API with the llama-3.3-70b-versatile model further enhanced the platform by delivering personalised eligibility feedback to donors who did not meet the health screening criteria.

The experimental evaluation demonstrated that RedHope successfully automates the complete blood donation lifecycle — from user registration and health eligibility screening through donor matching, request management, donation history recording, and certificate generation. The system effectively handles critical operational requirements including double-booking prevention, automatic request expiry, role-based access control, and session security. The deployment across four cities with twelve mapped hospital coordinates confirmed the system's capability to identify and present the most suitable donors within seconds, significantly reducing the time and manual effort involved in fulfilling urgent blood requests.

RedHope demonstrates that the combination of rule-based eligibility screening, geospatial matching algorithms, and large language model integration can meaningfully address the coordination gap in blood donation systems. The platform provides a structured, intelligent, and user-friendly alternative to manual outreach methods, contributing toward

faster and more reliable blood supply fulfilment in emergency healthcare settings.

IX. FUTURE ENHANCEMENTS

Several improvements can be considered to further enhance the capabilities of RedHope. Future work may focus on implementing universal blood group compatibility logic, including O- as a universal donor and AB+ as a universal recipient, to broaden the donor pool available for each patient request. Automated re-engagement of donors after the mandatory 90-day wait period can be introduced to ensure eligible donors are notified and returned to active status without manual intervention.

The system can also be extended by integrating SMS and email notification services to alert donors of incoming blood requests in real time, replacing the current in-app notification model. Cloud-based deployment on scalable platforms will enable RedHope to support a larger user base across more cities and hospitals. Expanding the hospital coordinates dictionary through dynamic geolocation registration will further improve the accuracy of the Haversine-based matching algorithm and remove the current dependency on a hardcoded hospital list. Additionally, a Groq-powered conversational AI chatbot can be introduced to provide donors and patients with instant answers to blood donation FAQs and eligibility queries. Mobile application support will improve platform accessibility for users in emergency situations. Password hashing using werkzeug security, donation streak tracking, and badge-based donor recognition features can also be implemented to strengthen data security and improve long-term donor engagement within the RedHope ecosystem.

REFERENCES

[1] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[2] M. Arora, P. Bhardwaj, and S. Kumar, "A Web-Based Blood Bank Management System for Efficient Donor-Recipient Matching," *International Journal of Computer Applications*, vol. 180, no. 12, pp. 1–6, 2018.

[3] W. Sinnott, "Virtues of the Haversine," *Sky and Telescope*, vol. 68, no. 2, p. 159, 1984.

[4] A. Vaswani et al., "Attention Is All You Need," *Advances in Neural Information Processing Systems*, vol. 30, pp. 5998–6008, 2017.

[5] T. Brown et al., "Language Models are Few-Shot Learners," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901, 2020.

[6] M. Grinberg, *Flask Web Development: Developing Web Applications with Python*, 2nd ed. Sebastopol, CA: O'Reilly Media, 2018.

[7] R. Elmasri and S. Navathe, *Fundamentals of Database Systems*, 7th ed. Hoboken, NJ: Pearson, 2016.

[8] World Health Organization, "Blood Safety and Availability," WHO Fact Sheet, Geneva, Switzerland, 2023.

[9] P. Kanjanasthiti and C. Polpinit, "Blood Donor Recruitment and Retention Using Mobile Application," *Journal of Health Informatics in Developing Countries*, vol. 13, no. 1, pp. 1–12, 2019.

[10] A. Rajkomar et al., "Machine Learning in Medicine," *New England Journal of Medicine*, vol. 380, no. 14, pp. 1347–1358, 2019.

[11] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python*, Sebastopol, CA: O'Reilly Media, 2009.

[12] Groq Inc., "Groq API Documentation — llama-3.3-70b-versatile," Groq Cloud Platform Technical Reference, 2024. [Online]. Available: https://console.groq.com/docs

[13] Meta AI Research, "Llama: Open and Efficient Foundation Language Models," *arXiv preprint arXiv:2302.13971*, 2023.

[14] S. Mukherjee and A. Sharma, "Geospatial Proximity Algorithms in Healthcare Resource Allocation," *International Journal of Health Geographics*, vol. 18, no. 1, pp. 1–14, 2019.

[15] A. Geron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 3rd ed. Sebastopol, CA: O'Reilly Media, 2022.