# **BLOOD BANK MANAGEMENT SYSTEM**

### 1.1. Project Overview

In the proposed system, the seeker will only have to provide blood group of the blood he or she is in search of, a list of all the possible donors. This will reduce the effort of seeker as he or she will not have to explore the whole list to select to contact which donor. More over to check that the seeker is authentic and genuine the request for the blood which will be put by the patient on the application will be approved by the hospital in which he is admitted. Information regarding blood donations camps would also be provided to end users.

The present project elucidates the following features :-

* Registering the Donors
* Modification of Donor Information
* Searching a Donor
* Life Saving Contacts (in major cities)
* Paging Services

**1.3. Feasibility Study :**

### Technical Feasibility:-

Technical feasibility assesses whether the development of a Disease Predictor Software is achievable from a technological standpoint. Here are key considerations:

### 1. Software Requirements

Hardware: Evaluate the availability and suitability of the required hardware components, such as servers, computational resources, and storage systems. Software Stack: Ensure access to necessary software tools, programming languages, and frameworks for software development.

### 2. Data Accessibility

Data Sources: Determine if relevant medical data, including patient records, diagnostic reports, and historical health information, is accessible.

Data Quality: Assess the quality, completeness, and consistency of available data for training and testing algorithms.

### 3. Algorithm Complexity

Algorithm Development: Analyze the complexity of designing accurate disease prediction algorithms, which may involve machine learning, data analytics, or statistical modeling.

Computational Resources: Ensure the availability of sufficient computational power to train and run these algorithms effectively.

### 4. Integration Capabilities

Interoperability: Evaluate the feasibility of integrating the software with existing healthcare systems, electronic health records (EHRs), or laboratory information management systems (LIMS).

Data Exchange Standards: Ensure compatibility with industry-standard data exchange formats and protocols.

### 5. Scalability

User Growth: Consider the ability of the software to scale as the user base grows, handling increased data volumes and user interactions without performance degradation.

Load Balancing: Implement load balancing mechanisms to distribute computational workloads efficiently.

### 6. Security and Privacy

Data Security: Implement robust security measures to protect sensitive medical data, ensuring compliance with data privacy regulations like HIPAA or GDPR. User Authentication: Incorporate secure user authentication and authorization mechanisms.

### 7. Testing and Validation

Quality Assurance: Develop comprehensive testing procedures to validate the accuracy and reliability of disease predictions.

Validation Data: Secure relevant datasets for testing and fine-tuning algorithms, including validation with clinical experts.

### Operational Feasibility :-

Operational feasibility evaluates whether a Disease Predictor Software can be effectively integrated into the existing healthcare ecosystem and meet operational needs. Here are key considerations:

#### 1. User Acceptance

User Training: Assess the ease of use and the training required for healthcare professionals to utilize the software effectively.

User Feedback: Gather input from potential users to understand their preferences and concerns.

#### 2. Integration with Healthcare Workflow

Workflow Compatibility: Ensure that the software aligns with existing clinical workflows, minimizing disruptions.

Data Flow: Evaluate how data will be collected, processed, and communicated within the healthcare system.

#### 3. Accessibility and Availability

Device Compatibility: Confirm that the software can run on a variety of devices and platforms, including desktop computers, tablets, and smartphones. Uptime and Reliability: Ensure high availability and reliability to support 24/7 access, especially in critical healthcare settings.

#### 4. Data Management

Data Handling: Develop efficient mechanisms for data input, storage, retrieval, and backup to manage patient information securely.

Data Governance: Establish data governance policies to maintain data accuracy, privacy, and compliance with regulations.

5. Scalability and Performance

Scalability Planning: Prepare for increased usage by designing the software to scale seamlessly, accommodating more patients and healthcare providers. Performance Optimization: Continuously monitor and optimize software performance to reduce response times and ensure efficient operation.

#### 6. Maintenance and Support

Software Updates: Plan for regular updates and patches to address bugs, security vulnerabilities, and feature enhancements.

Technical Support: Establish a support infrastructure to assist users with technical issues, questions, and troubleshooting.

#### 7. Cost Management

Budget Considerations: Assess the budget required for ongoing maintenance, support, and software enhancements.

Cost-Benefit Analysis: Determine the cost-effectiveness and potential financial benefits of implementing the software.

#### 8. Legal and Regulatory Compliance

Compliance Assurance: Ensure adherence to healthcare regulations, including data privacy laws (e.g., HIPAA or GDPR) and medical standards (e.g., HL7). Liability Management: Define legal responsibilities and liability in case of software-related issues.

### Economic Feasibility :-

Economic feasibility assesses whether developing a Disease Predictor Software is financially viable. It involves evaluating costs, revenue potential, and return on investment (ROI). Here are key considerations:

#### 1. Cost Estimation

Development Costs: Calculate expenses for software development, including hiring developers, acquiring hardware, and licensing software tools. Data Acquisition: Assess the cost of obtaining medical datasets, including patient records and research databases.

Infrastructure Costs: Account for ongoing expenses related to server maintenance, data storage, and IT support.

#### 2. Operating Costs

Maintenance: Evaluate ongoing expenses for software updates, bug fixes, and technical support.

Data Storage: Estimate costs associated with storing and managing large volumes of medical data securely.

Personnel: Consider staffing costs for IT professionals, data scientists, and support staff.

#### 3. Revenue Generation

Pricing Model: Decide on a pricing strategy, such as subscription fees, payper-use, or freemium, based on user segments and market competition. User Adoption: Estimate user adoption rates and project revenue based on different scenarios.

Partnerships: Explore potential collaborations with healthcare providers, insurance companies, or research institutions to generate additional revenue streams.

#### 4. Return on Investment (ROI)

ROI Analysis: Calculate the expected return on investment by comparing development and operating costs to projected revenues.

Break-even Point: Determine when the software is expected to break even and start generating profits.

Risk Assessment: Identify financial risks and uncertainties that may impact the ROI, such as changes in market demand or regulatory hurdles.

#### 5. Funding Sources

Capital Investment: Assess the availability of funds for development, whether from investors, grants, or internal resources.

Financial Planning: Develop a financial plan that outlines the budget allocation and funding strategy for the project.

### Scheduling Feasibility:-

Scheduling feasibility assesses the project's ability to be completed within a reasonable and predefined timeframe. This involves setting realistic timelines, resource allocation, and risk management strategies. Here are key considerations:

#### 1. Project Timeline

Define Milestones: Establish clear project milestones and deadlines for different phases, such as development, testing, and deployment.

Critical Path Analysis: Identify the critical path, which outlines the sequence of tasks that must be completed on time to avoid project delays.

#### 2. Resource Availability

Human Resources: Ensure that skilled developers, data scientists, and other required team members are available throughout the project.

Technical Resources: Assess the availability of necessary hardware, software, and infrastructure components.

External Dependencies: Consider dependencies on third-party services or data sources and their reliability.

#### 3. Risk Management

Risk Assessment: Identify potential risks that could lead to schedule delays, such as technical challenges, data quality issues, or resource shortages. Mitigation Strategies: Develop contingency plans and mitigation strategies for identified risks to minimize their impact on the schedule.

#### 4. Scope Management

Scope Definition: Clearly define the project scope, including features and functionalities, to prevent scope creep that can extend the project timeline. Change Control: Implement a change control process to evaluate and approve any modifications to the project scope.

#### 5. Agile Methodologies

Agile Frameworks: Consider using agile methodologies, such as Scrum or Kanban, to facilitate flexibility and adaptability in response to changing requirements.

Sprints: Plan and schedule development sprints to deliver incremental functionality and maintain a predictable development pace.

### Legal Feasibility :-

Legal feasibility involves assessing whether the development and deployment of a Disease Predictor Software comply with relevant laws, regulations, and ethical standards. This is crucial to avoid legal issues and ensure ethical use of medical data. Here are key considerations:

#### 1. Data Privacy and Security

Compliance with Regulations: Ensure compliance with data privacy laws, such as HIPAA (in the U.S.) or GDPR (in Europe), which govern the handling of patient data.

Data Encryption: Implement robust data encryption mechanisms to protect sensitive medical information during storage and transmission.

#### 2. Informed Consent and Ethics

Informed Consent: Establish procedures to obtain informed consent from individuals whose data is used for training and testing the software's algorithms.

Ethical Guidelines: Adhere to ethical guidelines for medical research and data usage, including respect for patient autonomy and confidentiality.

