PULSE CHAIN GIKI

A Secure and Intelligent Blood Donation Platform for Campus

Group Members:

Nabiha Nasir (514)

M Hamdoon (387)

Ahsan Naseer (314)

Irtaza Alyan (246)

Supervisor:

Laraib Afzal

Course:

Software Engineering

INTRODUCTION:

Pulse Chain is a centralized, campus-based blood donation management system designed to address delays, lack of coordination, and manual processes in traditional blood donation practices. The system provides a unified digital platform connecting **donors, recipients, hospitals, and administrators** within the campus community. It enables users to register as donors, request blood, search for compatible donors, and track donation activities. Dedicated dashboards for hospitals and administrators support monitoring, verification, and efficient coordination of blood donation operations. This project aims to enhance emergency response, improve transparency, and promote a reliable blood donation culture within the campus.

PRODUCT SCOPE:

Pulse Chain is a **web-based blood donation system** designed to support rapid and efficient blood donation within a campus. The system offers **role-based dashboards** for donors, recipients, hospitals, and administrators, enabling streamlined blood requests, donor management, and record tracking. It improves communication, reduces delays during emergencies, and ensures accurate management of blood-related data.

Name	Description
Donor	A user who has registered to donate blood.
Requester	A user who submits request for blood.
Admin	A person who manages the system with full privileges.
Hospital	Elevated access for verifying availability.
Dashboard	Interface shows important stats and actions.
Availability	Determines whether a donor is eligible to donate at a given time.

OVERVIEW:

DESCRIPTION

The system is a web-based blood donation platform consisting of separate modules for users, hospitals, and administrators. It supports user registration, login, donor enrollment, and activity tracking. The hospital module enables monitoring of donor availability, while the admin module oversees the entire system by managing donor records, blood requests, and overall operations.

PRODUCT PERSEPECTIVE

The product is a web-based blood donation system that enables seamless interactions across multiple components. It integrates key functionalities including user authentication, donor management, blood requests, and activity tracking. All modules share a centralized database that securely stores donor profiles, availability statuses, request histories, and statistics.

Modules

1. **User Module** – For donor registration, profile management, and activity tracking.
2. **Hospital Module** – To monitor donor availability and manage blood requests.
3. **Admin Module** – For overseeing the entire system, including donor lists, requests, and system operations.
4. **Search & Availability System** – Enables efficient searching of donors and real-time availability checks.
5. **History Tracking System** – Maintains detailed records of donations, requests, and system activity.

PRODUCT FUNCTION:

1:User Dashboard Functions

- Login and authentication.
- Register as a donor with blood group and availability.
- Request blood with required details.
- View recent activity and complete history of donations/requests.
- Dashboard counters for total donations and requests.
- Search for available donors.
- Update personal profile information.

2:Hospital Dashboard Functions

- View all donors and their availability.
- Mark donors as available or unavailable.
- Monitor user-submitted blood requests.
- Update request status (Approved / Pending / Rejected).
- Refresh data to get updated donor statuses and requests.

3:Admin Dashboard Functions

- View overall statistics: total donors, donations, active requests, and available donors.
- Manage donor and request records (edit, update, delete).
- Synchronize system-wide data and monitor logs/statistics.

User Characteristics

The system supports three types of users:

1. General Users

- Basic computer skills required.
- Ability to log in, register, and navigate dashboards.
- Includes students, staff, or citizens interested in donating blood.

2. Hospital Users

- Familiarity with hospital operations.
- Manage donor lists and update availability.
- Regularly monitor blood requests and donor status.

3. Administrators

- Experienced in system management and moderation.
- Monitor user activities and ensure data accuracy.
- Comfortable using advanced dashboard features.

Constraints

- A stable internet connection is required.
- All data must be verified to ensure accurate donor availability.
- Donor information relies on user honesty regarding blood group and availability.
- Each user account must be properly authenticated.
- Privacy and security regulations must be strictly followed.
- System functionalities depend on real-time server updates.

Assumptions and Dependencies

- Users provide accurate information during registration.
- Hospitals and administrators maintain correct donor availability status.
- Database and backend services are fully functional.
- Web browsers must be compatible with the system.

- Emergency blood requests are assumed to be handled promptly by administrators and hospitals.

STATE OF ART:

Traditional blood donation systems rely on paper-based or fragmented digital processes, often lacking **real-time tracking and efficient communication**. Existing solutions rarely provide unified platforms for donors, hospitals, and administrators, which can delay responses during emergencies. **Pulse Chain – GIKI** addresses these gaps with a **web-based, campus-focused system** featuring role-based dashboards, real-time donor availability, search functionality, and comprehensive history tracking, ensuring faster coordination and accurate record management.

LITERATURE REVIEW:

Project Name	**Key Features**	**Limitations**	**How Pulse Chain GIKI Improves**
Blood Bank and Donor Management System (BBDMS)	Automates donor registration, inventory tracking, and testing for general blood banks using centralized databases.	Relies on traditional hospital workflows without role-based dashboards or real-time campus-specific analytics; focuses on broad operations rather than university emergencies.	Adds four tailored dashboards with analytics for donors, recipients, hospitals, and admins, enabling campus-wide coordination and data visualization absent here
Location-based Mobile App for Blood Donor Search	Uses Google Maps and Scrum for real-time donor location between seekers and centers, boosting donor numbers by 39%.	Limited to mobile donor search and communication; no inventory management, hospital integration, or multi-dashboard analytics for stakeholders.	Incorporates geolocation within comprehensive dashboards and analytics, unlike this app's narrow focus, for full-cycle campus tracking from request to fulfillment
Online Blood Donation Management System	Client-server setup for donor registration, real-time inventory, hospital requests, and privacy-focused distribution.	Generic national scope without campus customization or advanced recipient/admin dashboards; emphasizes supply-demand balancing over user-specific analytics.	Provides GIKI-specific multi-role dashboards with analytics, overcoming OBDM's lack of targeted university features and stakeholder transparency .

SU Blood Donor Android App	Stores donor details (name, blood group, location) for quick access in blood collection centers; supports emergency donor replies.	Basic donor database without hospital/admin dashboards, analytics, or recipient tracking; assumes real-time access but lacks structured multi-user coordination.	Delivers analytics-driven dashboards for all roles, far beyond this app's simple storage, ensuring verified campus interactions and system oversight

4: EXTERNAL INTERFACES REQUIREMENTS:

USER INTERFACES

- **User Dashboard:** Allows multiple users to perform actions such as login, registration, blood requests, and profile updates.
- **Hospital Dashboard:** Enables hospital staff to manage donor availability and view/update donor statuses.
- **Admin Dashboard:** Provides administrators with tools to manage donor records, monitor system statistics, and synchronize data.
- **Responsiveness:** The interface must be functional across various devices (desktop, laptop, tablet, smartphone) and browsers.
- **Performance:** The website should load within 2–3 seconds on modern browsers and devices.

HARDWARE INTERFACES

- Compatible devices include desktops, laptops, tablets, and smartphones for all user roles (General Users, Hospital Users, and Administrators).
- A stable internet connection is required for system functionality.

SOFTWARE INTERFACES

- The primary interface is a web browser.
- All modules, including authentication, donor management, and request tracking, must be integrated and use a shared database for storing profiles, statuses, and history.
- External services may be required for additional features such as email notifications.

COMMUNCATION INTERFACES

- **Web-Based Interaction:** All communication between modules and users occurs via the web interface.
- **Real-Time Data Synchronization:** The backend supports instant updates to ensure donor availability, request status, and other relevant information are reflected across all dashboards in real time.
- **User Communication:** System features, such as the donor search functionality, facilitate communication between donors and requesters.
- **Notifications:** Alerts for blood requests and donor status updates are delivered reliably and promptly to relevant users.

FUNCTIONAL REQUIREMENTS:

Donor / Requester Dashboard Requirements

- **FR1:** The system should allow the user to register a new account.
- **FR2:** The system should allow the user to log in using valid credentials.
- **FR3:** The system should allow the user to log out securely.
- **FR4:** The system should allow the user to view their profile information including name, email, role, and membership date.
- **FR5:** The system should allow the user to edit and update their profile details.
- **FR6:** The system should allow the user to register as a blood donor.
- **FR7:** The system should allow the user to record blood donations.
- **FR8:** The system should allow the user to view their donation history.
- **FR9:** The system should allow the user to submit a blood request.
- **FR10:** The system should allow the user to search for available donors in their area.
- **FR11:** The system should allow the user to view their blood request history.
- **FR12:** The system should allow the user to track the number of blood requests made.
- **FR13:** The system should allow the user to view recent activity including donations and requests.
- **FR14:** The system should allow the user to view overall donation impact statistics.

Hospital Dashboard Requirements

- **FR15:** The system should allow hospital users to view donation statistics.
- **FR16:** The system should allow hospital users to view blood request statistics.
- **FR17:** The system should allow hospital users to check current blood availability by type.

Admin Dashboard Requirements

- **FR18:** The system should allow admin to manage different donor records.
- **FR19:** The system should allow admin to monitor system statistics.
- **FR20:** The system should allow admin to synchronize data across all dashboards.

Requirement ID	**FR-1**		**Requirement Type**		Functional		**Use Case #**		UC-01
Status	***New***		***Agreed-to***	**-**	***Baselined***	**-**	***Rejected***	**-**	
Parent Requirement #	None								
Description	This requirement defines the system's ability to create a new user profile by collecting basic information such as name, email, password, and role. The registration process must validate user inputs, prevent duplicate accounts, and securely store credentials**.**								
Rationale	Without a registration feature, users cannot access personalized functions such as donation history, blood requests, or donor registration. This is the foundational entry point for all system operations.								
Source	Stakeholder discussions and core system objectives.				**Source Document**	Initial System Requirements Specification (SRS) draft.			
Acceptance/Fit Criteria	The system must allow a user to create an account with valid information, reject invalid or duplicate email entries with clear messages, and confirm successful registration by creating the user profile in the database and redirecting them to the login page.								
Dependencies	The user registration module depends on the database system to store new user records.								
Priority	***Essential***		***Conditional***	**-**	***Optional***	**-**			
Change History									

Requirement ID	**FR-2**		**Requirement Type**		Functional		**Use Case #**		UC-01
Status	***New***		***Agreed-to***	**-**	***Baselined***	**-**	***Rejected***	**-**	
Parent Requirement #	None								
Description	This requirement covers the system's ability to let a registered user log in using their email and password. The system must verify the provided credentials, confirm that the account exists, and authenticate the user before giving access to protected features.								
Rationale	Login is necessary to maintain controlled access. It prevents unauthorized use and keeps user-specific data like donation history, requests, and personal details protected from misuse.								

Source	Discussions with stakeholders regarding access security and user data protection.			**Source Document**	The initial SRS draft compiled during the requirement-gathering phase.	
Acceptance/Fit Criteria	The system must correctly authenticate valid login attempts, reject incorrect or unregistered credentials with clear error messages, and take the user to their dashboard immediately after successful login.					
Dependencies	The login feature depends on the authentication component responsible for verifying stored user credentials.					
Priority	***Essential***		***Conditional***	-	***Optional***	-
Change History						

Requirement ID	**FR-3**		**Requirement Type**		Functional	**Use Case #**		UC-01
Status	***New***		***Agreed-to***	-	***Baselined***	-	***Rejected***	-
Parent Requirement #	None							
Description	This requirement defines the system's ability to log out a currently authenticated user. The system should end the active session, clear any temporary authentication data, and return the user to a safe public page so the account cannot be accessed without logging in again.							
Rationale	A secure logout process prevents unauthorized access when a user leaves their device unattended. It ensures that personal information, donation records, and request data remain protected once the user finishes using the system.							
Source	Stakeholder discussions emphasizing session			**Source Document**	The SRS draft was compiled during the requirement.			
Acceptance/Fit Criteria	The system must successfully terminate the user's session, prevent access to any protected page after logout, and redirect the user to the login screen or homepage without leaving behind cached or active session data.							
Dependencies	The logout feature depends on the session management component responsible for creating and ending user sessions.							
Priority	***Essential***		***Conditional***	-	***Optional***	-		
Change History								

Requirement ID	**FR-4**		**Requirement Type**		Functional	**Use Case #**		UC-01
Status	***New***		***Agreed-to***	-	***Baselined***	-	***Rejected***	-
Parent Requirement #	None							
Description	Users can access a dedicated profile page displaying personal details for verification and updates.							

