PROJECT
REPORT

ACCESS SHARE

Connecting People with Healthcare Resources and Assistive Devices

SUBMITTED BY:

Aayushi Kumari
Likitha Konam
Sruthilaya Banothu
Vatista Kachroo

TABLE OF CONTENTS

01
Executive Summary

02
Requirements

03
Design and Architecture

04
Snippets

05
Future Scope

06
References

EXECUTIVE SUMMARY

Access Share is a collaborative platform that aims to streamline the sharing and requesting of critical healthcare resources. Its primary objectives are to:

- Allow users to donate or share assistive devices.
- Enable recipients to find and request devices based on location or type.
- Provide a caregiving listing and request system where caregivers can advertise their
- services, and those seeking assistance can easily find them.
- Facilitate blood donation requests and responses, linking donors and recipients in real-time

The platform features a React-based front [2] styled with Tailwind CSS [3], and a FastAPI backend [1] employing SQLAlchemy Authors [5] and Alembic [6] for data modeling and migrations. By focusing on a clean user interface, robust API design, and careful data management, Access Share strives to improve community health and resource accessibility in a sustainable, user-friendly manner.

This report outlines the **system requirements**, **high-level design** and **architecture**, key **use cases** via diagrams, a few **visual snippets,** and **future enhancements** planned.

REQUIREMENTS

Key requirements for Access Share include:

1. Secure Registration and Login: Store hashed passwords, handle user sessions, and protect sensitive actions.

2. Device Listings and Requests: Donors post device details; recipients can browse and request them.

3. Caregiving Listings and Scheduling: Caregivers list services; seekers can filter by location or expertise.

4. Blood Donation Requests: Users specify blood type, location, and urgency; donors respond accordingly.

5. Status Updates: Listings transition from "available" to "requested" to "taken," re-flecting current availability.

6. Profile Management: Users can edit their profile details including username, password, and contact number, or choose to permanently delete their account.

DESIGN & ARCHITECTURE

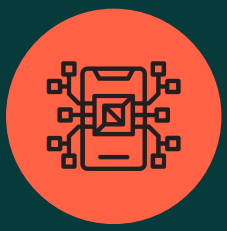
The Design & Architecture of Access Share emphasizes a layered approach:

- **Client Layer:** A React + Tailwind UI for interactive, responsive user experiences.
- **Server Layer:** FastAPI endpoints to handle RESTful communication, containing business logic for device management, caregiving workflows, and blood donation coordination.
- **Data Layer:** A PostgreSQL database with SQLAlchemy and Alembic ensuring reliable data storage and versioned schema migrations

Following figures illustrates three aspects:

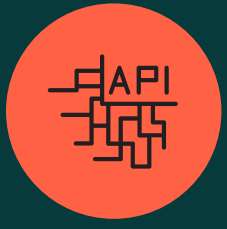
a - System Architecture

Shows the interaction between users, the frontend, backend services, and the database. Each layer is clearly separated and communicates through API requests and responses.



