Aid Hive

(Blood Donation Management System)

Under the supervision of

Prof. Imran Shafique

Bachelor of Science in CS/IT (2021-2025)

Department of Information Technology,

Government Islamia College,

Gujranwala

Aid Hive

A project presented to

Government Islamia College, Gujranwala

In partial fulfilment

of the requirement for the degree of

Bachelor of Science in Information Technology (2021-2025)

By

Abdur Rehman	2021-ICG-663
Ammad Ahmad	2021-ICG-97
Ali Hassan	2021-ICG-120
Hafiz Maaz Ahmad Siddiqi	2021-ICG-117

Bachelor of Science in CS/IT (2021-2025)

Department of Information Technology,

Government Islamia College,

Gujranwala

DECLARATION

We, the undersigned members of the project team, hereby confirm that the software system titled "Aid Hive: Blood Donation Management System", along with its documentation, is our own original work. We also confirm that this project, in whole or in part, has not been plagiarized from any source, nor has it been submitted previously for any degree, diploma, or qualification at this or any other university or institution of learning. In the event any part of this work is determined to be a plagiarized work or copy of any other project, we take full responsibility for the same. This project and report have been designed and prepared by our team from scratch under the guidance of our revered supervisor.

Signature: --------------------------	Signature: --------------------------
Abdur Rehman 073151	Ammad Ahmad 056503
Signature: --------------------------	Signature: --------------------------
Ali Hassan 056427	Hafiz Maaz Ahmed Siddiqi 056430

CERTIFICATE OF APPROVAL

It is to confirm that the final year design project (FYDP) of BS-IT "Blood Donation Management System" was prepared by **Abdur Rehman (2021-ICG-663), Ammad Ahmad (2021-ICG-97), Ali Hassan (2021-ICG-120),** and **Hafiz Maaz Ahmad Siddiqi (2021-ICG-117)** under the guidance of **"Prof. Imran Shafique"** in my view; it is complete enough, in scope and quality, for the award of Bachelors of Science in Information Technology.

Signature: --------------------------------------

FYDP Supervisor: Prof. Imran Shafique

Signatures (Faculty Advisory Committee (FAC)

Signatures			
Name			

Signature: --------------------------------------

Head of FYDP Coordination Office:

Signature: -------------------------------------- **Dated: ________________**

Chairperson, Department of Information Technology

Executive Summary

Aid Hive is a web-based application that connects donors of blood and recipients. It is built on the MERN stack, whereas MySQL is implemented as the database for secure and dependable data storage. The application has five primary pages: Home, Donate, Find Blood, Aid Hive Workers, and About Us, in addition to an Aid Hive Admin panel, which can be accessed by login. Donors and recipients can only access the system after filling up the registration form to provide authenticity and safety. The system is easy, effective, and user-friendly to save time, efficiently manage blood donations, and eventually save lives.

Acknowledgement

First and foremost, we express our gratitude to Allah Almighty whose blessings have enabled us to accomplish our Final Year Project successfully. We are truly grateful to our honorable supervisor, **Prof. Imran Shafique**, for his constant support, guidance, and encouragement. We are thankful for the assistance of **Mr. Usman Ahmed** for this project. We thank our teachers, classmates, and everyone who offered suggestions and feedback. Lastly, we appreciate our friends and families for their encouragement and support, which enabled us to carry out this project.

Signature: ---------------------------	Signature: ---------------------------
Abdur Rehman 073151	Ammad Ahmad 056503
Signature: ---------------------------	Signature: ---------------------------
Ali Hassan 056427	Hafiz Maaz Ahmad Siddiqi 056430

Table of Contents

List of Table

List of Figures

Chapter 1
Introduction

1. Introduction

1.1 Problem Statement

In critical cases, the majority of patients suffer from long delays in getting blood donations since there is no platform with real-time capabilities. The current platforms lack real-time donor location tracking, priority notifications, and instant communication between donors and recipients. Our project Aid Hive is likely to bridge this gap with a trustworthy platform that provides real-time cap abilities for connecting donors and patients.

1.2 Problem Solution

To overcome the issue of delayed blood donation, we have created Aid Hive as an online real-time platform where the patients and donors are matched first. Our system has real-time location tracking, emergency alert, and secure messaging in a manner that the patients are helped immediately without any delay. The donor and patient easily locate the donors nearby, order the blood, and get instant alerts. The donors can respond instantly too, and both can chat in real-time. Aid Hive also monitors donations and has an easy, simple-to-use dashboard. With these features, our project removes loopholes from current systems and makes blood donation faster, smarter, and reliable, particularly in emergency cases.

1.3 Objectives of the Proposed System

Our project Aid Hive is a blood donation management system developed using the MERN stack and MySQL database. The main aim of this system is to possess a quick, easy, and safe way of bringing donors and recipients together. Our aim is to provide patients with an easy way to discover compatible donors based on blood type and geography. Recipients and donors can be registered in the system by completing forms without going through a complex login or signup process. Logging in to manage the site is privileged only by the admin. The system also has an online location tracking through Google Maps, where patients can conveniently locate nearby donors or Aid Hive workers. Along with this, emergency requests will be relayed through real-time alerts and notifications, which will ensure prompt response and faster communication.

1.4 Scope

The functionality of our project Aid Hive is to provide an online blood donation management system that makes the process easier and faster. The system is built using the MERN stack, and MySQL is used as the database to store donor and receiver details. Our project contains five main pages: Home, Donate, Find Blood, Aid Hive Workers, and About Us. It also encompasses two principal buttons: Aid Hive Admin login and Live Location Tracking. They make the system easy to use such that it becomes extremely user-friendly and simple to operate.

1.5 System Components

The Aid Hive system is made up of different components that work together to provide blood donation management.

1.5.1 Pages of the System

The system has five main pages:

- **Home Page:** Gives an overview and easy navigation.
- **Donate Page:** Donors fill a form with their details to register as donors.
- **Find Blood Page:** Recipients fill a form to request blood.
- **Aid Hive Workers Page:** Shows information and live location of workers.
- **About Us Page:** Provides project and team information.

1.5.2 Buttons of the System

The system has only one main button:

- **Aid Hive Admin Button:** Only for admin login.

1.5.3 Donor and Recipient Forms

Donors and recipients can join by filling forms with their details. No login is required for them.

1.5.4 Admin Login

Login is only available for the admin. The admin can manage donor/recipient data and oversee the system.

1.5.5 Database (MySQL)

The system uses MySQL database to store donor details, blood requests, and history records safely.

1.6 Related System Analysis/Literature Review

There are several online systems for donating blood that are already in use, but they do not fulfil the emergency requirements efficiently. In Pakistan and other countries, none of these available systems are well-equipped or user-friendly. The following is a brief overview of some of these existing systems and how our project Aid Hive is better.

Table 1: Related System Analysis with proposed project solution

System / Website	**Main Features**	**Limitations**	**Aid Hive Improvements**
Sehat Kahani Blood Services	Connects hospitals and patients for arranging blood donations.	Focused on hospitals only, no real-time donor location, and not open for all users.	Aid Hive is open for everyone and allows direct connection between donor and recipient.
Red Cross Blood Services	Large-scale international donor management system with hospital integration.	Complicated system, not user-friendly for local or individual needs, requires multiple steps.	Aid Hive is lightweight, simple, and localized for quick use in all over the Pakistan.
Aid Hive (Our System)	Donor and recipient form filling, MySQL database, 5 web pages, and admin login.	Does not include automated push notifications or SMS alerts.	Focused on simplicity, visibility, and easy access.

1.7 Vision Statement

Aid Hive's vision is to create a reliable, accessible, and technology-driven blood donation management system that brings donors and recipients together in real time. Through integrating safe storage space for data, live tracking of location, and efficient matching of recipients and donors, the system aims to reduce delays in the event of an emergency, provide transparency, and promote a culture of voluntary blood donation. Aid Hive will, in the near future, extend its platform to hospitals, blood banks, and medical organizations and become an online trusted portal for lifesaving blood donation services.

1.8 System Limitation and Constraints

1.8.1 System Limitations:

1. The system does not provide real-time alerts or notifications. Users must check the platform themselves.
2. Only the admin has a login system. Donors and recipients can only fill forms; no personal accounts are created for them.
3. The system works only with an active internet connection. Without internet, it cannot be used.
4. The database is limited to MySQL, which may face challenges if the system is scaled to a very large number of users.
5. There is no direct integration with hospitals or blood banks at this stage.

1.8.2 System Constraints:

1. ***Technology Constraint:***
   The system is built using the MERN stack with MySQL database, so it is limited to web-based usage only.
2. ***Connectivity Constraint:***
   Continuous internet connection is required for form submission, database access.
3. ***Hardware Constraint:***
   Users need a device (desktop, laptop, or smartphone) with a browser and GPS support for location tracking.

4. ***User Constraint:***

   Only the admin has login access. Donors and recipients can interact with the system only through form filling.

5. ***Data Constraint*:**

   The system stores information only in the MySQL database, which requires regular maintenance and optimization when data grows larger.

1.9 Tools and Technologies

The following tools and technologies were used to develop Aid Hive (Blood Donation Management System):

1.9.1 Frontend

- **React.js:** For building the user interface.
- **CSS:** For styling and layout of web pages.

1.9.2 Backend

- **Node.js:** For server-side logic.
- **Express.js:** For connecting frontend with backend.

1.9.3 Database

- **MySQL:** To store and manage donor and recipient information.

1.9.4 Development Tools

- **Visual Studio Code (VS Code):** Code editor.
- **npm (Node Package Manager**): For installing and managing dependencies.
- **GitHub:** For version control and team collaboration.

Table 2: Tools and Technologies for Proposed Project

Category	**Technology / Tool**	**Purpose**
Frontend	React.js	To build the user interface of the system.
	CSS	For styling and layout of web pages.
	Framer Motion	For smooth animations and transitions.
Backend	Node.js	For server-side scripting and logic.
	Express.js	To handle APIs and connect frontend & backend.
Database	MySQL	To store and manage donor and recipient records.
Libraries / APIs	Axios	To handle HTTP requests between frontend and backend.
Development	Visual Studio Code (VS Code)	Code editor for development.
	npm (Node Package Manager)	To install and manage project dependencies.
	GitHub	For version control and collaboration.
Security	Bcrypt	For Hash Passwords.
	Prepared Statements	For SQL injection
	.env	Improves security for easily config. And deploy.

1.10 Project Planning

Table 3: Project Planning Gantt Chart

Start Date					January	Feb	March	April	May	June
Task No.	**Task**	**Supervisor**	**Start Date**	**End Date**						
1	Project Proposal	Prof. Imran Shafique			▨					
2	SRS and SDS	Prof. Imran Shafique				▨				
3	RTM	Prof. Imran Shafique				▨				
4	Frontend	Prof. Imran Shafique				▨	▨			
5	Backend	Prof. Imran Shafique					▨	▨		
6	DB Integration	Prof. Imran Shafique						▨	▨	
7	Test Plans	Prof. Imran Shafique							▨	
8	Test Report	Prof. Imran Shafique							▨	▨

1.11 Summary

The Aid Hive Blood Donation Management System has been created to counteract the problems of traditional blood donation such as latency, unverified information, and lack of convenience while searching for appropriate donors. It provides an online-based system where donors and recipients can register through forms, and the system can be easily managed by admins securely. It consists of five main pages (Home, Donate, Find Blood, Aid Hive Workers, About Us) and one main feature button (Admin Login). Built on the MRN stack with MySQL as the database, Aid Hive brings about efficiency, accessibility, and reliability in blood donation activity management.

Chapter 2
Requirements Analysis

2. Analysis

Analysis stage is involved with understanding the issue with the existing blood donation procedures and defining how the envisaged system will resolve them. It is involved with reviewing existing limitations, identifying the user requirements, and defining the functional and non-functional requirements of the Aid Hive Blood Donation Management System. This stage ensures that system design is based on clear-cut objectives, user requirements, and with regard for feasibility.

2.1 User Functions and Characteristics

The Aid Hive Blood Donation Management System is designed to serve three main categories of users: Donors, Recipients, and Admins. Each user type has specific functions and characteristics that define their role in the system.

1. Donor

- **Functions:**
  - Register on the website by filling out the donor registration form.
  - Provide required details such as name, age, gender, blood group, contact information, etc.
  - Become available in the system's database for recipients to search.
- **Characteristics:**
  - Donors are voluntary individuals who are willing to donate blood.
  - They require a simple and user-friendly registration interface.
  - Their data must be stored securely and made accessible to recipients when required.

2. Recipient

- **Functions:**
  - Register on the website by filling out the recipient form with necessary details (name, required blood group, contact number, etc.).

  - Search for available donors based on blood group and geographical location.
  - Request blood through the system in urgent situations.
- **Characteristics:**
  - Recipients are usually patients or attendants in urgent need of blood.
  - They expect quick search results and accurate donor information.
  - Their requests must be handled efficiently to avoid delays in emergencies.

3. Admin

- **Functions:**
  - Log in securely through the admin login page.
  - Manage donor and recipient records in the MySQL database.
  - Approve, monitor, or delete any donor or recipient entries if needed.
  - Ensure that the data remains updated, accurate, and secure.
- **Characteristics:**
  - Admins are authorized personnel with full system access.
  - They are responsible for maintaining the privacy and integrity of the system.
  - They must have technical knowledge to handle system operations effectively.

Table 4: System Functions and User Characteristics

User Type	Functions	Characteristics
Donor	• Fill out donor registration form • Provide details (Name, Age, Gender, Blood Group, Contact, etc.) • Appear in system database for matching with recipients	• Voluntary individuals willing to donate blood • Require a simple and user-friendly interface • Information must be stored securely and accessed quickly
Recipient	• Fill out recipient registration form • Search donors by blood group and location • Submit request in urgent cases	• Usually patients/attendants in urgent need of blood • Expect fast response and accurate donor details • Depend on reliable donor availability
Admin	• Log in securely using admin credentials • Manage donor and recipient records • Approve, update, or remove entries • Monitor blood requests and system activity	• Authorized personnel with full system access • Responsible for data accuracy, privacy, and security • Must have technical knowledge to operate system

2.2 Requirement Identifying Technique

Requirement identifying is the process of determining what the system must do to meet user needs effectively. For the Aid Hive Blood Donation Management System, the following techniques were used to identify and gather requirements:

1. **Interviews with Stakeholders**
    - Discussions were held with potential donors, recipients, and administrators to understand their expectations and challenges.
    - Questions focused on difficulties in current blood donation processes, preferred features, and usability requirements.
2. **Observation of Existing System**
    - Studied current blood donation practices in hospitals and blood banks.
    - Identified delays in donor-recipient communication, lack of centralized data, and inefficiency in emergency cases.
3. **Questionnaires/Surveys**
    - Distributed online and offline forms to gather inputs from volunteers and patients.
    - Collected data on desired features, accessibility preferences, and common issues faced.
4. **Brainstorming Sessions**
    - Group discussions among project members to decide which features are essential.
    - Prioritized functionalities like admin login and donor/recipient forms.
5. **Review of Existing Literature & Systems**
    - Studied similar blood donation management systems and software to identify best practices.

- Noted features that were missing or needed improvement.

6. **Use Case Technique**

- The Use Case technique helps analyze how users interact with the system in real-life scenarios.
- It allows the team to define clear, user-focused features by relating each user action to the corresponding system behavior.
- This approach ensures that functional requirements are accurately specified, complete, and aligned with actual user needs.

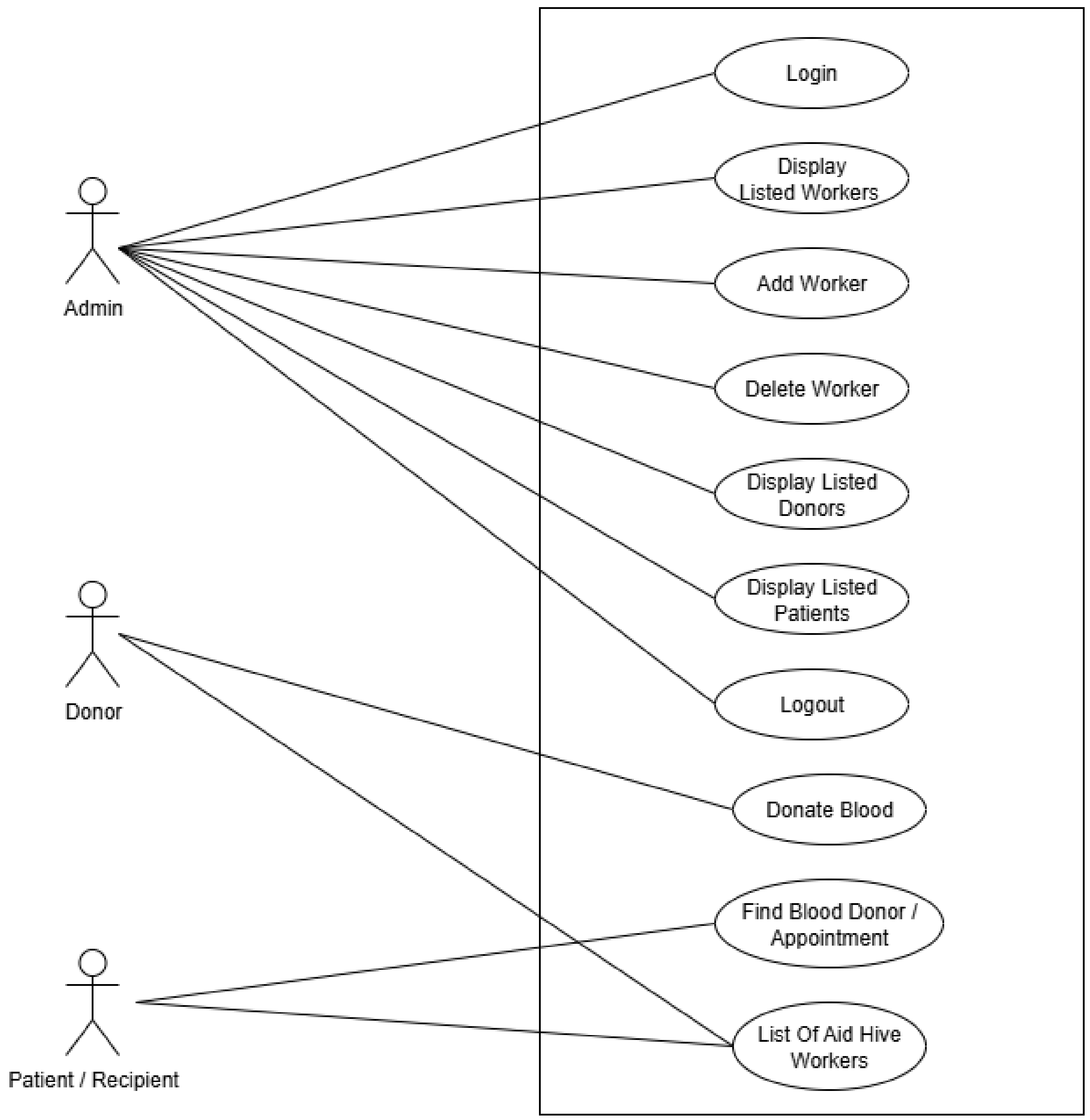
Outcome:

By using these techniques, the project team clearly defined functional and non-functional requirements, ensuring that the system effectively serves donors, recipients, and administrators.

Table 5: Use Case List

Use Case	Actors	Description
Login	Admin/User	Admin or user logs into the system using valid credentials.
Display Listed Workers	Admin	Admin views the list of registered Aid Hive workers.
Add Worker	Admin	Admin adds a new worker to the system.
Delete Worker	Admin	Admin deletes an existing worker from the system.
Display Listed Donors	Admin	Admin views the list of registered blood donors.
Display Listed Patients	Admin	Admin views the list of registered patients.
Logout	Admin/User	Admin or user logs out from the system.
Donate Blood	Donor	Donor initiates a blood donation request.
Find Blood Donor	Patient/Recipient	Patients search for available donors and request appointments.
List Of Aid Hive Workers	Patient/Recipient	Patients or recipients view the list of Aid Hive workers for assistance.

Figure 1: Use Case of Aid Hive



2.3 Functional Requirements

The most important functional requirements of the Aid Hive system. Requirements are grouped by system functions or features with which the users interact. They are developed to address user requirements (donor and recipient) and allow for smooth, consistent, and secure deployment of the application.