Rationale	Viewing profile information increases transparency and allows users to confirm their registered details are correct.						
Source	Inferred From Scope		**Source Document**	The SRS draft compiled during the requirement.			
Acceptance/Fit Criteria	Profile information displays correctly, is updated in real time, and accurately reflects the database records.						
Dependencies	Depends on FR1 to have profile data available for display.						
Priority	***Essential***		***Conditional***	-	***Optional***	-	
Change History							

Requirement ID	**FR-5**		**Requirement Type**		Functional		**Use Case #**		UC-01
Status	***New***		***Agreed-to***	-	***Baselined***	-	***Rejected***	-	
Parent Requirement #	None								
Description	Users can update information like email, password, or contact number via a profile management interface.								
Rationale	Personal data may change over time; users must be able to correct or update it.								
Source	Feedback from initial testing and stakeholder recommendations. inferred From Scope				**Source Document**		SRS v1.0, Section 2.5		
Acceptance/Fit Criteria	Changes are saved successfully, confirmation is displayed, and updated info is shown immediately.								
Dependencies	Relies on profile update service and validation logic.								
Priority	***Essential***		***Conditional***	-	***Optional***	-			
Change History									

Requirement ID	**FR-6**		**Requirement Type**		Functional		**Use Case #**		UC-01
Status	***New***		***Agreed-to***	-	***Baselined***	-	***Rejected***	-	
Parent Requirement #	None								
Description	Users can voluntarily provide their blood type, health information, and consent to appear in the donor registry.								
Rationale	Necessary to maintain an active, searchable list of eligible donors for emergencies.								
Source	Healthcare partner requirements and domain research				**Source Document**		SRS v1.0, Section 2.6		
Acceptance/Fit Criteria	Registration confirmation is received, donor is visible in the donor database, and validation of blood type is enforced.								

Dependencies	Requires donor management module and access to user profile data.						
Priority	***Essential***		***Conditional***	-	***Optional***	-	
Change History							

Requirement ID	**FR-7**	**Requirement Type**		Functional		**Use Case #**		UC-01	
Status	***New***		***Agreed-to***	-	***Baselined***	-	***Rejected***	-	
Parent Requirement #	None								
Description	The system must allow donors to log a new donation event, capturing details such as the date of donation, the volume donated, and the location or hospital where the procedure took place.								
Rationale	Tracking donation dates is medically mandatory to ensure donors do not donate again before the safe waiting period passes, protecting the health of the donor.								
Source	Medical Guidelines		**Source Document**		Donor Policy Doc				
Acceptance/Fit Criteria	System records date and volume (optional). Next eligible date is calculated automatically.								
Dependencies	FR6 (Donor Registration)								
Priority	***Essential***		***Conditional***	-	***Optional***	-			
Change History									

Requirement ID	**FR-8**	**Requirement Type**		Functional		**Use Case #**		UC-01	
Status	***New***		***Agreed-to***	-	***Baselined***	-	***Rejected***	-	
Parent Requirement #	None								
Description	The system should provide users with a chronological list of all their past blood donations, displaying relevant details like dates and locations for personal record-keeping.								
Rationale	Viewing history serves as a personal record for the user and acts as a motivational tool, showing them the consistent impact they have made over time through their donations.								
Source	User Experience		**Source Document**		UI Mockups v1				
Acceptance/Fit Criteria	The system retrieves all donation records associated with the logged-in user ID and displays them in a list sorted by date (newest first). If no history exists, a friendly placeholder message indicating 'No donations yet' should be displayed.								
Dependencies	FR7: Record blood donations								
Priority	***Essential***		***Conditional***	-	***Optional***	-			
Change History									

Requirement ID	**FR-9**	**Requirement Type**		Functional		**Use Case #**		UC-01	
Status	***New***		***Agreed-to***	-	***Baselined***	-	***Rejected***	-	

Parent Requirement #	None						
Description	The system must allow users to generate a formal request for blood by specifying the required blood group, the number of units needed, the hospital location, and the urgency level.						
Rationale	This feature is the primary utility for recipients. It ensures that structured, actionable information reaches potential donors or the admin, facilitating a quick response to medical emergencies.						
Source	Core Business Logics	**Source Document**	SRS Initial Draft v1.0				
Acceptance/Fit Criteria	The request form must capture all mandatory fields including blood type, units, and location. Upon submission, the system creates a new request record with a unique ID and sets its initial status to 'Open' or 'Pending' for donors to see.						
Dependencies	FR2: Log in using valid credentials						
Priority	***Essential***		***Conditional***	-	***Optional***	-	
Change History							

Requirement ID	**FR-10**	**Requirement Type**	Functional	**Use Case #**	UC-01				
Status	***New***		***Agreed-to***	-	***Baselined***	-	***Rejected***	-	
Parent Requirement #	None								
Description	The system should provide a search interface that allows users to filter the donor database by specific blood groups and geographical areas (city or neighborhood) to find suitable matches.								
Rationale	In emergency situations, finding a donor nearby is time-sensitive. This search capability connects requesters directly with relevant donors, significantly reducing the time to secure blood.								
Source	User Requirement	**Source Document**	SRS Initial Draft v1.0						
Acceptance/Fit Criteria	The search algorithm correctly filters the user database to return only those users who are registered donors matching the selected blood group and city. The results should display the donor's name and contact option clearly.								
Dependencies	FR6: Register as a blood donor								
Priority	***Essential***		***Conditional***	-	***Optional***	-			
Change History									

Requirement ID	**FR-11**	**Requirement Type**	Functional	**Use Case #**	UC-01				
Status	***New***		***Agreed-to***	-	***Baselined***	-	***Rejected***	-	
Parent Requirement #	None								
Description	The system must allow users to access a log of all previous blood requests they have submitted, showing details of the request and its current fulfillment status (e.g., Pending, Completed).								
Rationale	This helps users track the progress of their active requests and maintain a record of past medical needs, which facilitates better management of ongoing or future requirements..								

Source	User Feedback			**Source Document**	UI Mockups v1				
Acceptance/Fit Criteria	The page must list all requests initiated by the current user, displaying key details like date, blood type, and status. The status indicator should visually distinguish between 'Fulfilled' (e.g., green) and 'Pending' (e.g., yellow) requests.								
Dependencies	FR10: Submit a blood request								
Priority	***Essential***		***Conditional***	-	***Optional***	-			
Change History									

Requirement ID	**FR-12**		**Requirement Type**		Functional		**Use Case #**		UC-01
Status	***New***		***Agreed-to***	-	***Baselined***	-	***Rejected***	-	
Parent Requirement #	None								
Description	The system should feature a dashboard counter or metric that automatically calculates and displays the total number of blood requests initiated by the specific user account.								
Rationale	This provides immediate visual feedback to the user about their usage of the system and helps the system administrators monitor usage patterns to prevent potential spam or misuse.								
Source	System Analytics			**Source Document**	SRS Initial Draft v1.0				
Acceptance/Fit Criteria	The system performs a count of all records in the requests table linked to the user's ID. This number is displayed on the dashboard and must increment instantly whenever the user successfully submits a new blood request.								
Dependencies	FR10: Submit a blood request								
Priority	***Essential***		***Conditional***	-	***Optional***	-			
Change History									

Requirement ID	**FR-13**		**Requirement Type**		Functional		**Use Case #**		UC-01
Status	***New***		***Agreed-to***	-	***Baselined***	-	***Rejected***	-	
Parent Requirement #	None								
Description	The system must allow hospital or admin users to view aggregated statistical data regarding total donations received over specific time periods (e.g., weekly or monthly reports).								
Rationale	Hospitals need this high-level data to analyze donor engagement trends, manage inventory expectations, and plan future blood drive campaigns effectively based on supply history.								
Source	Hospital Admin			**Source Document**	Admin Panel Specs				
Acceptance/Fit Criteria	The analytics module must correctly query the donation database to aggregate counts based on the selected date range. The data should be rendered as a bar chart or line graph, accurately reflecting the total volume or number of donors.								
Dependencies	FR7: Record blood donations								
Priority	***Essential***		***Conditional***	-	***Optional***	-			
Change History									

Requirement ID	**FR-14**	**Requirement Type**	Functional	**Use Case #**	UC-01				
Status	***New***		***Agreed-to***	**-**	***Baselined***	**-**	***Rejected***	**-**	
Parent Requirement #	None								
Description	The system should enable hospital administrators to visualize data concerning the demand for different blood types, highlighting which groups are most frequently requested.								
Rationale	Understanding demand patterns allows hospitals to prepare for shortages. If a specific blood type is in high demand, they can issue targeted appeals to donors of that group.								
Source	Hospital Admin	**Source Document**	Admin Panel Specs						
Acceptance/Fit Criteria	The system aggregates all request data by blood group (A, B, O, AB) and calculates the percentage of total demand for each. This data must be presented visually (e.g., a pie chart) to clearly identify high-demand blood types.								
Dependencies	FR10: Submit a blood request								
Priority	***Essential***		***Conditional***	**-**	***Optional***	**-**			
Change History									

Requirement ID	**FR-15**	**Requirement Type**	Functional	**Use Case #**	UC-01				
Status	***New***		***Agreed-to***	**-**	***Baselined***	**-**	***Rejected***	**-**	
Parent Requirement #	None								
Description	The system must provide a real-time inventory view for hospital staff, categorizing available blood stock by blood group (A+, B-, etc.) to show current supply levels instantly.								
Rationale	This is critical for operational efficiency in trauma centers. Doctors and staff need to know immediately if the required blood type is available in stock before initiating procedures.								
Source	Medical Requirement	**Source Document**	Inventory Module Specs						
Acceptance/Fit Criteria	The system calculates the net stock (Total Donations minus Total Usage) for each blood group. The dashboard displays these values in real-time, with visual indicators (e.g., red text) highlighting any blood types that have fallen below a predefined threshold.								
Dependencies	FR7: Record blood donations AND FR10: Submit a blood request								
Priority	***Essential***		***Conditional***	**-**	***Optional***	**-**			
Change History									

Requirement ID	**FR-16**	**Requirement Type**	Functional	**Use Case #**	UC-01				
Status	***New***		***Agreed-to***	**-**	***Baselined***	**-**	***Rejected***	**-**	
Parent Requirement #	None								
Description	The system should display a dynamic "Recent Activity" feed on the user dashboard, summarizing the latest actions such as recent donations made or updates on submitted requests.								

Rationale	This improves the User Experience (UX) by providing a quick snapshot of the most relevant recent events, saving the user time from navigating to separate history pages to check updates.						
Source	UX Design Team	**Source Document**	Dashboard Wireframe				
Acceptance/Fit Criteria	The system retrieves the 5 most recent timestamps from both the donation and request logs. These events are merged, sorted chronologically, and displayed on the dashboard with clear descriptions (e.g., "You donated blood on [Date]").						
Dependencies	FR7: Record blood donations AND FR10: Submit a blood request						
Priority	***Essential***		***Conditional***	-	***Optional***	-	
Change History							

Requirement ID	**FR-17**		**Requirement Type**		Functional		**Use Case #**		UC-01
Status	***New***		***Agreed-to***	-	***Baselined***	-	***Rejected***	-	
Parent Requirement #	None								
Description	The system must calculate and display an impact summary for the donor, such as "Total Liters Donated" or an estimated "Lives Saved," based on their donation history.								
Rationale	Showing the tangible impact of their actions serves as a powerful psychological reward and gamification element, encouraging donors to maintain a regular donation schedule.								
Source	Marketing / Engagement		**Source Document**		User Retention Strategy				
Acceptance/Fit Criteria	The system applies a predefined formula (e.g., 1 unit saves 3 lives) to the user's total donation count. The result is calculated dynamically and displayed as a prominent badge or counter on the user's profile to incentivize further donations.								
Dependencies	FR7: Record blood donations								
Priority	***Essential***		***Conditional***	-	***Optional***	-			
Change History									

Non-Functional Requirements & System Attributes

- **Availability:** The system shall maintain an uptime of **99% or higher**.
- **Performance:** User requests shall be processed within **2 seconds** under normal load.
- **Data Security:** All user data and blood donation records shall be **encrypted** and stored securely.
- **Access Control:** The system shall enforce **strict authentication and authorization** mechanisms.
- **Usability:** The user interface shall be **responsive** and functional across multiple devices and browsers.
- **Scalability:** The design shall **support growth**, accommodating an increasing number of users and transactions.

- **Data Integrity & Recovery:** The system shall **ensure data integrity** and support **backup and recovery** in case of failures.
- **Reliability:** Notifications for blood requests and donor status shall be **timely and dependable**.
- **Compliance:** The system shall **adhere to relevant health regulations and privacy laws**.

