Chapter 1

Introduction

The Organ Donation and Procurement Network Management System is a centralized platform designed to streamline organ donation, procurement, and transplantation. It integrates donors, recipients, hospitals, and regulatory bodies to ensure efficient organ tracking and utilization. By automating donor-recipient matching, maintaining medical records, and enabling real-time tracking, the system minimizes organ wastage and improves healthcare accessibility. Additionally, it incorporates AI for compatibility prediction and integrates RDBMS with NoSQL for seamless data management.

1.1 Existing System

The current organ donation and transplantation process faces several challenges:

- Lack of Centralized Data: Donor and recipient information is scattered across multiple locations, leading to inefficiencies in matching suitable candidates.
- Manual Processes: Tracking organ availability and coordinating between hospitals is often manual, increasing the chances of errors and delays.
- Organ Wastage: Due to poor tracking systems, many organs go unused despite the urgent need.

1.2 Proposed System

The proposed Organ Donation and Procurement Network Management System aims to overcome the inefficiencies of the current process by providing a centralized, automated, and technologically advanced solution. This system integrates donors, recipients, hospitals, and regulatory bodies to ensure seamless organ donation, procurement, and transplantation while adhering to legal and ethical standards.

Key Features:

- **Centralized Database**: Maintains comprehensive medical records of donors, recipients, hospitals, and medical professionals, ensuring accessibility and data consistency.

- **Real-time Tracking**: Continuously monitors organ availability, transportation, and storage to minimize delays and prevent organ wastage.
- **Regulatory Compliance**: Automates record-keeping to ensure strict adherence to legal and ethical standards, enhancing transparency and public trust.
- **Integration of RDBMS and NoSQL**: Utilizes a hybrid database approach to manage structured (donor-recipient details, hospital records) and unstructured (medical reports, feedback) data efficiently.
- **Healthcare Accessibility**: Streamlines organ procurement and transplantation to reduce waiting times, ensuring timely medical interventions for patients in critical need.
- **Public Awareness and Engagement**: Includes educational tools and awareness programs to encourage voluntary organ donation and increase participation.
- **Secure and Interoperable System**: Implements APIs to enable seamless data exchange between hospitals, government agencies, and transplant coordinators, ensuring efficient communication and decision-making.

By incorporating artificial intelligence, real-time tracking, and a hybrid database structure, this system creates a transparent, efficient, and reliable framework for organ donation and procurement. It significantly reduces organ wastage, improves healthcare accessibility, and strengthens regulatory compliance, ultimately saving more lives.

Chapter 2

Requirement Specification

2.1 Software Requirements

- Operating System: Windows, Linux, or macOS
- Database Management Systems: MySQL/PostgreSQL (for RDBMS), MongoDB (for NoSQL)
- Programming Languages: Python (for backend), JavaScript (for frontend)
- Frameworks: Flask
- IDE: Visual Studio Code

2.2 Hardware Requirements

- Server: Minimum 4-core CPU, 16 GB RAM, 500 GB SSD (for hosting the application)
- Client Systems: Minimum dual-core processor, 4 GB RAM, and 20 GB free disk space
- Network: Stable internet connection with at least 1 Mbps speed

2.3 Functional Requirements

1. **Donor Management**
    - **Add/Edit/Delete Donor Information:** Allow staff to manage donor profiles, including personal details and associated organs available for donation.
2. **Patient Management**
    - **Add/Edit/Delete Patient Information:** Manage patient profiles with details such as name, address, phone number, date of birth, and age, along with their organ requirements stored in the received relationship.
3. **Doctor Management**
    - **Add/Edit/Delete Doctor Information:** Maintain doctor profiles, including ID, name, specialization, and telephone number, ensuring accurate records of their expertise and hospital affiliation.

- **Patient Assignment:** Assign doctors to patients using the attended by relationship, based on specialization and availability, ensuring appropriate medical care.

4. **Organ Inventory Management**
   - **Organ Availability Tracking:** Monitor organ details such as ID and name in the organ entity, along with availability status to ensure efficient allocation.
   - **Donor-Organ Mapping:** Link organs to specific donors using the donates relationship, ensuring traceability and proper matching with patient needs.
5. **Transaction Management**
   - **Transaction Logging:** Record all transactions in the transaction ID field, capturing details like the date of donation and the status of organ procurement and transplantation.
   - **Organ Allocation:** Facilitate organ allocation to patients using the donates to and receives relationships, prioritizing based on urgency and compatibility.