2.3.1 Functional Requirement 1

Table 6 :Donor Registration

Identifier:	FR-1
Title:	Donor Registration
Description:	Allow donors to register via an online form by providing personal and contact details
Input Fields:	Name, Age, Gender, Blood Group, Contact Number, etc.
Processing:	Validates form data and stores donor information securely in the MySQL database.
Output:	Confirmation message of successful registration; donor added to the database for recipient search.
Dependencies:	MySQL database for storing data; web form frontend for input; server-side backend for processing.
Priority:	High

2.3.2 *Functional Requirements 2*

Table 7 :Recipient Registration

Requirement ID:	FR-2
Requirement name:	Recipient Registration
Description:	Allows recipients to register their details and blood requests via an online form.
User Type:	Recipient
Input Fields:	Name, Required Blood Group, Contact Number, etc.
Processing:	Validates form data and stores recipient information securely in the MySQL database.
Dependencies:	MySQL database for storing data; web form frontend for input; server-side backend for processing.
Priority	High

2.3.3 *Functional Requirement 3*

Table 8 : Search Donors

Requirement ID:	FR-3
Requirement name:	Search Donors
Description:	Enables recipients to search for available donors based on blood group and location.
User Type:	Recipient
Input Fields:	Blood Group, Location (City/Area)
Processing:	Matches recipient search criteria with donor records stored in the MySQL database.
Output:	List of matching donors with contact details and location info.
Dependencies:	MySQL database for storing donor data; frontend search interface; backend processing for query matching.

2.3.4 Functional Requirement 4

Table 9 : Admin login

Requirement ID:	FR-4
Requirement name:	Admin Login
Description:	Provides secure login access for administrators to manage system operations.
User Tye:	Admin
Input Fields:	Username, Password
Processing:	Validates admin credentials against records in the MySQL database and grants access to the admin dashboard.
Output:	Access to the admin panel for managing donors, recipients, and system activities.
Dependencies:	MySQL database for storing admin credentials; secure backend authentication; frontend login interface.

2.3.5 Functional Requirement 5

Table 10 : Manage Records

Requirement ID:	FR-5
Requirement name:	Manage Records
Description:	Allows administrators to view, approve, update, or remove donor and recipient records in the system.
User Tye:	Admin
Input Fields:	Record ID, Action Type (Approve / Update / Delete), Updated Data (if applicable)
Processing:	Performs the selected action on the chosen record in the MySQL database and ensures data integrity.
Output:	Confirmation of successful action (record approved, updated, or deleted).
Dependencies:	Admin login authentication; MySQL database; backend processing for CRUD operations.

2.3.6 *Functional Requirement 6*

Table 11 : Database Management

Requirement ID:	FR-7
Requirement name:	Database Management
Description:	Stores and manages all donor, recipient, and admin information securely in the MySQL database.
User Tye:	All Users
Input Fields:	Donor/Recipient/Admin details entered via forms or admin panel
Processing:	Validates, stores, retrieves, updates, and deletes records as per user actions; ensures data integrity and security.
Output:	Accurate storage and retrieval of user information; updates reflected in the system in real-time.
Dependencies:	MySQL database; backend server for database operations; secure access control.

2.4 Non-Functional Requirements

These are quality requirements for how well the Aid Hive system works and performs, not for what the system performs. They are reliability, usability, performance, and security expectations.

2.4.1 *Reliability*

It should run without incessant crashing. It should recover from sudden mistakes in a graceful manner without losing information. It should have backup protocols so that essential operations such as posting of blood orders are not lost.

2.4.2 *Useability*

Aid Hive must be simple and accessible to everyone, even to non-technical individuals. The most essential functionalities must be within reach in two or three steps, and the interface must be based on known standards and guidelines with accessibility attributes.

2.4.3 Performance

The system should react to user activities in 2 seconds. It should support 500 simultaneous users and remain responsive even on low-bandwidth cellular networks.

2.4.4 Security

The Aid Hive system ensures security by applying multiple protection mechanisms. User passwords are stored safely using bcrypt hashing. The login process is protected against brute-force attacks through express-rate-limit. SQL injection risks are minimized by using prepared statements with placeholders. Additionally, environment variables are managed with dotenv to keep sensitive information secure.

2.5 External Interface Requirements

Here's some external interface requirements are as follows:

2.5.1 User Interfaces Requirements

The web application shall provide an intuitive, responsive, and accessible user interface for the following:

- **Donors can:**
  - Register/login
  - Update their profiles
  - See blood requests
- **Recipients can:**
  - Search/filter donors by blood group and location
  - Submit a blood request
  - Track request status
- **Admin Panel:**
  - Firstly, login then connect to Aid Hive
  - Accept and reject requests
  - Manage users (donors and recipients)

UI will be built using HTML, Tailwind, JavaScript (React in MERN), and will be fully responsive for all screen sizes. Forms will include validation, tooltips, and error handling.

2.5.2 *Software interfaces*

The Aid Hive Blood Donation Management System uses multiple software components to ensure smooth functioning and interaction between different modules. The key software interfaces are:

1. **Front-End Interface (MERN Stack – React & Node.js)**
    - Provides a user-friendly interface for donors and recipients to fill registration forms and search for blood.
    - React ensures dynamic web pages and smooth navigation.
    - Node.js handles requests and communicates with the backend.
2. **Back-End Interface (Express.js)**
    - Acts as the middleware to process requests between the front-end and the database.
    - Handles form submissions, validation, and API requests.
3. **Database Interface (MySQL)**
    - Stores donor, recipient, and admin data.
    - Interfaces with the backend to allow insertion, deletion, updating, and retrieval of records.
    - Ensures data consistency and integrity.
4. **Admin Panel Interface**
    - Allows only admins (through login) to manage donor and recipient records.
    - Provides access to view, approve, or remove entries.

2.5.3 *Hardware interfaces*

Since Aid Hive is a web-based application, it does not require specialized hardware. However, certain minimum hardware requirements are necessary for users (donors, recipients, and admins).

1. User Side (Donors & Recipients)

- **Device Type:** Desktop computer, laptop, tablet, or smartphone.
- **Minimum Requirements:**
  - Processor: 1.5 GHz or higher
  - RAM: 2 GB or higher
  - Storage: 200 MB free space for browser cache
  - Internet: Stable connection (at least 1 Mbps)
- **Interface:** Users interact with the system through a web browser.

2. Admin Side

- **Device Type:** Desktop computer or laptop.
- **Minimum Requirements:**
  - Processor: 2.0 GHz Dual-Core or higher
  - RAM: 4 GB or higher
  - Storage: 500 MB free space
  - Internet: Stable broadband connection
- **Interface:** Admin interacts with the system through a secure login portal.

3.5.4 Communications interfaces

The Aid Hive system relies on standard communication protocols to interact between users, admin, and the backend system.

1. Web Communication (HTTP/HTTPS)

- Users (donors and recipients) and admins access the system through standard web browsers using HTTP/HTTPS protocols.
- HTTPS ensures that all data transmitted between users and the server is encrypted and secure.

2. API Communication

- The front-end communicates with the back-end via RESTful APIs implemented in Express.js.
- APIs handle requests such as form submissions, donor searches, admin actions, and live location updates.

3. Database Communication

- The back-end communicates with the MySQL database using standard SQL queries for inserting, updating, retrieving, and deleting records.

2.6 Summary

In this chapter, we studied how the Aid Hive Blood Donation Management System interacts with its users and interfaces. Donors and recipients register exclusively through simple online forms, while only the admin has login access to manage records. The system relies on standard web communication (HTTP/HTTPS), a user-friendly front-end, and secure MySQL database connections. This analysis guarantees that the system remains easy to use, reliable, and responsive, especially in urgent blood donation scenarios where timely access to donors is critical.

Chapter 3

Design and Architecture

3. System Design

The system is based on a MERN stack with a MySQL database with an aim to handle the blood donations optimally. It has a React.js frontend that offers five pages (Home, Donate, Find Blood, Aid Hive Workers, and About Us) and a single critical button for Admin access. Donors and recipients are restricted to interact with the system by submitting registration forms, whereas the admin has to log in to view and manage the complete system features. The backend, developed using Node.js and Express.js, manages all API requests, user login, and database communication with MySQL.

3.1 Design considerations

The design of the system takes into account the scalability, security, and usability. User interfaces are minimalistic and intuitive such that the donor and the recipient can easily register and view information, and the admins get a secure admin login to run the system. The backend is modular with separate API routes for blood requests such that the system is easy to maintain and issues are properly segregated. There is secure and organized storage of data with MySQL.

Assumptions and Dependencies:

Here are the Assumptions and Dependencies in points format:

Assumptions:

- Donors and recipients have basic internet access.
- Users can accurately fill out registration forms.
- Admins are authorized personnel and manage the system responsibly.
- Users understand how to navigate the website.

Dependencies:

- MERN stack for frontend (React) and backend (Node.js + Express).
- MySQL database for structured and reliable data storage.
- Stable server environment for backend operations.
- Secure API endpoints for data communication.

- Accurate user input and timely admin updates to maintain data integrity.

Limitations:

- Donors and recipients can only interact with the system through registration forms; no advanced user features are available.
- Admin access is required for most operations, so system functionality is limited without admin intervention.
- The system does not support offline usage.
- Scalability may be limited by server capacity and database performance under high user load.
- Data accuracy depends on correct input from users and timely updates from admins.

Risks:

- **Data Security Risk:** Unauthorized access to user or donor information if authentication is compromised.
- **Data Accuracy Risk:** Incorrect or incomplete information submitted by users may affect system reliability.
- **System Downtime:** Server or database failures can make the system temporarily unavailable.
- **Scalability Risk:** High user traffic may slow down the system or overload the server.
- **Admin Dependency:** Most critical operations depend on timely admin actions; delays may affect system efficiency.

3.2 Design Models

The system design is modeled using component-based and layered architecture to clearly the interaction between frontend, backend, and database.

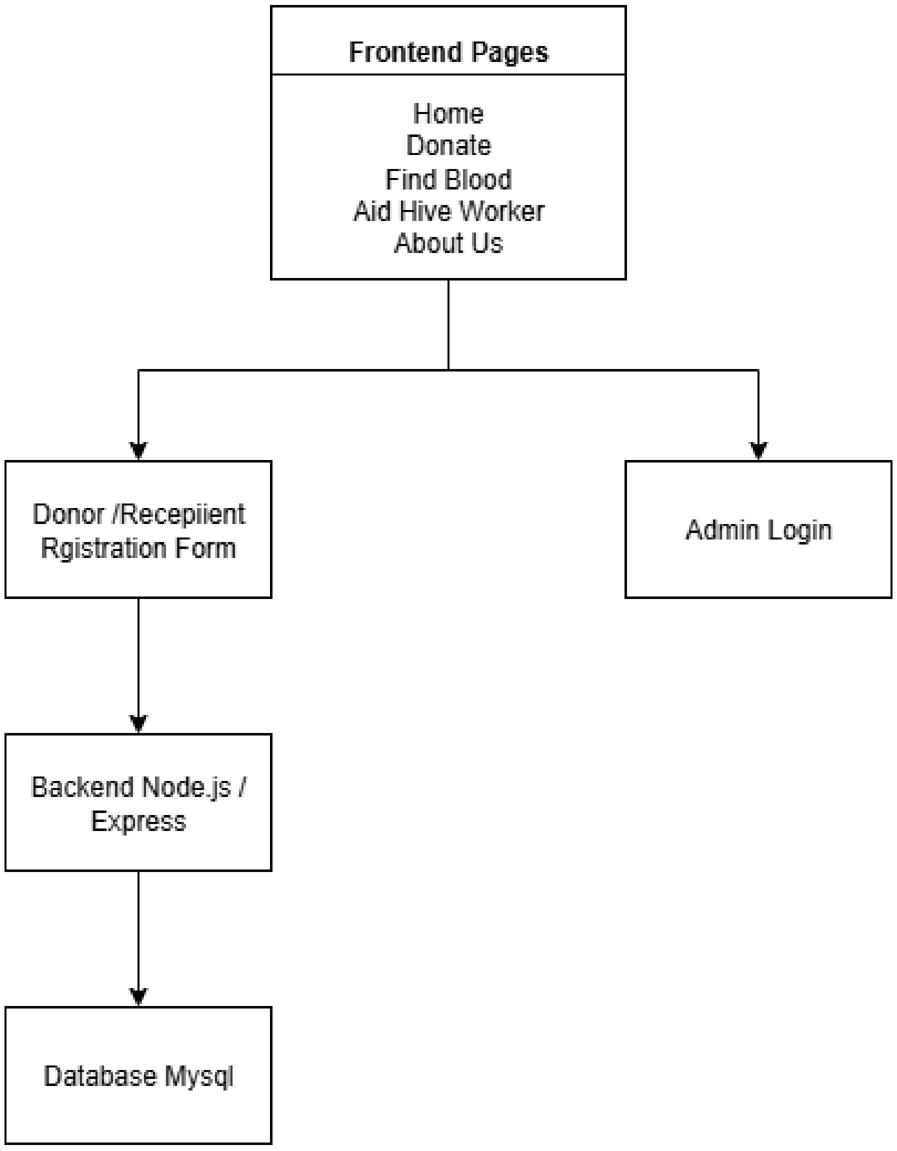
1. **Component Model:**

- **Frontend (React):** Handles user interface, pages (Home, Donate, Find Blood, Aid Hive Workers, About Us), buttons (Aid Hive Admin), and forms (Donor/Recipient registration).
- **Backend (Node.js + Express):** Processes API requests, manages authentication, handles business logic, and communicates with the MySQL database.
- **Database (MySQL):** Stores structured data including users, donations, blood requests, and worker information.

2. **Interaction Model:**
    - Donors/Recipients: Registration → Search/Donate → System Response.
    - Admin: Login → Manage Users/Donations/Workers → System Response.
3. **Data Flow Model:**
    - Users submit forms on the frontend → Backend APIs process the requests → Data stored or retrieved from MySQL → Responses sent back to frontend.
    - Admin actions (login, approvals, updates) follow the same flow with higher privileges.
    - This model ensures modularity, maintainability, and clear separation of concerns while supporting secure database interactions.

3.2.1 Functional Component Diagram

Figure 2: Component Diagram



1. User Module

The User Module is designed to manage interactions for both donors and recipients through registration forms. Users can access services without the need for login, keeping the process simple and user-friendly.

- **Registration Form:**
    - Donors and recipients provide personal details such as name, contact information, blood group, etc.

The form ensures all necessary information is collected to facilitate blood donation and requests.

- **Form Submission:**
    - Once submitted, the backend processes the data and stores it securely in the MySQL database.
- **Interaction with System:**
    - Donors can submit blood donation details.
    - Recipients can request blood and view available options.
    - Users cannot access administrative features; their interaction is limited to the forms and available search functionality.
- **Data Handling:**
    - All user-submitted data is validated for accuracy and stored in the database for further processing by the admin.

This module emphasizes ease of use, secure data collection, and minimal user barriers, allowing donors and recipients to participate in the system quickly and efficiently.

2. Request Module

The Request Module handles blood requests from recipients and coordinates them with available donors. It ensures that blood requirements are efficiently captured and processed.

- **Request Form:**

  - Recipients fill out a form specifying their blood group, required quantity, and location.

- **Form Submission:**
  - The backend receives the request data and stores it in the MySQL database.
  - Data validation ensures that all required fields are completed correctly.
- **Processing Requests:**
  - Admin can view submitted requests and match them with available donors.
  - The system updates request status (e.g., pending, approved, completed) for tracking.
- **User Interaction:**
  - Recipients can submit multiple requests but cannot directly access donor information.
  - The module ensures that requests are processed securely and efficiently while keeping user interaction simple.

This module provides a structured and reliable way to manage blood requests while maintaining data integrity and supporting the overall blood donation workflow.

3. Admin Panel

The Admin Panel serves as the main control hub for managing the blood donation system. Access is restricted and requires login using a valid username and password. Only authorized admins can perform operations within the panel.

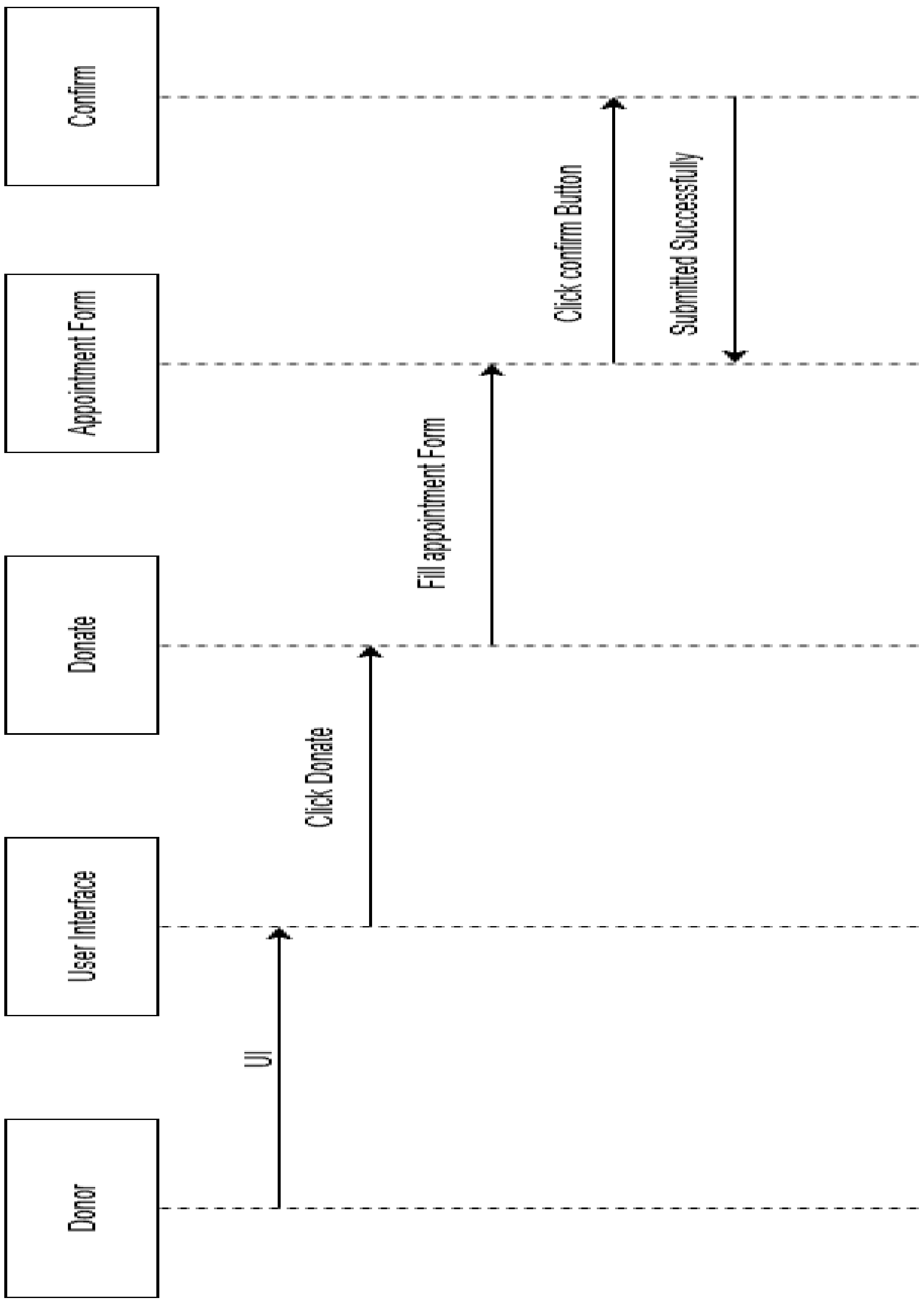
Key Features:

- **Login Authentication:**
  - Admin enters username and password to access the panel.
  - Ensures that only authorized personnel can manage the system.
- **User Management:**
  - View donor and recipient details submitted via registration forms.

  - Approve, update, or remove user records as necessary.
- **Request Management:**
  - Monitor and process blood requests submitted by recipients.
  - Assign donors to fulfill requests and update request status (pending, approved, completed).
- **Donation Tracking:**
  - Maintain records of donations submitted by donors.
  - Ensure accurate and secure storage in the MySQL database.

