Malnad College Of Engineering

Under the auspices of M.T.E.S ®

(An Autonomous Institution Affiliated to VTU, Belgaum)

P.B. No. 21, Hassan 573 202, Karnataka

Full Stack Development (23IS553)

TOPIC: Community Blood Donor & Finder

Submitted By:

NAME	USN
Brunda C K	4MC23IS021
Deeksha R S	4MC23IS029
Ganavi Prasad	4MC23IS034
Geetha H	4MC23IS035
Manya G Karle	4MC23IS059

Under The Guidance of:

Mr. Krishna Swaroop A

Assistant Professor, ISE

Department of Information Science and Engineering

Malnad College of Engineering

Hassan - 573 202

2024-2025

CONTENTS

Sl. no	Title	Page no
1	Abstract	1
2	Introduction	2
3	Objectives of the project	3
4	System requirements	3-4
5	System design	4-8
6	Database design	8-11
7	Implementation	11-13
8	Screenshots	13-17
9	Testing	18
10	Results	18
11	Conclusion	18-19
12	Future enhancements	19
13	References	19

.

1 ABSTRACT

The Blood Donor Web Application is a web-based full-stack system developed using Django, aimed at digitalizing, simplifying, and streamlining the process of connecting voluntary blood donors with individuals in urgent need of blood. Traditional methods of searching for donors, such as circulating messages on WhatsApp, posting on social media, or calling known contacts, are highly unorganized, time-consuming, and often unreliable during medical emergencies. These outdated methods lack verification, real-time availability information, and structured donor data, which leads to critical delays. To address this real-world healthcare challenge, this project introduces an efficient, automated, and database-driven platform that enables donors to register with their blood group, location, last donation date, and availability. Recipients can raise structured blood requests, specify urgency, and instantly view compatible donors filtered by blood group.

This system ensures secure authentication for all users, provides accurate donor information, and reduces the time-consuming manual processes involved in emergency blood searches. Core features include donor registration, donor availability status, city-based filtering, blood group matching, request submission, request management, and map-based donor visualization. The platform is designed with Django's MTV architecture to ensure modularity, scalability, and ease of development. The centralized database stores all donor and request details, ensuring consistency, improved accessibility, and efficient retrieval. This application significantly enhances transparency, reduces delays, and improves emergency response effectiveness. Overall, the Blood Donor Web Application delivers a reliable, scalable, and life-saving solution that modernizes traditional donor search systems and supports hospitals, patients, and the public during critical healthcare situations.

2 INTRODUCTION

Blood is one of the most essential components of human life, required in countless medical situations such as accidents, surgeries, childbirth complications, anemia treatments, cancer therapies, and organ transplants. Despite the tremendous need for blood, the process of finding compatible and available blood donors remains one of the most challenging aspects of healthcare management. In many regions, there is still no proper digital system that connects donors and

recipients in an organized, fast, and reliable manner. This creates critical delays, especially during emergencies where minutes can determine life or death.

The domain of blood donation was selected for this project because it addresses a highly sensitive and real-world healthcare problem faced globally. Lack of a centralized system causes poor coordination between donors and recipients and leads to unnecessary time loss during critical situations. Additionally, many voluntary donors are willing to help but have no platform to register and make themselves easily reachable

In the real world, multiple challenges exist in the traditional donor search system:

• Lack of real-time availability

• No structured donor database

• No history of donation or eligibility checks

• No emergency prioritization

• Communication delays

• High dependency on manual networks

• No location-based donor suggestions

• Lack of verification and reliability

The proposed Blood Donor Web Application addresses all these issues by implementing a digital, centralized, and systematic approach. Django's MTV architecture supports efficient development and provides built-in tools for authentication, form handling, database operations, and secure access. The system ensures clean separation of logic, user interface, and data storage, making the application scalable for future upgrades.

3 OBJECTIVES OF THE PROJECT

The Blood Donor Web Application is designed with several core objectives focused on modernizing and digitalizing the blood donation workflow. The project aims to eliminate the inefficiencies, errors, and delays caused by outdated manual donor search processes. The main objectives include:

.

• To develop a complete online platform for managing blood donation activities

• To simplify the process of finding compatible donors during medical

• To automate manual processes

• To provide real-time donor availability

• To offer secure authentication and access control

• To maintain a centralized database

• To reduce human errors, manipulation, and outdated information

• To make blood request handling transparent and traceable

• To create a scalable and future-ready system

4. SYSTEM REQUIREMENTS

4.1 SOFTWARE REQUIREMENTS

Programming Language: Python 3.x

Framework: Django 4.x

Database: SQLite / MySQL

Frontend Technologies: HTML5 , CSS3 , JavaScript

IDE : Visual Studio Code , PyCharm Community Edition

Other Dependencies: Django ORM

4.2 HARDWARE REQUIREMENTS

Minimum sysem configuration:

• Processor: Intel Core i3 or above

• RAM: Minimum 4 GB (Recommended 8 GB for faster load times)