PERFORMANCE REQUIREMENTS

1 **Real-Time Data Updates**

- The system shall reflect donor availability and blood request status **instantly** across all dashboards (user, hospital, admin).
- Hospitals and admins shall see **live updates** when users register, donate, or submit requests.

2 **Fast Search and Retrieval**

- The donor search feature shall return results **within seconds**, even under high load.
- Blood request history and donation logs shall be **quickly accessible** from the user dashboard.

3 **Concurrent User Handling**

- The system shall support **multiple users accessing and updating data simultaneously** without delays or data conflicts.
- Admin and hospital dashboards shall remain **responsive** during peak usage.

4 **Low Latency for Critical Actions**

- Actions such as submitting a blood request or updating donor status shall be completed with **minimal latency** to ensure timely response in emergencies.

5 **Efficient Dashboard Refresh**

- Dashboards shall **refresh and sync data seamlessly**, with **no noticeable lag**, particularly for hospital staff monitoring blood availability.

6 **Scalability**

- The system shall **scale efficiently** to handle increasing numbers of users, donors, and hospitals **without performance degradation**.

4+1 ARCHITECTURE VIEW MODEL

LOGICAL VIEW (CLASS DAIGRAM)

This view represents the object-oriented structure and functional requirements, focusing class hierarchies and interactions.

Key Classes:

- **User (Abstract):** Base entity containing authentication details (Name, Email, Role).
- **Donor:** Extends User; stores Blood Group, Availability, and Impact Score.
- **Hospital/Admin:** Extends User; manages inventory and system oversight.
- **Blood Request:** Encapsulates urgency, blood type, and status (Pending/Approved).

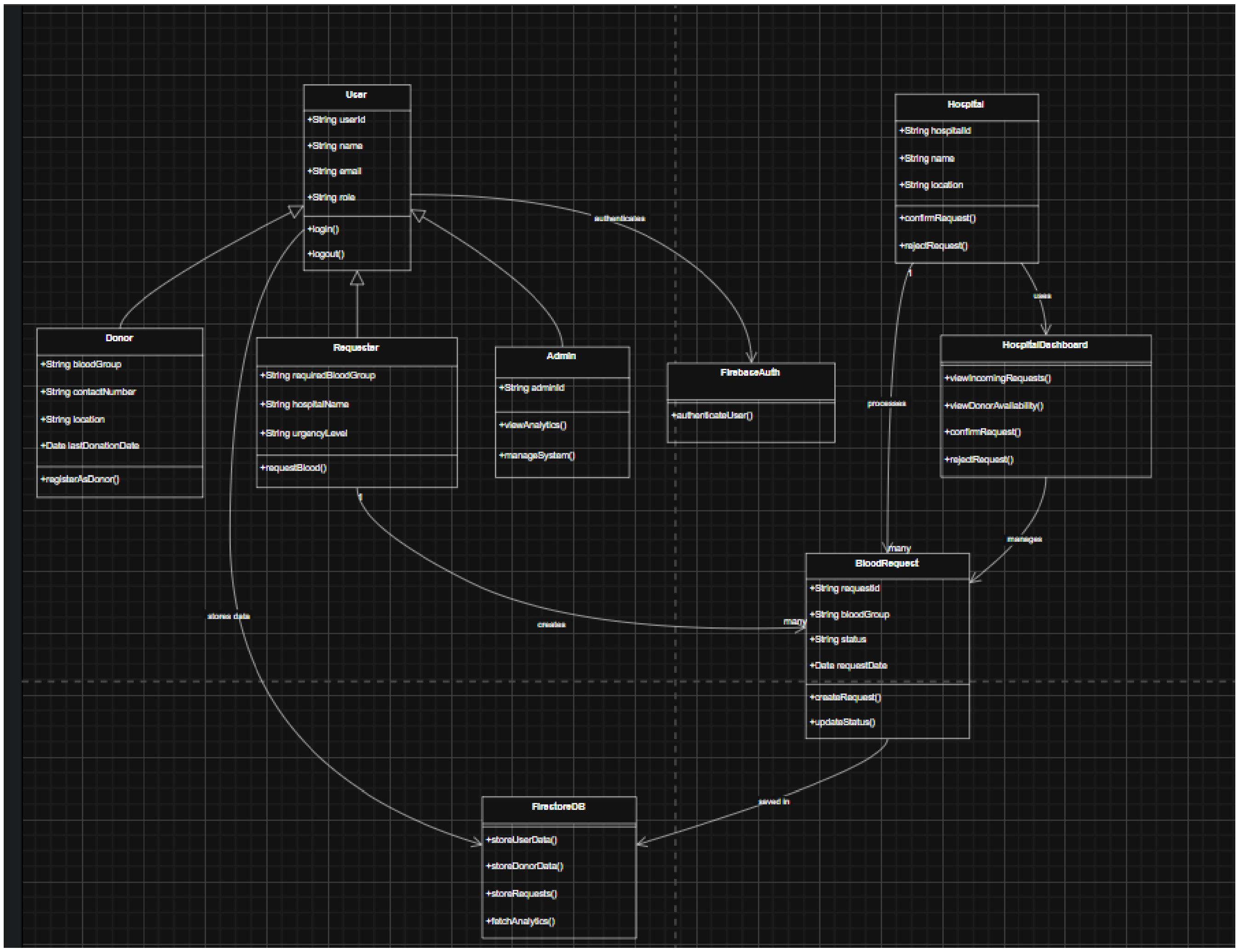


Image 1

DEVELOPMENT VIEW (COMPONENT DAIGRAM)

This view details the software organization, structured as a 3-Tier Layered Architecture to ensure modularity.

- **Presentation Layer (Frontend):** Responsive User, Hospital, and Admin dashboards accessible via web browsers.
- **Business Logic Layer (Backend):**
  - **Auth Module:** Manages secure login and session validation.
  - **Core Services:** Handles algorithms for donor searching, request processing, and statistical aggregation.
  - **Data Access Layer:** A unified interface interacting with the Shared Database to store profiles, logs, and transaction history.

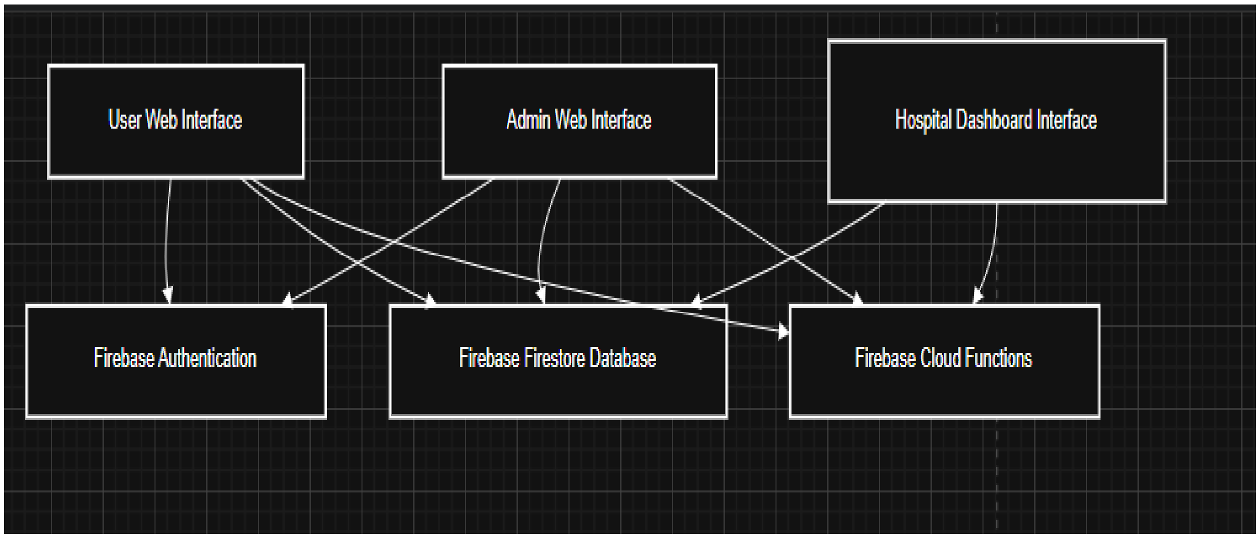


Image 2

PROCESS VIEW (Activity Daigram)

This view addresses the system's **runtime behavior, concurrency, and performance attributes**.

1. **Real-Time Synchronization**
   - Concurrent processes ensure that when a donor updates availability, **hospital and admin dashboards** reflect the change **instantly**.
2. **Query Optimization**

- Search algorithms are **optimized** to filter donors by location and blood group **within seconds**, even under high load.

3. **Session Management**

   - Background processes manage **secure session lifecycles**, ensuring automatic termination upon logout to maintain security.

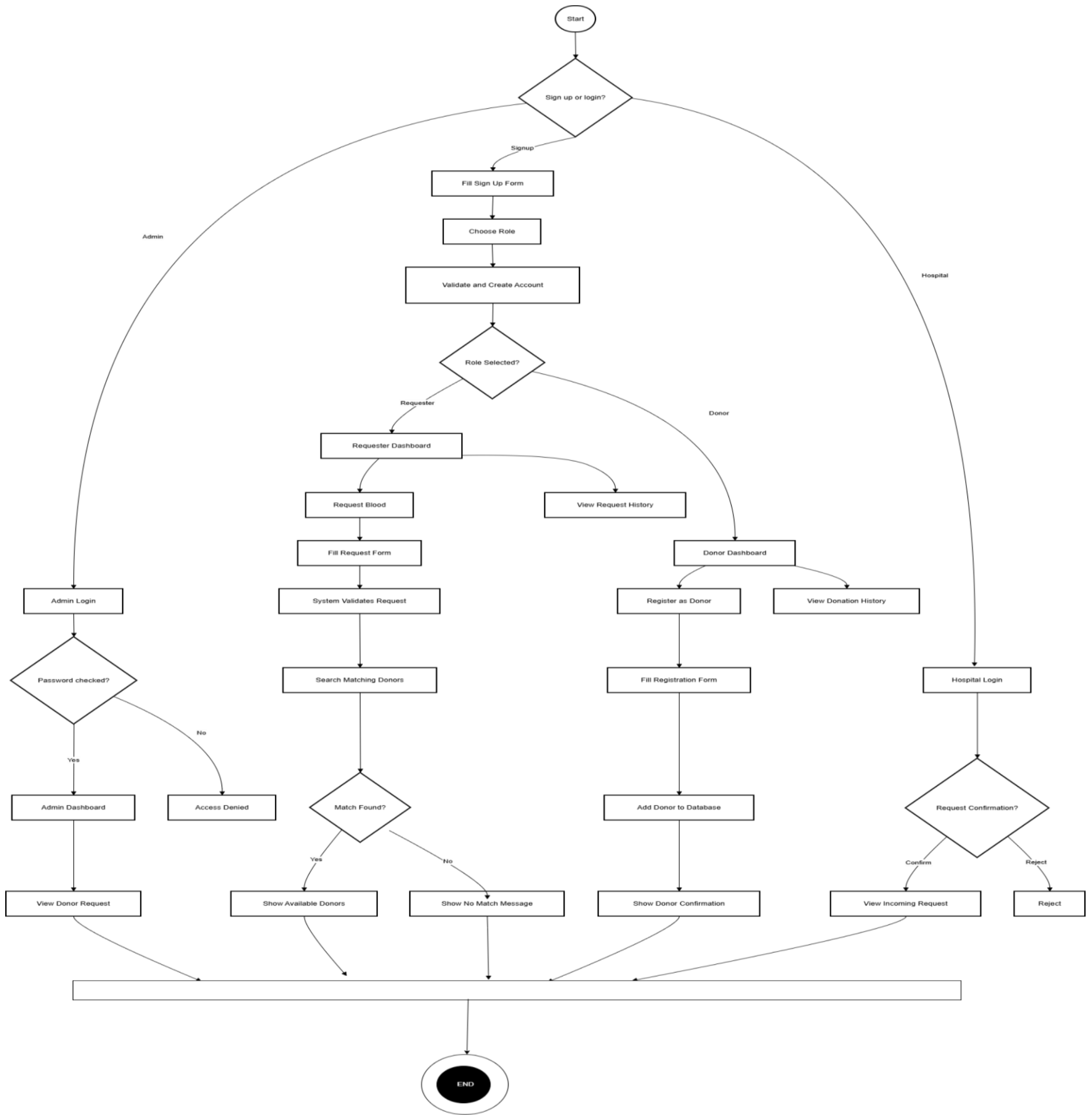


Image 3

PHYSICAL VIEW (DEPLOYMENT DAIGRAM)