3.2.1 Donor Registration Sequence Diagram

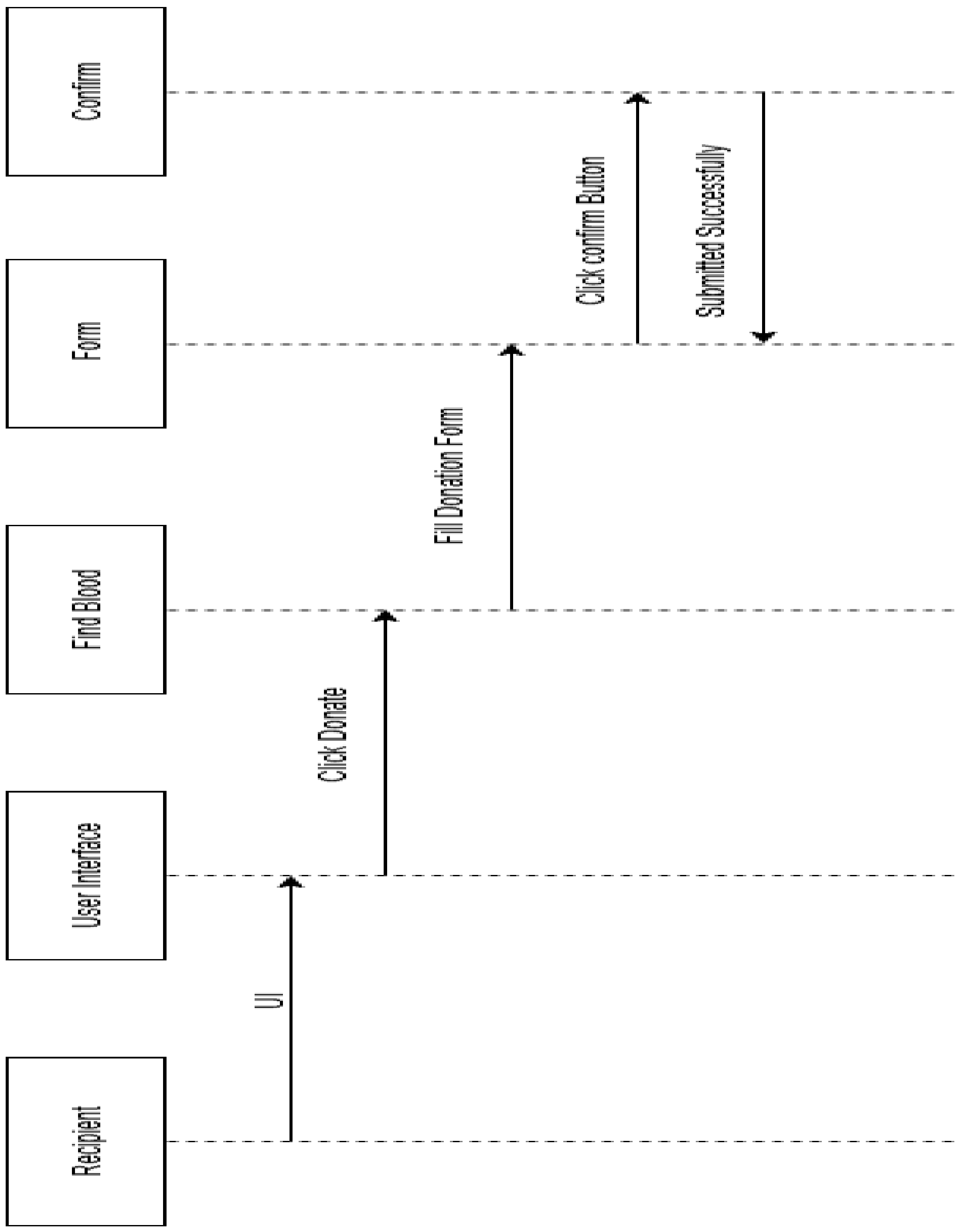
Figure 3: Donor Registration Sequence Diagram



- A donor decides to donate blood.
- The donor goes to the user interface (website or app).
- They choose the "Donate" option.
- The system then shows an appointment form.
- The donor fills out the form with their details.
- After submitting, the system asks the donor to confirm the appointment.
- The donor confirms the submission.
- The donation process is now successfully initiated.

3.2.2 Recipient Registration Sequence Diagram

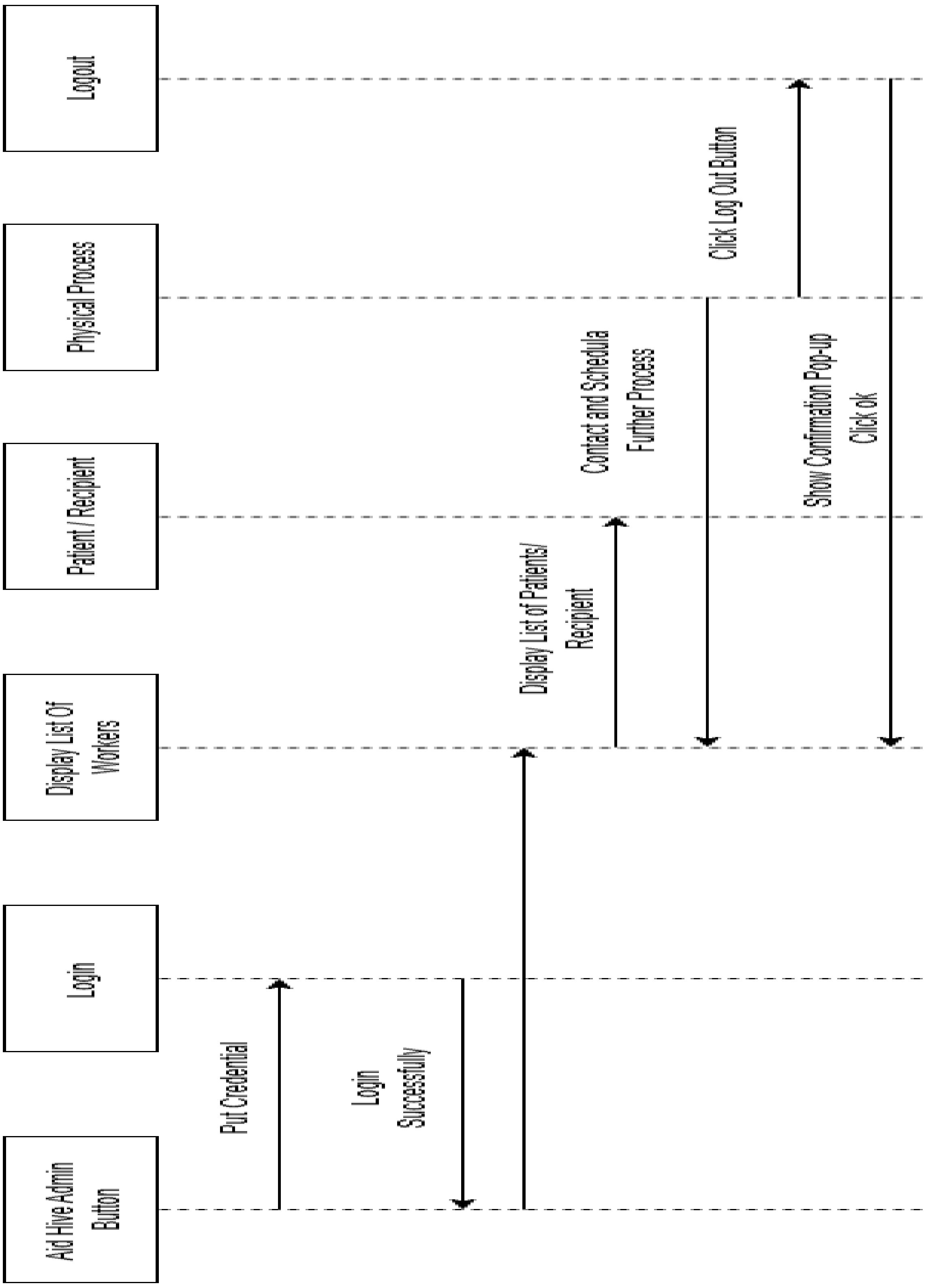
Figure 4: Recipient Registration Sequence Diagram



- A recipient needs blood and goes to the user interface (website or app).
- They use the "Find Blood" option.
- The system displays a form to fill in their blood requirements and details.
- The recipient fills out the form and submits it.
- The system then asks them to confirm the request.
- The recipient clicks confirm, and their blood request is successfully sent.

3.2.3 Admin Panel Sequence Diagram

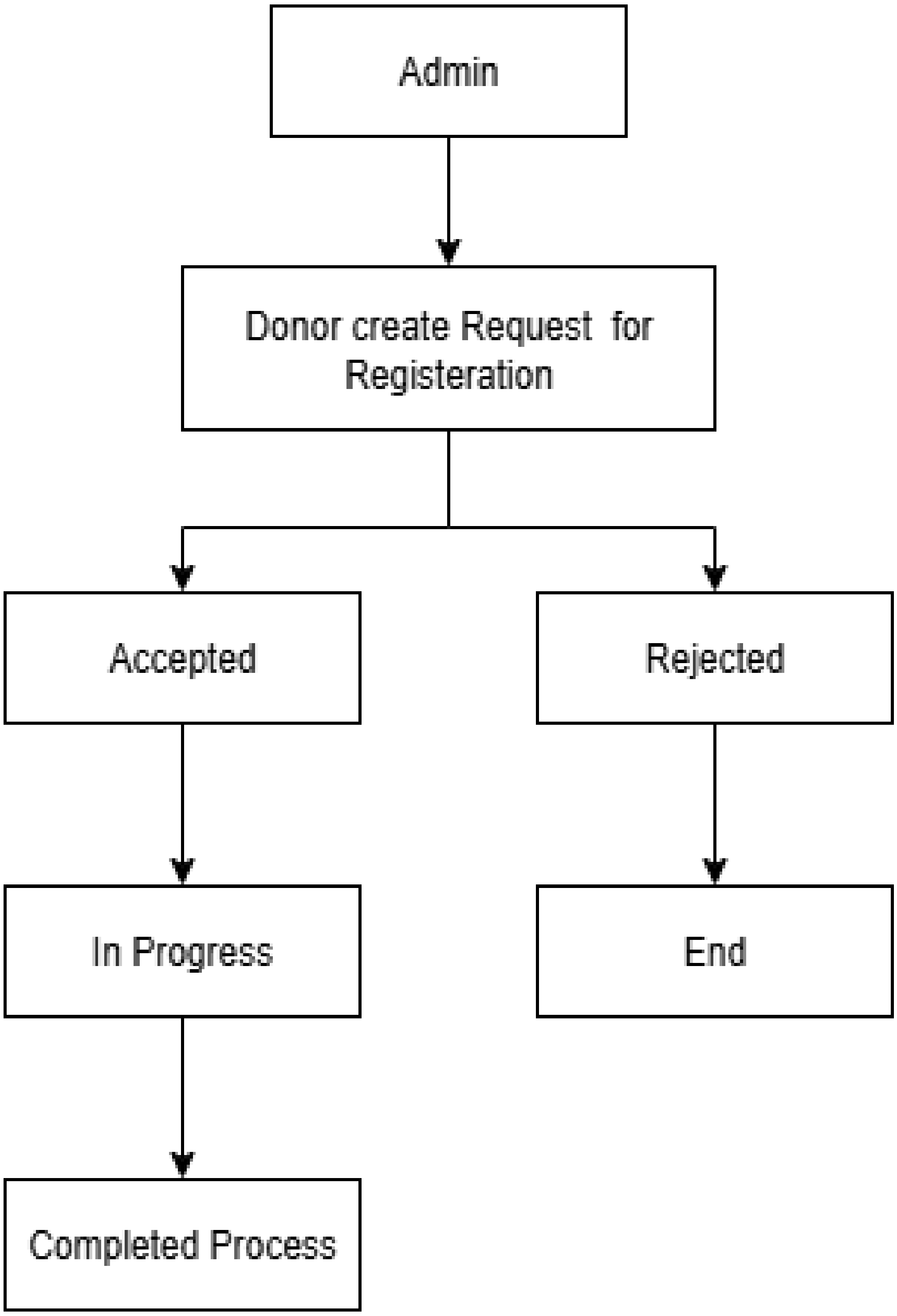
Figure 5: Admin Panel Sequence Diagram



- User clicks on the Aid Hive Admin Button
- User is prompted to login
- After login, system displays list of workers
- A patient or recipient is involved in the process
- A physical process (real-world action like aid or treatment) takes place
- User logs out of the system

3.2.2 Donor State Transition Diagram

Figure 6:Donor State Transition Diagram of Aid Hive



Explanation:

1. **Admin (Initial State)**

   The process begins with the admin logged into the system.

2. **Donor Creates Request for Registration**

   A donor fills the registration form on the platform. The request is forwarded to the admin for verification/approval.

3. **Decision State: Accepted or Rejected**

   If accepted: the donor's details are stored in the MySQL database, and the donor can now participate in the system (donating/responding to blood requests).

   If rejected: the process ends (the donor cannot access the system).

4. **In Progress (For Accepted Requests)**

   Once accepted, the donor's status moves to In Progress, meaning they are now an active donor in the system.

5. **Completed Process**

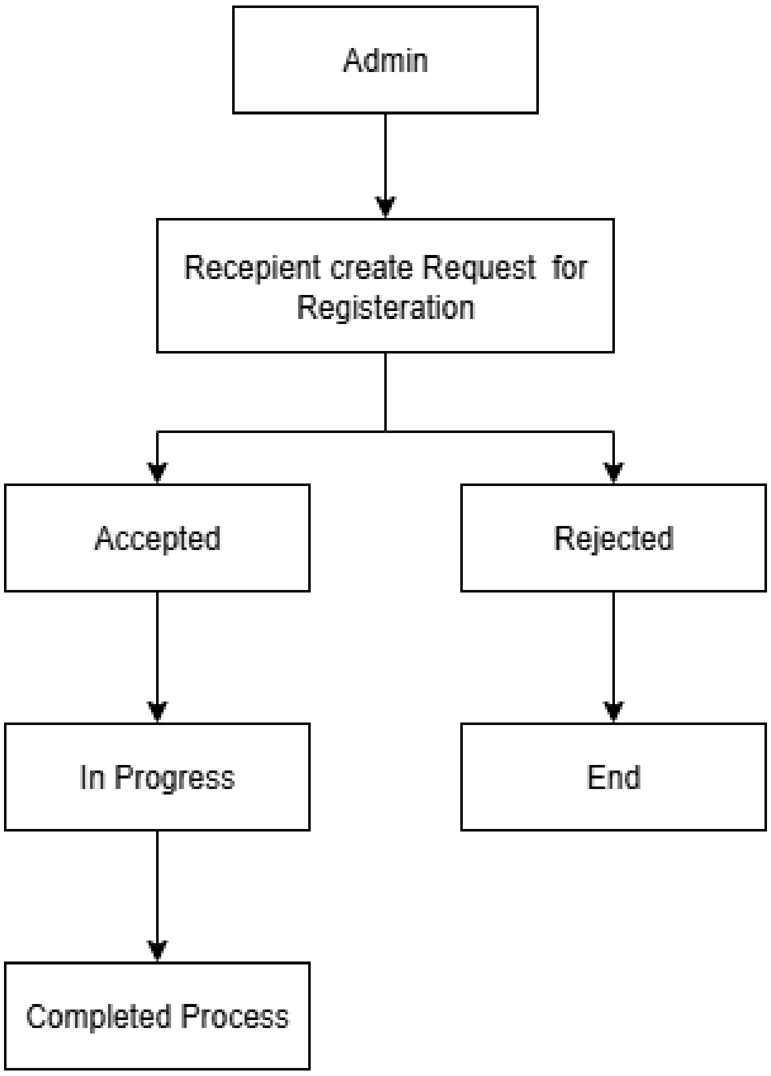
   When all verification steps are finalized, the donor registration is marked as Completed, and the donor is fully onboarded.

6. **End (For Rejected Requests)**

   If the request is not valid, the process terminates without registration.

3.2.4 *Recipient State Transition*

Figure 7 : Recipient State Transition Diagram



Explanation:

1. **Recipient (Start State)**

   A recipient (patient or person in need of blood) starts the process by accessing the system via the frontend UI.

2. **Aid Hive Worker**

   The request is forwarded to the Aid Hive Worker module, which handles filtering and management of requests.

3. **Listed Donor**

   The system retrieves and displays a list of donors from the MySQL database who match the required blood type and selected city.

4. **Check Status – Available or Not**

   The system checks the availability status of listed donors:

   - **Available:** Donor can be contacted for donation.
   - **Not Available:** Donor cannot respond at the moment.

3.3 Architectural Design

The system follows a three-tier architecture consisting of frontend, backend, and database layers to ensure modularity, maintainability, and clear separation of concerns.

1. **Frontend Layer (React.js):**

   - Provides the user interface for donors, recipients, and admins.
   - Donors and recipients interact through registration forms and search functionality.
   - Admin accesses feature only after login using username and password.
   - Pages include Home, Donate, Find Blood, Aid Hive Workers, and About Us, with button OF Aid Hive Admin.

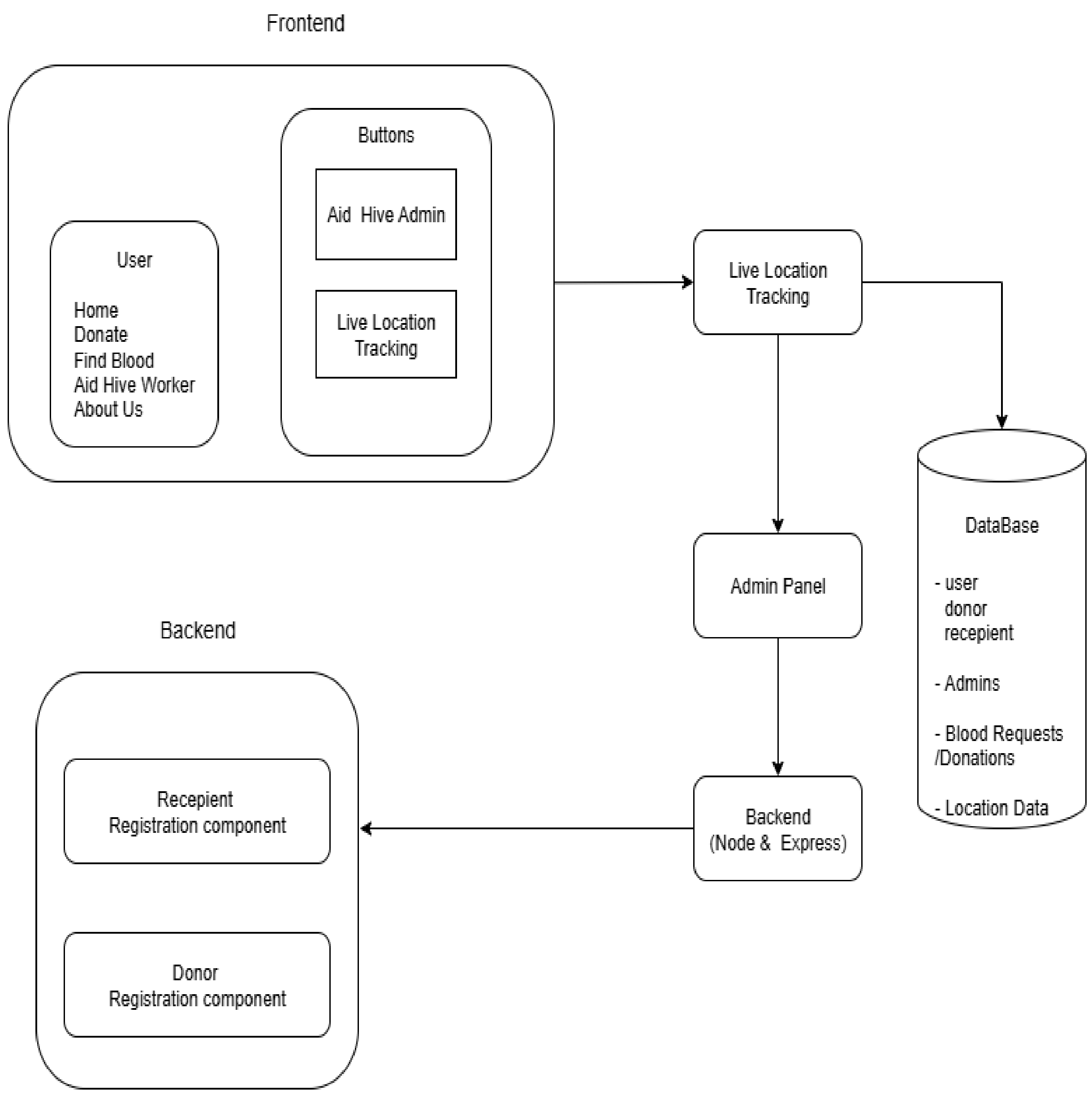
2. **Frontend Layer (React.js):**
   - Provides the user interface for donors, recipients, and admins.
   - Donors and recipients interact through registration forms and search functionality.
   - Admin accesses feature only after login using username and password.
   - Pages include Home, Donate, Find Blood, Aid Hive Workers, and About Us, with button OF Aid Hive Admin.
3. **Backend Layer (Node.js + Express.js):**
   - Handles all API requests from the frontend.
   - Processes form submissions, user requests, and donation tracking.
   - Performs authentication and authorization for the admin.
4. **Database Layer (MySQL):**
   - Stores structured data including users (donors, recipients, admin), blood requests, donations, and worker information.
   - Provides secure and reliable storage to support system operations.

Data Flow:

- Users submit forms → Backend validates and stores data → Database updates → Admin monitors and manages requests.

This architecture ensures secure, efficient, and scalable operations while keeping user interactions simple and admin operations centralized.