#### 3. Liability and Legal Responsibilities

Liability Assessment: Determine liability in case of inaccurate disease predictions, and establish protocols for handling potential legal disputes. Legal Agreements: Draft clear terms of service and user agreements to define user responsibilities and limitations of liability.

#### 4. Intellectual Property Rights

Patents and Copyrights: Investigate potential patent or copyright issues related to the algorithms or software components used in disease prediction.

Licensing: Ensure that you have the necessary licenses to use third-party software libraries and tools.

#### 5. Regulatory Approvals

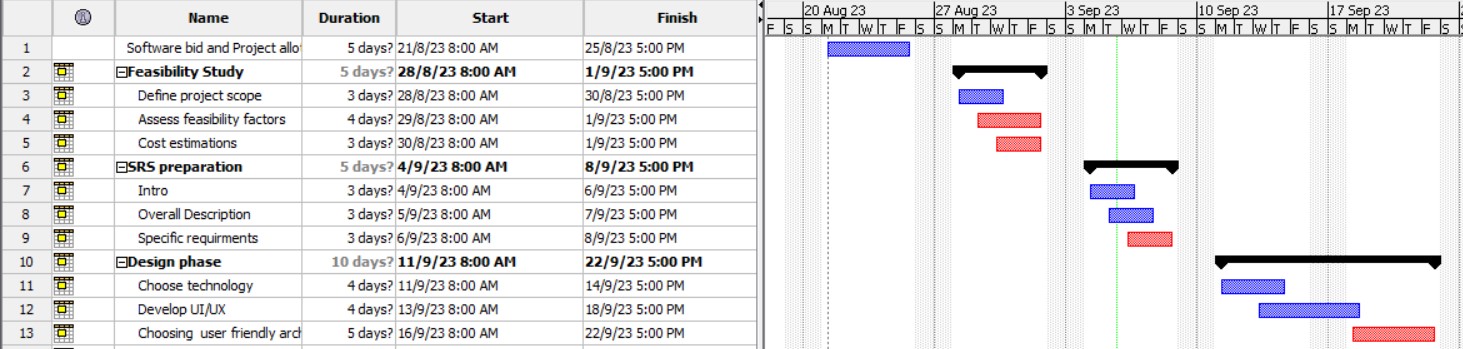
ICMR Approval (if applicable): If the software is considered a medical device, assess the need for regulatory approval from agencies like in India. Food and Drug Administration (ICMR) or equivalent organizations in other countries. Compliance Documentation: Prepare documentation demonstrating compliance with relevant healthcare and medical device regulations.

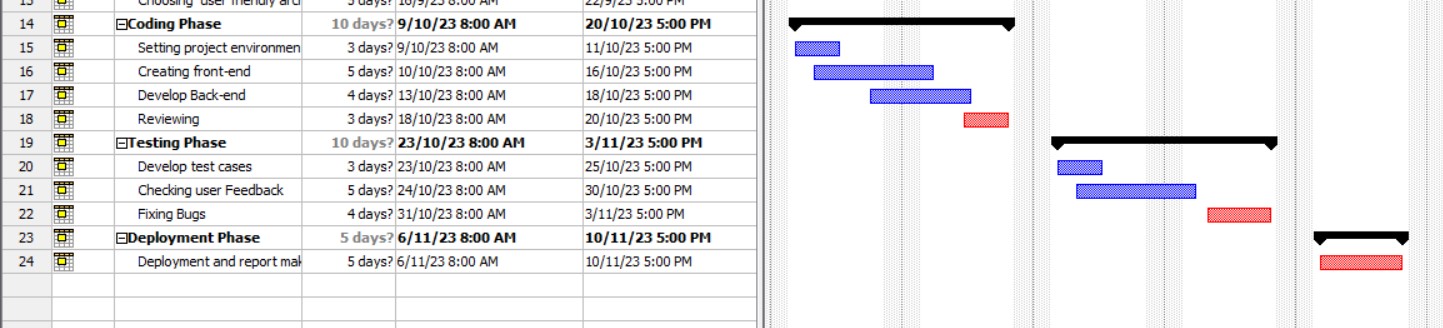
#### 6. User Data Ownership

Data Ownership: Clarify data ownership rights, especially when users contribute personal health data to the software.

Data Retention Policies: Establish data retention policies and procedures, specifying how long user data will be stored and for what purposes.