• Storage: Minimum 20 GB free disk space

• Operating System: Windows / macOS / Linux

5. SYSTEM DESIGN

5.1 Architecture Diagram

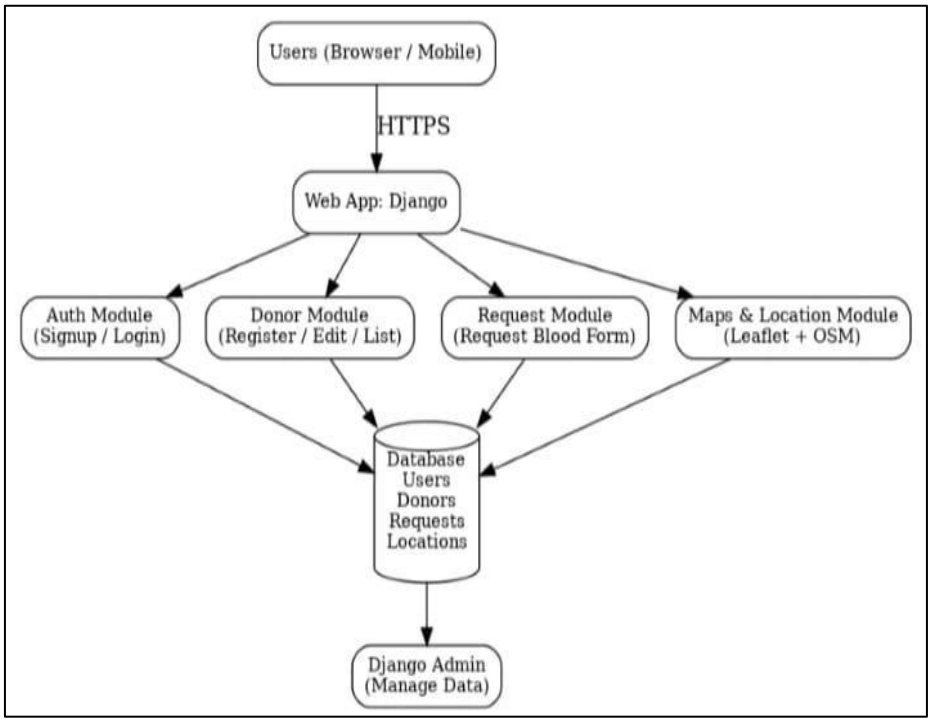


Figure: System Architecture of the Blood Donor Web Application

5.2 Explanation of Architecture

The architecture of the Blood Donor Web Application is designed using Django's MTV pattern, ensuring secure user interaction, proper data flow, and efficient management of donor and request information. The system is divided into interconnected modules that work together to deliver a smooth user experience and maintain accurate records.

Users (Browser/Mobile Layer)

- End-users interact with the system through web browsers on mobile or desktop devices.
- Users include donors, recipients, and admins.
- All communication between users and the server happens securely over HTTPS to protect sensitive information such as login credentials and personal details.

Web Application Layer (Django Framework)

This forms the core processing layer of the system. Django handles routing, authentication, form processing, validations, and database operations.

Major Modules in the System

a. Authentication Module (Signup/Login)

- Handles user account creation, login, and session management.
- Ensures only registered users can access donor registration and request features.
- Uses Django's authentication system for password hashing and secure login.
- Prevents unauthorized access to important pages.

b. Donor Module (Register / Edit / List)

- Allows users to register themselves as donors with details like blood group, city, availability, and last donation date.
- Enables donors to update their information when needed.
- Fetches and displays all donors based on filters such as blood group and location.
- Stores donor location details (latitude/longitude) for map-based searching.

c. Request Module (Blood Request Form)

- Used by patients/recipients to request blood by specifying required blood group, units, hospital name, and urgency.
- Validates request details to ensure accuracy before saving.
- Displays all active requests on the dashboard to allow faster matching with donors.
- Helps users view recent and urgent blood requests easily.

d. Maps & Location Module (Leaflet + OpenStreetMap)

- Provides map-based visualization of donors using Leaflet JS and OSM.
- Helps users locate nearest donors based on city or coordinates.
- Enhances the usability of the system by providing a geographic view of blood availability.

Database Layer

- Stores all essential data, including Users, Donors, Requests, and Locations.
- Ensures consistency through Django ORM and relational models.
- Automatically handles foreign key relationships between users and their donor/request records.
- Supports fast retrieval for filtering donors and displaying requests.

Django Admin Panel (Data Management Layer)

- Provides backend access for administrators to manage users, donors, and blood requests.
- Used to update or delete incorrect data and monitor system activity.
- Ensures that system information stays clean, updated, and organized.

Data Flow Explanation

1. Input Stage (User Interaction)

- Users enter data through forms such as signup, donor registration, or blood request submission.
- All input is sent via HTTPS to ensure security.

2. Processing Stage (Django Views & Business Logic)

- Django views validate data, apply rules (e.g., donor eligibility), and prepare responses.

- Filtering logic identifies matching donors for a given city and blood group.
- System checks last donation date and marks donor availability accordingly.

3. Decision-Making Stage

- The system determines:

– Which donors match the required blood group

– Which donors are available and eligible

– What requests need to be displayed on dashboards

- Urgent requests are prioritized.

4. Storage Stage

- All donor and request information is stored in the database for retrieval, updates, and future use.
- Admins can review and manage stored data via the Django Admin panel.

User Request Flow (Models → Views → Templates) Models:

- Define structure of Users, Donors, Requests, and Location data.
- Contain validation rules and relationships.

Views:

- Handle all business logic such as saving forms, filtering donors, and managing requests.
- Connect user actions to model operations.

Templates:

- Render HTML pages for home, dashboard, donor list, and forms.
- Display filtered donor results, active requests, and maps.

Interaction With the Database

1. Relational structure:

- Each donor and request is linked to a user.
- City, blood group, and availability fields are indexed for fast search.