Figure 8: Architectural Design of Aid Hive



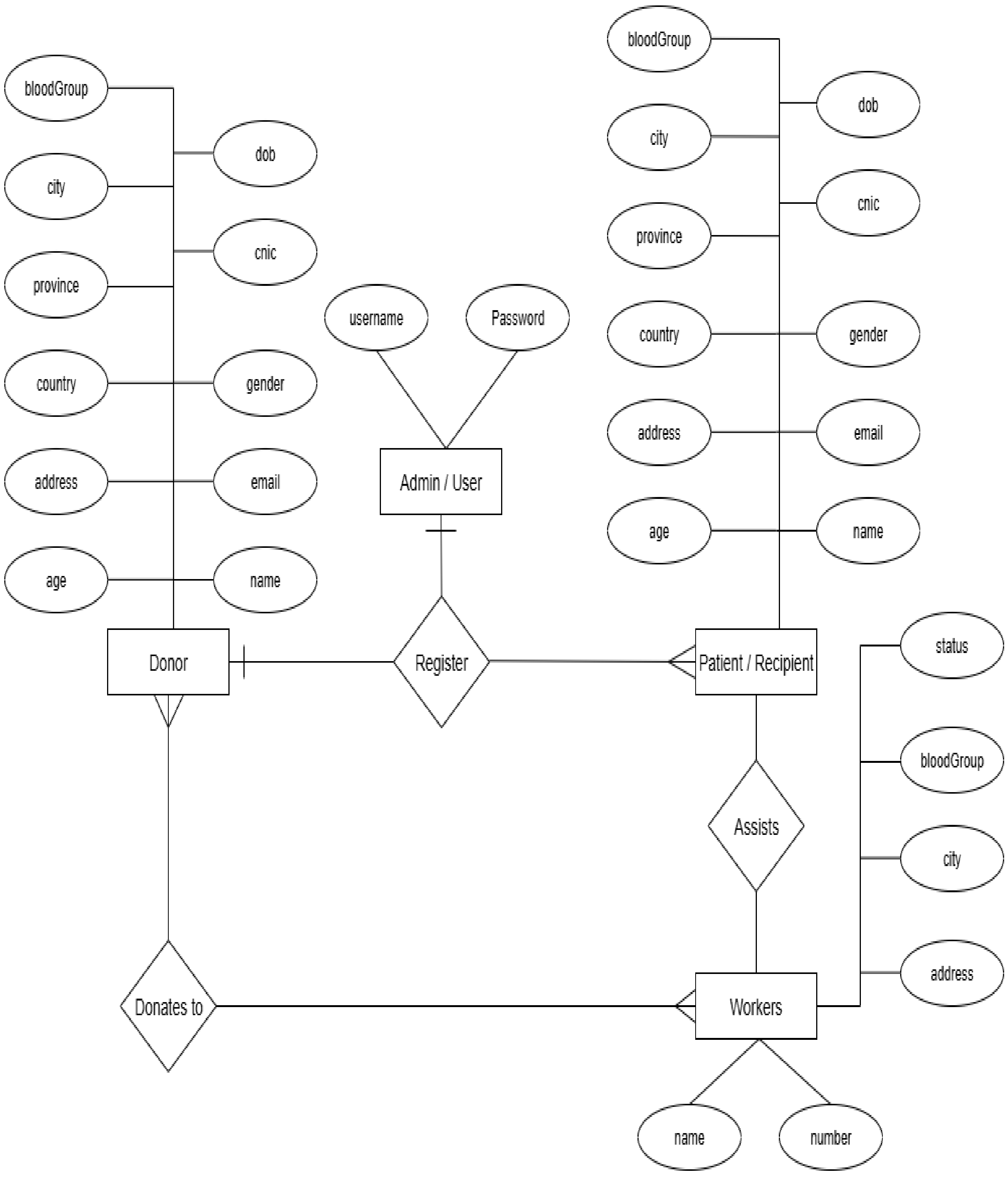
3.4 Data Design

- The Aid Hive: Blood Donation Management System has a MySQL relational database for storing and managing all system data. The security, integrity, and scalability are given priority while designing the system. The database schema has been designed in such a way that every entity like donors, recipients, admins, and blood requests are well defined with the relationships between them.
- Normalization is applied to avoid data redundancy and ensure data consistency.
- Primary keys are used for unique identification of records, and foreign keys maintain relationships between entities.
- The database supports role-based access control, allowing different access privileges for donors, recipients, and admins.

Key Entities:

1. Users: Stores information about donors, recipients, and admins.
2. Blood Requests: Manages requests made by patients/recipients.
3. Donations: Tracks donation details and donor contributions.
4. Aid Hive Workers: Maintains records of approved/registered workers.

Figure 9 : ER Diagram



3.4.1 Data Dictionary

Alphabetical List of System Entities or Major Data with Types and Descriptions:

Table 12: Data Dictionary Table

Terminology Description	**Terminology Description**
Blood Request	MySQL table representing a blood donation request
Blood Type	VARCHAR – Type of blood (A+, A-, B+, B-, O+, O-, AB+, AB-)
Donor	MySQL table representing a blood donor
Donor ID	INT (Primary Key, Auto Increment) – Unique identifier for each donor
Recipient	My SQL Table representing a person requesting blood
Request Status	Enum - Status of blood request (Pending, Approved, Completed)
User	My SQL Table storing general user information

Structured Approach: Functions and Function Parameters:

Table 13 : Functions and Function Parameters

API Endpoint	Description	HTTP Method	Parameters
/Api/donors	Retrieves available donors based on location & blood type	GET	(bloodType: String, location: Object)
/Api/requests/create	Creating a new blood donation request	POST	(recipientID: ObjectId, bloodType: String, location: Object)
/Api/requests/update	Updates request status	PUT	(requestID: ObjectId, status: String)

3.5 User Interface Design

The User Interface (UI) of the system is designed to be simple, intuitive, and user-friendly for donors, recipients, and admins. It ensures that users can perform their tasks efficiently while keeping navigation straightforward.

Key Pages and Features:

1. **Home Page:**
   - Displays system overview, purpose, and quick navigation links to other pages.
   - Accessible to all users without login.
2. **Donate Page:**

- Donors can fill out a donation registration form including name, blood group, contact, and donation details.
- Form submission sends data to the backend for storage.

3. **Find Blood Page:**
    - Recipients can submit blood request forms specifying blood group, quantity.
    - Requests are sent to the backend for processing by admin.
4. **Aid Hive Workers Page:**
    - Displays information about workers involved in blood donation and aid activities.
    - Admin can monitor live location of workers.
5. **About Us Page:**
    - Provides general information about the platform and its objectives.
6. **Buttons:**
    - Aid Hive Admin Button: Redirects to login page for admin authentication.
7. **Forms:**
    - Donor and Recipient forms are simple and mandatory fields are validated to ensure complete and accurate data collection.
    - No login is required for donors or recipients to interact with forms.

Design Principles:

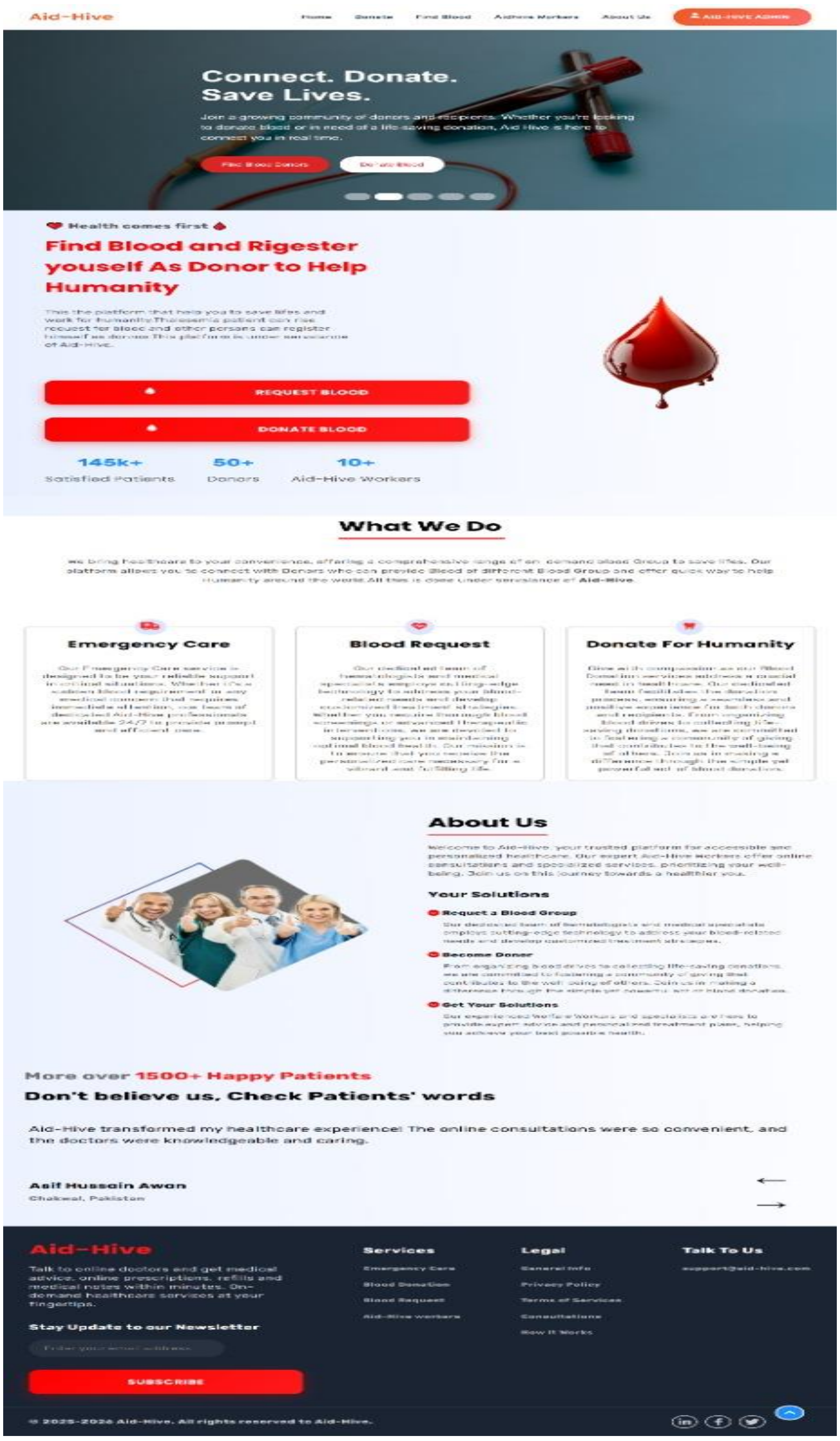
- Minimalist and clear layout to reduce confusion.
- Forms with proper labels and placeholders for easy data entry.
- Responsive design for accessibility on desktops, tablets, and mobile devices.
- Admin interface includes authentication and dashboards for effective management.

3.5.1 Screen Images

Following is the list of screen shots of UI mockups of Aid Hive illustrating the visual design of different components of the Aid Hive.

The subsequent images illustrate the landing page of the Aid Hive. That is the page that the user finds when he opens the app.

Figure 10 : Landing Page



Aid-Hive

Home Donate Find Blood Aidhive Workers About Us AID-HIVE ADMIN

Connect. Donate.
Save Lives.

Join a growing community of donors and recipients. Whether you're looking to donate blood or in need of a life-saving donation, Aid Hive is here to connect you in real time.

Find Blood Donors | Donate Blood

Health comes first

Find Blood and Rigester youself As Donor to Help Humanity

This the platform that help you to save lifes and work for humanity.Thalassemia patient can rise request for blood and other persons can register himself as donor.This platform is under servaliance of Aid-Hive.

REQUEST BLOOD

DONATE BLOOD

145k+ Satisfied Patients | 50+ Donors | 10+ Aid-Hive Workers

What We Do

we bring healthcare to your convenience, offering a comprehensive range of on demand blood Group to save lifes. Our platform allows you to connect with Donors who can provide Blood of different Blood Group and offer quick way to help Humanity around the world.All this is done under servaliance of **Aid-Hive**.

Emergency Care

Our Emergency Care service is designed to be your reliable support in critical situations. Whether it's a sudden blood requirement or any medical concern that requires immediate attention, our team of dedicated Aid-Hive professionals are available 24/7 to provide prompt and efficient care.

Blood Request

Our dedicated team of hematologists and medical specialists employs cutting-edge technology to address your blood-related needs and develop customized treatment strategies. Whether you require thorough blood screenings or advanced therapeutic interventions, we are devoted to supporting you in maintaining optimal blood health. Our mission is to ensure that you receive the personalized care necessary for a vibrant and fulfilling life.

Donate For Humanity

Give with compassion as our Blood Donation services address a crucial need in healthcare. Our dedicated team facilitates the donation process, ensuring a seamless and positive experience for both donors and recipients. From organizing blood drives to collecting life-saving donations, we are committed to fostering a community of giving that contributes to the well-being of others. Join us in making a difference through the simple yet powerful act of blood donation.

About Us

Welcome to Aid-Hive, your trusted platform for accessible and personalized healthcare. Our expert Aid-Hive workers offer online consultations and specialized services, prioritizing your well-being. Join us on this journey towards a healthier you.

Your Solutions

Requet a Blood Group
Our dedicated team of hematologists and medical specialists employs cutting-edge technology to address your blood-related needs and develop customized treatment strategies.

Become Doner
From organizing blood drives to collecting life-saving donations, we are committed to fostering a community of giving that contributes to the well-being of others. Join us in making a difference through the simple yet powerful act of blood donation.

Get Your Solutions
Our experienced Welfare Workers and specialists are here to provide expert advice and personalized treatment plans, helping you achieve your best possible health.

More over **1500+ Happy Patients**

Don't believe us, Check Patients' words

Aid-Hive transformed my healthcare experience! The online consultations were so convenient, and the doctors were knowledgeable and caring.

Asif Hussain Awan
Chakwal, Pakistan

Aid-Hive

Talk to online doctors and get medical advice, online prescriptions, refills and medical notes within minutes. On-demand healthcare services at your fingertips.

Stay Update to our Newsletter

Enter your email address

SUBSCRIBE

Services
Emergency Care
Blood Donation
Blood Request
Aid-Hive workers

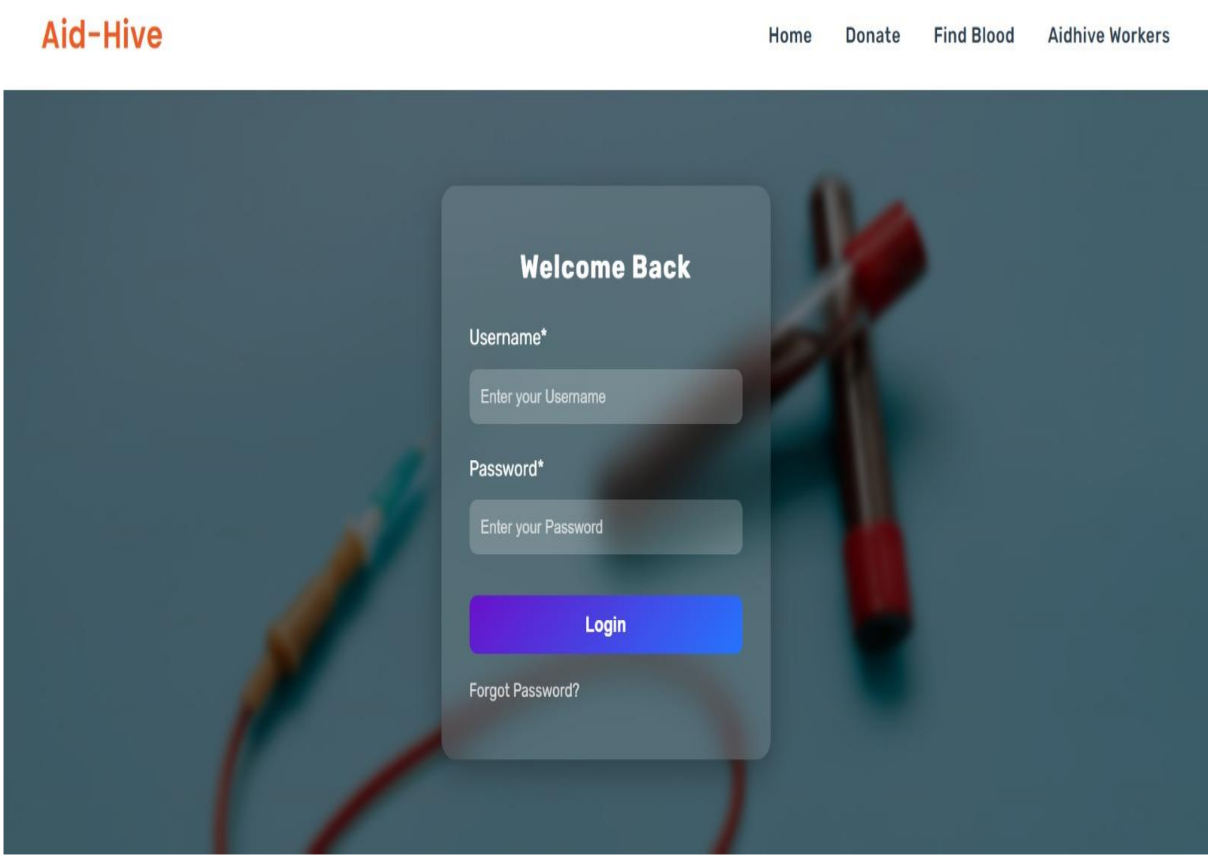
Legal
General Info
Privacy Policy
Terms of Services
Consultations
How It Works

Talk To Us
support@aid-hive.com

© 2025-2026 Aid-Hive. All rights reserved to Aid-Hive.

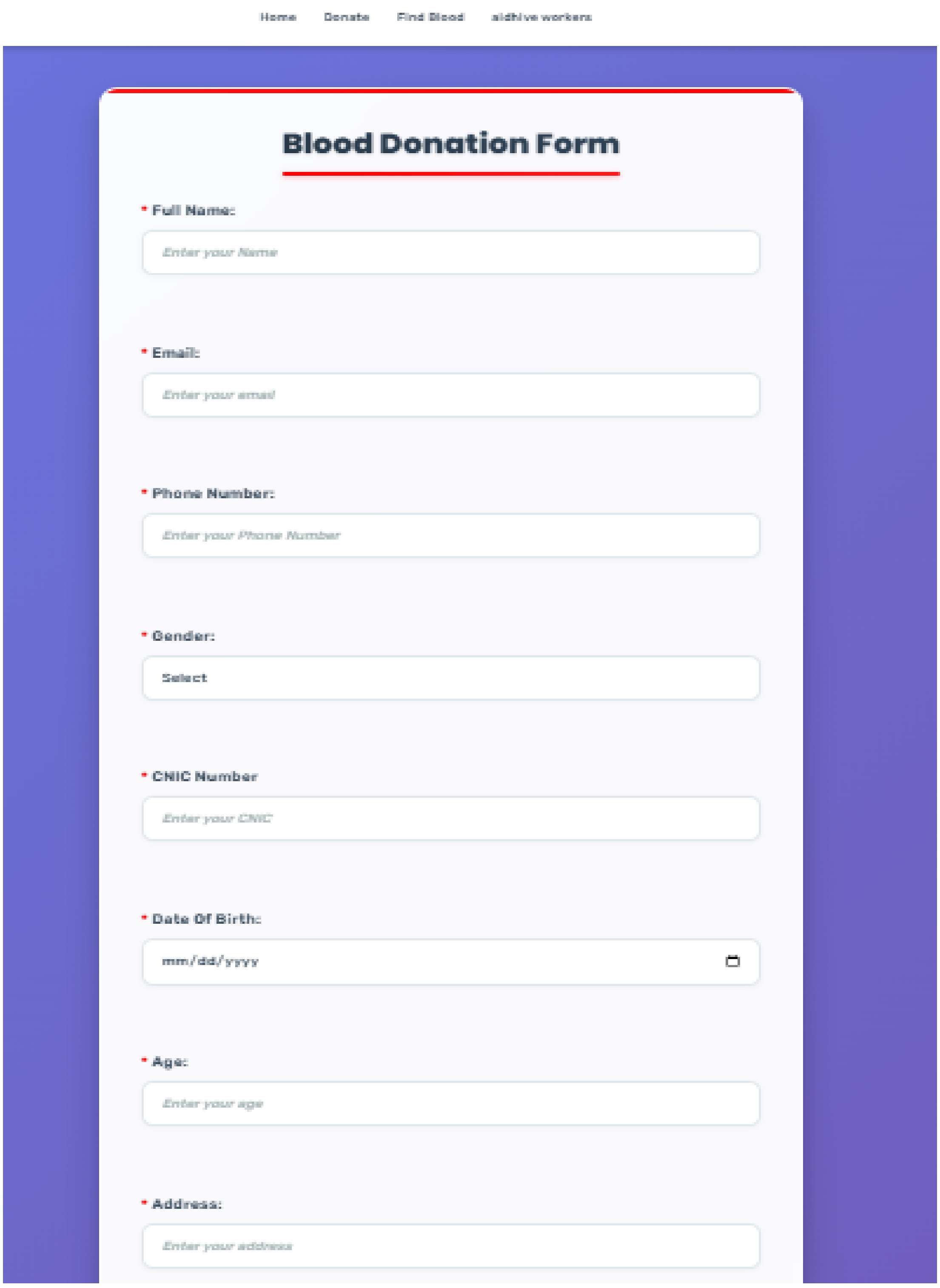
The following image show case the Admin Login Form up of the Aid Hive.