b- ER Diagram

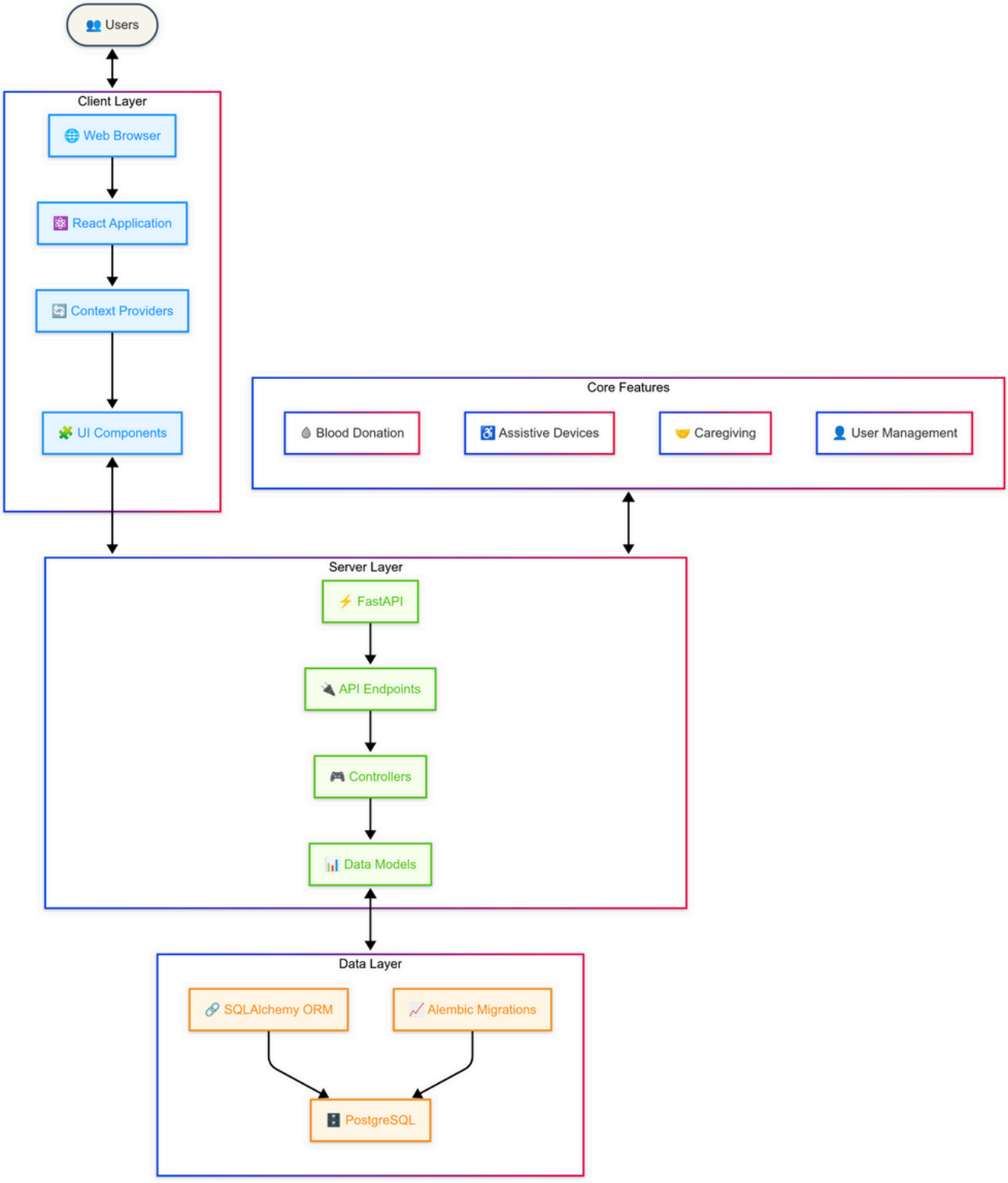
Displays the main entities such as User, AssistiveDeviceListing, CaregiverListing, and BloodDonationRequest, along with how they relate through foreign keys and ownership.