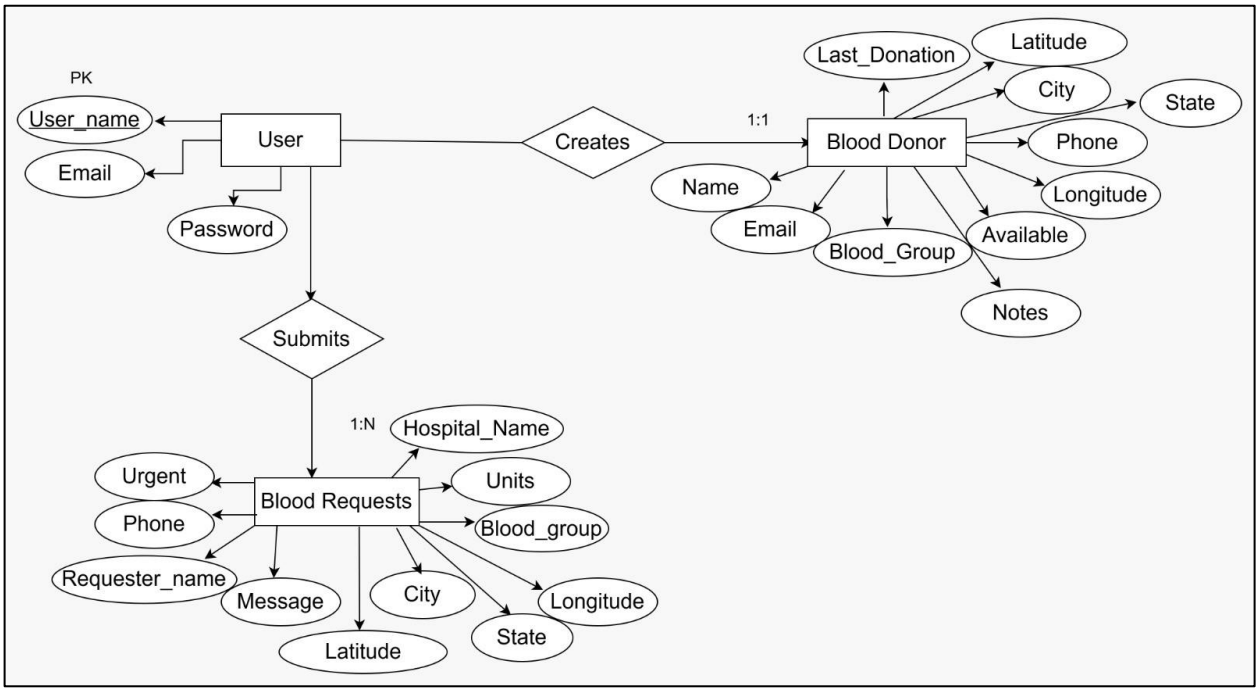
2. Data Integrity:

- Django ORM ensures consistent, validated entries.
- Only authenticated users can modify donor or request data.

3. Security:

- Passwords are hashed.
- CSRF protection is enabled for all forms.
- Admin access is restricted.

6. DATABASE DESIGN



Tables with attributes

Table Name	Attributes	Key Type	Description
USER	user_name,email, password	PK:user_name	Stores basic authentication details of the users. Each user can log in and access the system.
BLOOD DONOR	donor_id, name, email, blood_group, phone, city, state, latitude, longitude, last_donation, available, notes	PK:donor_id FK:email→USER.email (or username → USER)	Stores detailed donor information including personal data, blood group, availability, and last donation. Each USER can have one DONOR profile (1:1).
BLOOD REQUESTS	request_id, requester_name, phone, email, blood_group, units, hospital_name, city, state, latitude, longitude, urgent, message	PK: request_id FK:email→USER.email	Stores all blood request submissions made by users. One user can submit multiple blood requests (1:N).

Keys Used

The database schema of the Blood Donor Application uses both Primary Keys (PK) and Foreign Keys (FK) to maintain data integrity and define relationships between tables.

- Primary Keys (PK):

Each table has a unique primary key to identify its records:

- user_name in the USER table
- donor_id in the BLOOD DONOR table • request_id in the BLOOD REQUESTS table
- Foreign Keys (FK):

Foreign keys establish links between the related tables:

- email / user_name in BLOOD DONOR references USER table
- email / user_name in BLOOD REQUESTS references USER table

This ensures that donor profiles and blood requests are always linked to valid registered users.

Explanation for each table

1. USER Table

This table stores the basic authentication details of the users such as user_name, email, and password. It is used to allow users to sign up and log in to access the system. The user_name acts as the primary key.

2. BLOOD DONOR Table

This table stores detailed donor information including donor_id, name, email, blood_group, city, state, phone, latitude, longitude, last_donation, availability, and notes. The donor_id is the primary key and the email/user_name is a foreign key that links each donor to a specific user. Each user can have only one donor profile (1:1).

3. BLOOD REQUESTS Table

This table stores all blood request submissions made by users. It includes request_id, requester_name, phone, email, blood_group, units, hospital_name, city, state, latitude, longitude, urgent, and message. The request_id is the primary key, and the email/user_name acts as a foreign key referencing the USER table. One user can submit multiple blood requests (1:N).

7. IMPLEMENTATION

Models Overview

Models represent the core database structure of the Blood Donor Web Application.

The main models include:

a) Donor Model

Stores donor details such as name, blood group, contact number, city, state, availability, and last donation date. This model helps in filtering and fetching donor data efficiently. b) BloodRequest Model

Stores information about each blood request, including requester name, required blood group, city, hospital name, urgency flag, and message. This model maintains complete records of requests raised by users.

c) User Model

Django's built-in User model is used for authentication. It handles username, password (hashed), and email.