##### 1.4. Gantt Chart

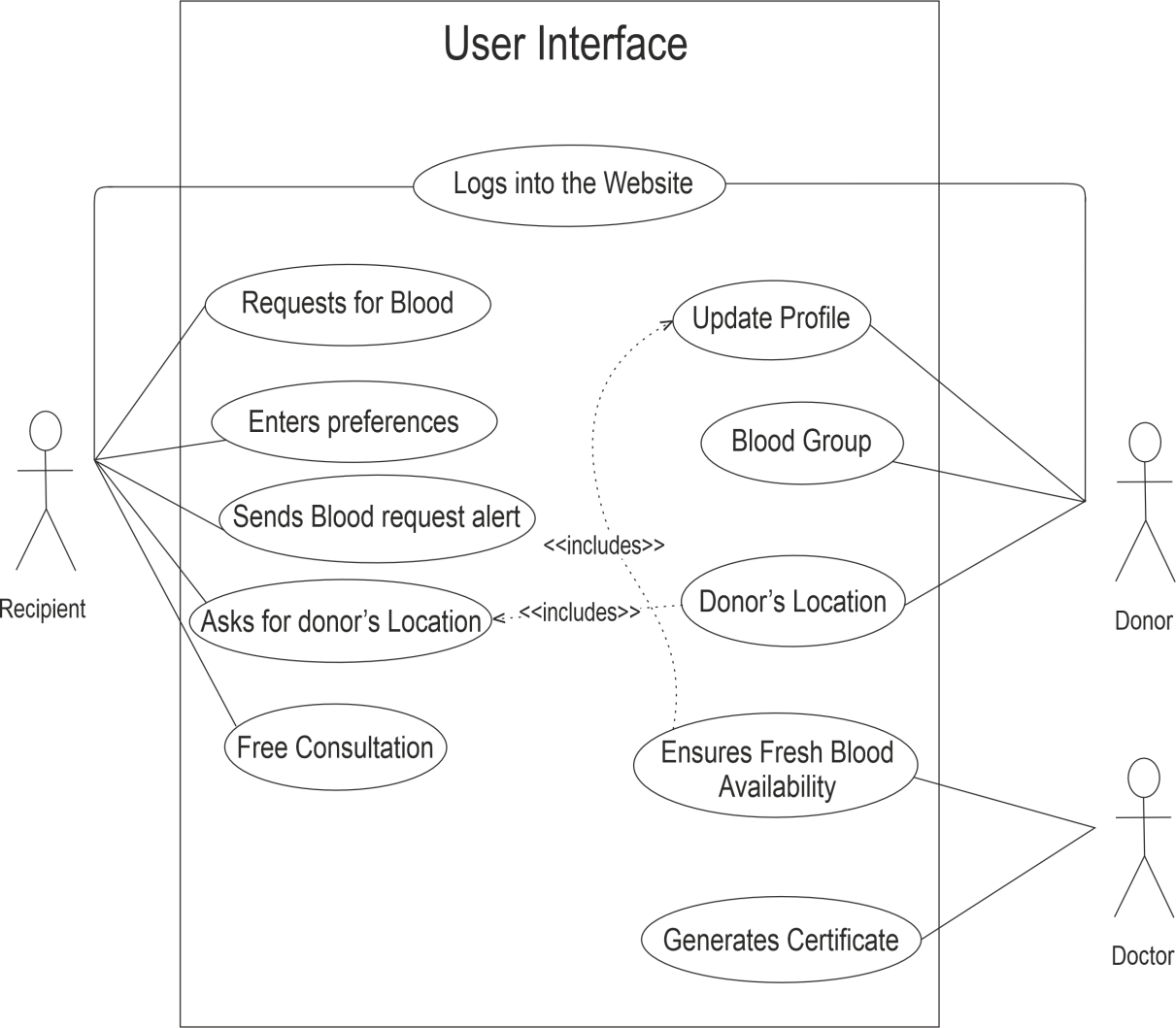




**FIG 1.1**

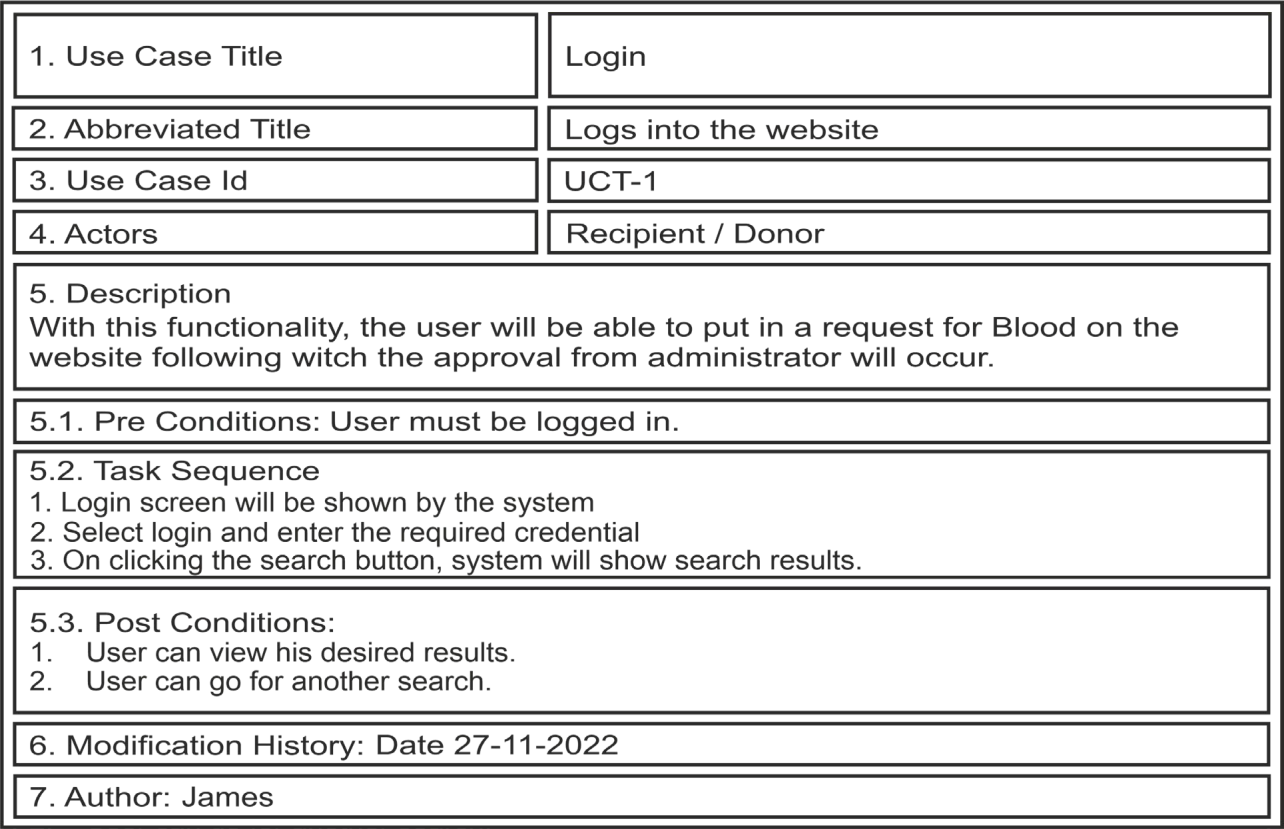
## 2. Analysis Phase

### 2.1.1 Use Cases



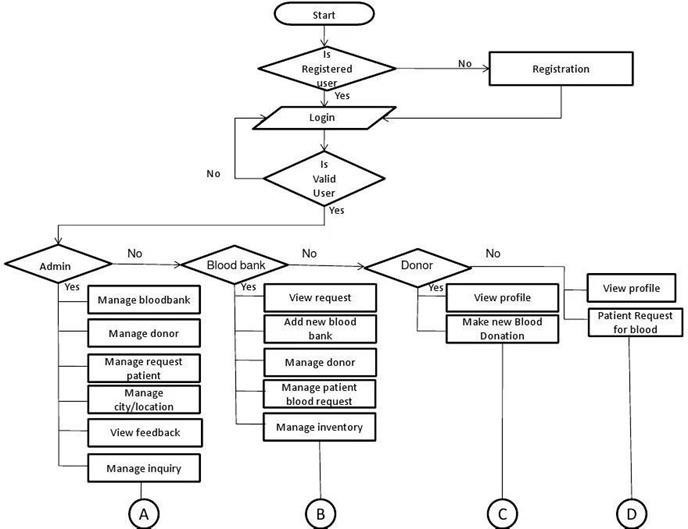
**FIG 2.1**

### 2.2.2 Use Case Templates



**FIG 2.2**

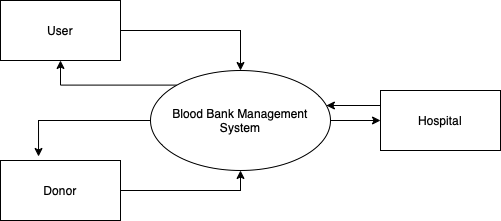
### 2.1. Swimlane Diagrams



**FIG 2.3**

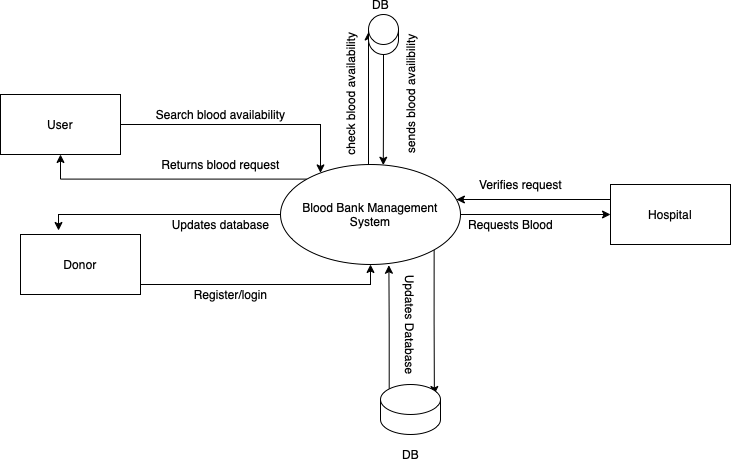
### 2.2. Data Flow Diagrams

#### 2.2.1. DFD Level 0



**FIG 2.4**

#### 2.2.2. DFD Level 1



**FIG 2.5**

## 1.3 OVERVIEW

**Existing System**

The present system is a system where the patient when searches for the blood of particular blood type will be provided with the list of the possible donors. This is a very inappropriate way as the information shown is very cluttered. The victim at such a key moment of time will have to communicate with the donors that whether he is in the such a situation to come and donate blood. This will waste a lot of time at such a crucial moment where in this much time someone can lose his or her life.

**Proposed System**

In the proposed system the seeker will only have to provide blood group of the blood he is in search of, a list of all the possible donors sorted on the of basis distance from his or her will be shown. Gradually the radius of prompt will also increase until a donor is confirmed. This will reduce the effort of seeker as he will not have to explore the whole list to select to contact which donor. More over to check that the seeker is authentic and genuine, the request for the blood which will be put by the patient on the application will be approved by the hospital in which he is admitted. Information regarding blood donations camps would also be provided to end users.

In this project mainly 3 modules are there.

1. Admin
2. Donors
3. Acceptors

Admin: This module focuses on the both donors & acceptors. Each member in a donor and acceptor is given a user id and password, which identifies him uniquely. The member is given a login form. He enters the login details user id and password.

The options given to:

· Change Password

· Maintain donor details

· Maintain acceptor details

· Update donor details

· Update acceptor details

· Logout Donor: Each member in a Donor is given a user id and password, which identifies him uniquely. The member is given a login form. he enters the login details user id and password.