This view maps the software components to the hardware infrastructure required for Deployment

- **Client Tier:** End-user devices (Smartphones, Laptops, Tablets) running modern web browsers.
- **Application Server Tier:** Hosts the backend logic and API endpoints, handling HTTP/HTTPS requests over the internet.
- **Data Tier:** A secure, encrypted database server responsible for persisting sensitive user data and medical records

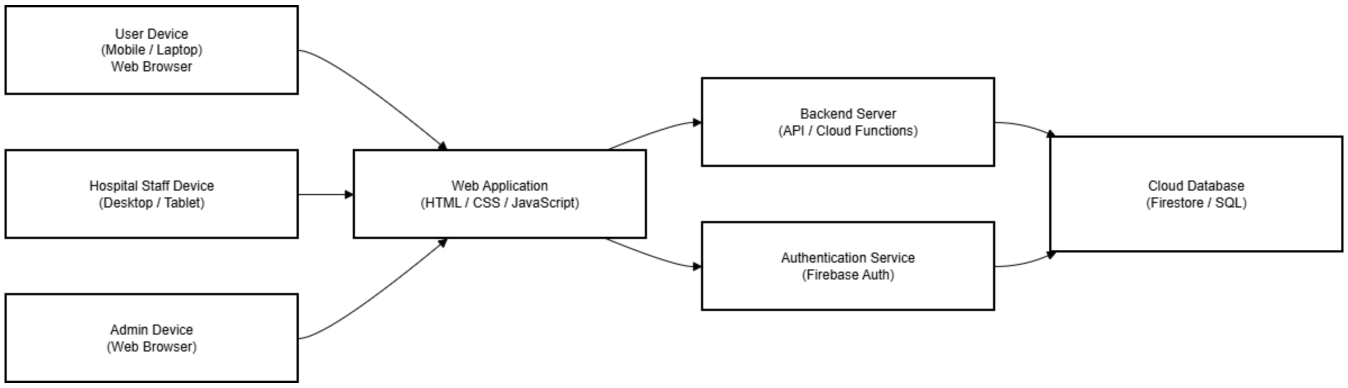


Image 4

USE CASE DAIGRAM

This view captures the functionality of the system from the perspective of external actor defining the boundaries and interactions of the Pulse Chain system.

Primary Actors:

1. **Donor** – The person who donates (blood, organs, money, or resources depending on the system).
2. **Requester –** The person who requests the donation (patient or beneficiary).
3. **Hospital Staff** – Medical personnel who manage requests, donations, and related processes.

Secondary Actors (support the system, not direct users):

1. **Backend System –** The server, database, and logic layer that processes requests, stores data, and communicates between primary actors.
2. **Admin –** Oversees the system, manages users, monitors activities, and handles approvals.

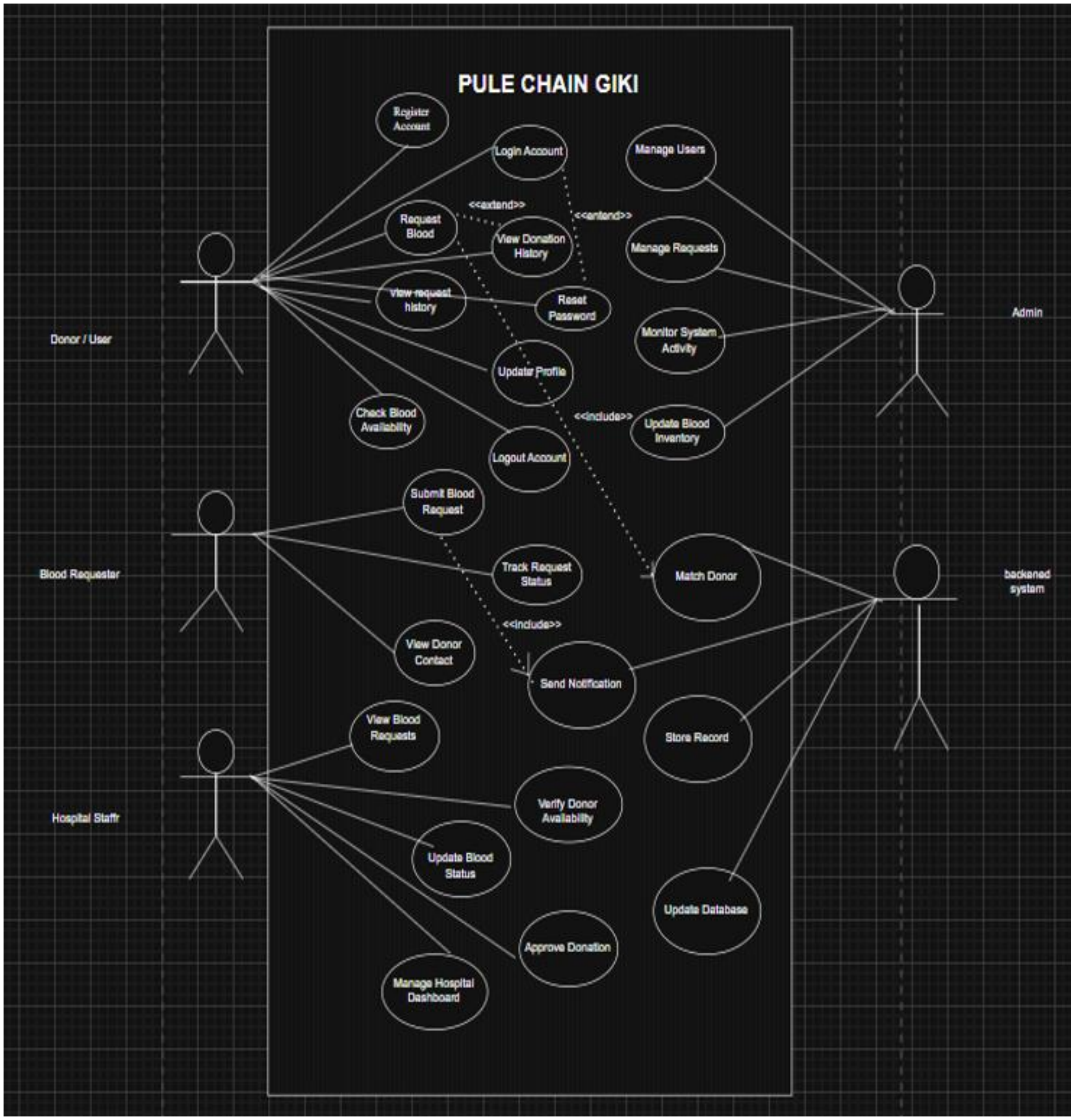
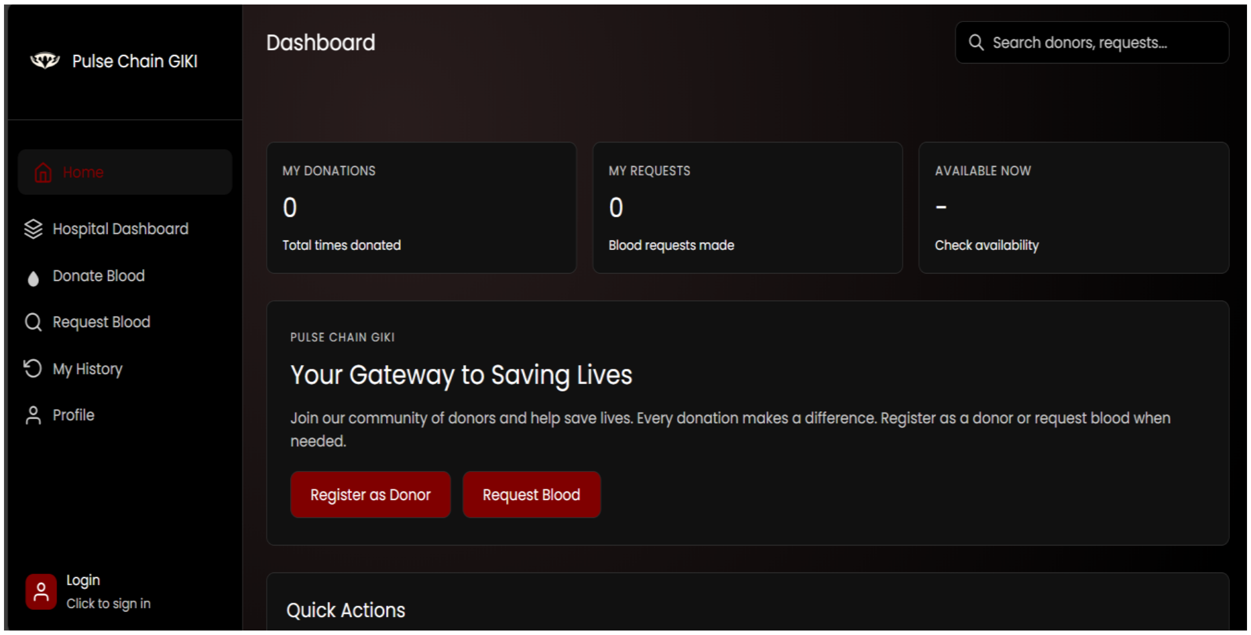

PULE CHAIN GIKI
Register Account
Login Account
Manage Users
<<extend>>
<<extend>>
Request Blood
View Donation History
Manage Requests
View request history
Reset Password
Monitor System Activity
Donor / User
Admin
Update Profile
<<include>>
Check Blood Availability
Update Blood Inventory
Logout Account
Submit Blood Request
Match Donor
Track Request Status
Blood Requester
backened system
<<include>>
View Donor Contact
Send Notification
View Blood Requests
Store Record
Hospital Staffr
Verify Donor Availability
Update Blood Status
Update Database
Approve Donation
Manage Hospital Dashboard

Image 5

USER INTERFACE



Pulse Chain GIKI

Dashboard

Search donors, requests...

Home
Hospital Dashboard
Donate Blood
Request Blood
My History
Profile

MY DONATIONS
0
Total times donated

MY REQUESTS
0
Blood requests made

AVAILABLE NOW
-
Check availability

PULSE CHAIN GIKI

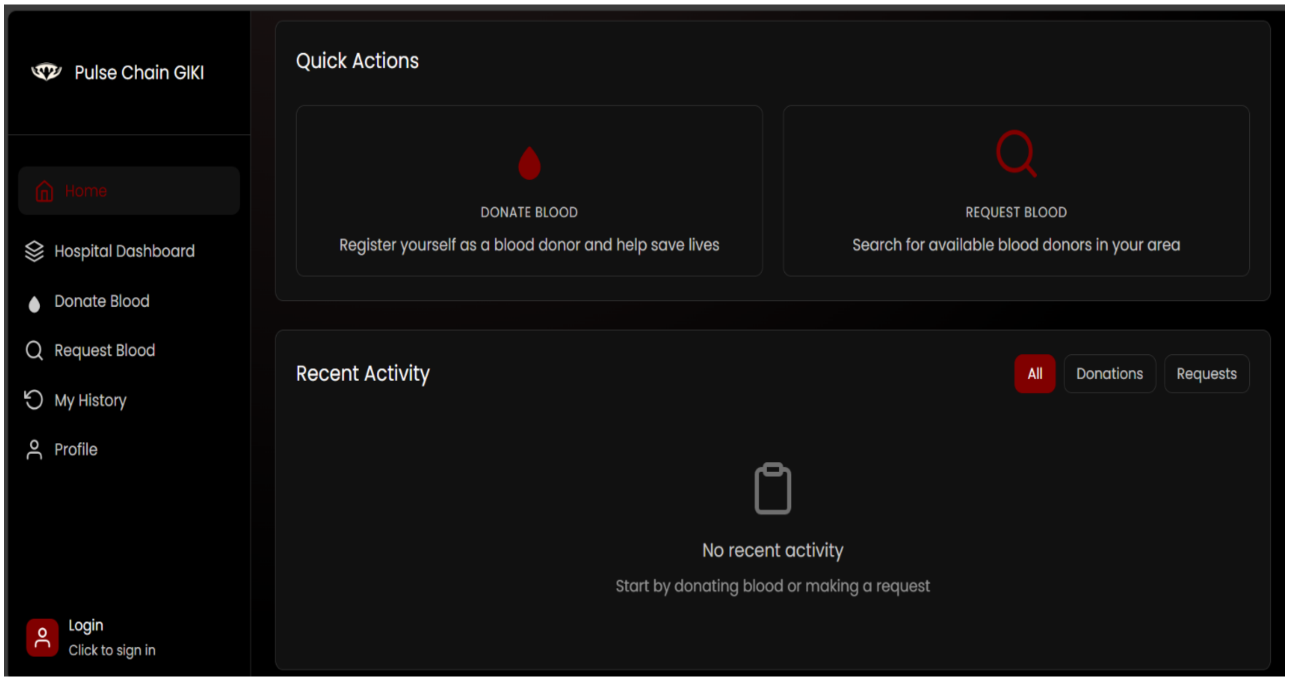
Your Gateway to Saving Lives

Join our community of donors and help save lives. Every donation makes a difference. Register as a donor or request blood when needed.