Figure 11 : Admin Login Form



The following image show case the Donor Registration Form up of the Aid Hive.

Figure 12 : Donor Registration Form

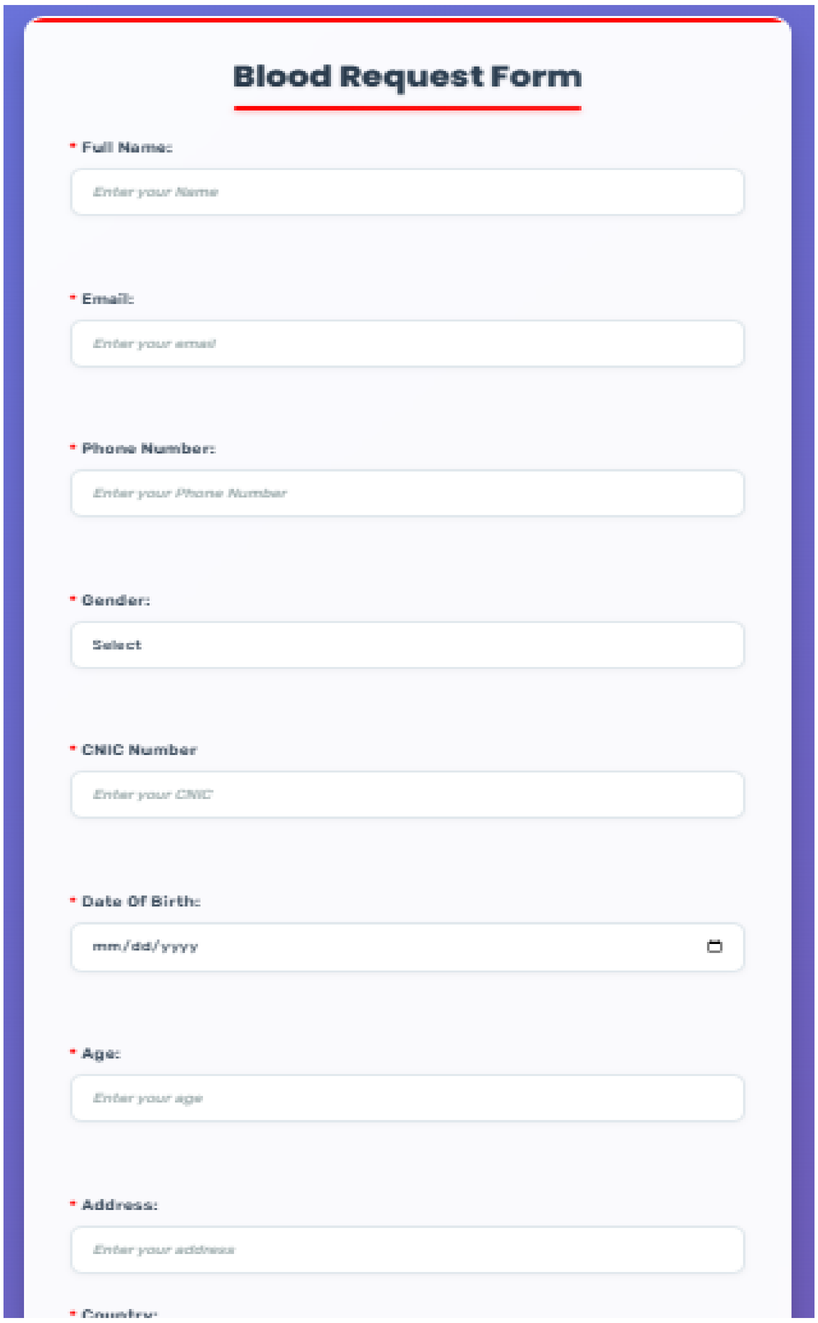


Blood Donation Form

* Full Name: Enter your Name
* Email: Enter your email
* Phone Number: Enter your Phone Number
* Gender: Select
* CNIC Number Enter your CNIC
* Date Of Birth: mm/dd/yyyy
* Age: Enter your age
* Address: Enter your address

The following image show case the Recipient Registration Form up of the Aid Hive.

Figure 13 : Recipient Registration Form



Blood Request Form

* Full Name:
Enter your Name

* Email:
Enter your email

* Phone Number:
Enter your Phone Number

* Gender:
Select

* CNIC Number
Enter your CNIC

* Date Of Birth:
mm/dd/yyyy

* Age:
Enter your age

* Address:
Enter your address

* Country:

3.5.2 Screen Objects and Actions

Use Case 1: Donor Registration

Table 14: Use case 1

Use Case ID	UC-01
Actor	Donor
Description	Allows a donor to register and provide their details for blood donation.
Preconditions	Donor has internet access and opens the Donate page.
Postconditions	Donor's information is stored in the MySQL database.

Main Flow:

1. Donor navigates to the Donate page.
2. Donor fills out the registration form.
3. Donor submits the form.
4. Backend validates and stores the information in the database.
5. System displays a confirmation message.

Alternate Flow:

- If the form is incomplete or contains invalid data, the system shows an error message and prompts the donor to correct it.

Use Case 2: Recipient Registration

Table 16: Use case 1

Use Case ID	UC-02
Actor	Recipient
Description	Allows a Recipient to register and provide their details for blood donation.
Preconditions	Recipient has internet access and opens the find blood page.
Postconditions	Recipient's information is stored in the MySQL database.

Main Flow:

1. Recipient navigates to the find blood page.
2. Recipient fills out the registration form.
3. Recipient submits the form.
4. Backend validates and stores the information in the database.
5. System displays a confirmation message.

Alternate Flow:

- If the form is incomplete or contains invalid data, the system shows an error message and prompts the Recipient to correct it.

Use Case 3: Admin Processing Blood Requests

Table 17: Use case 2

Use Case ID	UC-02
Actor	Admin
Description	Allows the admin to view, manage, and approve blood requests submitted by recipients.
Preconditions	Admin must be logged in using username and password.
Postconditions	Blood request status is updated in the database and donor assignment is recorded.

Main Flow:

1. Admin logs in to the Admin Panel.
2. Admin navigates to the Request Management section.
3. Admin views pending blood requests submitted by recipients.
4. Admin assigns available donors to requests and updates status (approved/pending/completed).
5. Backend updates the database and sends confirmation to the recipient.

Alternate Flow:

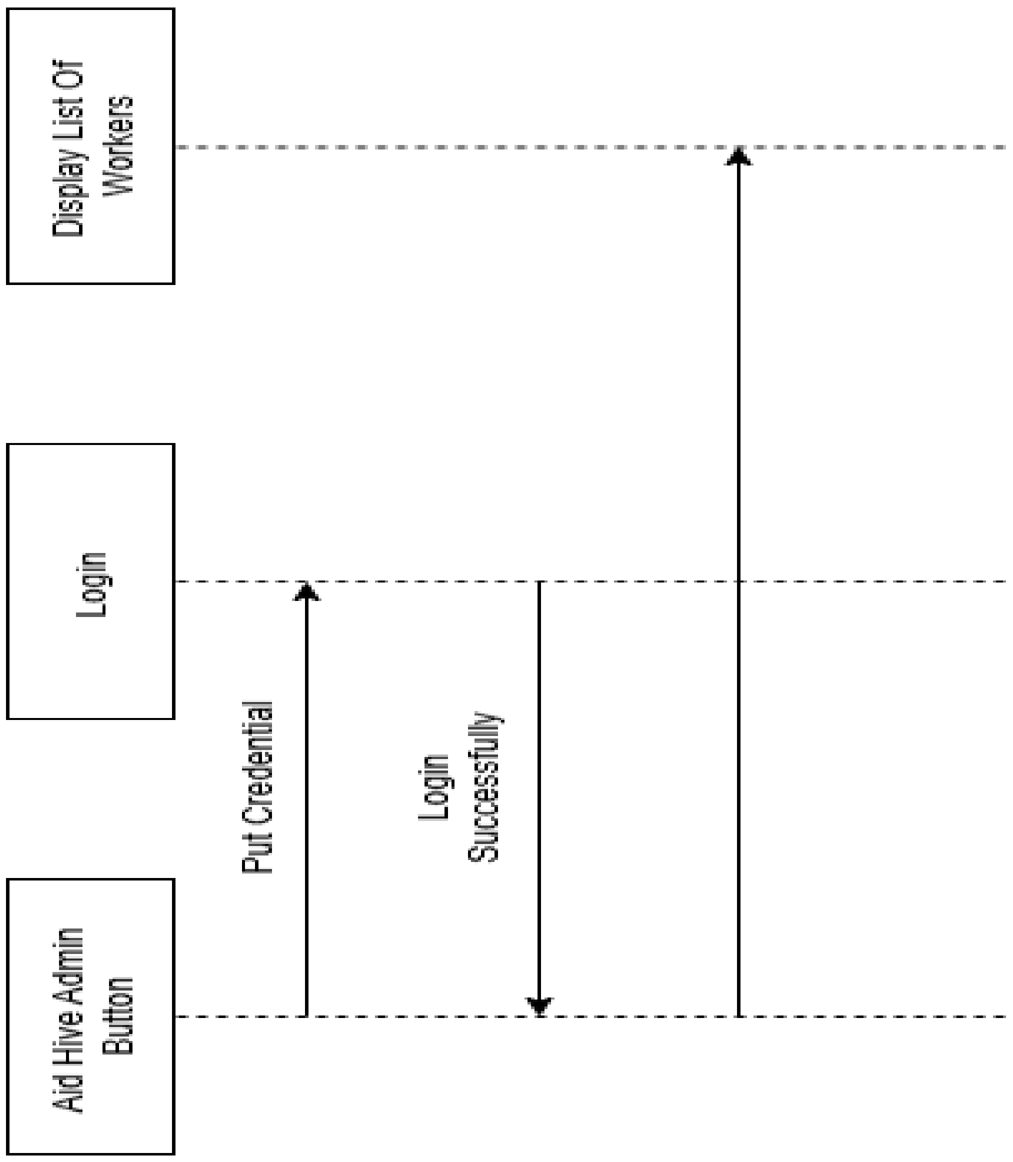
- If no matching donors are available, the system keeps the request in pending status and notifies the admin.

3.6 Behavioral Model

Sequence Diagrams:

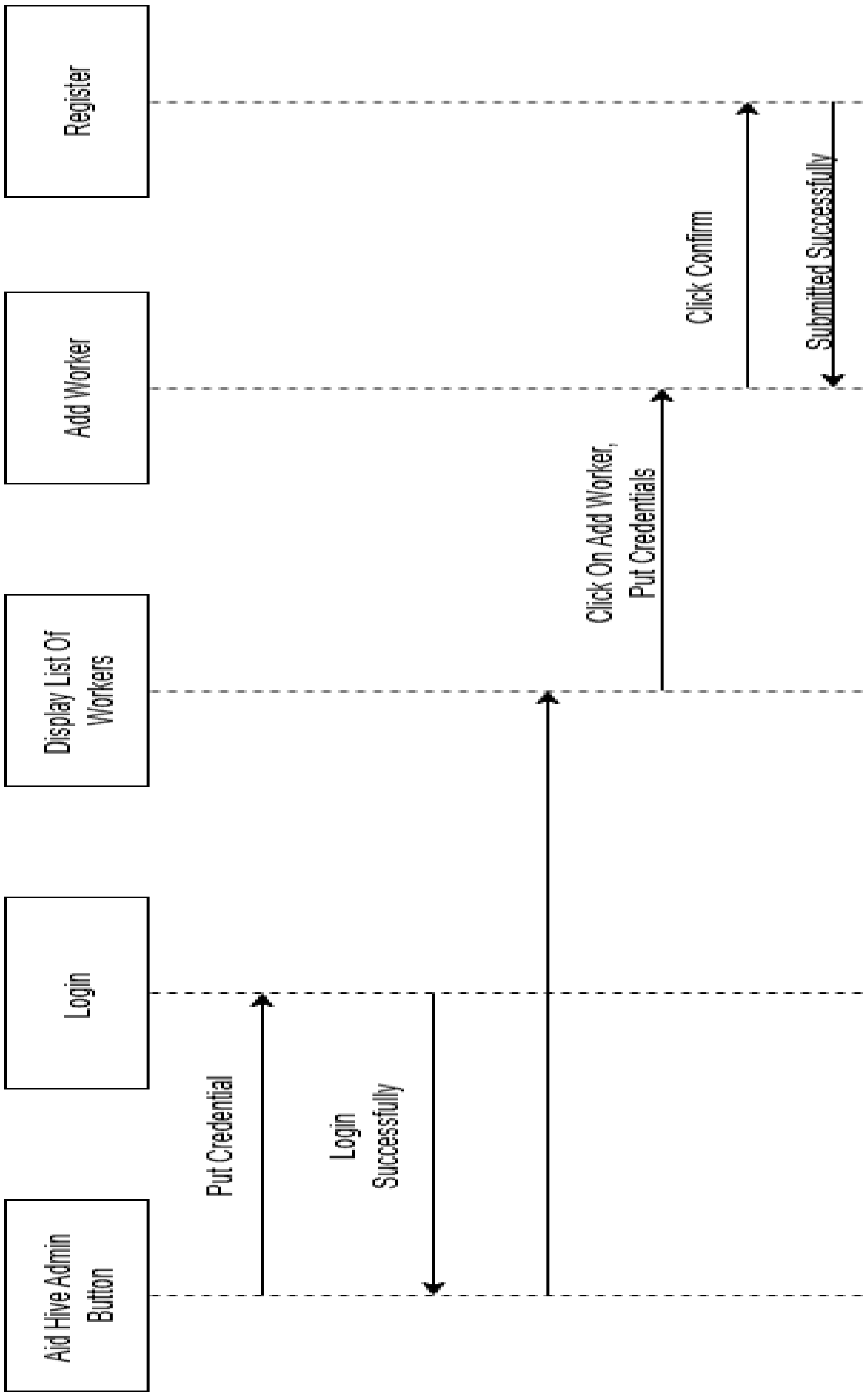
3.6.1 Admin login

Figure 14: Admin login



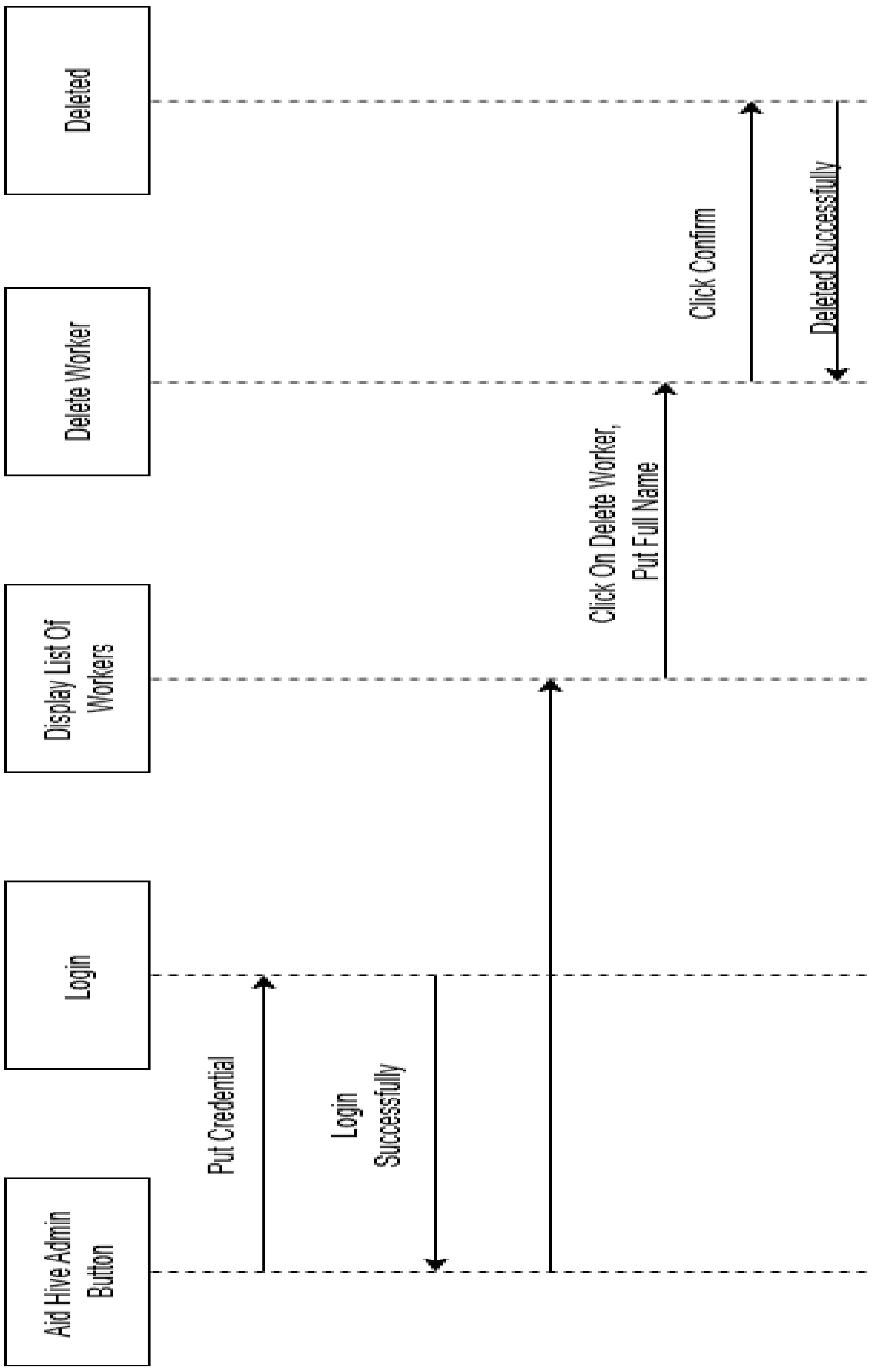
3.6.2 List of Worker / Add Worker

Figure 15: List of Worker / Add Worker



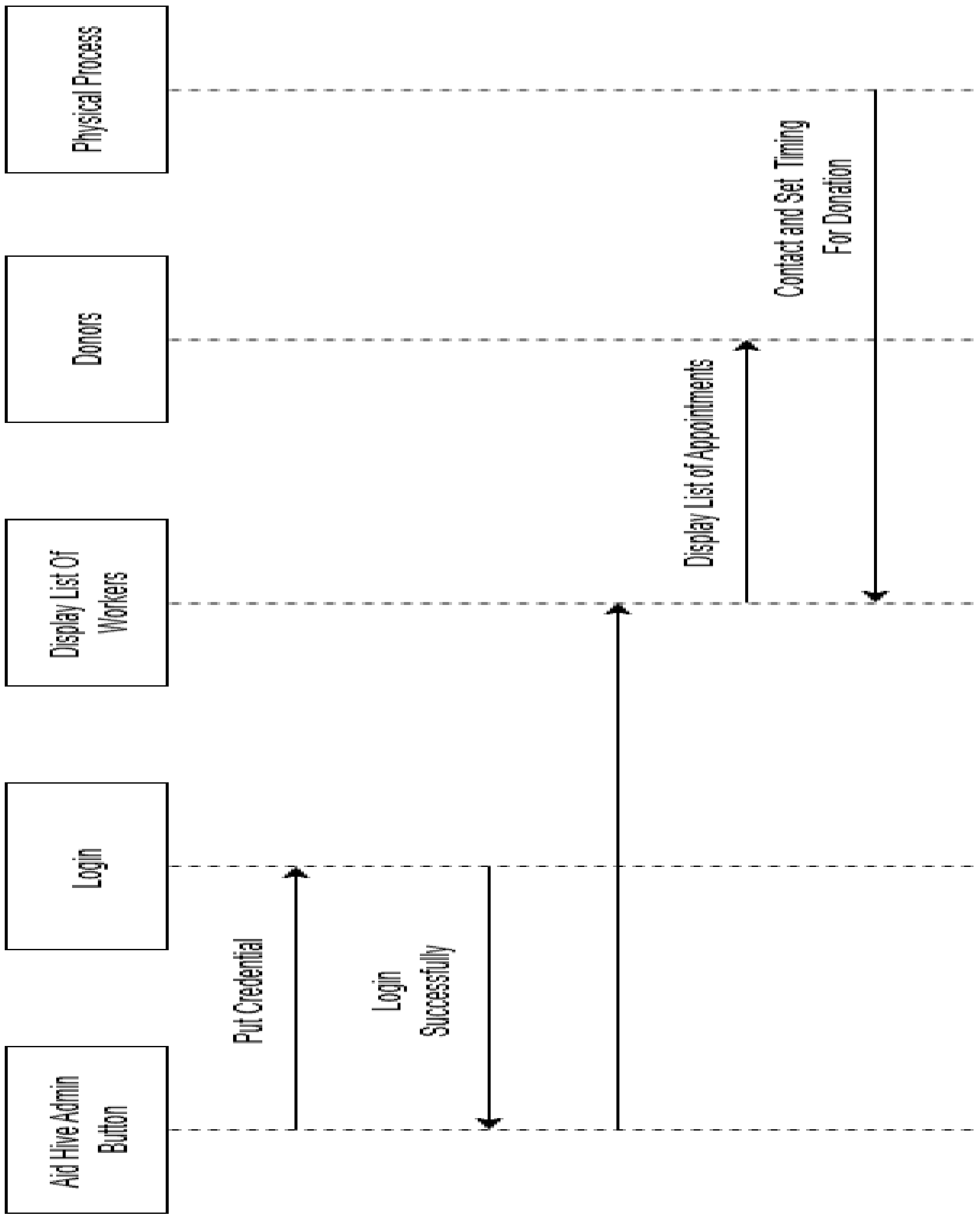
3.6.3 Delete Worker

Figure 16: Delete Worker



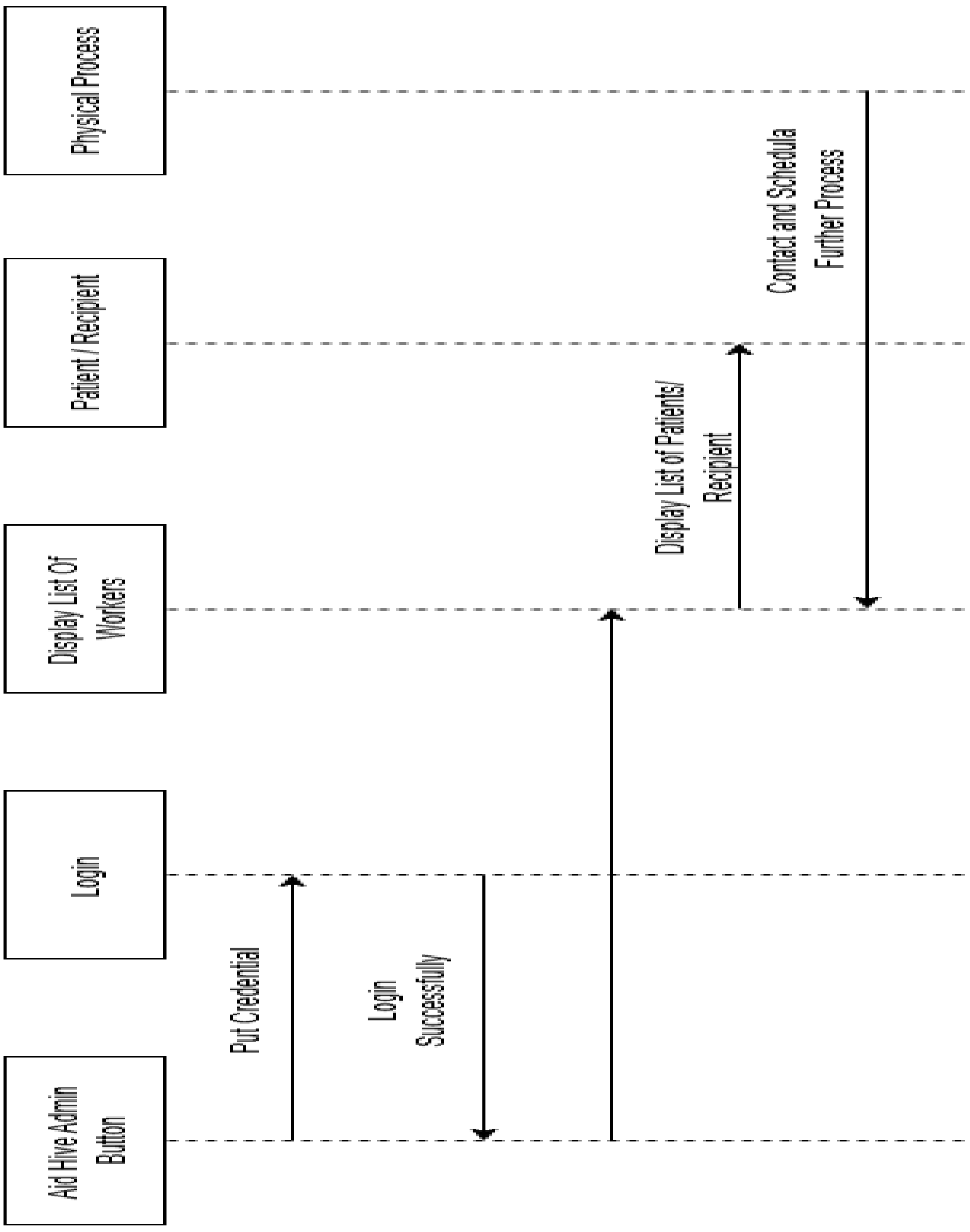
3.6.4 Display Listed Donors

Figure 17: Display Listed Donor



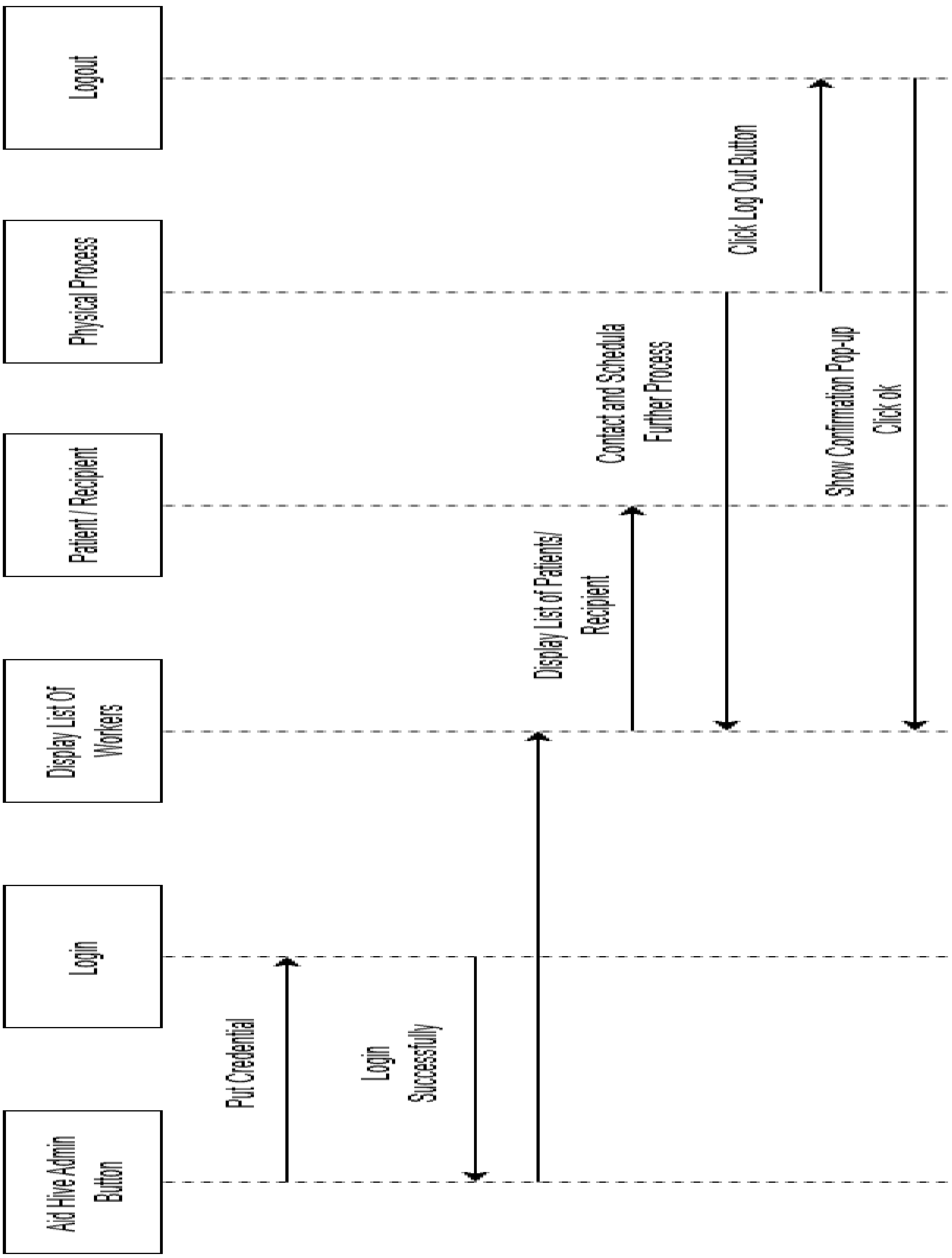
3.6.5 Display Listed Recipient

Figure 18 : Display Listed Recipient



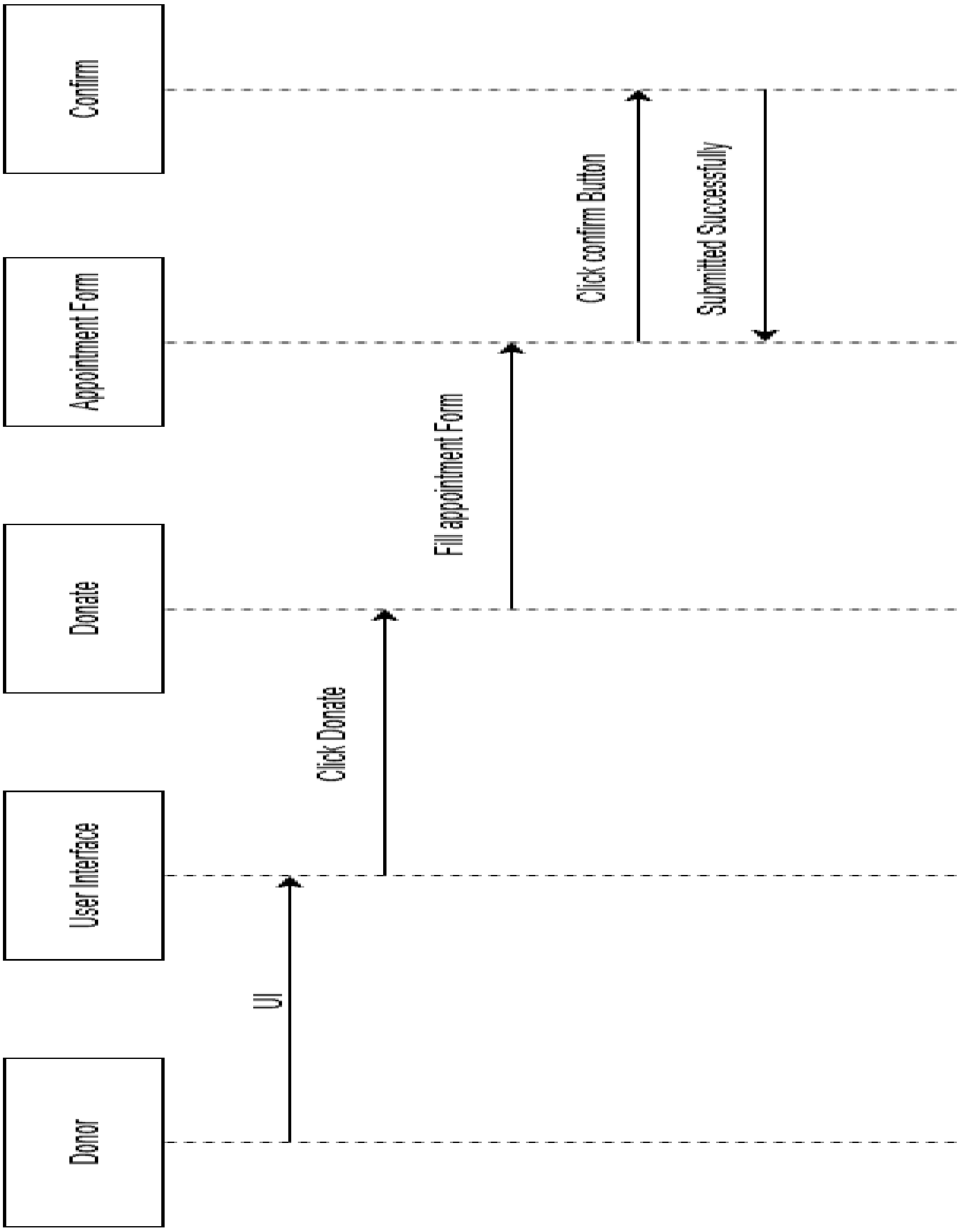
3.6.6 Logout

Figure 19: Logout



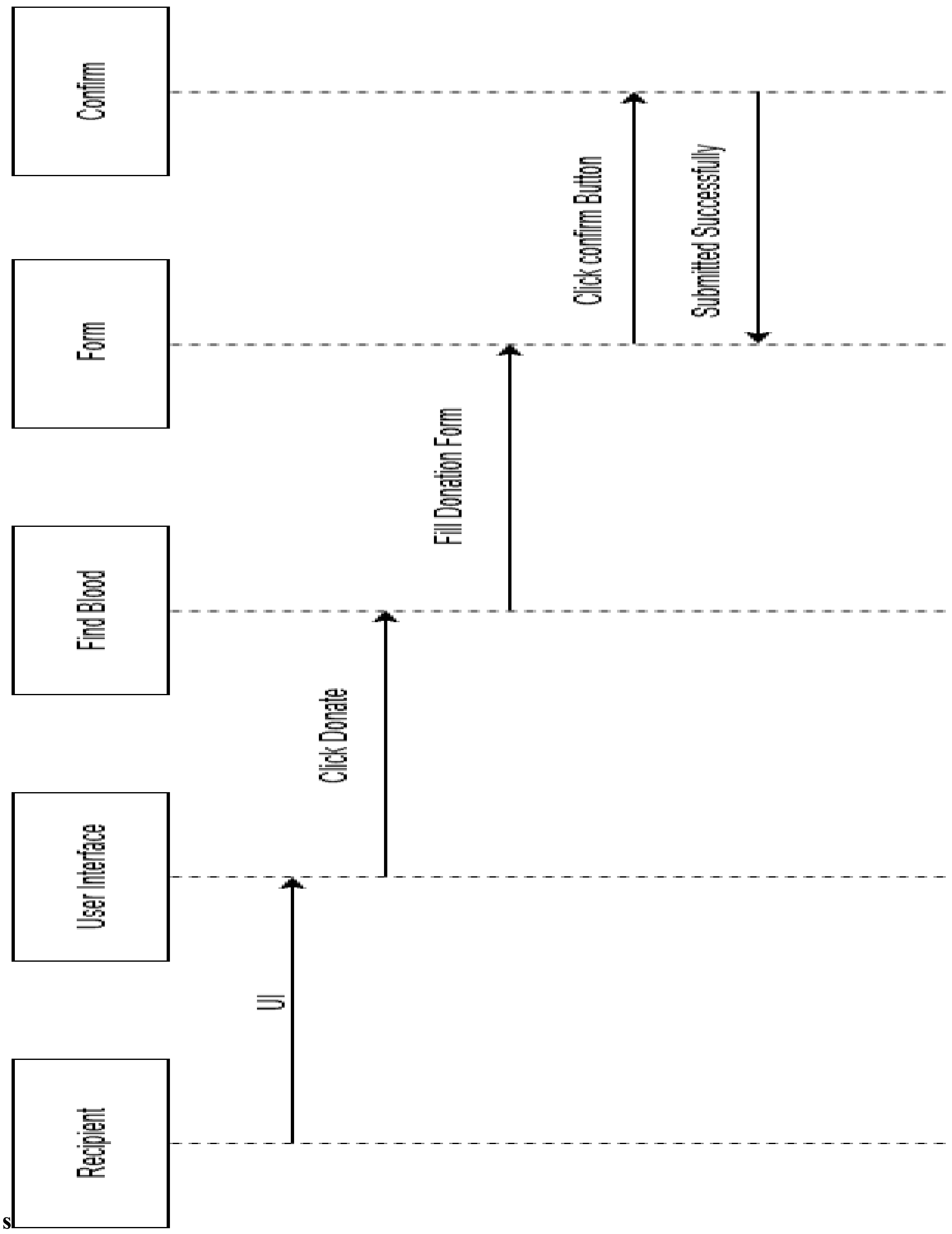
3.6.7 Donor Registration

Figure 20: Donor Registration



3.6.8 Recipient Registration

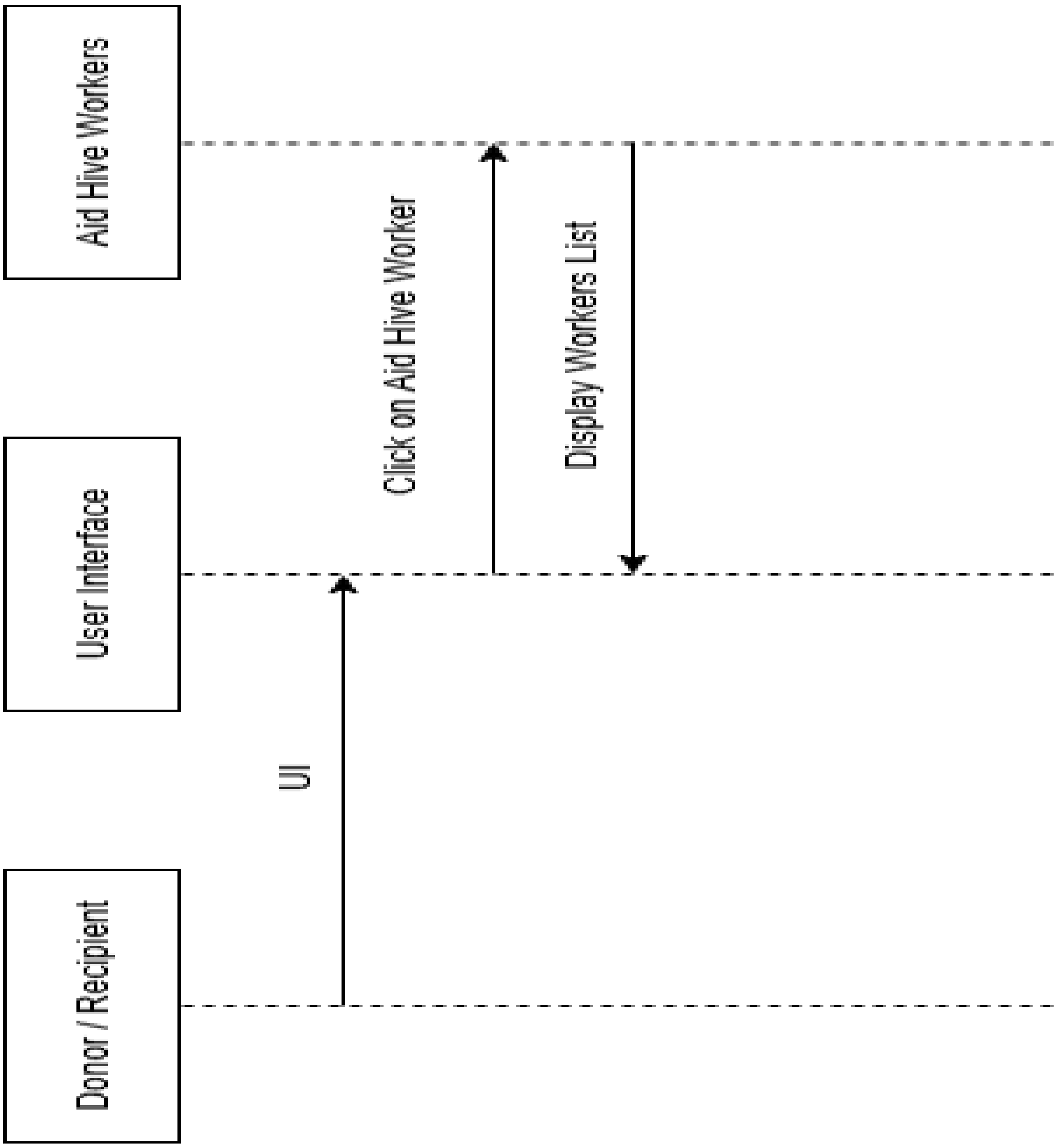
Figure 21: Recipient Registration



s

3.6.9 Donor / Recipient Display Worker List

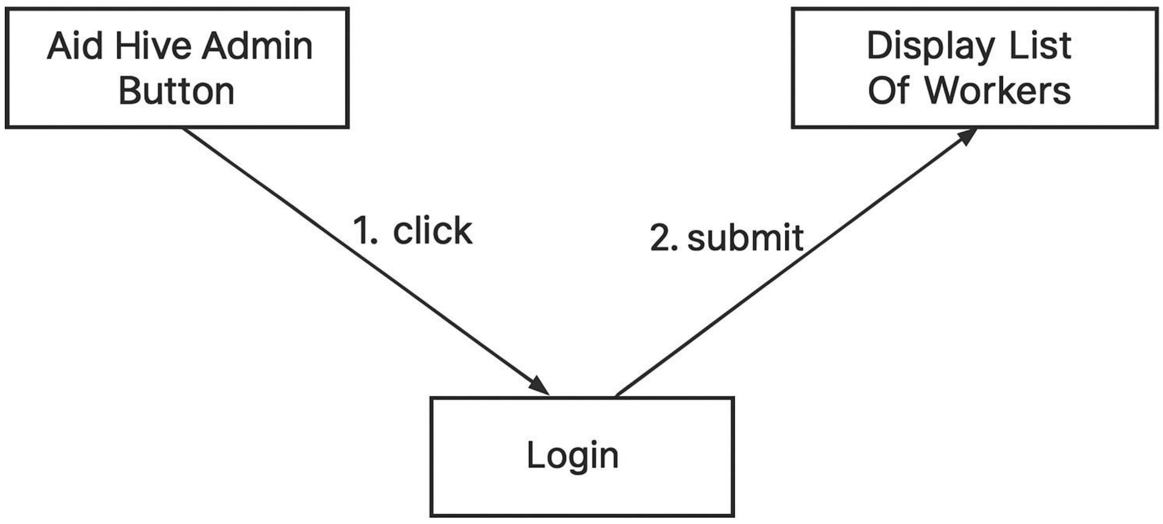
Figure 22: Donor / Recipient Display Worker List