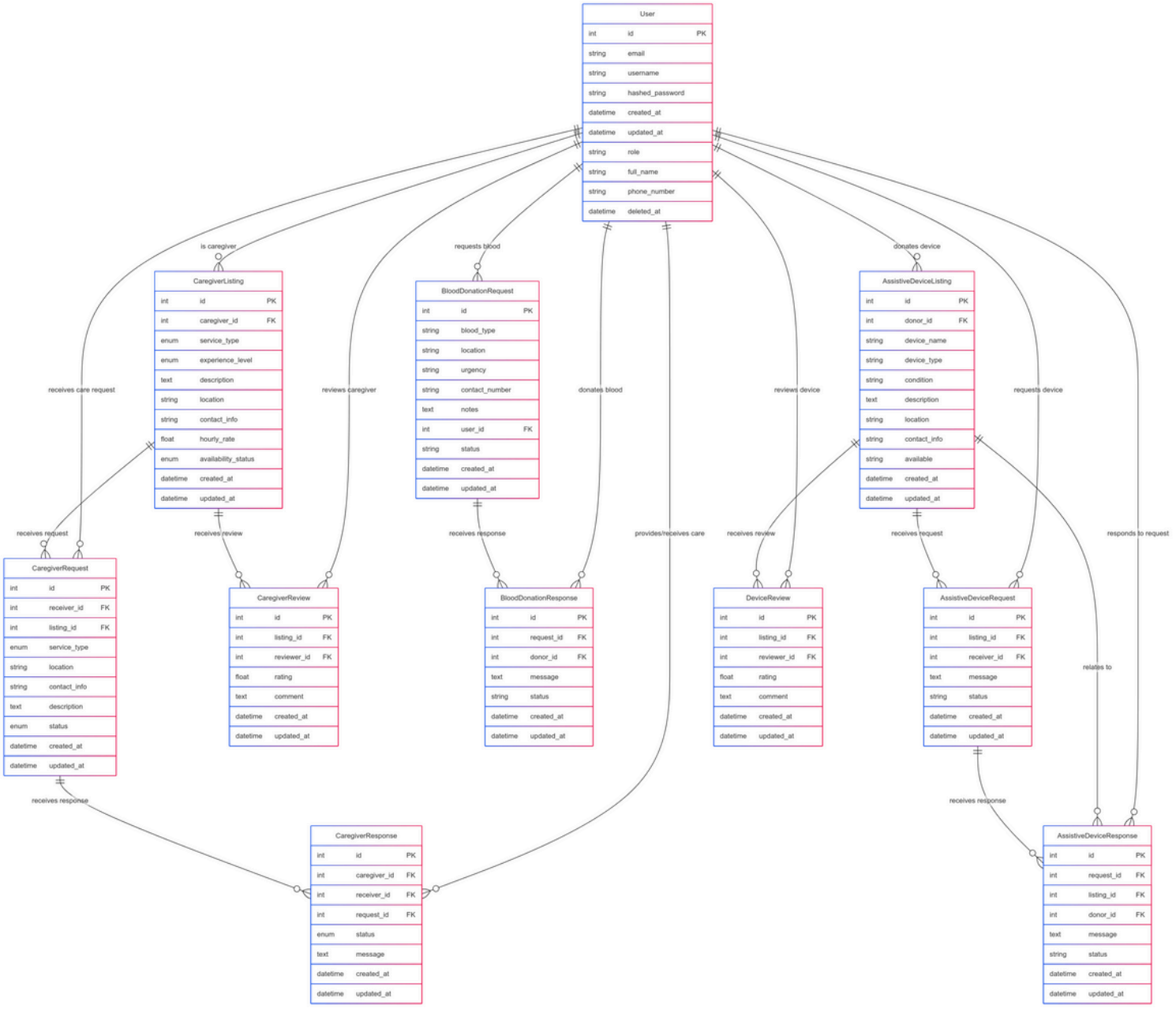
c- Assistive Device Sequence Diagram

Outlines the step-by-step flow for assistive device interactions—from donor listing, recipient request, status updates, to confirmation—across the UI, API, and database.