Register as Donor | Request Blood

Login
Click to sign in

Quick Actions



Pulse Chain GIKI

Home
Hospital Dashboard
Donate Blood
Request Blood
My History
Profile

Quick Actions

DONATE BLOOD
Register yourself as a blood donor and help save lives

REQUEST BLOOD
Search for available blood donors in your area

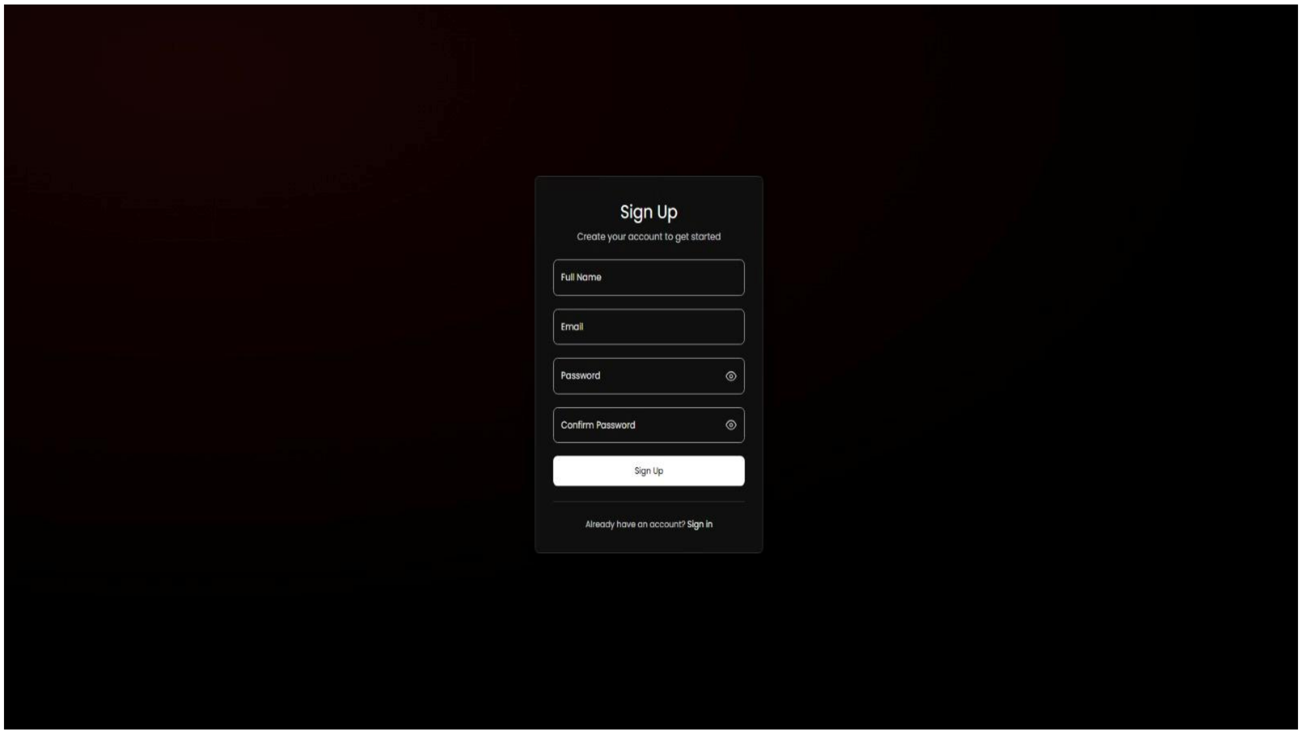
Recent Activity

All | Donations | Requests

No recent activity
Start by donating blood or making a request

Login
Click to sign in

Landing page



Sign Up

Create your account to get started

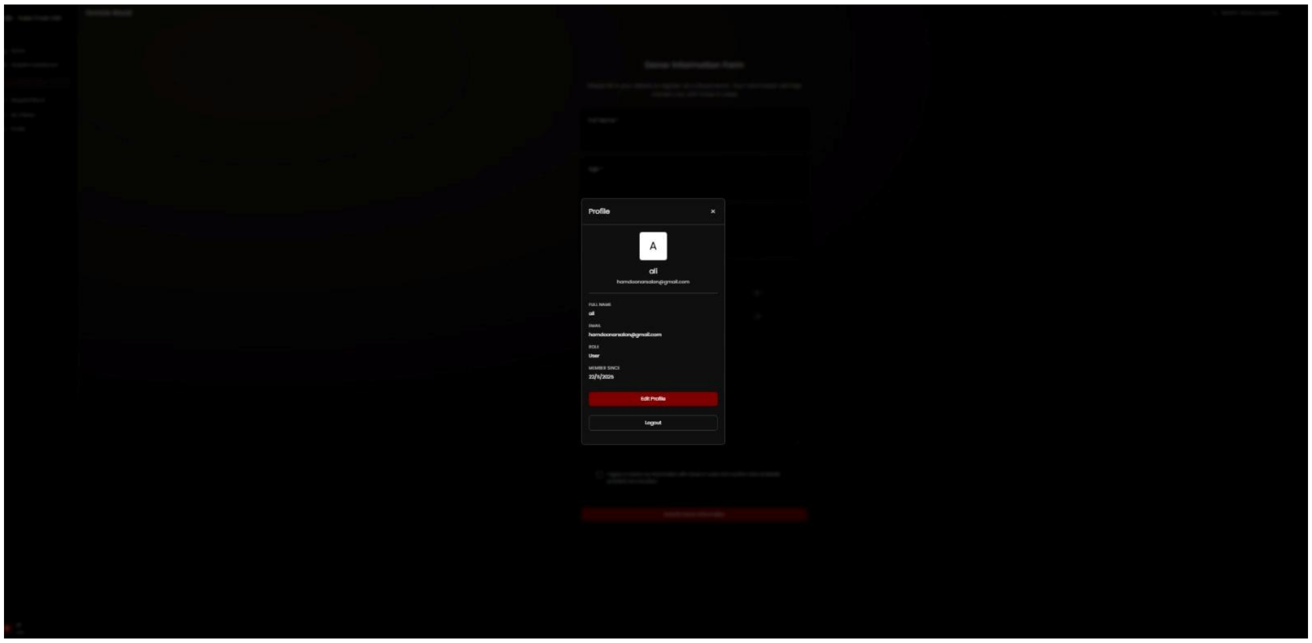
Full Name

Email

Password

Confirm Password

Sign-Up



Profile

A

ali

hamdaanarsalan@gmail.com

FULL NAME

ali

EMAIL

hamdaanarsalan@gmail.com

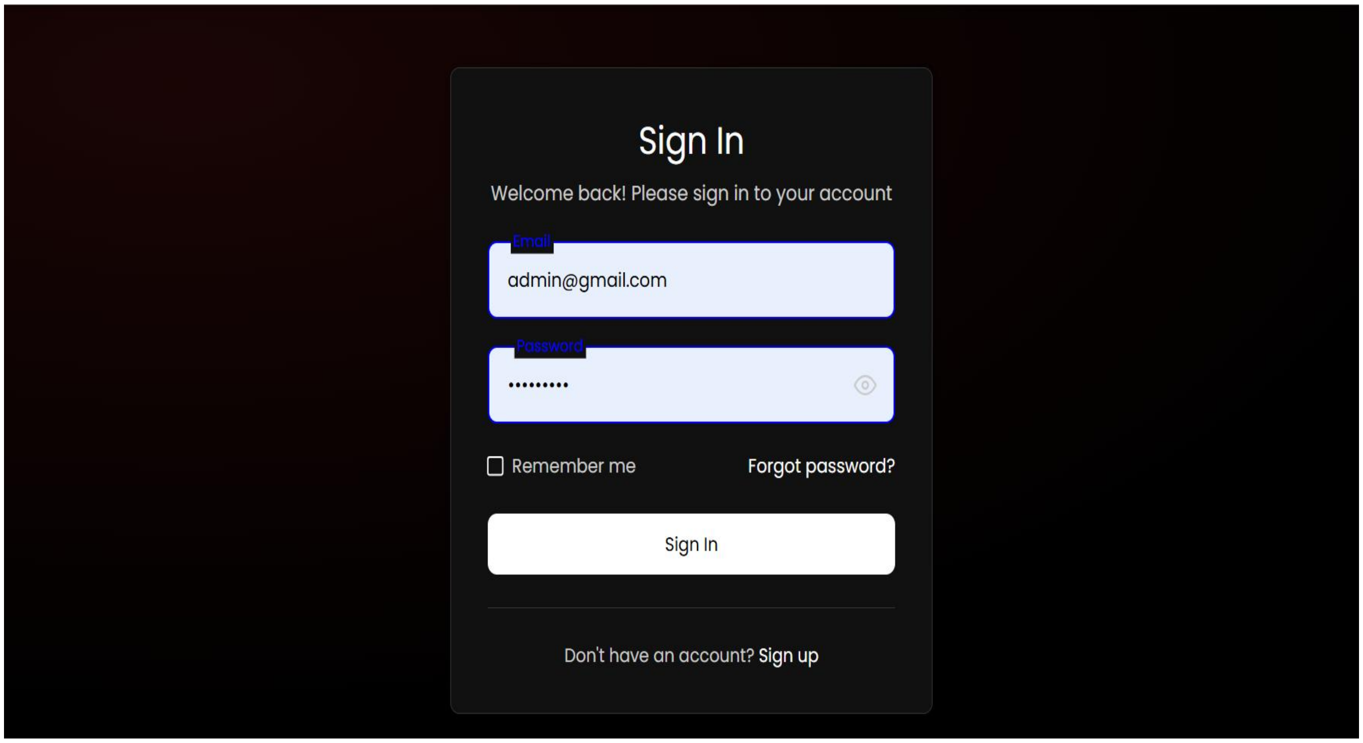
ROLE

User

MEMBER SINCE

22/11/2025

Account Registered


Sign In

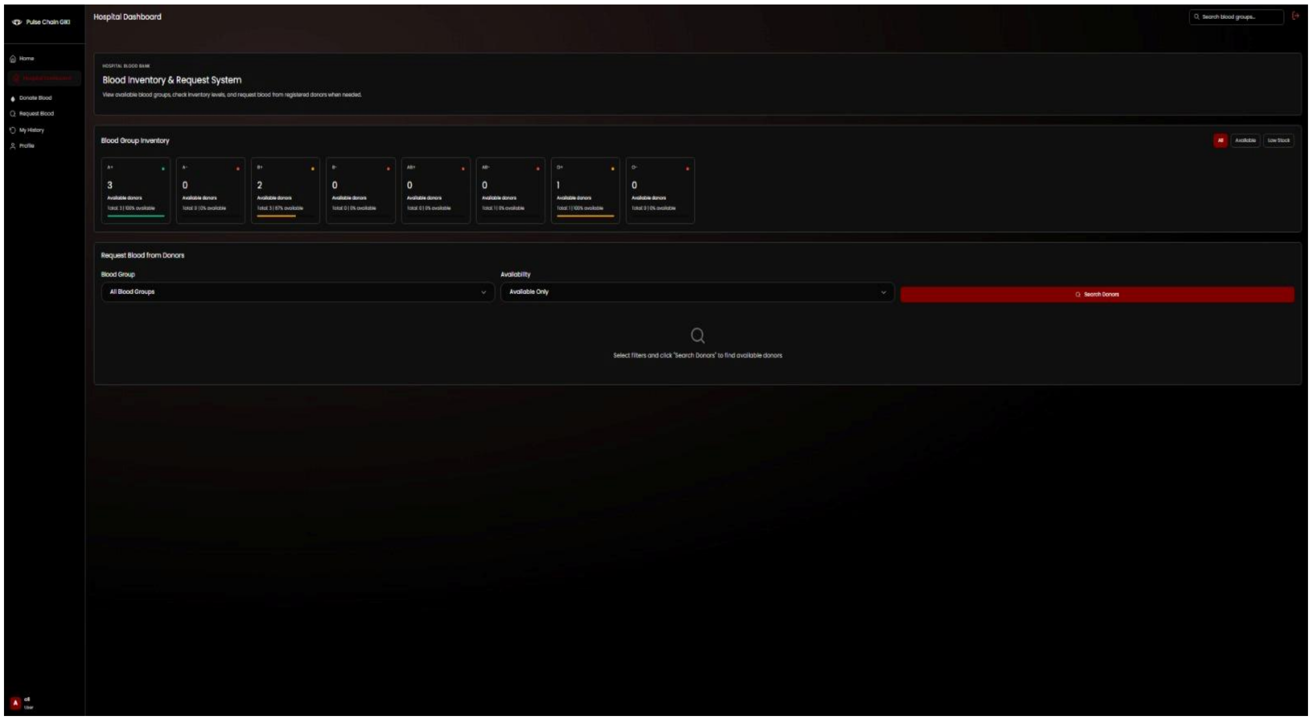
Welcome back! Please sign in to your account

Email: admin@gmail.com

Password: •••••••••

Remember me   Forgot password?