Collaboration Diagrams:

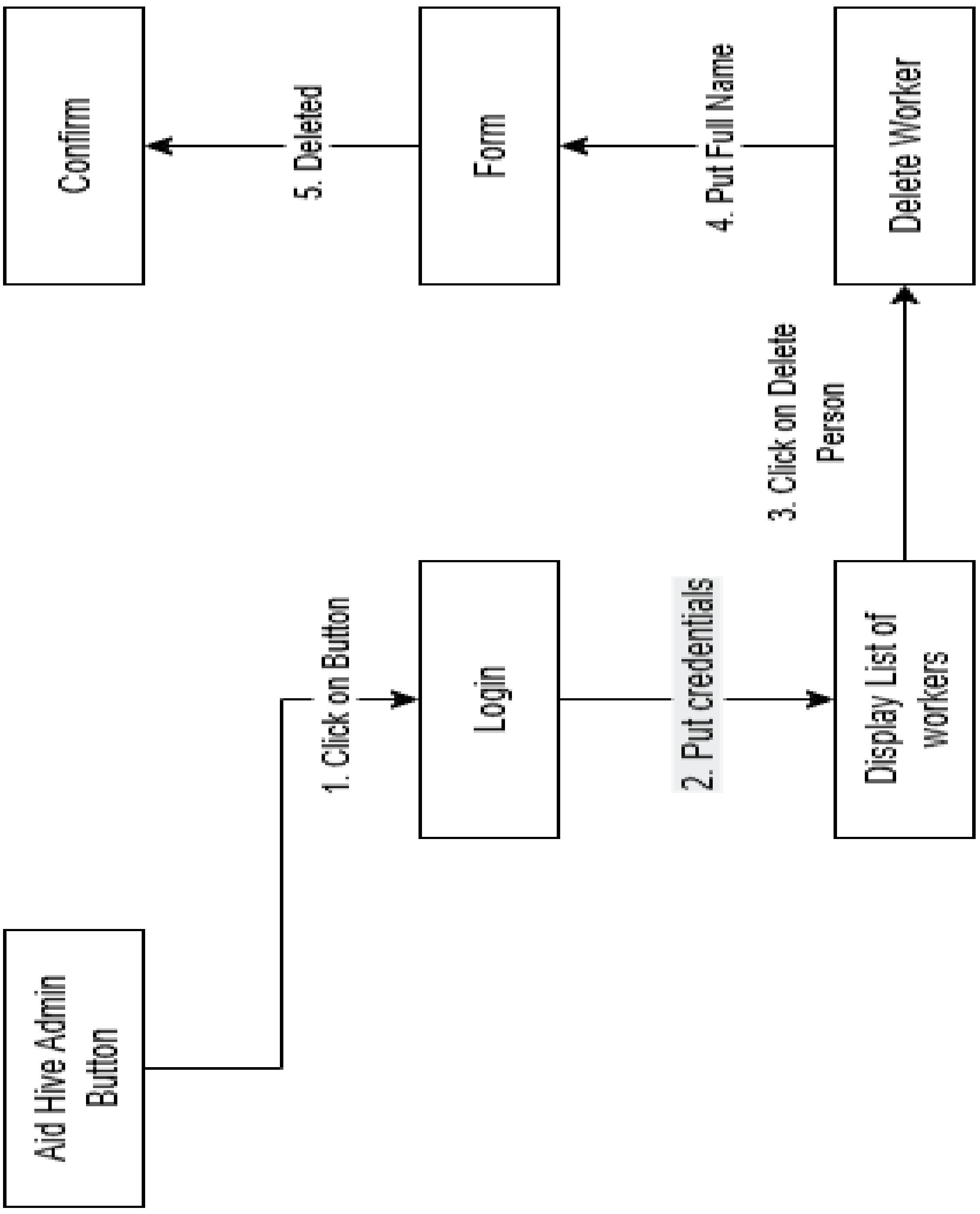
3.6.10 Admin Login

Figure 23 :Admin Login



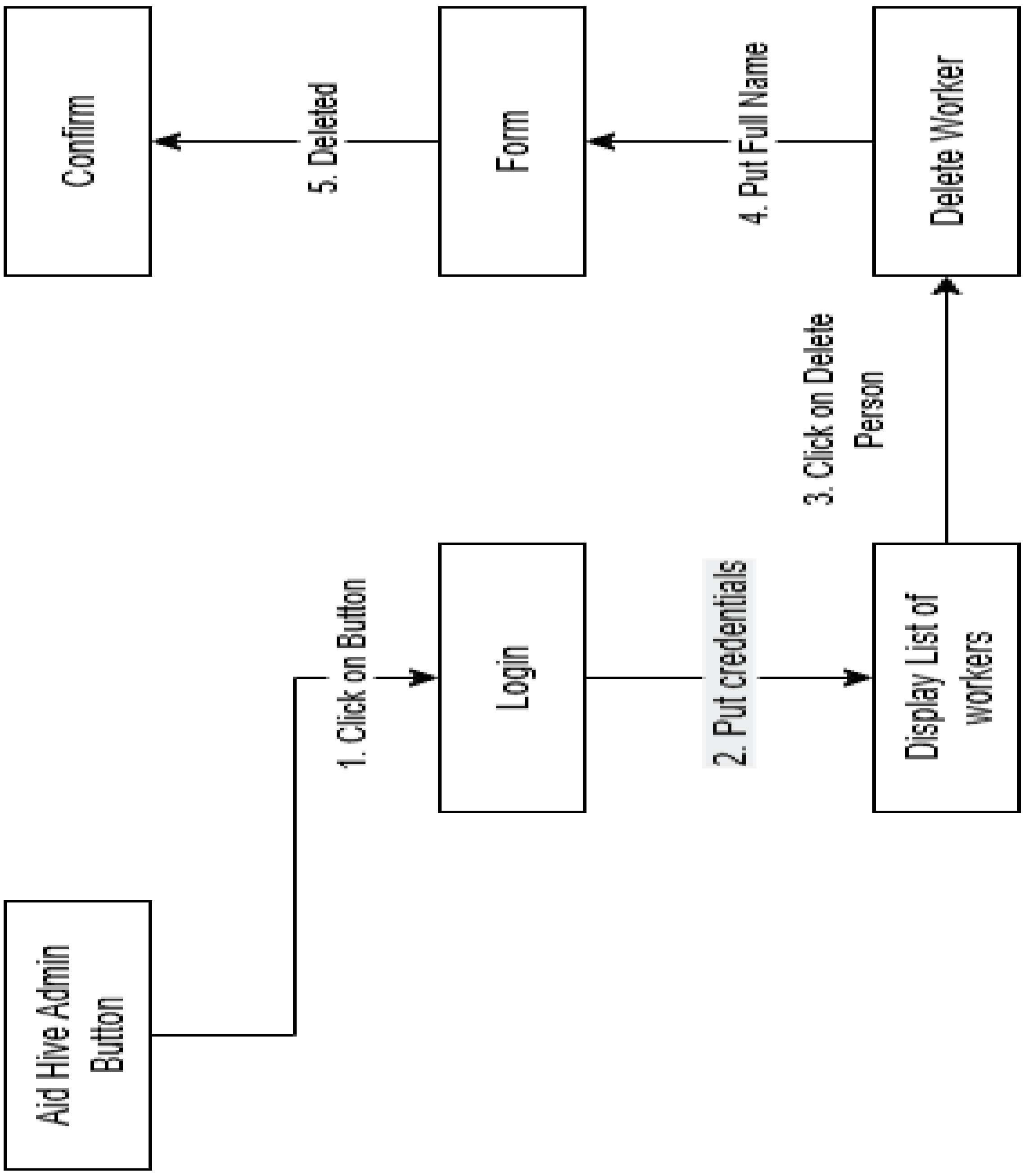
3.6.11 List of Worker / Add Worker

Figure 24: List of Worker / Add Worker



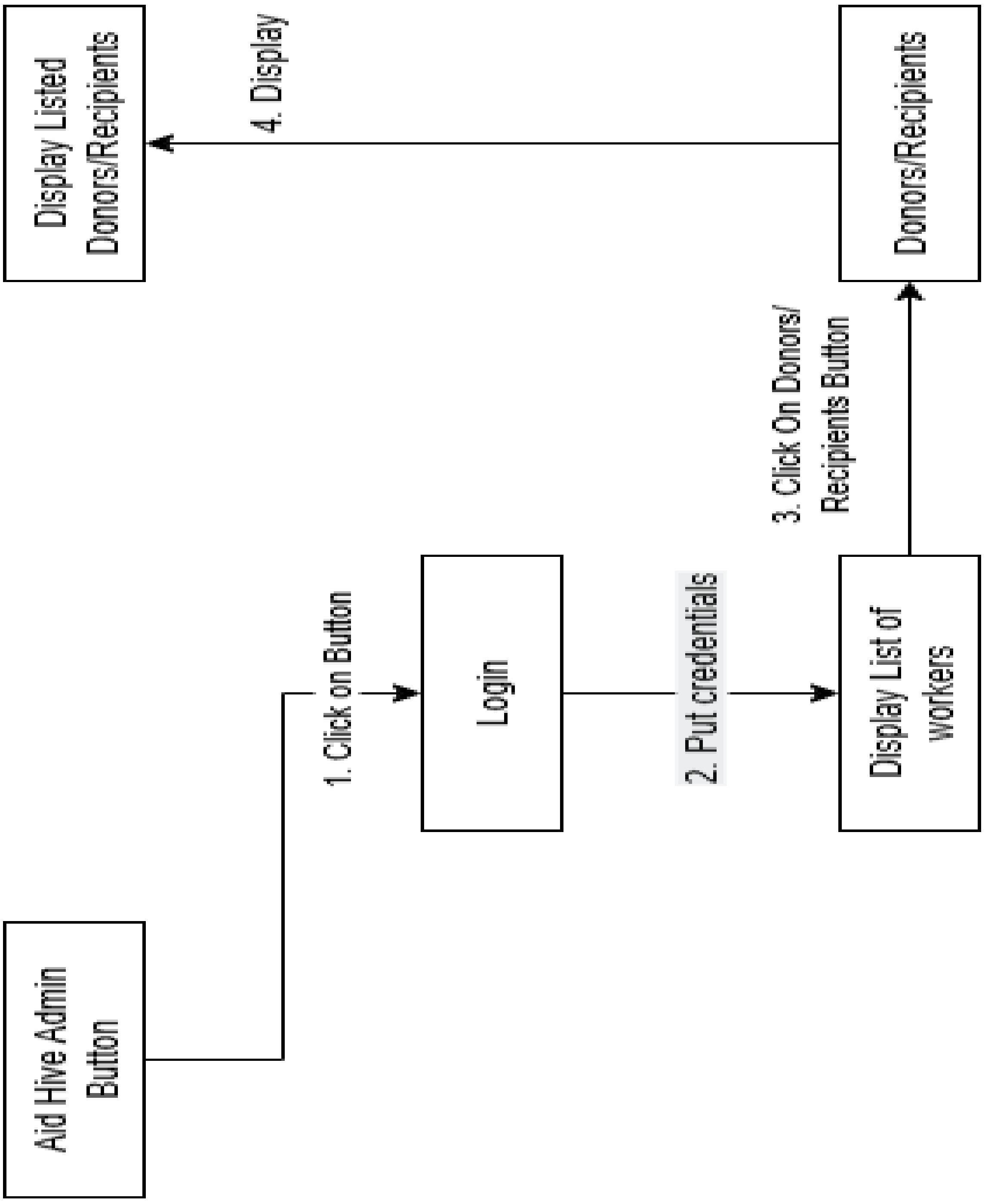
3.6.12 Delete Worker

Figure 25: Delete Worker



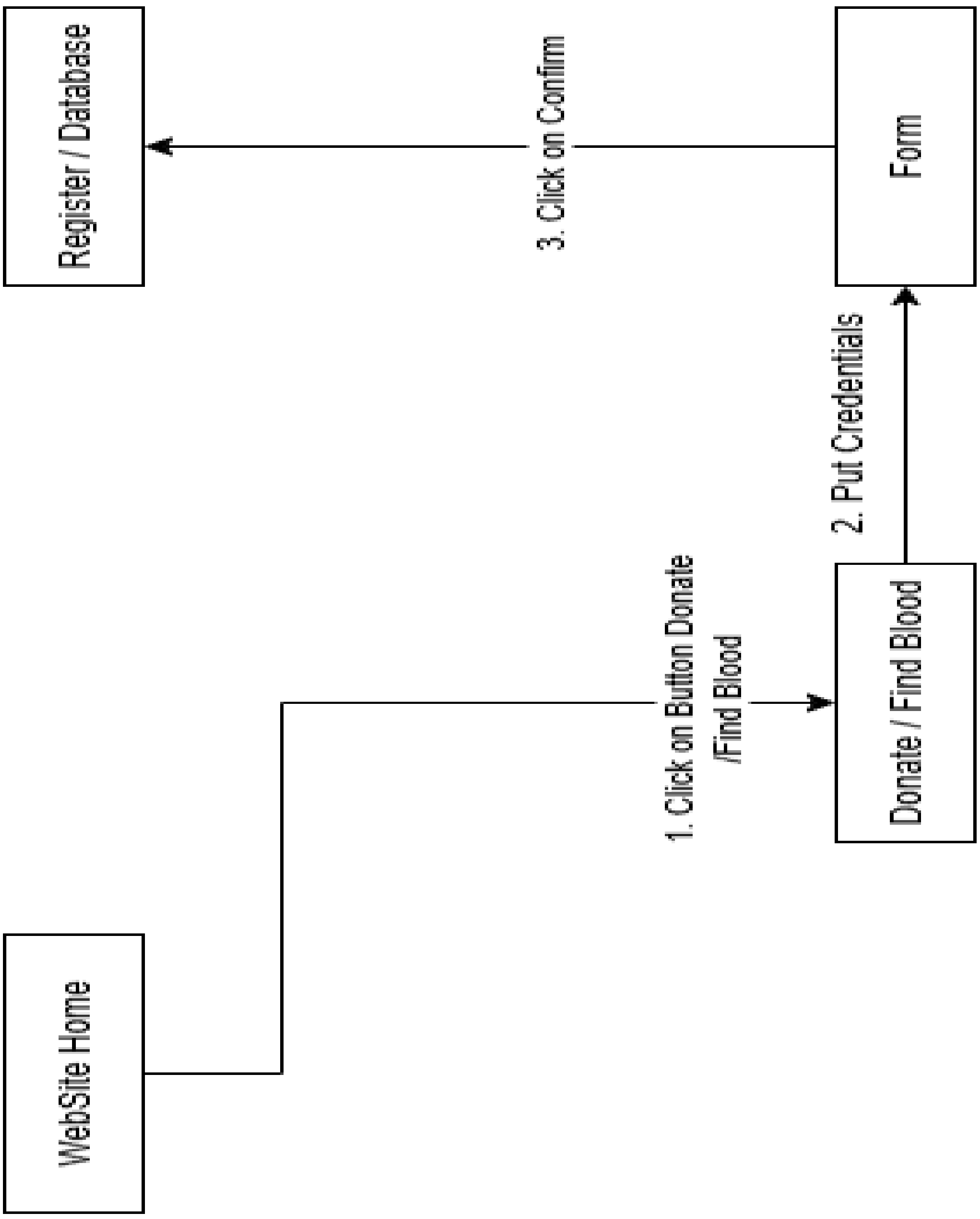
3.6.13 Display Listed Donor / Recipient

Figure 26: Display Listed Donor / Recipient



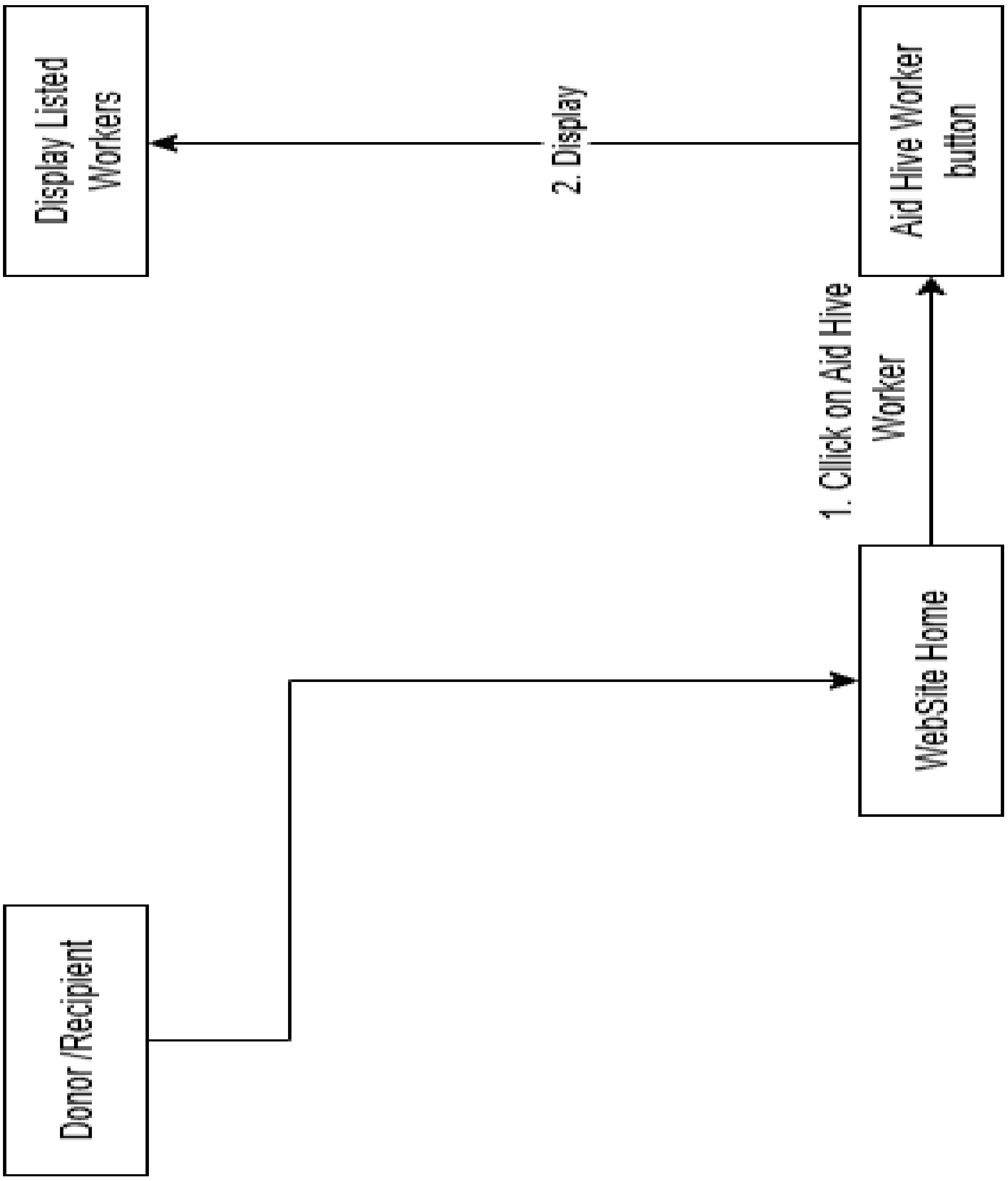
3.6.14 Donor / Recipient Registration

Figure 27: Donor / Recipient Registration



3.6.15 Donor / Recipient Display Worker List

Figure 28: Donor / Recipient Display Worker List



3.7 Design Decisions

The design decisions focus on simplicity, security, and proper role-based access:

1. **MERN Stack with MySQL:**
   - Frontend: React.js
   - Backend: Node.js + Express.js
   - Database: MySQL for structured data storage
2. **Form-Based Interaction for Users:**
   - Donors and recipients interact only through registration and blood request forms.
   - No login is required for users to submit data.
3. **Admin Login Authentication:**
   - Admin can access the system only via username and password.
   - Ensures secure and authorized management of the system.
4. **Three-Tier Architecture:**
   - Separation of frontend, backend, and database layers improves maintainability and scalability.
5. **User-Friendly Interface:**
   - Simple navigation and clear forms make donor and recipient interactions easy.
   - Admin panel is designed for efficient management and monitoring.
6. **Data Validation and Security:**
   - Form inputs are validated before submission.
   - Sensitive data is secured through backend and database protocols.