The options given to each member in a staff are:

· Change password

· Find a Blood group

· Why to donate blood

# **2.** OVERALL DESCRIPTION

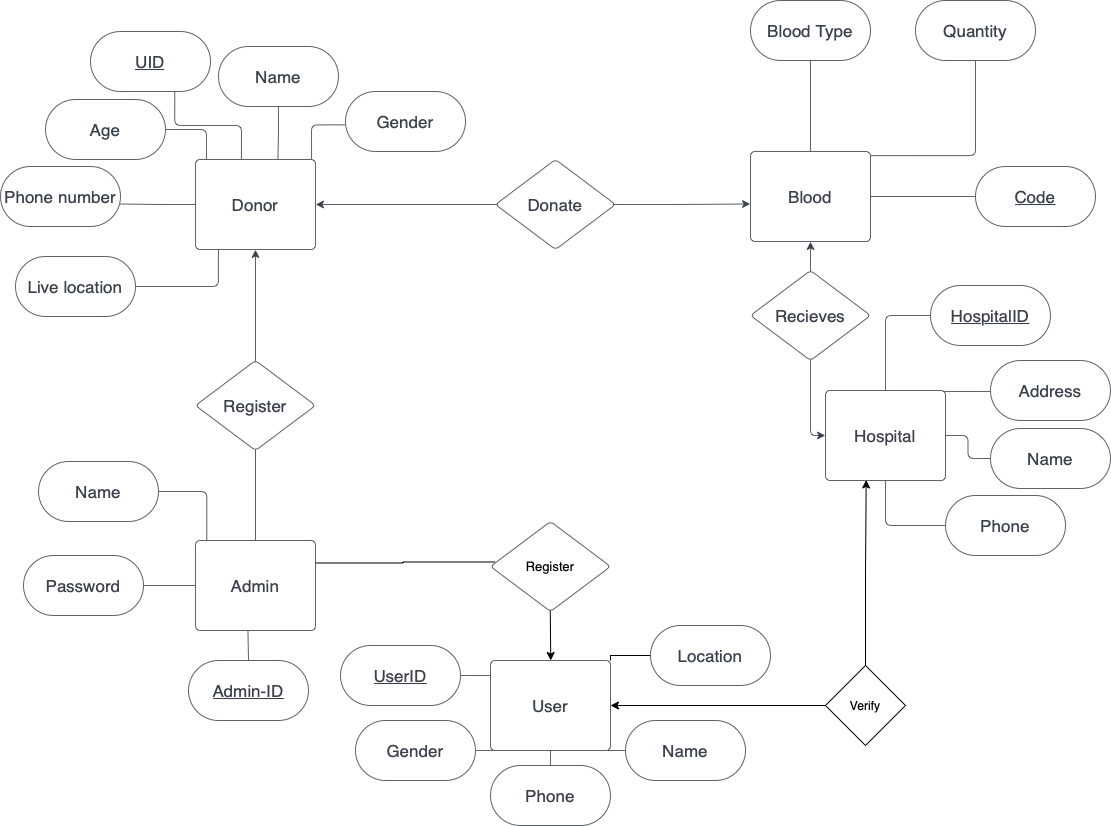
## 1.2 PRODUCT PRESPECTIVE

* To provide a means for the blood bank to publicize and advertise blood donation programs.
* To provide an efficient donor and blood stock management functions to the blood bank by recording the donor and blood details.
* To improve the efficiency of blood stock management by alerting the blood bank staffs when the blood quantity is below it par level or when the blood stock has expired.
* To provide pure blood with no wastages blood is been collected in different types of packs. They are double, triple, and triple (AS), quadruple pack.
* To provide synchronized and centralized donor and blood stock database.
* To provide immediate storage and retrieval of data and information.

## 1.3 PRODUCT FEATURES

* Login Interface
* User should enter the valid username and password to get access to its profile.
* Donor Profile
* User will be able to see its Account No., The receipts of the blood donated to the bank, Donation to the Bank, Need of the Blood to the Bank and Request for Blood.
* Blood Stock Management
* It will show the Blood Detailed of the specific bottle with its Full Donor Detail or Account No. if he/she is registered to the Bank.
* Report
* It will be available on the Admin’s Profile and will show the Availability of the Blood Groups with its no. of available bottle as per admin’s choice to view the report as Month, Day, or Year.

## PRODUCT FUNCTION



ER DIAGRAM

**FIG 2.6**

Benefits of blood bank management system project:

1. Seeker can get the list of donor’s distance wise, blood group wise from the central inventory.
2. Donors can view the blood donation camp organizing at different places.
3. As it is an automated application, its index page will encourage the donor to donate the blood.
4. The donor will be provided with a certificate with a unique QR code on it which will help us know that when was the last time this donor had donated blood.
5. Blood bank in charge is getting rid from manual procedure. Now they don’t have to manually fill entries and keep a record of them.
6. The probability of error would be minimal.
7. This will keep also help us to keep a check that whether the recipient is genuine or not which will be done in the following way: Whenever a person will put a request on the application for the need of particular blood type then that request would need to be approved by the hospital only in that case he will be able to get to see the list of all donors. 8. The GUI of this would be very convenient and easy to use.

# **3.** FUNCTIONAL REQUIREMENTS

**3.1**  **Donor registration:**

Given that the user has accessed application, then the user should be able to register through the application. The donor software operator must provide first name, last name, blood or plasma group, location, contact number, address and password.

**3.2 Login:**

Given that the user has registered then the end user should be able to login to the application.

There will be admin who will have his own login username and password and he will be the person who will be in charge of this whole application.

**3.3 Search:**

User should be able to request for blood at emergency situation where theuser would have to just specify the blood type he is need for and the whole list of donors who have registered in the application.

**3.4 Enquiries and Doubts:**

There will be separate page where the software users can post all their doubts and could ask for any sort of help if in need.

**3.5 Logout:**

User will be able to logout of the application using a dedicated button.

# **4.** NON-FUNCTIONAL REQUIREMENTS

* Availability
  1. The system should be available at all times, meaning the user can access it using application.
  2. In case of a of a hardware failure or database corruption, a replacement page will be shown. Also, in case of a hardware failure or database corruption, backups of the database should be retrieved from the application data folder and saved by the administrator.
  3. It implies 24 x 7 availability.

* Security
  1. The system will keep all transactions that includes any customer information, confidential. ii. The system must automatically log out all customers after a certain period of inactivity.

* Reliability

As the system provide the right tools for problem solving it is made in such a way that the system is reliable in its operations and for securing the sensitive details.

* Performance

Application will be lightweight and must send request immediately. The location data must be accurate

# **5.** SOFTWARE REQUIREMENTS

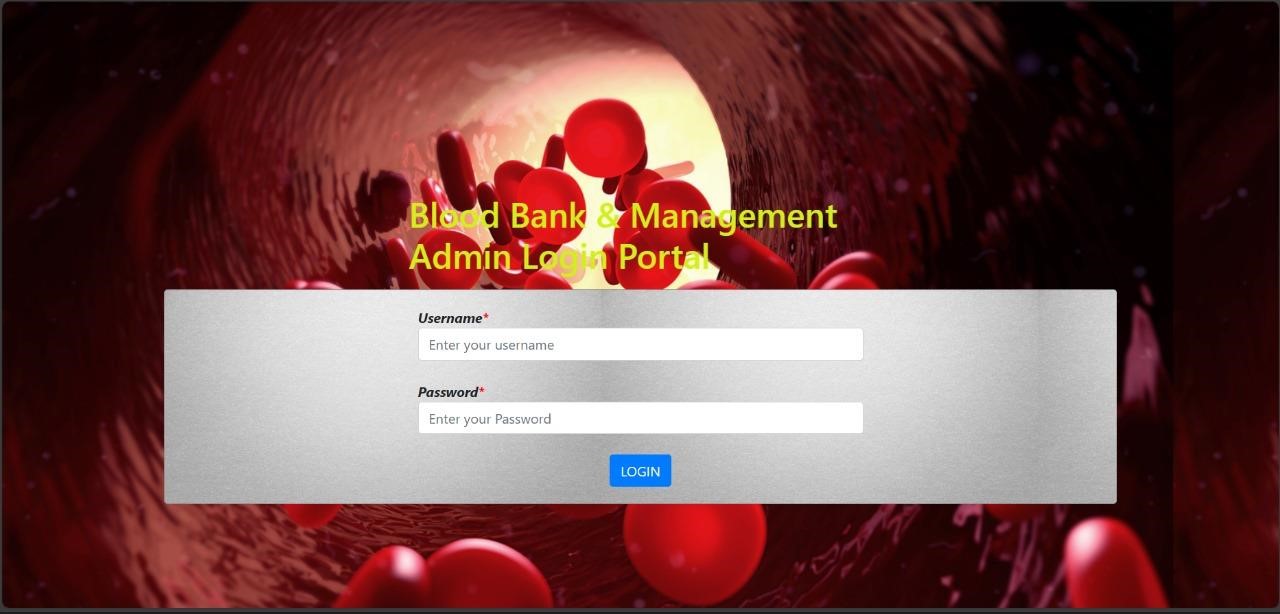
* Operating System: Android OS, Android Emulators
* Front End: HTML,CSS,PHP, Java
* Back End: My SQL
* Development Tool: Android Studio