Admin/Sign-In



Hospital Dashboard

HOSPITAL BLOOD BANK

Blood Inventory & Request System

View available blood groups, check inventory levels, and request blood from registered donors when needed.

Blood Group Inventory

A+	A-	B+	B-	AB+	AB-	O+	O-
3	0	2	0	0	0	1	0
Available donors	Available donors	Available donors	Available donors	Available donors	Available donors	Available donors	Available donors
Total: 3 | 100% available	Total: 0 | 0% available	Total: 3 | 67% available	Total: 0 | 0% available	Total: 0 | 0% available	Total: 1 | 0% available	Total: 1 | 100% available	Total: 0 | 0% available

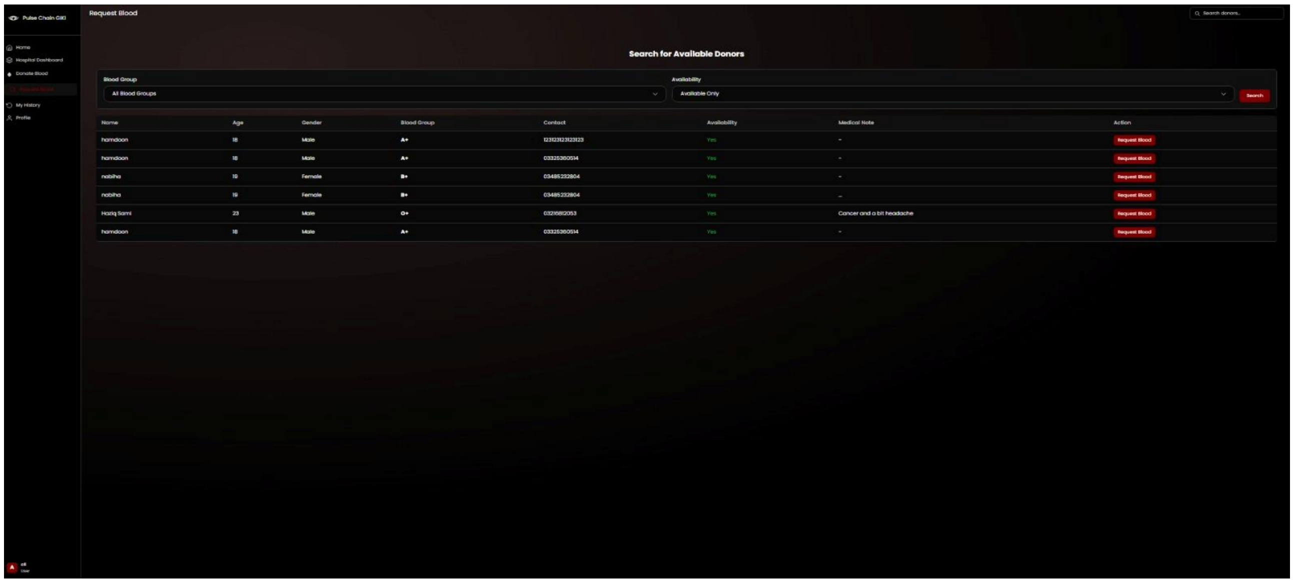
Request Blood from Donors

Blood Group: All Blood Groups

Availability: Available Only

Select filters and click "Search Donors" to find available donors

Blood Inventory Dashboard



Request Blood

Search for Available Donors

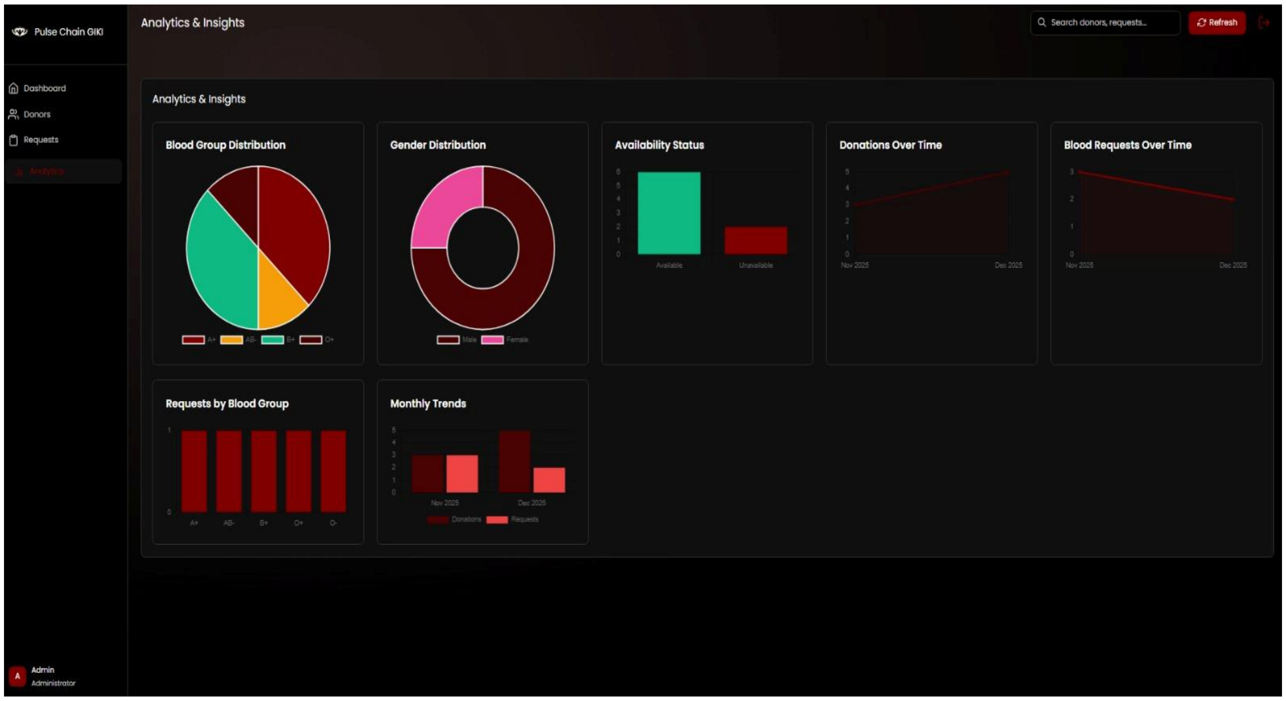
Blood Group: All Blood Groups

Availability: Available Only

Search

Name	Age	Gender	Blood Group	Contact	Availability	Medical Note	Action
hamdoon	18	Male	A+	123123123123123	Yes	-	Request Blood
hamdoon	18	Male	A+	03325360514	Yes	-	Request Blood
nabiha	19	Female	B+	03485232804	Yes	-	Request Blood
nabiha	19	Female	B+	03485232804	Yes	_	Request Blood
Haziq Sami	23	Male	O+	03216812053	Yes	Cancer and a bit headache	Request Blood
hamdoon	18	Male	A+	03325360514	Yes	-	Request Blood

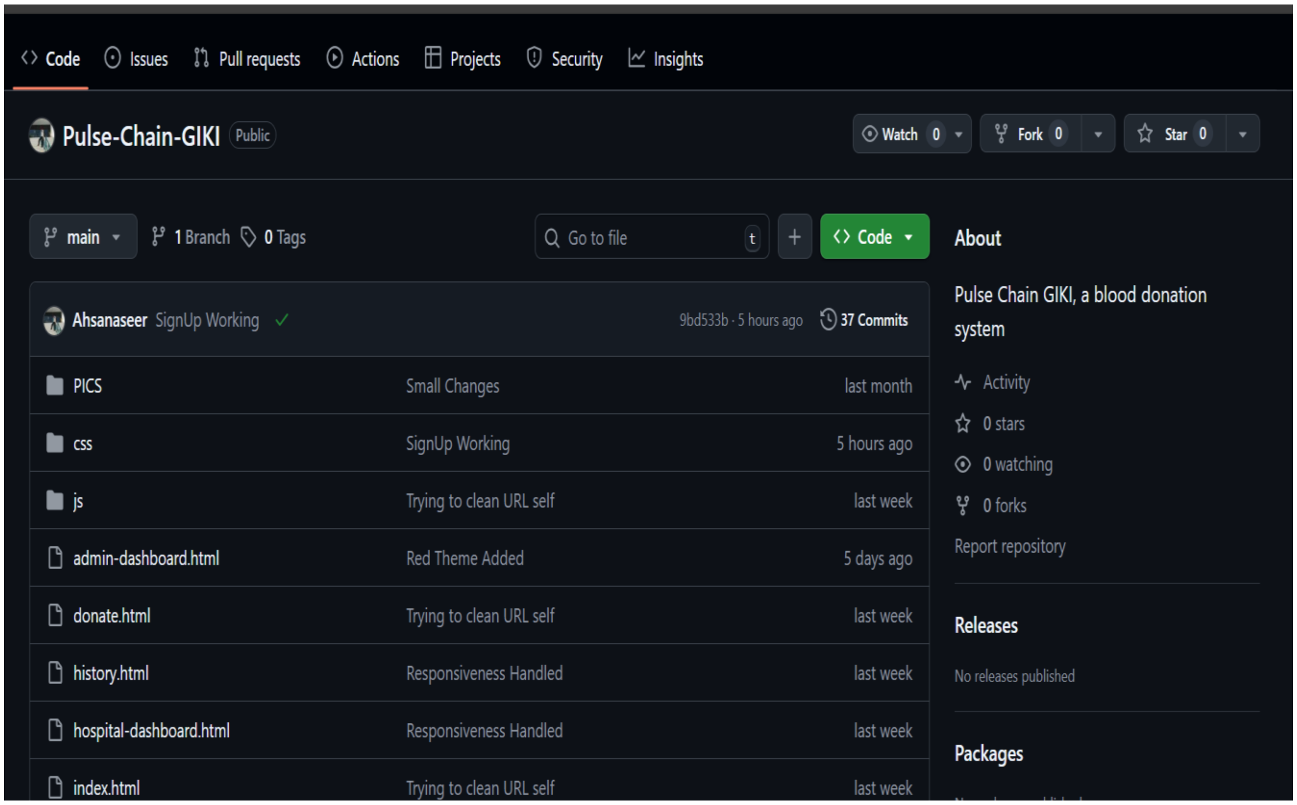
Request Blood



Analytics and Insights

GITHUB COMMITS:

https://github.com/Ahsanaseer/Pulse-Chain-GIKI.git



Code · Issues · Pull requests · Actions · Projects · Security · Insights

Pulse-Chain-GIKI Public

Watch 0 · Fork 0 · Star 0

main · 1 Branch · 0 Tags · Go to file · Code

Ahsanaseer SignUp Working ✓ 9bd533b · 5 hours ago 37 Commits

PICS	Small Changes	last month
css	SignUp Working	5 hours ago
js	Trying to clean URL self	last week
admin-dashboard.html	Red Theme Added	5 days ago
donate.html	Trying to clean URL self	last week
history.html	Responsiveness Handled	last week
hospital-dashboard.html	Responsiveness Handled	last week
index.html	Trying to clean URL self	last week