2.4 Non-Functional Requirement

- **Performance:** The system must support up to 500 concurrent users with response times of under 2 seconds for critical operations like organ matching and transaction processing.
- **Scalability:** The system should seamlessly scale to handle increased data and support multiple organ donation centers.
- **Availability:**
  - Ensure 99.9% uptime with scheduled downtime under 2 hours per quarter.
  - Use load balancing and failover mechanisms to ensure uninterrupted access.
- **Reliability:**
  - The system should operate stably without crashes or data loss.
  - Implement robust error handling and regular backups to enable data recovery within 2 hours.
- **Usability:** Provide an intuitive, mobile-responsive user interface that supports multiple languages.

- **ACID Properties:** Ensure atomicity (full transactions), consistency (valid states), isolation (no conflicts in concurrent operations), and durability (permanent data after commits).

Chapter 3

Entity Relationship Diagram

3.1 Preamble

The Entity-Relationship (ER) Diagram represents the database structure for an Organ Donation and Procurement Network. This diagram models the key entities involved in the organ donation and transplantation process, including Donors, Patients, Hospitals, Medical Professionals, and Organs. The purpose of this ER diagram is to provide a clear and organized view of how these entities interact, ensuring a streamlined system for managing organ donations and transplants.

Entities:

1. DONOR: Stores information about organ donors, including their personal details and donation history, ensuring accurate tracking of available organs.
2. PATIENT: Represents individuals in need of organ transplants, storing their personal and medical details.
3. HOSPITAL: Maintains data on hospitals that facilitate organ transplants, tracking their contact details and locations.
4. MEDICAL_PROFESSIONAL: Contains details of doctors and medical professionals responsible for transplant procedures, including their specialization and hospital affiliation.
5. ORGANS: Tracks the available organs for donation, ensuring proper allocation to patients in need.
6. DONATES_TO: Represents the relationship between Donors and Patients, recording details of successful organ transplants.
7. WORKS_IN: Establishes a relationship between Medical Professionals and Hospitals, indicating where each professional is employed.
8. ADMITTED_TO: Links Patients to Hospitals, storing the admission details of patients undergoing transplant procedures.

Relationships:

- Each Donor may donate multiple Organs, and these donations are tracked through the DONATES_TO relationship.
- A Patient may receive organs from multiple Donors, and these transactions are recorded with details such as the date of donation.
- Each Hospital hosts multiple Patients, managing their admission and treatment.
- Medical Professionals work in hospitals and are responsible for conducting transplants.

This ER diagram effectively captures the core functionalities of an organ donation network, ensuring efficient tracking, management, and allocation of organs for transplant procedures.

3.2 Entities & Attributes

- DONOR: Stores details of organ donors.
  - Attributes: Donor_ID (Primary Key), D_Fname, D_Lname, D_Address, D_Dob, D_PhNo.
- PATIENT: Stores details of transplant patients.
  - Attributes: Patient_ID (Primary Key),P_Name, P_Fname, P_Lname, P_Address, P_PhNo, P_Dob, P_Age.
- HOSPITAL: Maintains information about hospitals.
  - Attributes: Hospital_ID (Primary Key), Hospital_Name, Hos_Address, Hos_Ph_No.
- MEDICAL_PROFESSIONAL: Stores data on medical professionals performing transplants.
  - Attributes: Professional_ID (Primary Key), Name, Specialization.
- ORGANS: Represents organs available for transplantation.
  - Attributes: Organ_ID (Primary Key), Organ_Name.
- DONATES_TO: Represents the donation relationship between Donors and Patients.
  - Attributes: Transaction_ID (Primary Key), Donor_ID (Foreign Key), Patient_ID (Foreign Key), Date_of_Donation, Amount_Paid.
- WORKS_IN: Links Medical Professionals to Hospitals.
  - Attributes: Professional_ID (Foreign Key), Hospital_ID (Foreign Key).

- ADMITTED_TO: Stores the relationship between Patients and Hospitals.
    - Attributes: Patient_ID (Foreign Key), Hospital_ID (Foreign Key), Date_of_Admitted.