Purpose of Models:

- Maintain clean, structured data
- Ensure relations and validations
- Support all system operations through Django ORM

URL Routing Overview

URL routing directs the user's browser request to the appropriate view.

Common routes include:

- /login/ → Login page
- /signup/ → User registration
- /register-donor/ → Donor registration form
- /request-blood/ → Blood request form
- /donors/ → Donor list page
- /dashboard/ → Main dashboard

Purpose of Routing:

- Handles navigation
- Connects pages to backend logic
- Keeps the application modular and organized

Important Functionalities

a. Donor Registration: Allows donors to enter name, blood group, city, last donation date, and availability. The system validates fields before saving data.
b. Blood Request Submission: Recipients fill form with required blood group, city, hospital name, urgency, and message. Data is stored and used for matching with donors.
c. Donor Filtering: Filters donors by:

- Blood group
- City
- Availability status

Only matching donors are shown on the results page.

d) Authentication: Users can log in and out securely using Django's authentication system
e) Donor List Display: Displays donor details in table/card format for recipients to contact easily.

Special Logic

a) Availability Check: Donors with availability = "No" are automatically excluded from search results.

b) Last Donation Date Validation: Donors who donated recently (within 90 days) can be marked as temporarily unavailable.

c) Urgent Requests: If a request is marked "urgent," it is highlighted for priority viewing.

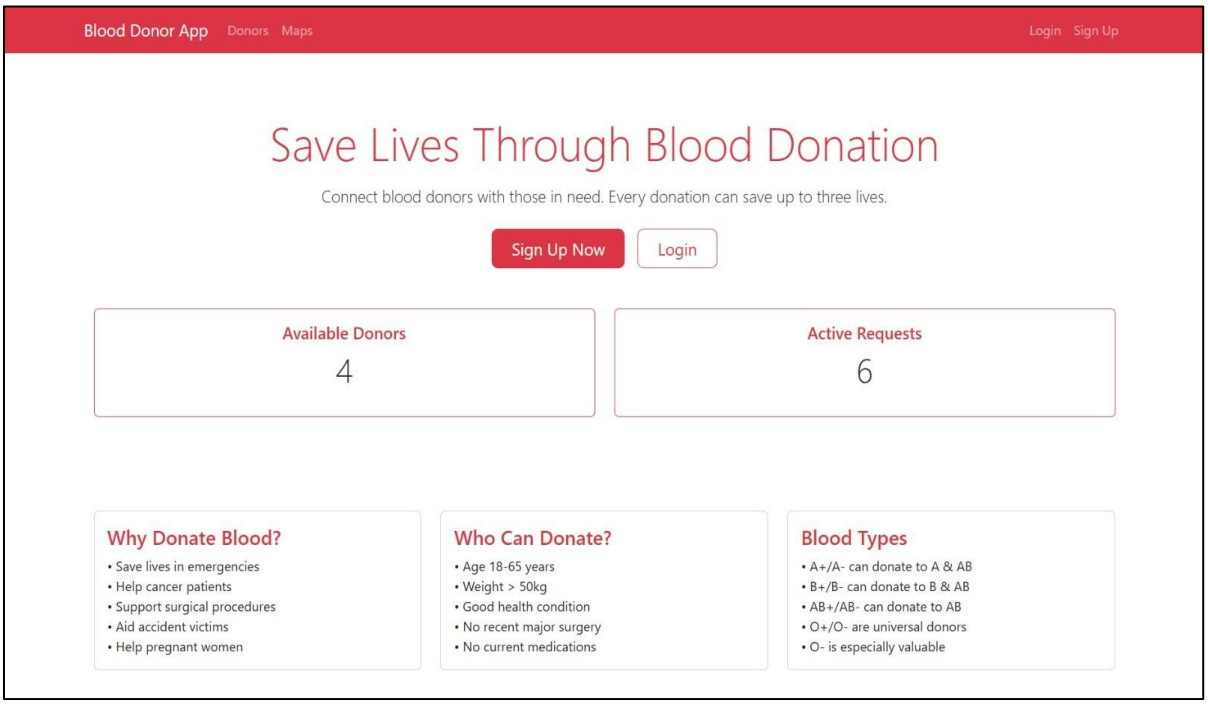
d) Form Validation: The system checks for:

- Correct blood group format
- Valid phone number
- Required fields filled

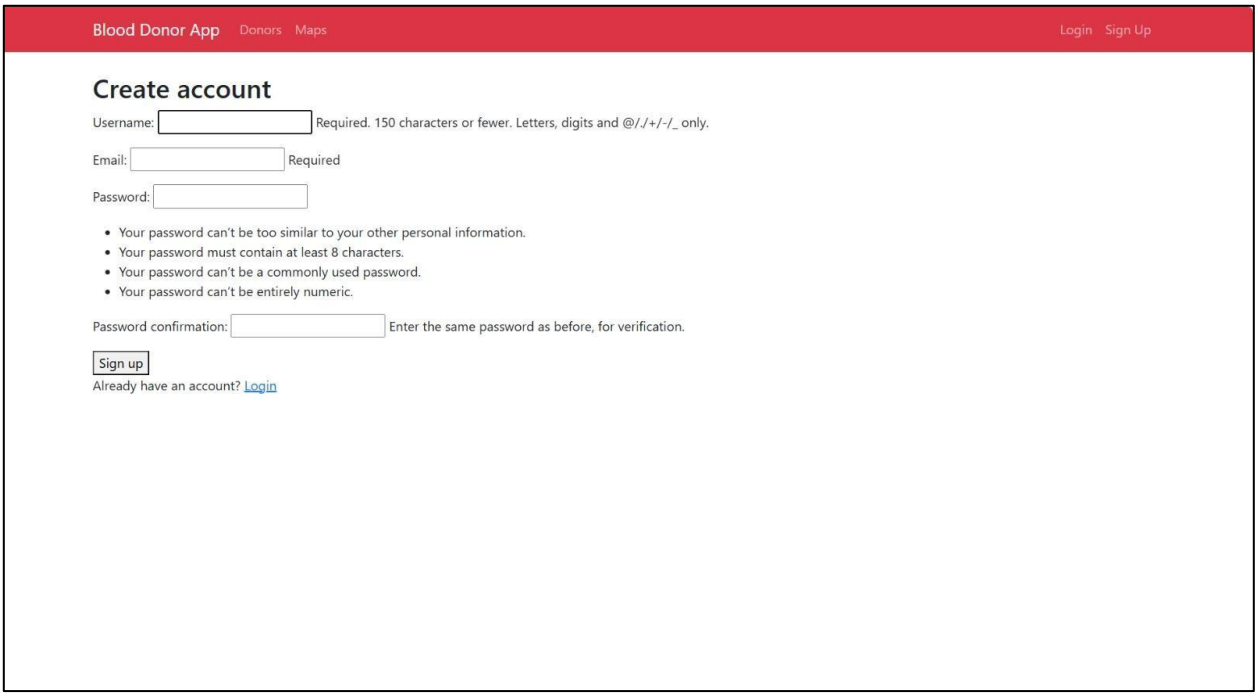
8. SCREENSHOTS

Include:

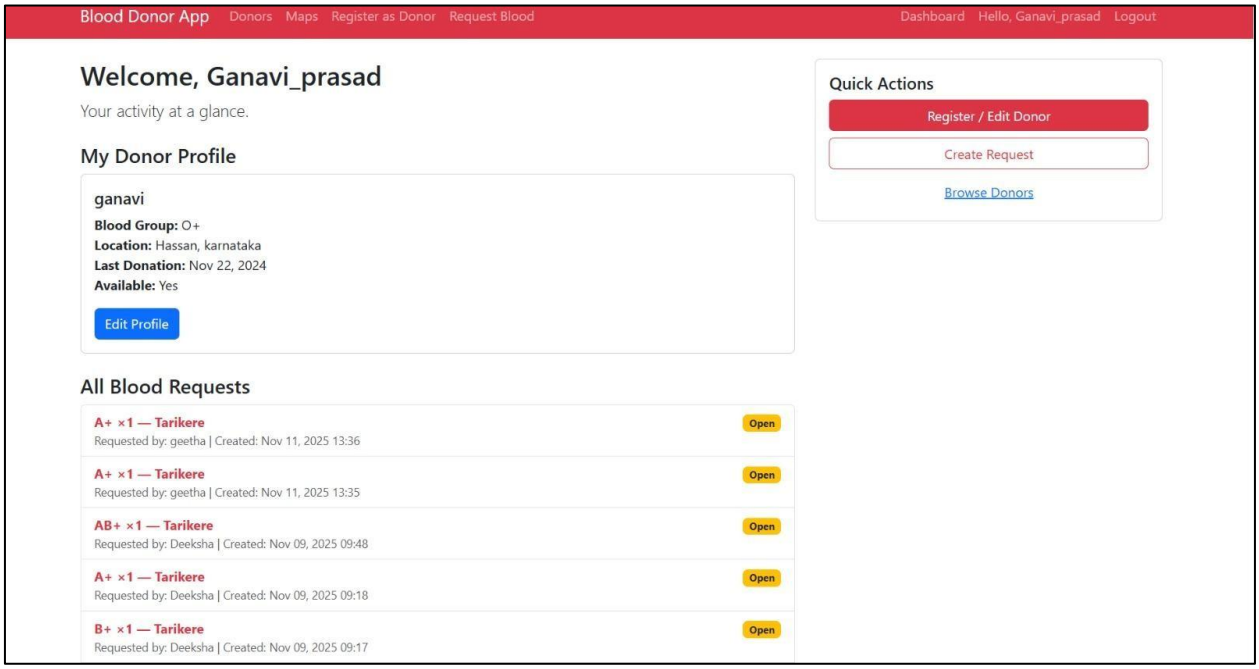
- Homepage

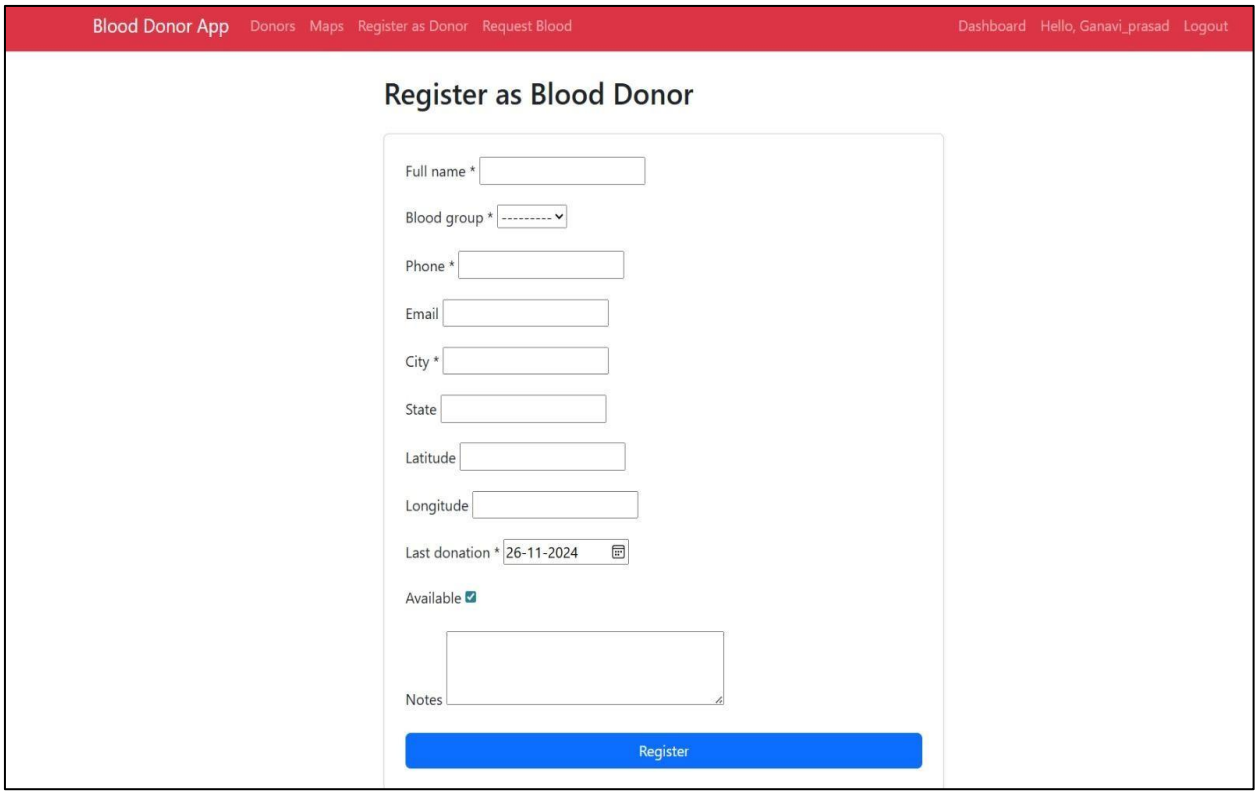


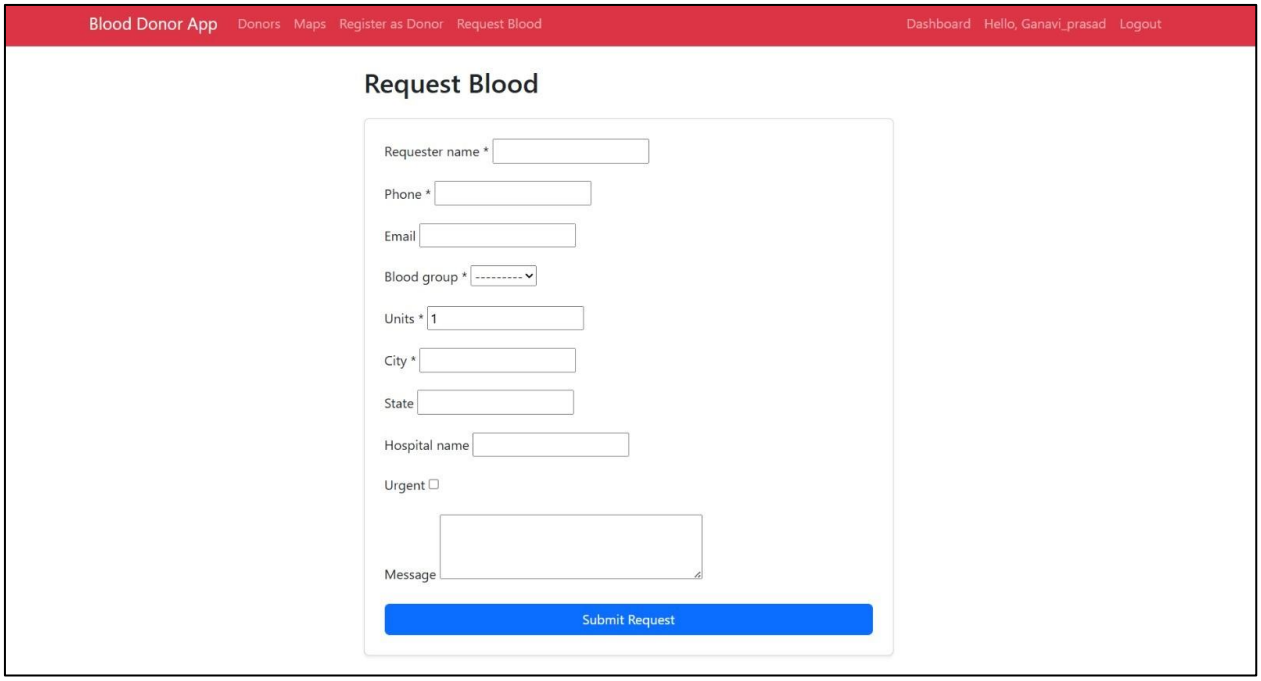
- Login/Signup



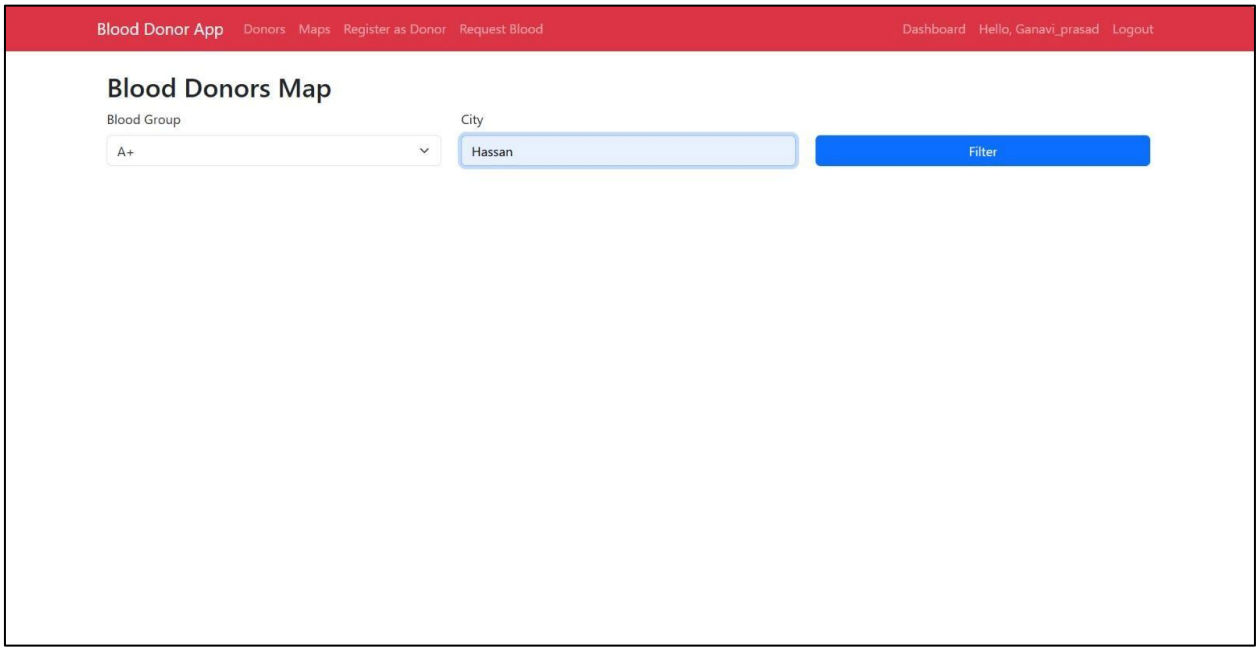
- Important pages

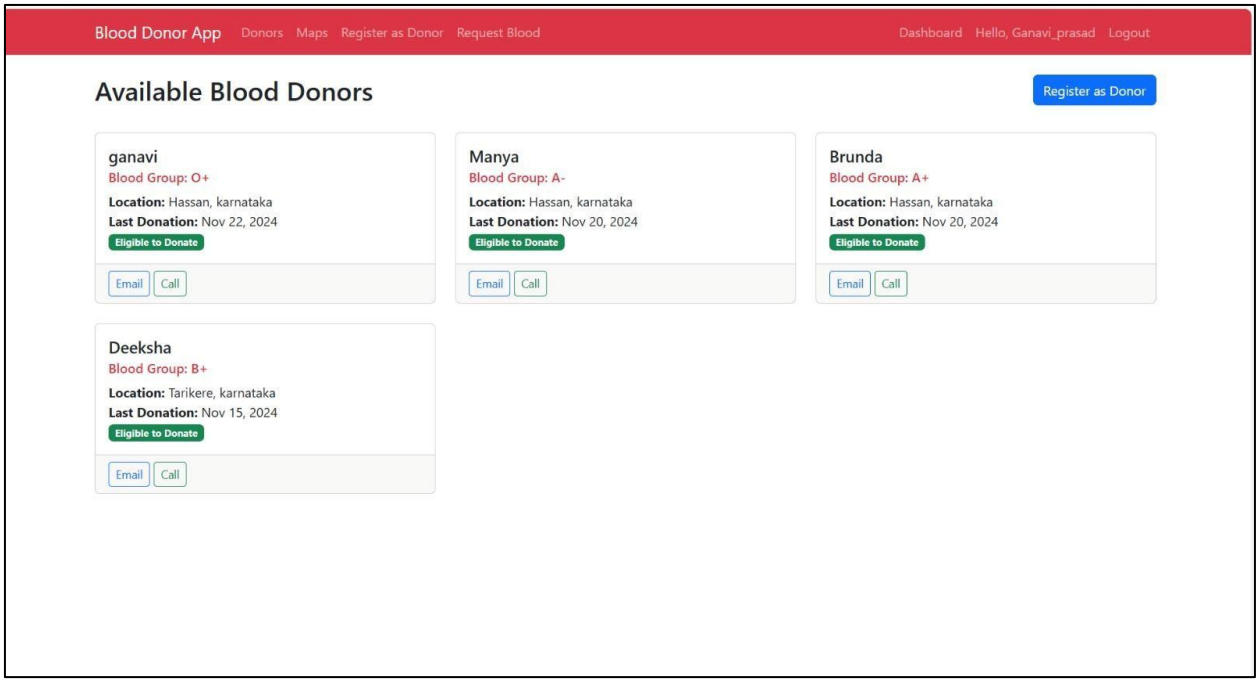

Blood Donor App
Donors
Maps
Register as Donor
Request Blood
Dashboard
Hello, Ganavi_prasad
Logout
Welcome, Ganavi_prasad
Your activity at a glance.
My Donor Profile
ganavi
Blood Group: O+
Location: Hassan, karnataka
Last Donation: Nov 22, 2024
Available: Yes
Edit Profile
Quick Actions
Register / Edit Donor
Create Request
Browse Donors
All Blood Requests
A+ ×1 — Tarikere
Requested by: geetha | Created: Nov 11, 2025 13:36
Open
A+ ×1 — Tarikere
Requested by: geetha | Created: Nov 11, 2025 13:35
Open
AB+ ×1 — Tarikere
Requested by: Deeksha | Created: Nov 09, 2025 09:48
Open
A+ ×1 — Tarikere
Requested by: Deeksha | Created: Nov 09, 2025 09:18
Open
B+ ×1 — Tarikere
Requested by: Deeksha | Created: Nov 09, 2025 09:17
Open