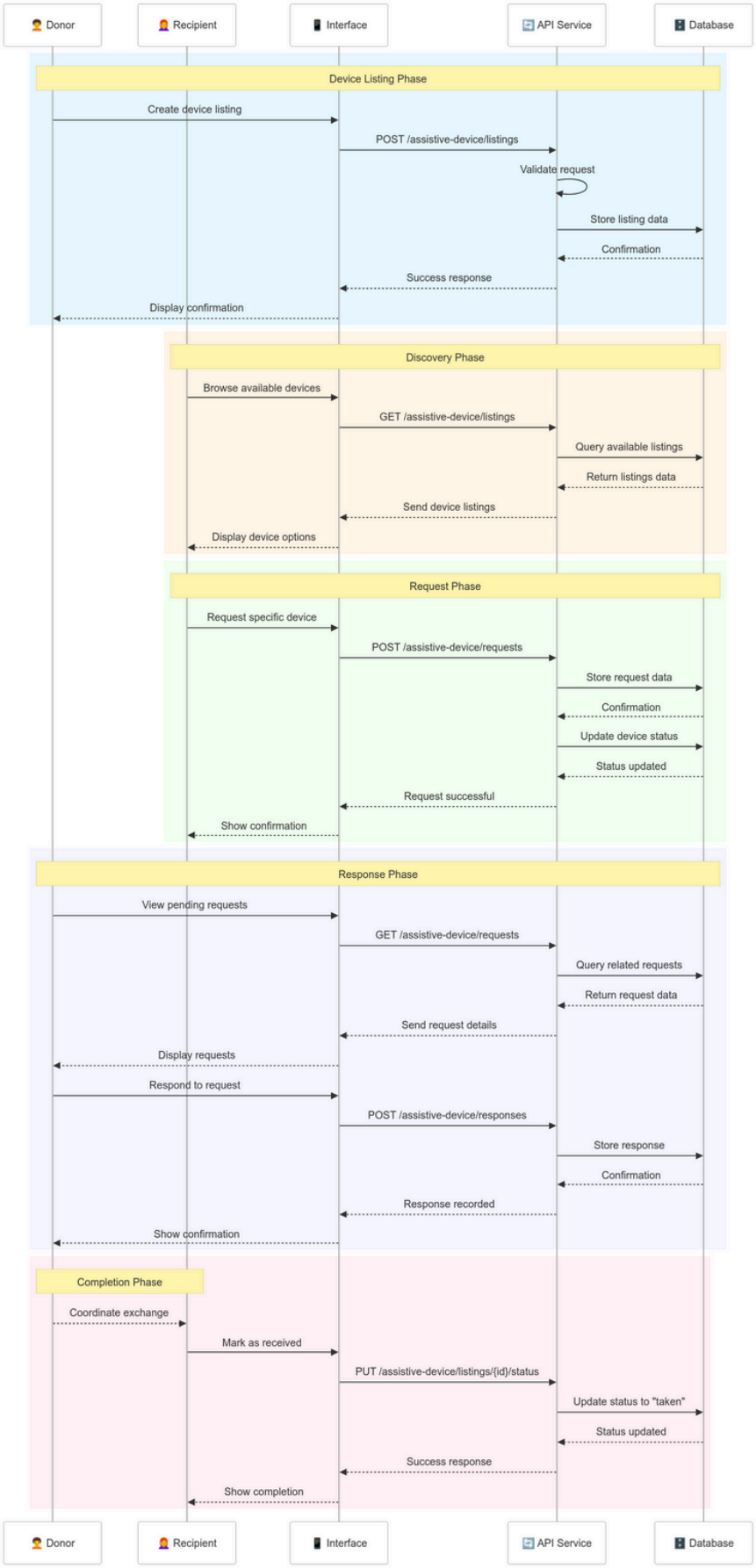
SYSTEM ARCHITECTURE



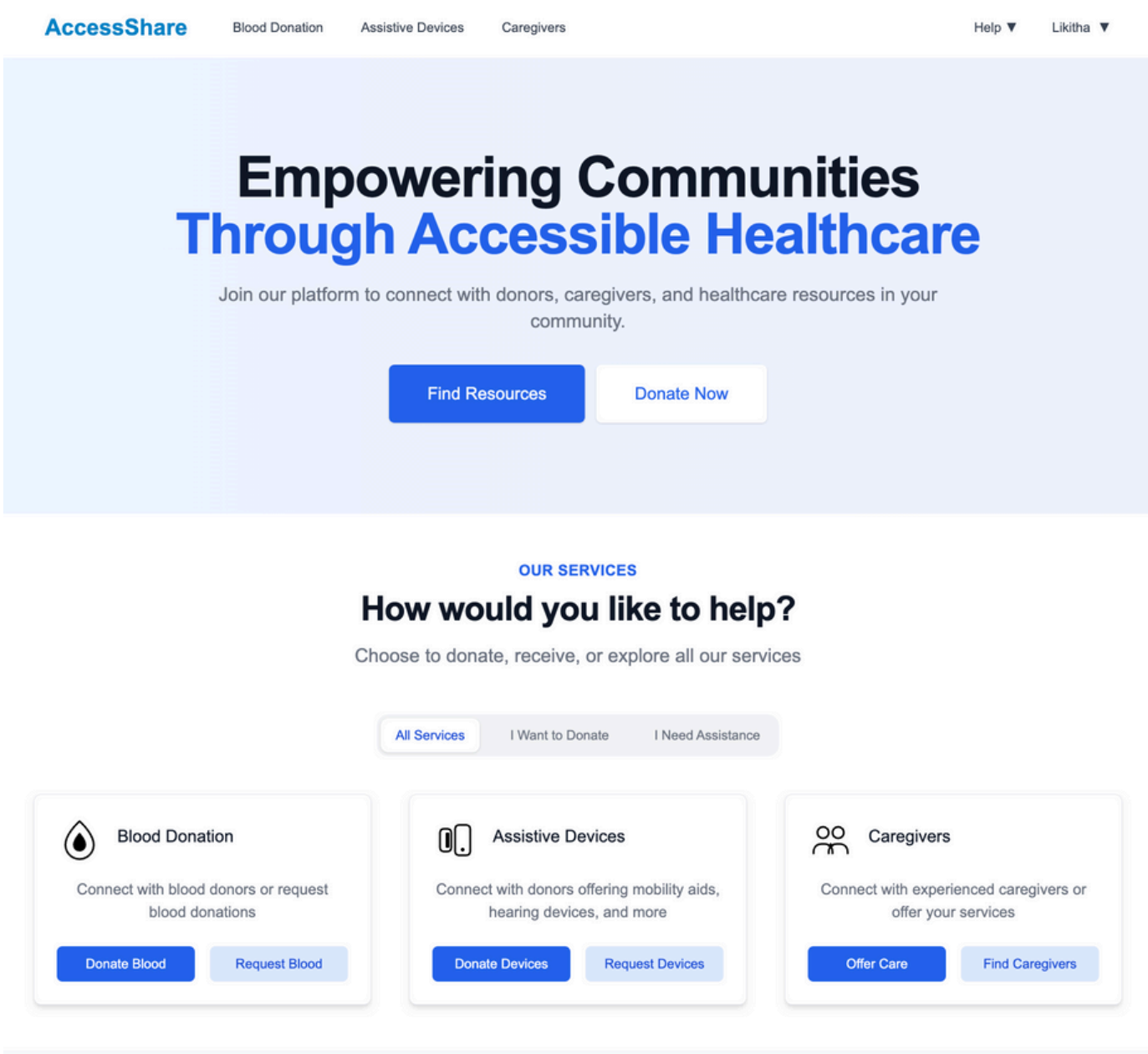
ER DIAGRAM


User
int id PK
string email
string username
string hashed_password
datetime created_at
datetime updated_at
string role
string full_name
string phone_number
datetime deleted_at
is caregiver
requests blood
donates device
CaregiverListing
int id PK
int caregiver_id FK
enum service_type
enum experience_level
text description
string location
string contact_info
float hourly_rate
enum availability_status
datetime created_at
datetime updated_at
BloodDonationRequest
int id PK
string blood_type
string location
string urgency
string contact_number
text notes
int user_id FK
string status
datetime created_at
datetime updated_at
AssistiveDeviceListing
int id PK
int donor_id FK
string device_name
string device_type
string condition
text description
string location
string contact_info
string available
datetime created_at
datetime updated_at
receives care request
reviews caregiver
donates blood
reviews device
requests device
receives request
receives review
receives response
provides/receives care
receives review
receives request
responds to request
CaregiverRequest
int id PK
int receiver_id FK
int listing_id FK
enum service_type
string location
string contact_info
text description
enum status
datetime created_at
datetime updated_at
CaregiverReview
int id PK
int listing_id FK
int reviewer_id FK
float rating
text comment
datetime created_at
datetime updated_at
BloodDonationResponse
int id PK
int request_id FK
int donor_id FK
text message
string status
datetime created_at
datetime updated_at
DeviceReview
int id PK
int listing_id FK
int reviewer_id FK
float rating
text comment
datetime created_at
datetime updated_at
AssistiveDeviceRequest
int id PK
int listing_id FK
int receiver_id FK
text message
string status
datetime created_at
datetime updated_at
relates to
receives response
receives response
CaregiverResponse
int id PK
int caregiver_id FK
int receiver_id FK
int request_id FK
enum status
text message
datetime created_at
datetime updated_at
AssistiveDeviceResponse
int id PK
int request_id FK
int listing_id FK
int donor_id FK
text message
string status
datetime created_at
datetime updated_at