# **6.** HARDWARE REQUIREMENTS

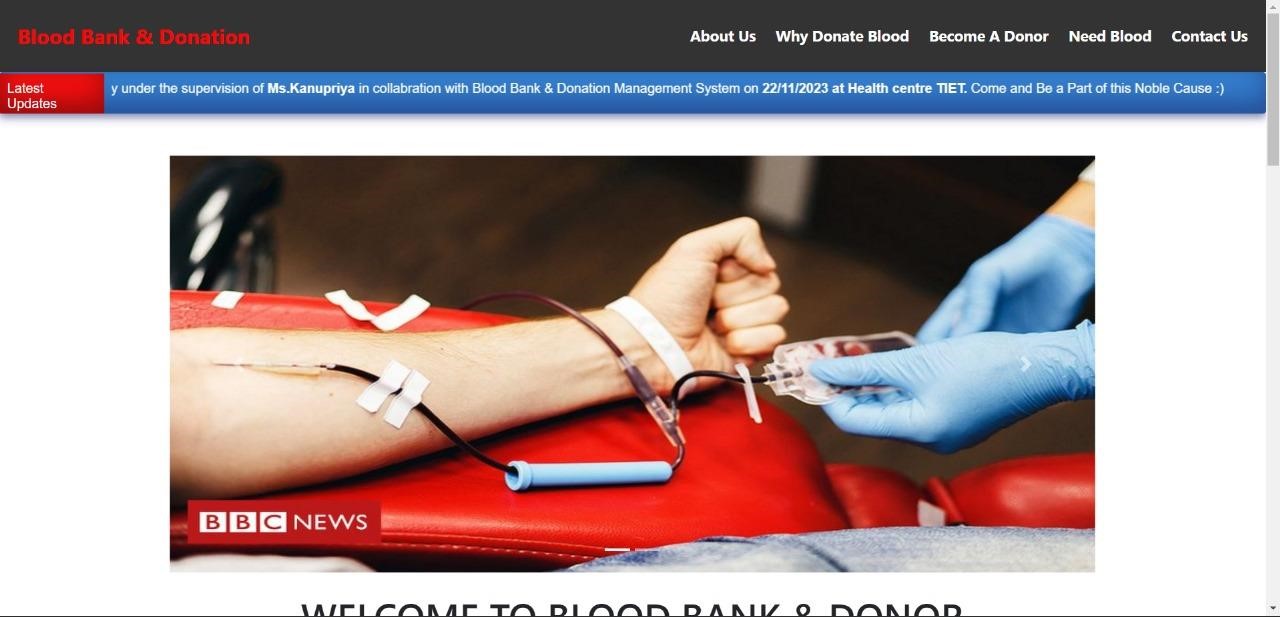
* Processor: Intel Core i5
* RAM: 4GB
* Hard Disk: 512GB