3.8 Summary

The Aid Hive Platform System Design is all about designing an efficient, secure, and accessible system. The donors and the recipients are in interaction only with the blood request forms and registration forms, whereas the admins use username and password login to enter the system. The system is deployed using a MERN stack along with MySQL for storing structured data and a three-tier architecture with layers of frontend, backend, and database.

Chapter 4

Implementation

4. Implementation

This chapter covers the implementation of the Aid Hive Blood Donation Management System. It explains how the designed modules were developed using MERN stack with MySQL database integration.

4.1 Code Repository

For version control, collaborative development, and efficient project management, the Aid Hive blood donation web application used Git as the main version control system. The repository included all source code, frontend and backend components, documentation, and related project resources. Using Git allowed the team to track changes, resolve conflicts, and collaborate effectively while working with the MERN stack. The repository served as a single source of truth, ensuring that all team members worked with the latest version of the system and enabling smooth coordination throughout the development process.

Git Repository Link

https://github.com/Abdur-Rehman153/Aid Hive.git.

4.2 Summary

The Implementation chapter elaborates that creating an efficient and secure blood donation portal with the MERN stack and MySQL as the database. The frontend was implemented using React.js and made responsive and easy to use, while the backend from Node.js and Express.js managed data operations, API routing, and admin features. Essential features like donor registration, recipient blood requests. Admin access is maintained through username and password login, and the system checks all form submissions for data integrity. All modules were thoroughly tested for performance, security, and systems requirements for real-time donation and recipient connection.

Chapter 5

Testing and Evaluation

5. Introduction

Testing and Evaluation of Aid Hive is designed to ensure the system works satisfactorily and meets user expectations. Throughout the development process, testing is used to confirm all the functionality from user sign-up and logon to donor matching and blood requests work correctly under different conditions.

Key principles guiding the testing process include:

- Conducting independent testing that does not rely on other test results.
- Writing short and clear statements to validate expected outcomes.
- Using test data properly with setup and teardown methods.
- Avoiding hardcoded values to make tests adaptable.
- Ensuring tests are repeatable and reliable each time they are executed.

5.1 Unit Testing (UT)

Unit testing entails testing separate components or functions of the Aid Hive system to ensure that each component works as it is supposed too independently. These are done early in development to catch small issues before they become big issues.

5.1.1 *Donor Registration*

Table 15: User Registration

Testcase ID	UT-1
Requirement ID	FR-1 (Donor-Registration)
Title	Register New Donor
Description	Tests if a new donor can register successfully
Objective	Ensure system stores new donor data correctly
Driver/precondition	User is not already registered
Test steps	1. Open registration form 2. Enter valid info 3. Submit 4. Check database
Input	Ali Hassan, contact.abhassan@gmail.com, Gujranwala, Hafizabad Road Sharukh colony, B+, 1234
Expected Results	Form loads, data accepted, success message, new record exists
Actual Result	From loads Data accepted Redirected with success User found in database
Remarks	Pass

5.1.2 Recipient Registration

Table 16: Recipient Registration

Testcase ID	UT-2
Requirement ID	FR-2 (Recipient Registration)
Title	Register New Recipient
Description	Tests if a new recipient can register successfully
Objective	Ensure system stores new recipient data correctly
Driver/precondition	User is not already registered
Test steps	1. Open registration form 2. Enter valid info 3. Submit 4. Check database
Input	Sara Khan, sara.khan@gmail.com, Lahore, Gulberg, O-, 5678
Expected Results	Form loads, data accepted, success message, new record exists
Actual Result	Success + redirect Error shown Error shown
Remarks	Pass

5.1.3 Admin Login

Table 17: Admin Login

Testcase ID	UT-3
Requirement ID	FR-2 (Admin Login)
Title	Admin Login
Description	Tests if admin can login with correct credentials
Objective	Ensure admin access works
Driver/precondition	Admin account exists
Test steps	1. Open admin login 2. Enter credentials 3. Submit
Input	admin@aidhive.com, password123
Expected Results	Login successful, dashboard opens
Actual Result	Successful login
Remarks	Pass

5.1.4 Find Blood

Table 18: Find Blood

Testcase ID	UT-4
Requirement ID	FR-4 (Find Blood)
Title	Search Blood Donors
Description	Tests if the system returns correct donors based on blood group & city
Objective	Ensure blood search works accurately
Driver/precondition	Donors exist in database
Test steps	1. Open Find Blood page 2. Enter blood group & city 3. Submit search
Input	Blood Group: B+, City: Gujranwala
Expected Results	List of matching donors displayed
Actual Result	Blood Matched
Remarks	Pass

5.2 Functional Testing (FT)

Functional testing is done to make sure that all the critical functionalities and user-oriented modules of Aid Hive function as desired. The goal is to make sure the system is as per the functional specifications and that the system is usable by all the stakeholders (Donor, Recipient, Admin) efficiently.

5.2.1 Donor Registration

Table 19: Donor Registration

Testcase ID	FT-1
Requirement ID	FR-1 (Donor Registration)
Title	Verify Donor Registration
Description	Ensures a donor can register successfully with valid inputs
Objective	Confirm donor data is stored correctly
Driver/precondition	User is not already registered
Test steps	Open form → Enter valid info → Submit → Verify database
Input	Ali Hassan, contact.abhassan@gmail.com, Gujranwala, Hafizabad Road Sharukh colony, B+, 1234
Expected Result	Form submits, success message, new record created
Actual Result	Works as expected
Remarks	Pass

5.2.2 Recipient Registration

Table 20: Recipient Registration

Testcase ID	FT-2
Requirement ID	FR-3 (Recipient Registration)
Title	Verify Recipient Registration
Description	Ensures a recipient can register successfully with valid inputs
Objective	Confirm recipient data is stored correctly
Driver/precondition	User is not already registered
Test steps	Open form → Enter valid info → Submit → Verify database
Input	Sara Khan, sara.khan@gmail.com, Lahore, Gulberg, O-, 5678
Expected Result	Form submits, success message, new record created
Actual Result	Successfully registered
Remarks	Pass

5.2.3 *Admin Login*

Table 21: Admin login

Testcase ID	FT-3
Requirement ID	FR-4 (Admin Login)
Title	Verify Admin Login
Description	Ensures admin can login with correct credentials
Objective	Confirm admin access works properly
Driver/precondition	Admin account exists
Test steps	Open login → Enter credentials → Submit
Input	admin@aidhive.com, password123
Expected Result	Admin dashboard opens
Actual Result	Successfully Login
Remarks	Pass

5.2.4 *Admin Panel Controls*

Table 22: Admin Control

Testcase ID	FT-5
Requirement ID	FR-5 (Admin Dashboard)
Title	View Users and Deactivate Account
Description	Admin views users/requests and deactivates accounts
Objective	Ensure administrative functions perform correctly
Driver/precondition	Admin logged in
Test steps	Login → View → Deactivate
Input	User ID = 1098
Expected Result	User status changes to inactive
Actual Result	Successfully updated
Remarks	Pass

5.3 Integration Testing (IT)

Integration testing was done after all the separate unit tests were successfully finished. Integration testing consisted of testing interactions of sets of modules to ensure that the overall flow of the features gives the desired outputs.

5.3.1 Request Submission Flow

Table 23: Test Case 1

Testcase ID	IT-1
Requirement ID	FR-1, FR-2
Title	User Registration to Request Flow
Description	Ensures system returns accurate donors based on blood group & city
Objective	Confirm blood search works correctly
Precondition	Donor exists in database
Test steps	Open Find Blood → Enter blood group & city → Submit
Input	Blood Group: B+, City: Gujranwala and further fields
Expected Result	Matching donors displayed
Actual Result	Works as expected
Remarks	Pass

5.3.2 User Login Blocked

Table 24: Test Case 2

Testcase ID	IT-2
Requirement ID	FR-1, FR-5
Title	Admin Deactivates User → Prevent Login
Description	Ensure admin block prevents user access via login
Objective	Validate integration of admin actions and authentication logic
Precondition	User exists; admin has access
Test steps	User tries login
Input	Username: admin, password: admin1234
Expected Result	Login fails with "Wrong Username or password" message
Actual Result	As expected,
Remarks	Pass

5.3.3 Matches with User History

Table 25: Test Case 3

Testcase ID	IT-6
Requirement ID	FR-5, FR-2
Title	Admin Review of User History
Description	Admin checks if user request logs match with system-wide records
Objective	Ensure transparency and data syncing between user and admin views
Precondition	Admin and recipient both logged in
Test steps	User submits → Admin reviews logs
Input	User ID = 3012, Request ID = 1204
Expected Result	Admin sees same request data as recipient
Actual Result	Matched perfectly
Remarks	Pass

5.4 Performance Testing (PT)

Following the testing of each module and integration, performance testing was done to determine the behavior of the system under stress, heavy load, and concurrent usage states. These were performed on response time, reliability, and stability.

5.4.1 Login API Response Time

Table 26: Test Case 1

Testcase ID	PT-1
Requirement ID	FR-1
Title	Login API Performance
Description	Test login response time with multiple simultaneous users
Objective	Ensure system handles authentication quickly under load
Precondition	4 users attempt login (valid & invalid credentials)
Test steps	Simulate login requests → Measure response time
Input	Email/password for 4 users
Expected Result	Response time < 1 second
Actual Result	~350ms average
Remarks	Pass

5.4.2 Request Creation Under Load

Table 27: Test Case 2

Testcase ID	PT-2
Requirement ID	FR-2
Title	Request for Submission Load Performance
Description	Assess how request creation behaves under simultaneous usage
Objective	Ensure the system remains responsive with multiple users
Driver/precondition	100 recipients submit blood requests at once
Test steps	Simulate load → Submit requests → Record time
Input	Request payloads with blood type, location, urgency
Expected Result	< 2 seconds per request
Actual Result	~1.5 seconds
Remarks	Pass

5.4.3 Admin Panel Performance

Table 28: Test Case 3

Testcase ID	PT-5
Requirement ID	FR-5
Title	Admin Dashboard Load Time
Description	Test how fast admin panel loads user and request data
Objective	Confirm UI responsiveness and database query performance
Driver/precondition	Admin logged in
Test steps	Access dashboard → View user & request tables
Input	Admin access to system database
Expected Result	Load time < 1.5 seconds
Actual Result	~910ms
Remarks	Pass

5.5 Summary

This chapter describes the testing of the Aid Hive Blood Donation Management System. Different types of testing such as unit, functional, integration, and performance were applied on modules like authentication, donor–recipient requests, donor matching, and admin features. The system worked correctly, responded efficiently, and fulfilled the functional requirements, showing that Aid Hive is reliable and ready for deployment.

Chapter 6
System Conversion

6. Introduction

The System Conversion from Aid Hive updates the conventional, manual procedures of blood donations to a centralized, web-based system. Major operations like donor registration, management of blood requests, matching of donors and recipients are relocated to an efficient online platform, thereby accelerating the process, making it more efficient and friendly to use.

6.1 Conversion Method

For the Aid Hive system, a direct conversion method is used. This means the manual and standalone processes of blood donation (such as paper-based donor lists, offline request handling, and manual matching) are replaced immediately by the new centralized web-based system.

The direct conversion approach was chosen because:

- The system is relatively new and not dependent on heavy legacy infrastructure.
- It ensures all users (donors, recipients, and admin) interact with a single updated platform without relying on outdated methods.

Direct Conversion

The Aid Hive system adopts a Direct Conversion method, where the existing manual blood donation process is completely replaced by the new web-based system in a single step. Once the system is deployed, all stakeholders—including donors, recipients, and administrators—will begin using the new platform comfortably, without relying on old manual methods.

This approach was selected because it provides:

- Immediate transition to a centralized system for blood donation.
- Elimination of duplicate processes, ensuring everyone uses one updated platform.
- Faster adoption by all users, as the old paper-based procedures are discontinued.

Parallel Conversion

The Aid Hive system uses a Parallel Conversion method, in which the new web-based platform runs side by side with the existing manual process for a limited period of time. During this phase, blood donation requests, donor registrations, and recipient matching are handled both manually and through the Aid Hive portal.

This method was chosen because it:

- Provides a safety net, as the old system continues to function until the new system is fully reliable.
- Helps detect and fix any technical issues in the Aid Hive system without interrupting ongoing blood donation activities.
- Allows users to gradually adapt to the new system, reducing training challenges and resistance.

Pilot Conversion

The Aid Hive system can also be deployed using a Pilot Conversion method, where the new web-based blood donation platform is first introduced to a small group of users or a specific region before being rolled out organization-wide. In this approach, a limited number of donors, recipients, and administrators are given access to the Aid Hive system to perform real-world tasks such as donor registration, blood request posting, and location-based donor searches.

Suitability for Aid Hive

For the Aid Hive system, the most suitable approach is a Pilot Conversion, where the platform is first introduced to a limited group of users such as a single hospital, a specific blood bank, or a small community of donors and recipients. In this pilot phase, the selected group will use the Aid Hive system for donor registration, blood requests and while the rest of the users continue with the manual process.

Phased Conversion

In the Phased Conversion approach, the Aid Hive system is implemented gradually in stages, rather than all at once. Specific modules such as Donor Registration, Blood Request Posting and Admin Panel are rolled out one after another, allowing each component to be tested and stabilized before moving on to the next.

Suitability for Aid Hive

The Phased Conversion method is suitable for Aid Hive because it allows the system to be introduced step by step instead of all at once. Since Aid Hive contains multiple critical modules such as Donor Registration, Blood Request Management and Admin Panel rolling them out in phases ensures that each feature is properly tested, validated, and accepted by users before moving forward.

6.2 Summary

The Aid Hive system was converted from a manual blood donation process into a centralized, web-based platform using a structured conversion strategy. Different methods such as Direct, Parallel, Pilot, and Phased Conversion were considered, with Phased Conversion identified as the most suitable due to its low risk, gradual rollout, and ease of user adoption. This ensured that modules like Donor Registration, Blood Requests and Admin Panel were introduced step by step, minimizing errors. The conversion process successfully established a reliable, user-friendly, and efficient system.

Chapter 7
Conclusion

7. Introduction

The Aid Hive Blood Donation Management System, built on the MERN stack with MySQL, streamlines blood donation through five pages Home, Donate, Find Blood, Aid Hive Workers, and About Us with one key feature: admin panel. Donors and recipients only interact by filling registration forms, while data management and approvals are handled securely by the admin. This simple yet effective structure makes the system user-friendly, reliable, and well-organized for managing blood donation activities.

7.1 Evaluation

Table 29 : Evaluation Table

Objectives	Status
Allow donors to update their availability status directly from their dashboard.	Completed
Keep user data secure and private, complying with standard security practices.	Completed
Ensure the system is fully responsive and works well on desktops, laptops, and mobile devices.	Completed
Provide a clear and user-friendly interface using React.js and Tailwind CSS.	Completed
Allow admins to manage donor and recipient requests.	Completed
Provide role-based dashboards for both donors and recipients with personalized features.	Completed
Store and manage donation and request history for all registered users.	Completed

7.2 Traceability Matrix

Following is a table of all the requirements for Aid Hive with their brief description and design specifications.

Table 30 : Requirement Traceability Matrix

Requirement ID	**Requirement Description**	**Design Specification**	**Code**	**Test ID**
R1	The system must allow users to store and retrieve blood donor/recipient profiles and request data.	Component: *User Authentication, Profile Management*	auth.js, user.js, profileRoutes.js	T01
R2	The system must recommend blood donors based on location and blood type.	Component: *AI-Powered Recommendation Engine*	matchEngine.js, donorFilter.js	T02
R3	The system must track the status of blood requests (pending, fulfilled, etc.).	Component: *Request Status Tracking*	requestStatus.js, requestRoutes.js	T04
R4	The system must allow users to view their donation history and track past activities.	Component: *Donation History Management*	history.js, userDashboard.js	T05
R5	The system must provide a user-friendly interface for donation and request submission.	Component: *User Interface Design*	Form Component.js requestForm.css	T06
R6	History management for tracking past donations.	Design: *Database Logging (My sql)*	mysql.js, donationLogs.js	T07

7.3 Conclusion

In conclusion, Aid Hive offers a straightforward yet efficient platform for linking blood donors and recipients with the system remaining secure and controlled via the admin panel. Recipients and donors only communicate through the completion of registration forms, and admin controls all records and approvals via a safe login. The organized mechanism enables user-friendliness, reliability, and proper management of blood donation activities, making the system a feasible solution for processing the donation procedure.

Correlation Table: Objectives vs. Functionality

Table 31 : Correlation Table

Defined Objective	**Functionality Provided**
Strong Security Purposes	Password hashing by bcrypt, login-brute force protection
Easy access and usability across all devices	Fully responsive frontend using React.js and Tailwind CSS
Track and manage blood donation history	Profile management with past donation forms stored in My SQL
Simplified blood request process	Structured request forms with validation and user-friendly submission interface

7.4 Future Work

Even though the Aid Hive system has fulfilled its original objectives, there is always potential for future growth and expansion. Among the proposed advancements are:

- **Mobile Application Development**:
  To improve accessibility and convenience for users while they are on the go, a cross-platform mobile application version will be developed.
- **AI-Based Donor Matching**:

This method uses computer programs with artificial intelligence to suggest the best donors based on urgency and historical data.

- **Multilingual Support**:

Providing multilingual support will make the platform more accessible to users from various geographical locations.

- **Hospital Integration**:

Enabling real-time donor data viewing and mass donation campaign management for hospital accounts.

- **Offline Alert System**:

In places with poor internet connectivity, use SMS-based alert systems.

References

MySQL Documentation. Retrieved from

https://www.mysql.com/docs/manual/

React. (n.d.). React.js Documentation. Retrieved from

https://react.dev/

Tailwind CSS. (n.d.). Tailwind CSS Documentation. Retrieved from

https://tailwindcss.com/

Express. (n.d.). Express.js Documentation. Retrieved from

https://expressjs.com/

Node.js. (n.d.). Node.js Documentation. Retrieved from

https://nodejs.org/docs/latest/api/documentation.html

Google. "OAuth 2.0 for Web Applications." Internet:

https://developers.google.com/identity/protocols/oauth2/

Appendix A

Use case Description Template

Appendix-A Use Case Description (Fully Dressed Format)

The Table A-1 below indicates a comprehensive use case template

Table 32 : Show the detail use case template and example

Use Case ID	UC-1
Use Case Name	Register as a Blood Donor
Actors	Donor (User), System (Aid Hive Web App)
Description	This use case allows a new user to register as a blood donor by filling in their personal information, blood type, and availability. Once registered, they can be matched with recipients.
Trigger	The donor clicks on the "Donate" button on the homepage or navbar.
Preconditions	The donor must have internet access and a valid email address.
Postconditions	The donor's data is saved in MySQL, and they receive a confirmation email via Node mailer. They are redirected to the donor dashboard.
Normal Flow	Donor navigates to the "Donor Registration" page. System displays the registration form. Donor enters full name, email, password, blood type, location, and availability. Donor clicks the "Submit" button. Frontend (React) performs validation on all fields (e.g., required fields, email format, password strength) If valid, form data is sent via Axios to the Express.js backend. Backend hashes the password using crypt, stores the data in My SQL. System sends a confirmation email using Node mailer. System responds with a success message and redirects to the dashboard.
Alternative Flows	AF-1: *(At Step 3)* If the user enters an already registered email address, the system displays an error message: *"Email already exists."* AF-2: *(At Step 5)* If any field is invalid (e.g., empty or wrong format), the system highlights the error and disables submission.
Business Rules	Only users 18 years and above can register. Blood type must be selected from a fixed dropdown list. Donors must confirm availability status (Yes/No). Email format must match RFC 5322 standards. Password must have at least 8 characters including uppercase, lowercase, and numbers. Backend checks for existing accounts to prevent duplicates.