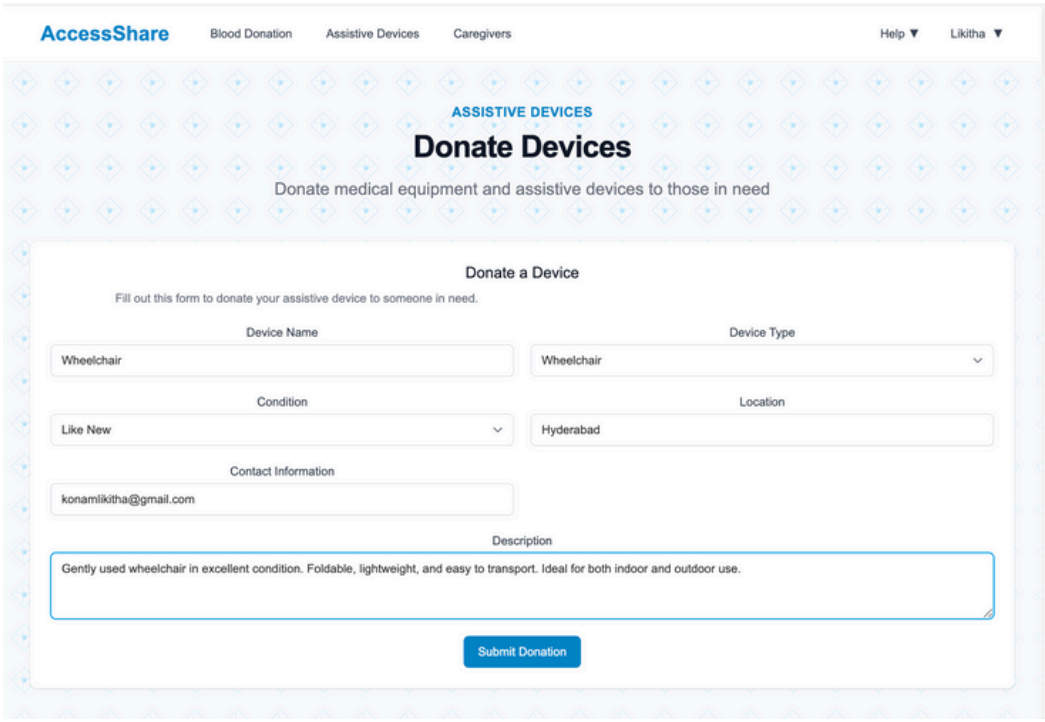
ASSISTIVE DEVICE SEQUENCE DIAGRAM



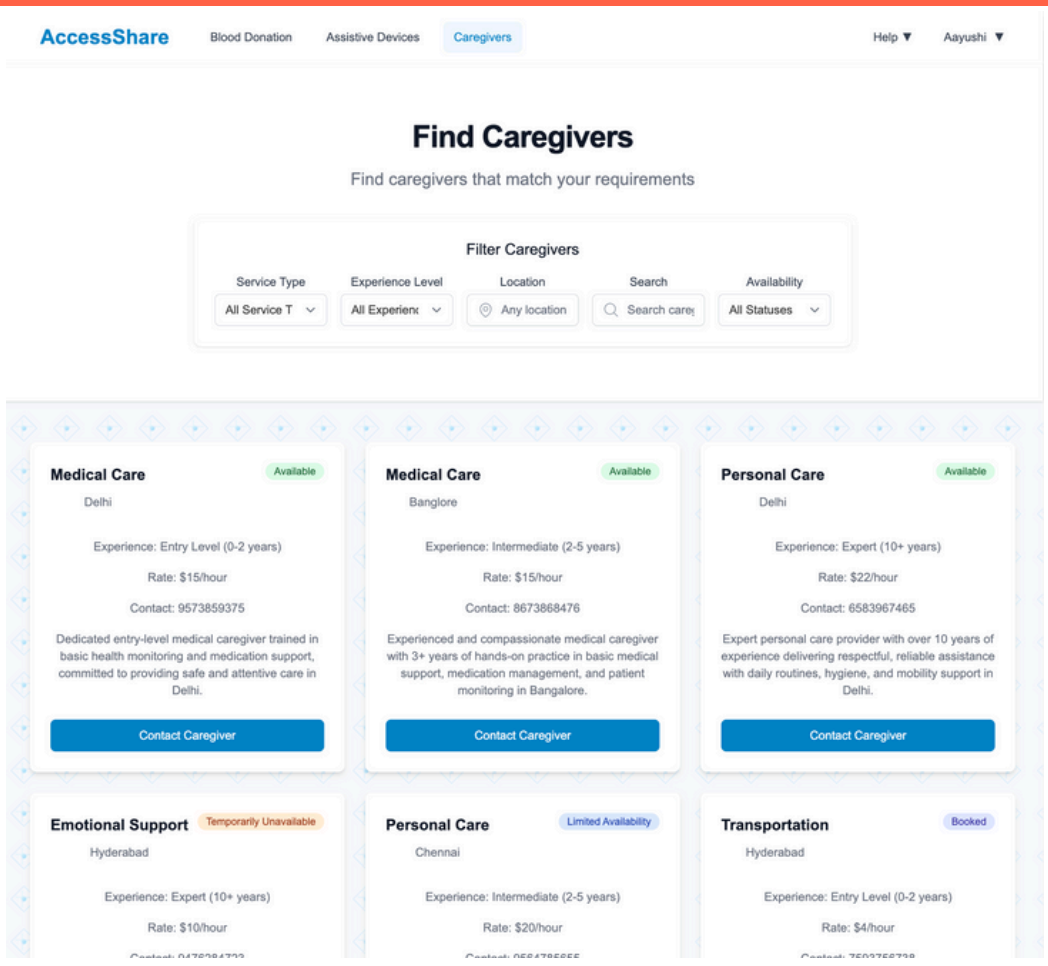
Snippets



Landing Page



Device Donation Page



Caregiver Listings Page

FUTURE SCOPE

While the current implementation addresses key community needs, the platform can evolve further in functionality, accessibility, and reach. This section outlines the proposed enhancements that will strengthen its impact, improve user engagement, and support more comprehensive resource sharing across diverse user groups.

01 — Notification Engine

Introduce real-time notifications for listing updates, incoming requests, and approvals. This keeps users informed without needing to constantly check the platform, and ensures time-sensitive actions—like device handovers or caregiver responses—are not missed.

02 — Advanced Authentication

Implement OAuth 2.0 and JWT-based flows for improved account security and flexibility. Users will be able to log in using trusted services like Google or institutional email, and manage sessions securely from multiple devices.

03 — Wanted Requests

Enable users to post specific needs even when no listing is available. This feature allows the community to respond directly to unmet needs—encouraging donors to step forward with relevant resources and filling critical gaps in availability.

04 — Integrated Payments

For caregiving services that go beyond voluntary support, add payment integration via trusted gateways. This allows caregivers to be compensated where appropriate and helps standardize pricing, tracking, and verification.

05 — Analytics Dashboard

Develop role-based dashboards for admins, donors, and caregivers. Track usage metrics, resource impact, and user activity trends to measure performance, guide outreach, and support data-driven improvements across the platform.

REFERENCES

1.Ramírez, S. (2025)
Documentation: https://fastapi.tiangolo.com/

2. React Team (2025)
Documentation: https://react.dev/

3. Tailwind Labs (2025)
Documentation: https://tailwindcss.com/docs

4. TanStack Team (2025),
Documentation: https://tanstack.com/query/latest

5. SQLAlchemy Authors (2025)
Documentation: https://docs.sqlalchemy.org/

6. Alembic Authors (2025)
Documentation: https://alembic.sqlalchemy.org/