**2.4. User Story Card**

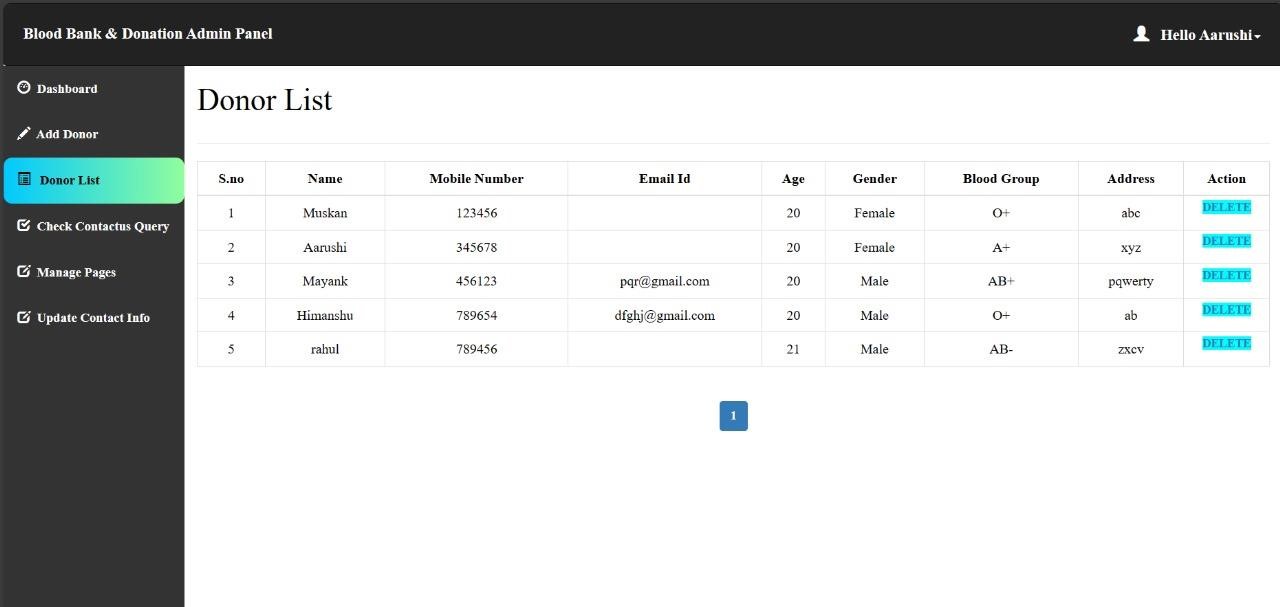
## Admin Login Page



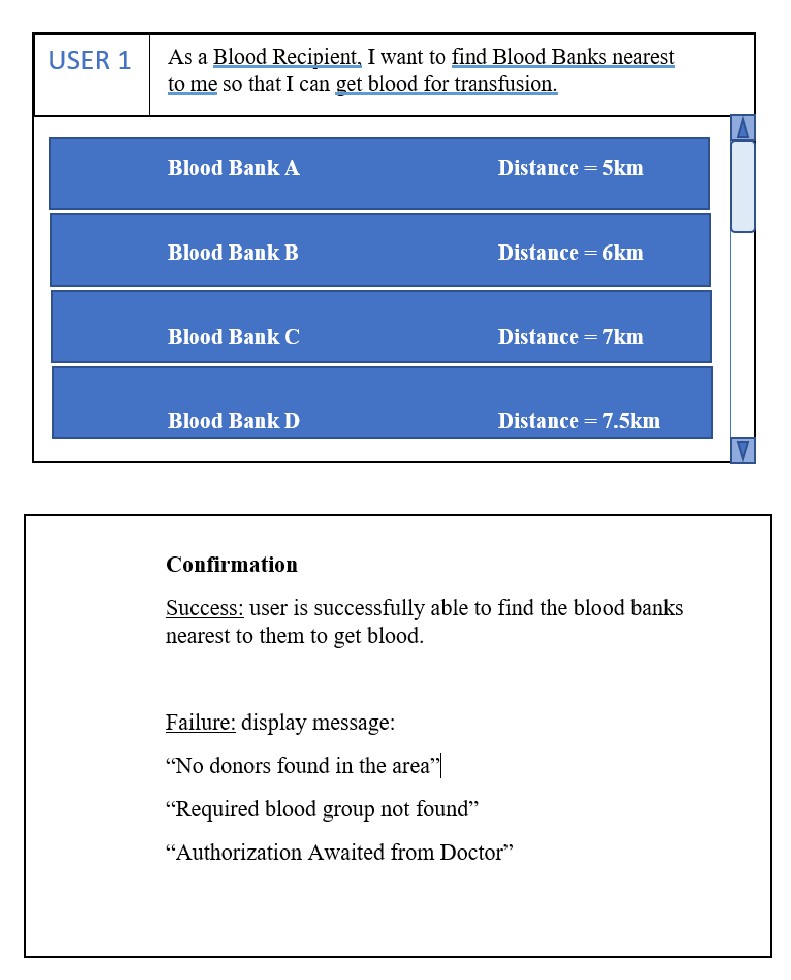
## Viewer Page



## Donor Database

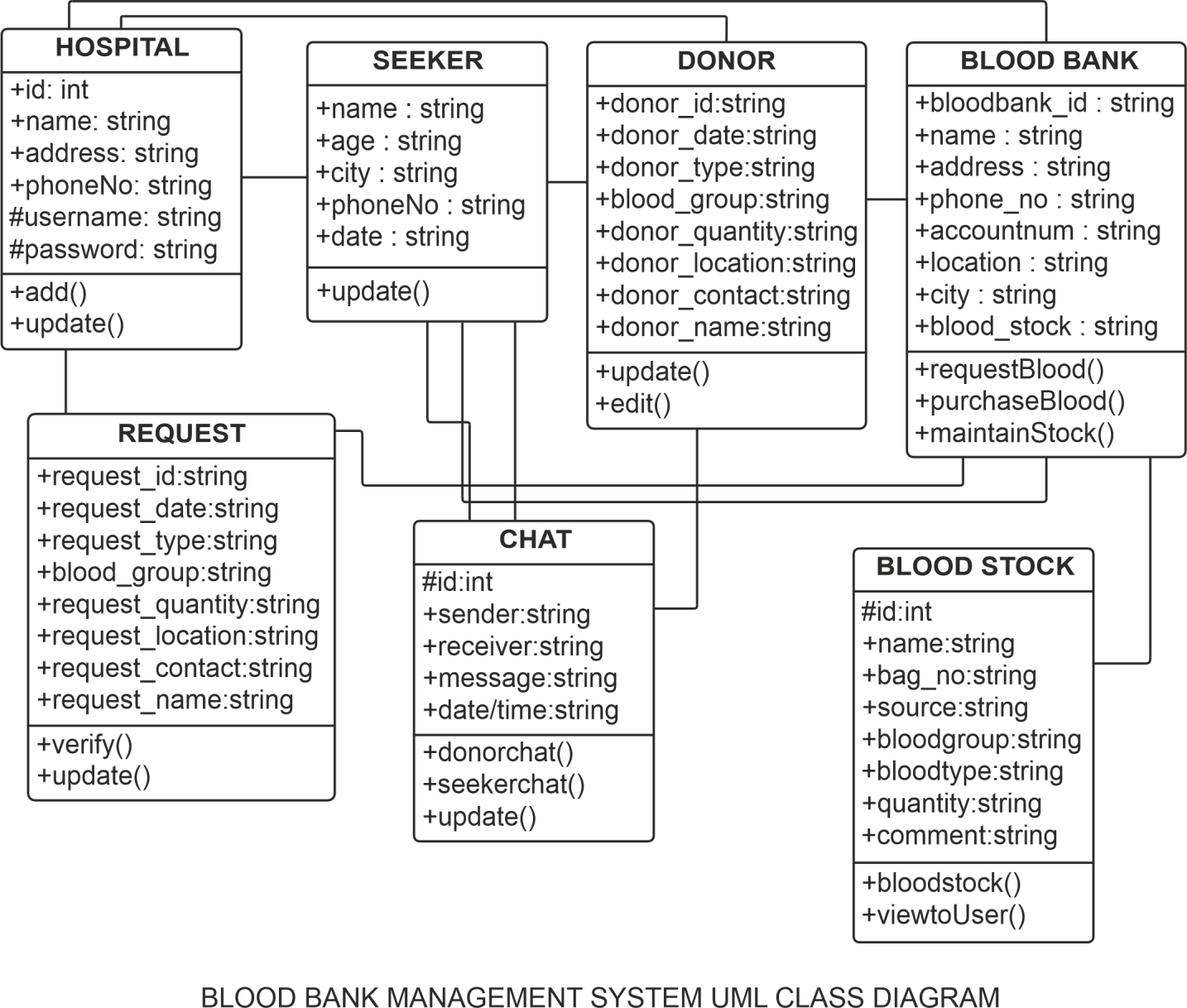


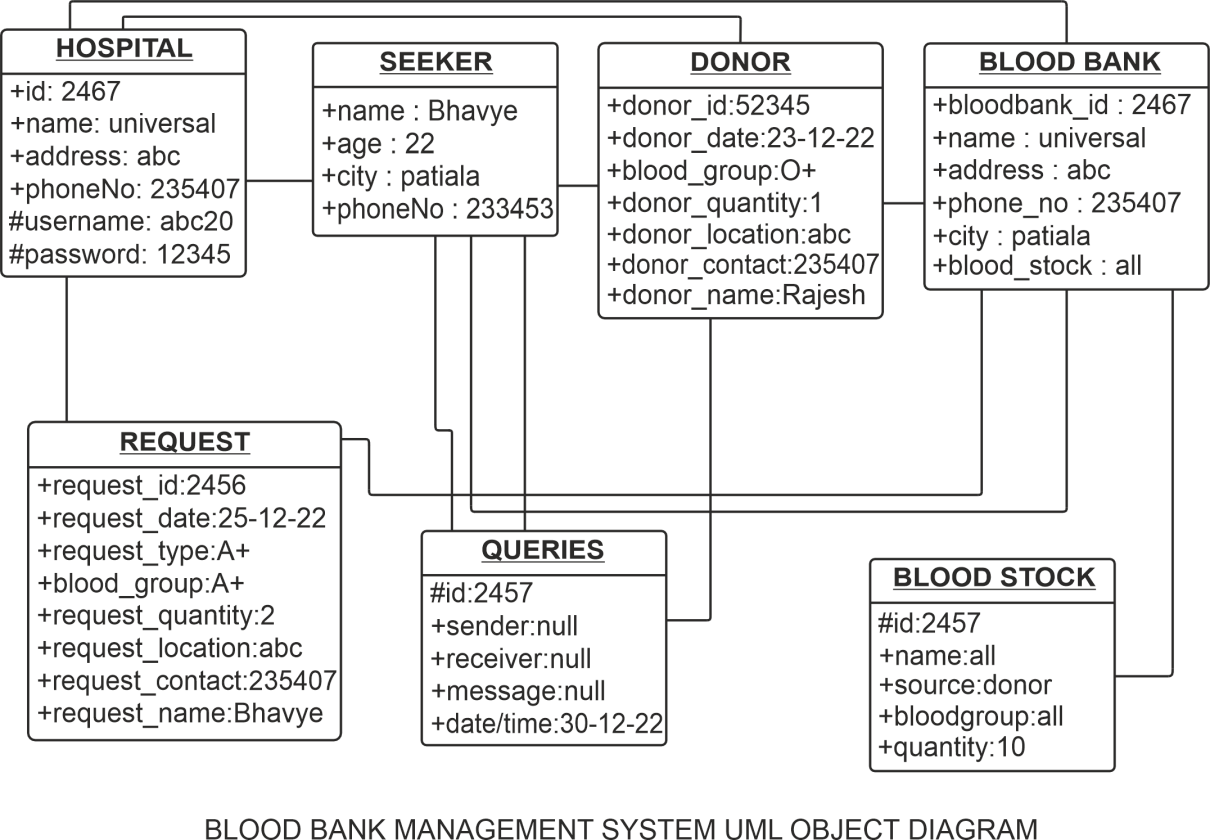
## Previous Idea



## 3. Design Phase

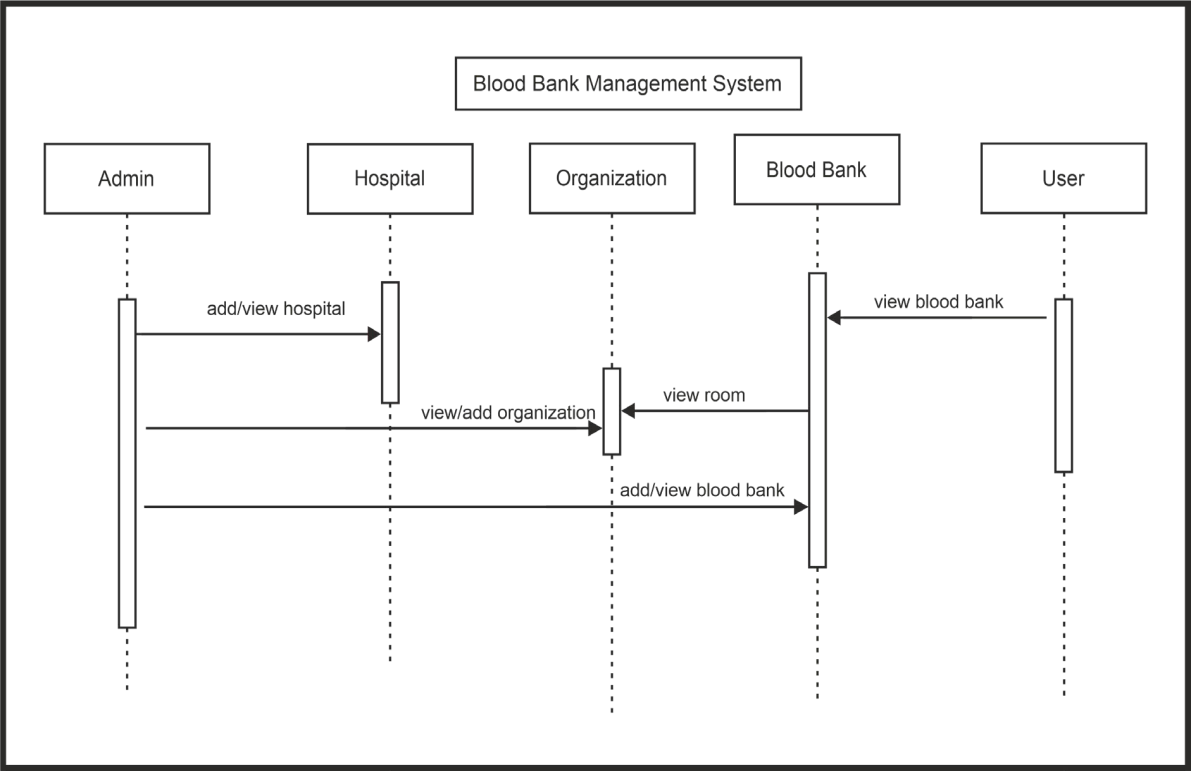
**3.1. Class Diagram and Object Diagram**