About

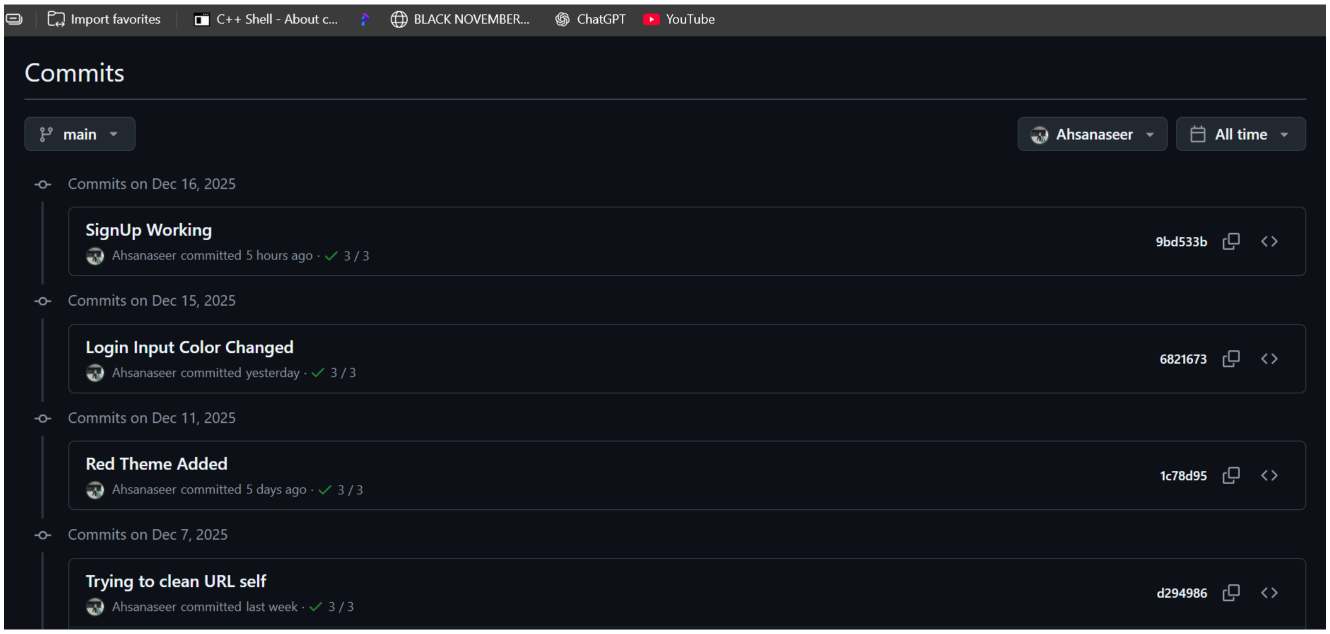
Pulse Chain GIKI, a blood donation system

Activity
0 stars
0 watching
0 forks
Report repository

Releases

No releases published

Packages



Commits

main · Ahsanaseer · All time

Commits on Dec 16, 2025

SignUp Working
Ahsanaseer committed 5 hours ago · ✓ 3 / 3 — 9bd533b

Commits on Dec 15, 2025

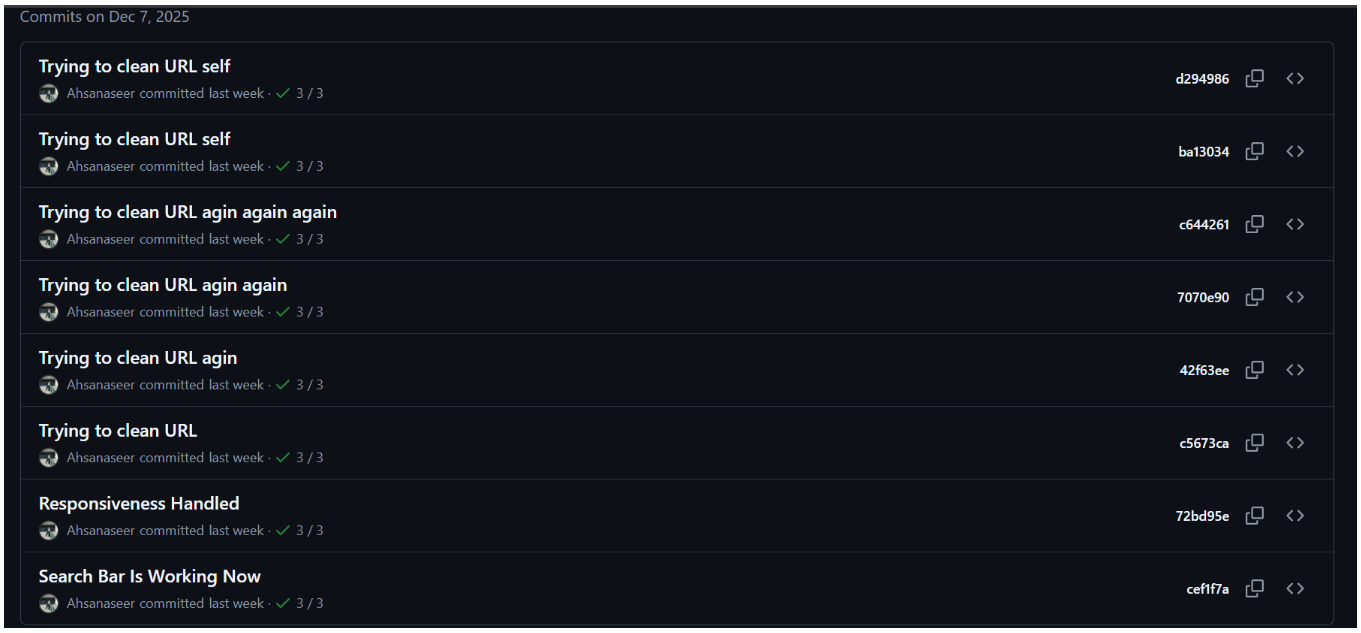
Login Input Color Changed
Ahsanaseer committed yesterday · ✓ 3 / 3 — 6821673

Commits on Dec 11, 2025

Red Theme Added
Ahsanaseer committed 5 days ago · ✓ 3 / 3 — 1c78d95

Commits on Dec 7, 2025

Trying to clean URL self
Ahsanaseer committed last week · ✓ 3 / 3 — d294986


Commits on Dec 7, 2025

Trying to clean URL self
Ahsanaseer committed last week · ✓ 3 / 3 d294986

Trying to clean URL self
Ahsanaseer committed last week · ✓ 3 / 3 ba13034

Trying to clean URL agin again again
Ahsanaseer committed last week · ✓ 3 / 3 c644261

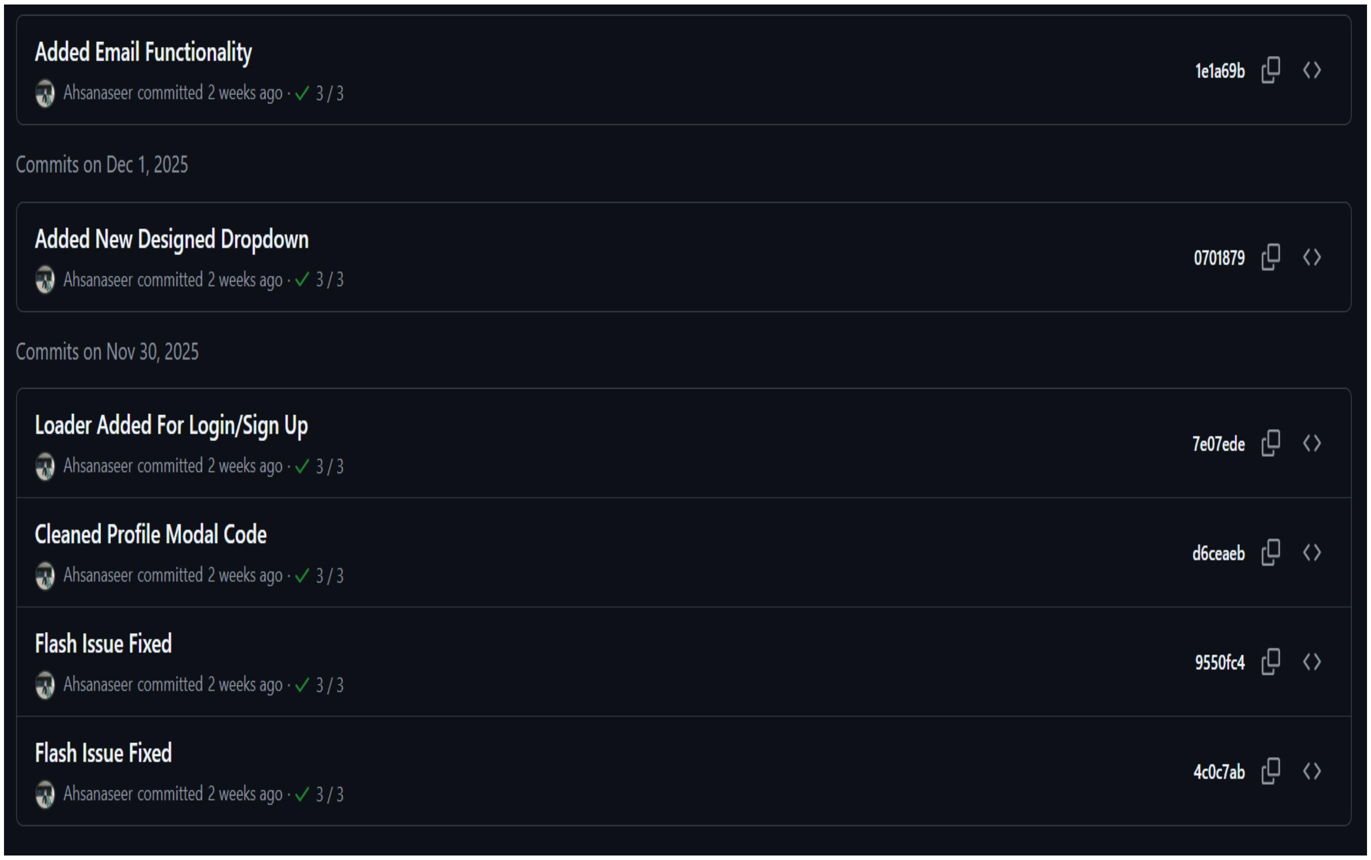
Trying to clean URL agin again
Ahsanaseer committed last week · ✓ 3 / 3 7070e90