Blood Donor App
Donors
Maps
Register as Donor
Request Blood
Dashboard
Hello, Ganavi_prasad
Logout
Register as Blood Donor
Full name *
Blood group * ---------
Phone *
Email
City *
State
Latitude
Longitude
Last donation * 26-11-2024
Available
Notes
Register

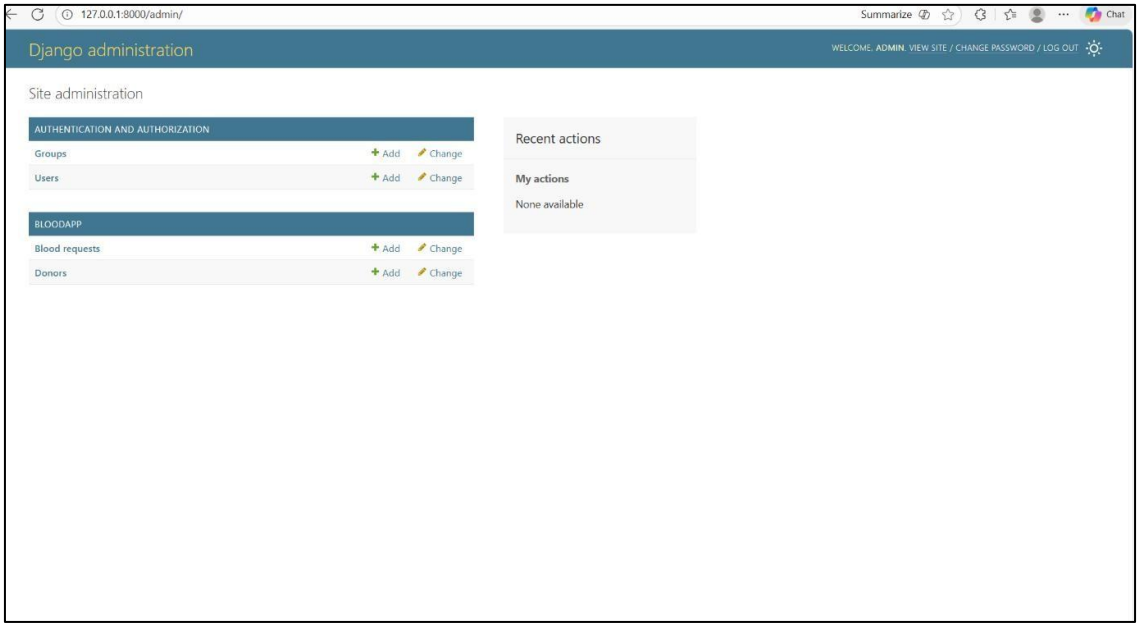

Blood Donor App
Donors
Maps
Register as Donor
Request Blood
Dashboard
Hello, Ganavi_prasad
Logout
Request Blood
Requester name *
Phone *
Email
Blood group *

Units *
1
City *
State
Hospital name
Urgent
Message
Submit Request


Blood Donor App
Donors
Maps
Register as Donor
Request Blood
Dashboard
Hello, Ganavi_prasad
Logout
Blood Donors Map
Blood Group
A+
City
Hassan
Filter



- Admin panel



9. TESTING

Basic test cases carried out to ensure correct functioning of the Blood Donor Application include:

- Form validation
- Login

- Donor registration
- Blood request creation
- Filtering donors by blood group and city
- CRUD operations on donor/request records
- Error handling for invalid inputs
- Authentication restriction for protected pages

10. RESULTS

The Blood Donor Application successfully meets the main objectives of the project. The system allows users to register as donors, submit blood requests, and search for available donors based on blood group and city. All major features were tested and found to be functioning correctly, including authentication, form validation, donor filtering, and request management.

The platform provides a clear dashboard displaying active requests and donor availability, helping users quickly find suitable matches during emergencies. Overall, the project improves accessibility, reduces manual searching efforts, and offers a structured way to manage blood donation information.

11. CONCLUSION

The project demonstrates how full-stack development can be effectively used to build a practical and socially useful application. The Blood Donor App provides a reliable and user-friendly interface for both donors and recipients, ensuring accurate data handling and easy access to essential information.

By simplifying donor registration and blood request processes, the system helps bridge the gap between donors and people in urgent need. The project achieves its intended goals and showcases the importance of digital platforms in supporting critical healthcare requirements.

12. FUTURE ENHANCEMENTS

Possible improvements include:

- Adding SMS/Email notifications to alert donors instantly
- Integrating live map tracking for nearer donors

- Adding hospital verification system
- Implementing OTP-based user verification
- Creating a mobile app version for better accessibility

13. REFERENCES

- Django Documentation
- Python Official Documentation
- Bootstrap Framework Docs
- Online tutorials (StackOverflow)
- Research articles on blood donation system