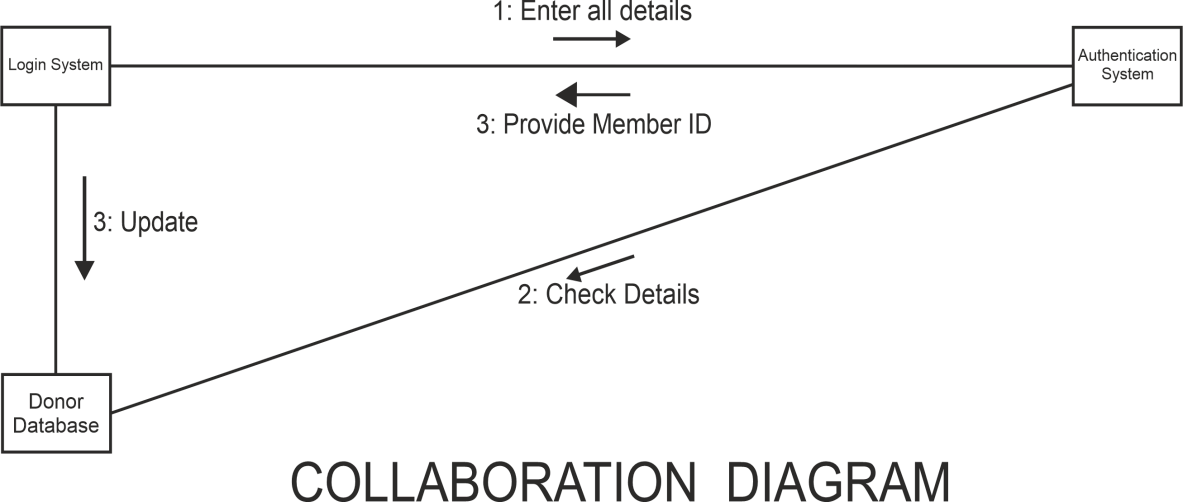
**FIG 3.1**

### 3.2. Sequence Diagram



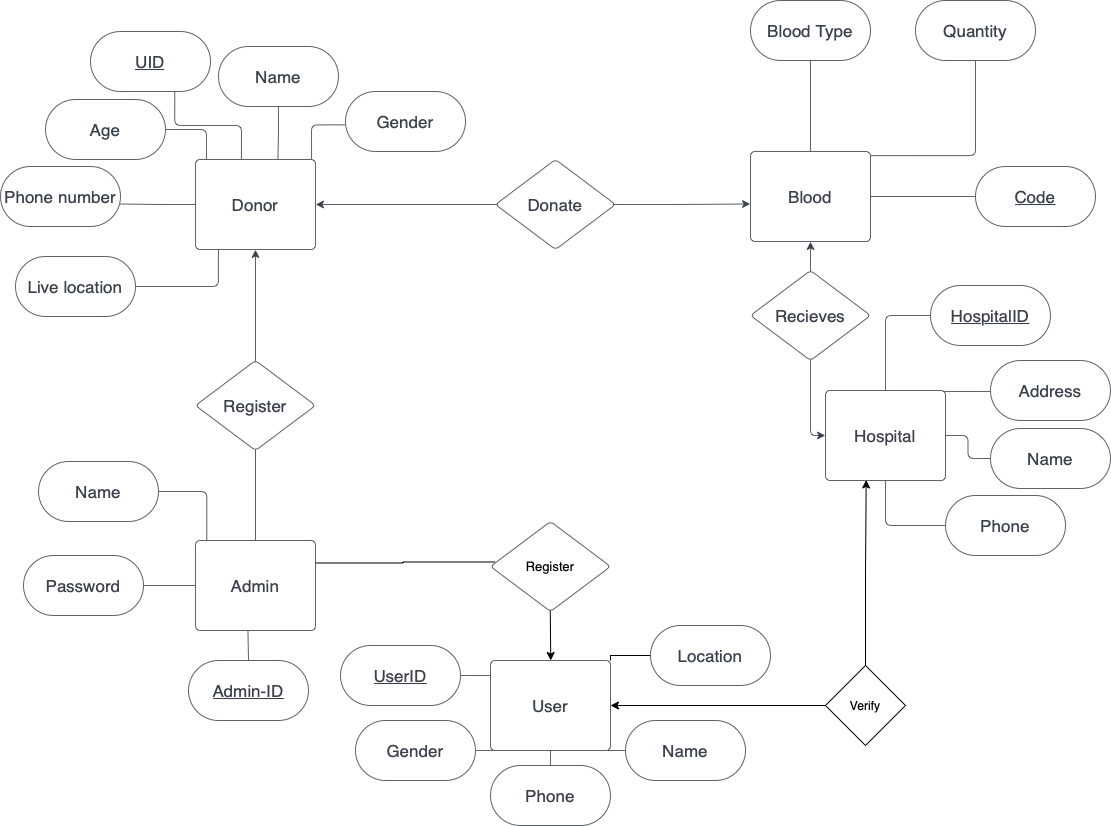
**FIG 3.2**

### 3.3. Collaboration Diagram



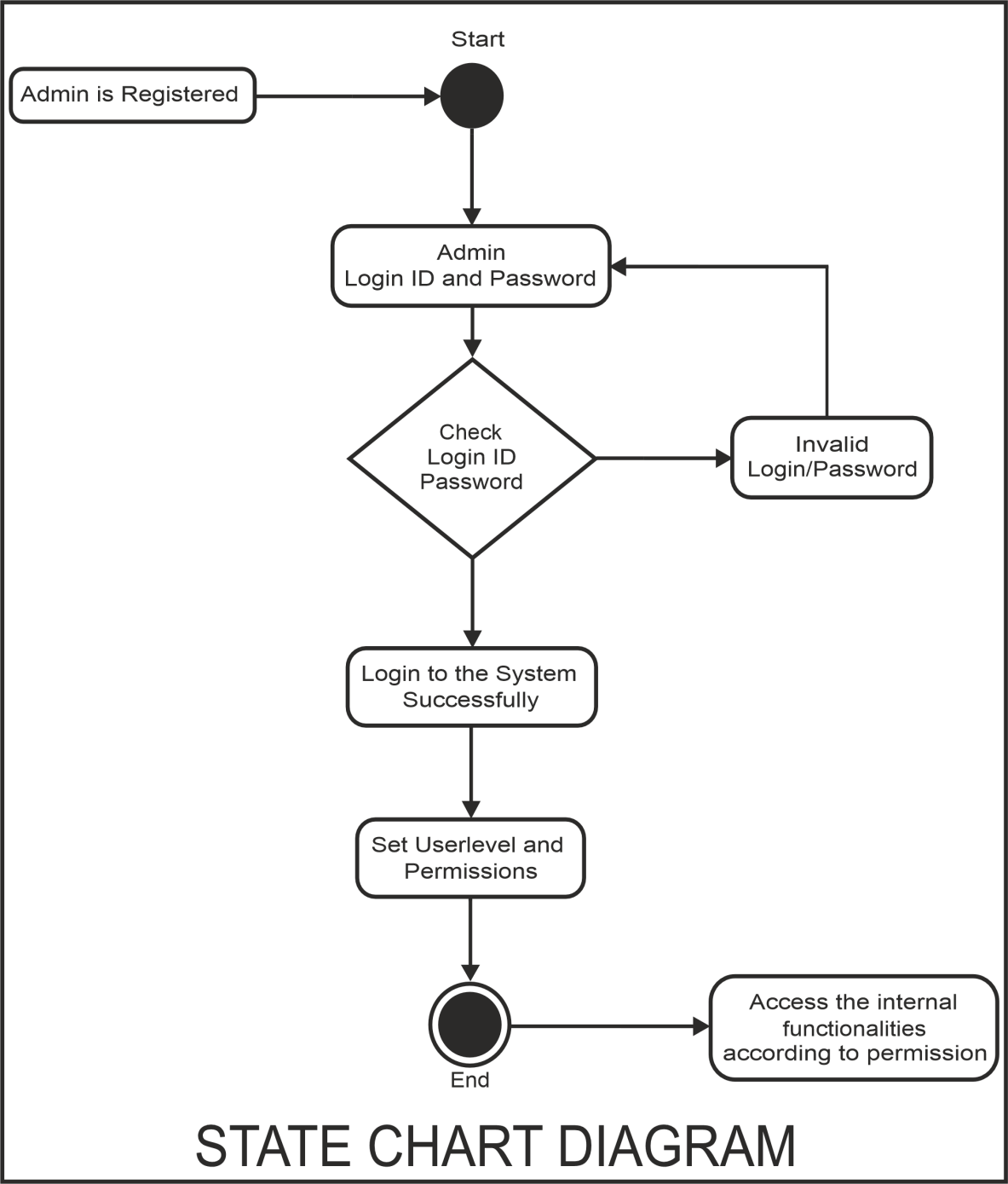
**FIG 3.3**

### 3.4. Database Design: ER Diagram



**FIG 3.4**

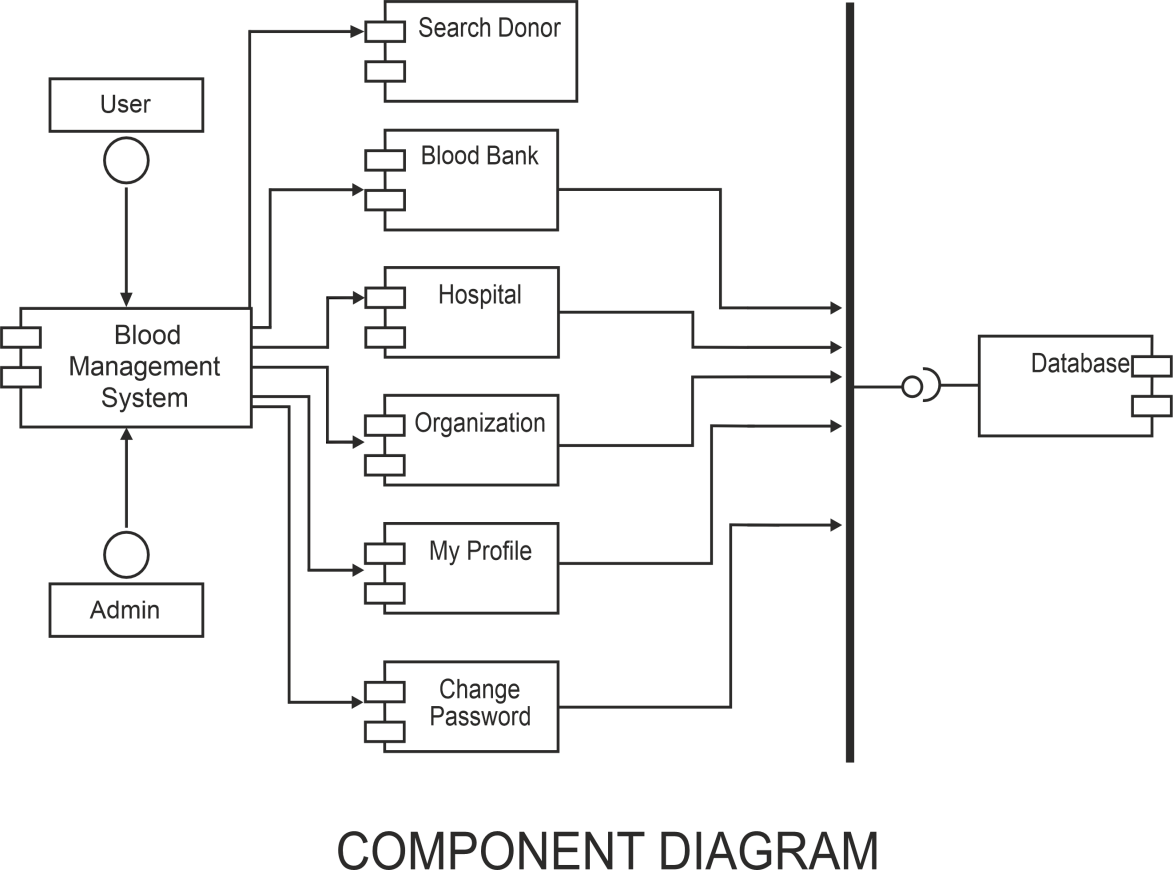