Trying to clean URL agin
Ahsanaseer committed last week · ✓ 3 / 3 42f63ee

Trying to clean URL
Ahsanaseer committed last week · ✓ 3 / 3 c5673ca

Responsiveness Handled
Ahsanaseer committed last week · ✓ 3 / 3 72bd95e

Search Bar Is Working Now
Ahsanaseer committed last week · ✓ 3 / 3 cef1f7a


Added Email Functionality
Ahsanaseer committed 2 weeks ago · ✓ 3 / 3 1e1a69b

Commits on Dec 1, 2025

Added New Designed Dropdown
Ahsanaseer committed 2 weeks ago · ✓ 3 / 3 0701879

Commits on Nov 30, 2025

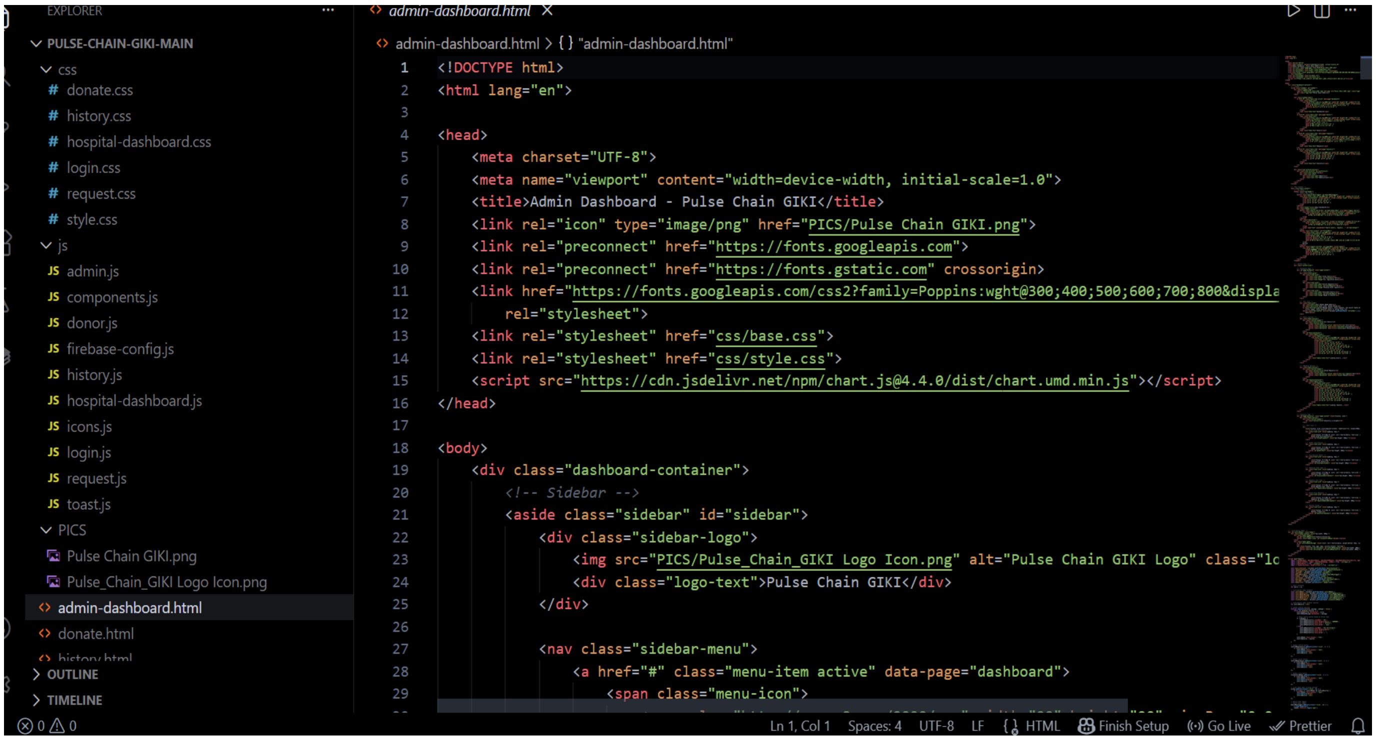
Loader Added For Login/Sign Up
Ahsanaseer committed 2 weeks ago · ✓ 3 / 3 7e07ede

Cleaned Profile Modal Code
Ahsanaseer committed 2 weeks ago · ✓ 3 / 3 d6ceaeb

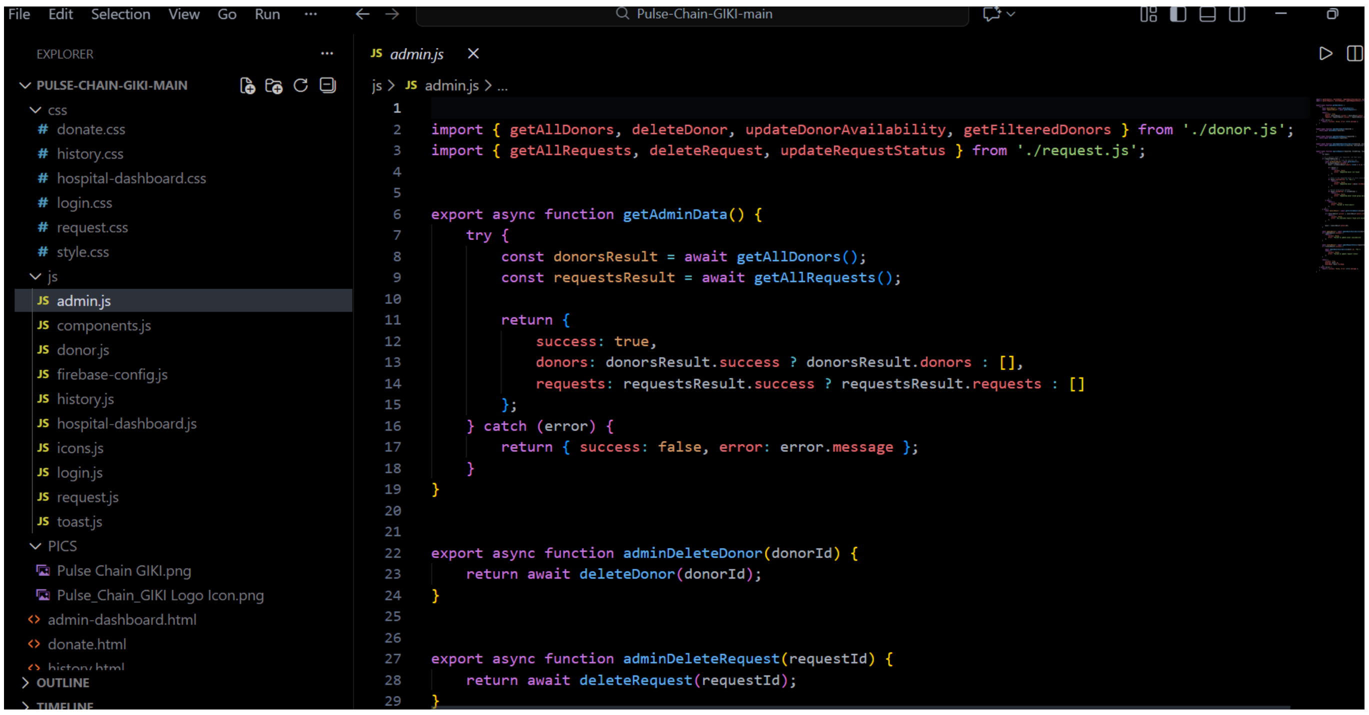
Flash Issue Fixed
Ahsanaseer committed 2 weeks ago · ✓ 3 / 3 9550fc4

Flash Issue Fixed
Ahsanaseer committed 2 weeks ago · ✓ 3 / 3 4c0c7ab

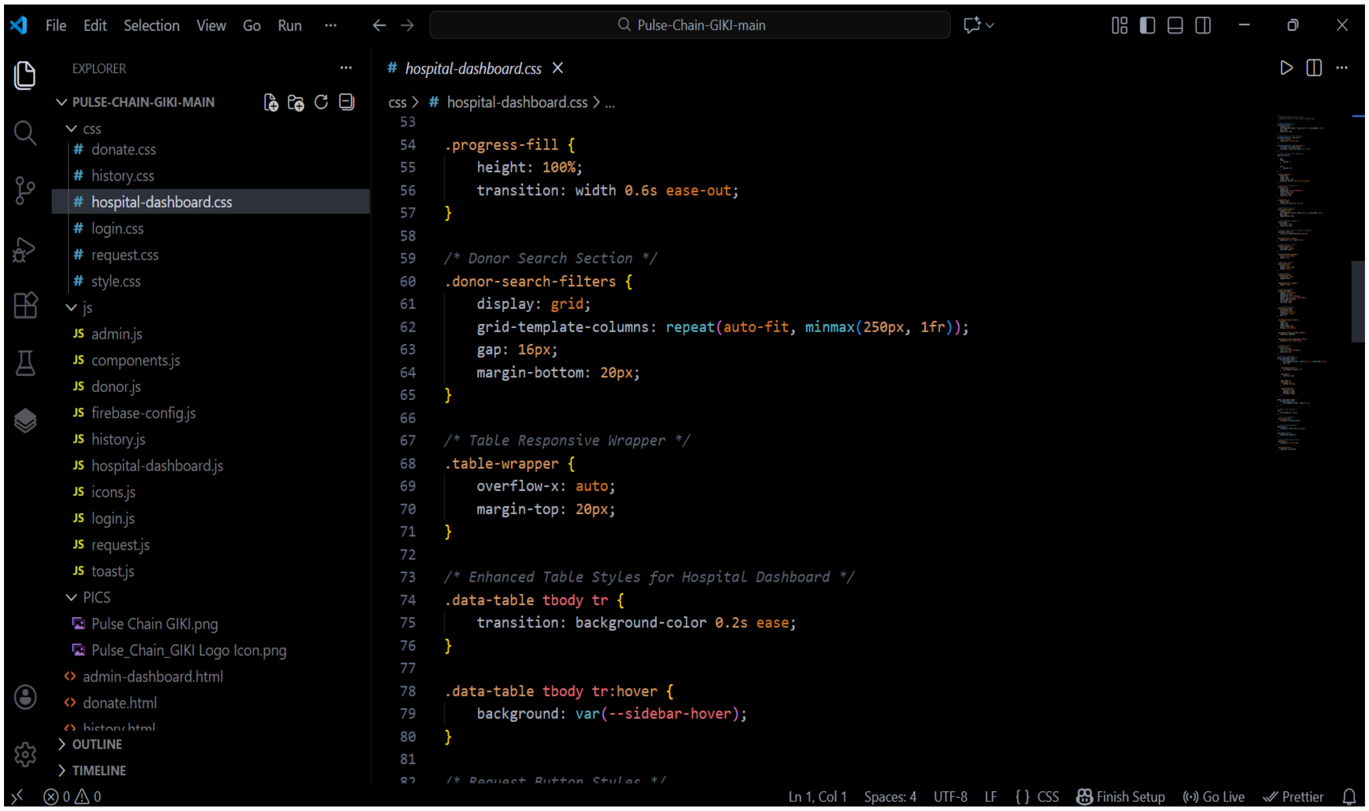
CODE SCREENSHOTS



```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Admin Dashboard - Pulse Chain GIKI</title>
    <link rel="icon" type="image/png" href="PICS/Pulse Chain GIKI.png">
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500;600;700;800&displa
        rel="stylesheet">
    <link rel="stylesheet" href="css/base.css">
    <link rel="stylesheet" href="css/style.css">
    <script src="https://cdn.jsdelivr.net/npm/chart.js@4.4.0/dist/chart.umd.min.js"></script>
</head>

<body>
    <div class="dashboard-container">
        <!-- Sidebar -->
        <aside class="sidebar" id="sidebar">
            <div class="sidebar-logo">
                <img src="PICS/Pulse_Chain_GIKI Logo Icon.png" alt="Pulse Chain GIKI Logo" class="lo
                <div class="logo-text">Pulse Chain GIKI</div>
            </div>

            <nav class="sidebar-menu">
                <a href="#" class="menu-item active" data-page="dashboard">
                    <span class="menu-icon">
```



```javascript

import { getAllDonors, deleteDonor, updateDonorAvailability, getFilteredDonors } from './donor.js';
import { getAllRequests, deleteRequest, updateRequestStatus } from './request.js';


export async function getAdminData() {
    try {
        const donorsResult = await getAllDonors();
        const requestsResult = await getAllRequests();

        return {
            success: true,
            donors: donorsResult.success ? donorsResult.donors : [],
            requests: requestsResult.success ? requestsResult.requests : []
        };
    } catch (error) {
        return { success: false, error: error.message };
    }
}


export async function adminDeleteDonor(donorId) {
    return await deleteDonor(donorId);
}


export async function adminDeleteRequest(requestId) {
    return await deleteRequest(requestId);
}
```



```css

.progress-fill {
    height: 100%;
    transition: width 0.6s ease-out;
}

/* Donor Search Section */
.donor-search-filters {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));
    gap: 16px;
    margin-bottom: 20px;
}

/* Table Responsive Wrapper */
.table-wrapper {
    overflow-x: auto;
    margin-top: 20px;
}

/* Enhanced Table Styles for Hospital Dashboard */
.data-table tbody tr {
    transition: background-color 0.2s ease;
}

.data-table tbody tr:hover {
    background: var(--sidebar-hover);
}

```

WORK DIVISION

Nabiha Nasir

Documentation Lead | Requirement Engineer | System Analyst

• Complete **project documentation** (SRS, final report, specifications)
• Requirements gathering and requirement analysis
• Functional and non-functional requirements documentation
• Use case descriptions and requirement traceability
• **All system diagrams** (Use Case, Class, Activity, Component, Deployment, 4+1 Architecture)
• System overview, scope definition, and literature review
• Documentation review, formatting, and final compilation

Ahsan Naseer

Frontend Developer | Backend Developer | Full Stack Developer

• Complete system development using **HTML, CSS, and JavaScript**
• Frontend UI implementation and dashboard development
• Backend development using **Firebase**
• Authentication and user management

• Database design and real-time data handling
• Integration of frontend with backend services
• Deployment and testing support

M. Hamdoon

Project Coordinator | Jira Manager
• **Jira project management** and sprint handling
• Task creation, assignment, and progress tracking
• Sprint planning and timeline coordination
• Team collaboration and workflow management
• Monitoring project milestones and deliverables

Irtaza Alyan

Documentation Support | Requirement Assistant
• Assistance in **documentation writing and refinement**
• Supporting requirement descriptions and validation
• Review of SRS and report sections
• Formatting and proofreading support