### 3.5. State Chart Diagram



**FIG 3.5**

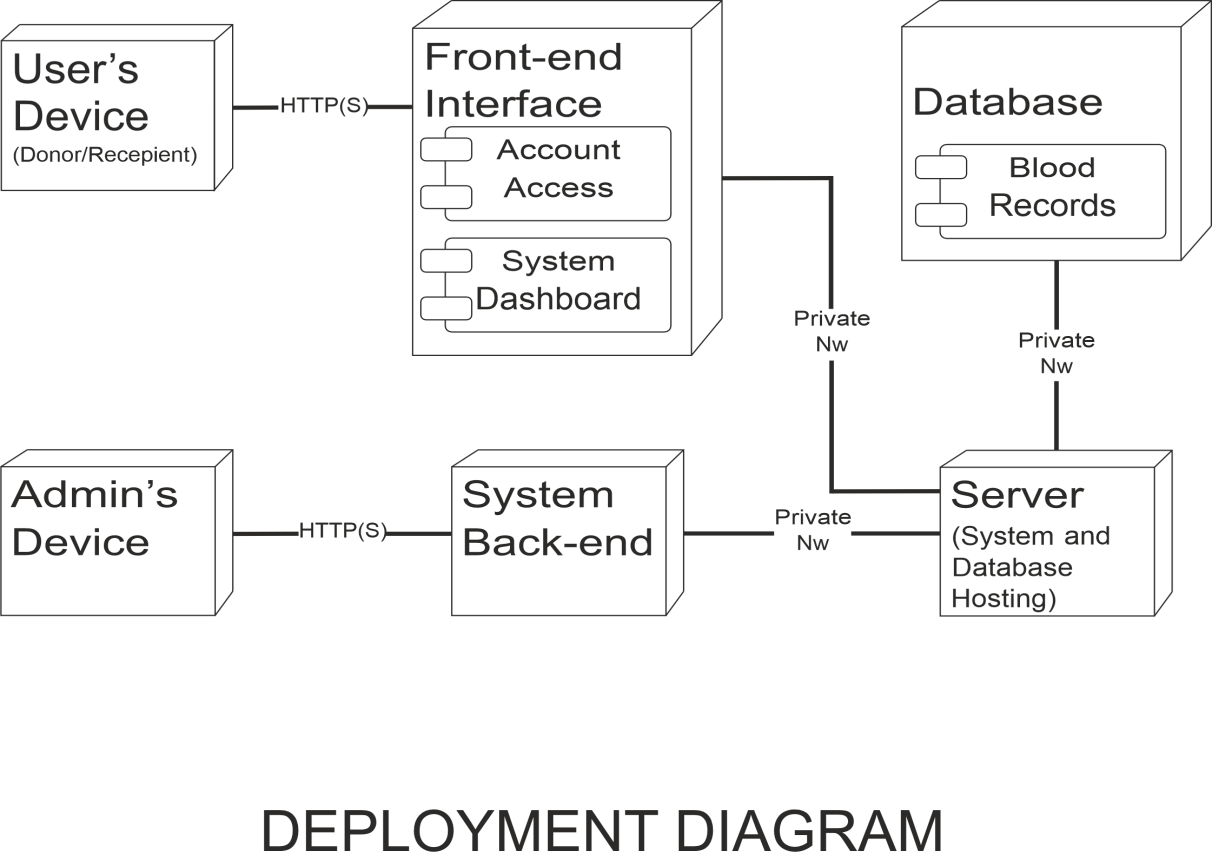
## 4. Implementation

### 4.1. Component Diagram



**FIG 4.1**

### 4.2. Deployment Diagram



**FIG 4.2**

### 4.3. Screenshots of working project