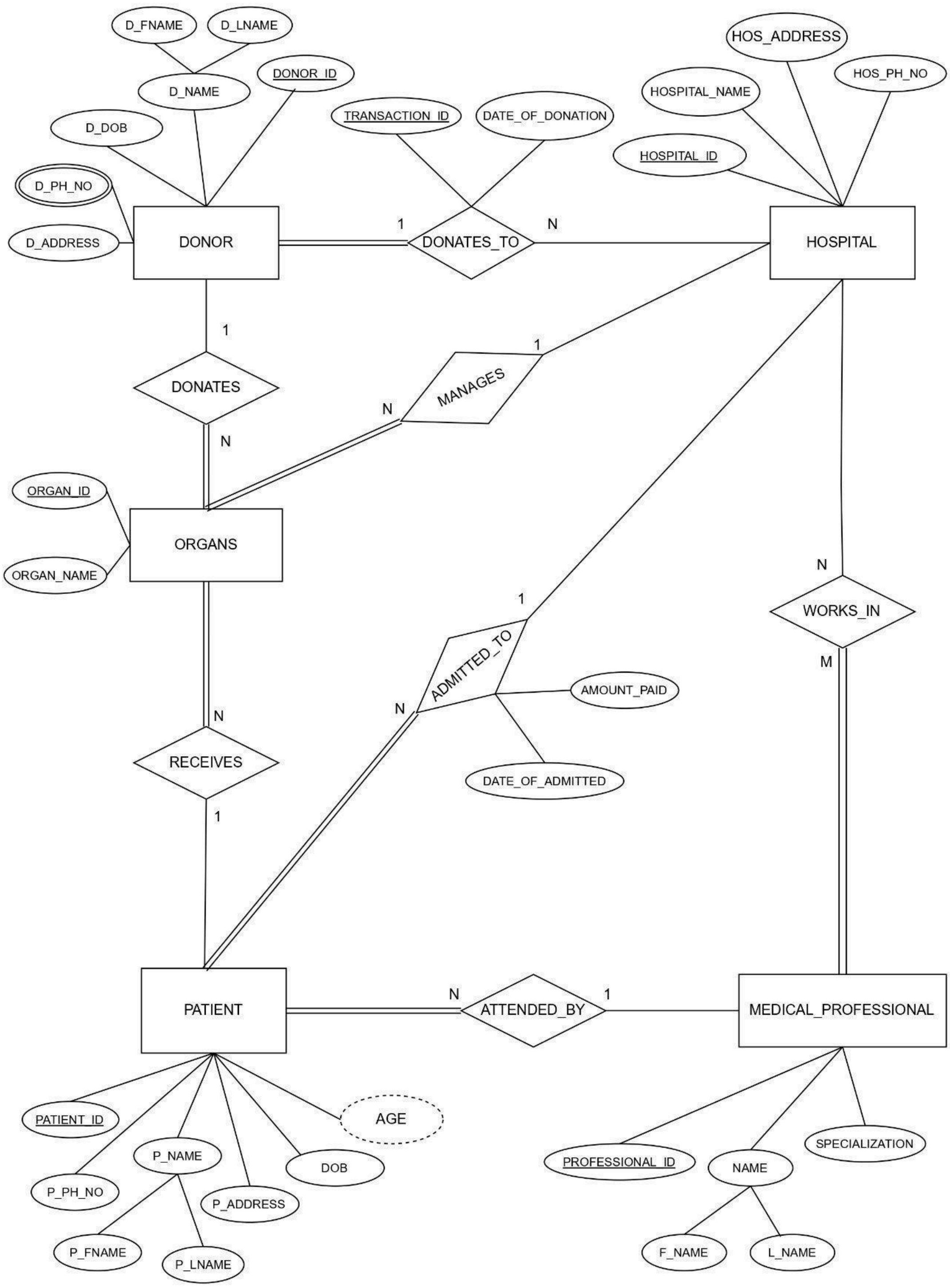

fig 3.1:ER-Diagram of Organ Donation and Procurement Management System

3.3 Mapping of Strong Entities

Each entity is mapped to a relational table with its attributes and primary key.

1. Tables for Strong Entities:

1. **DONOR** (Donor_ID **PK**, D_FNAME, D_LNAME, D_ADDRESS, D_DOB)
2. **PATIENT** (Patient_ID **PK**, P_FNAME, P_LNAME, P_ADDRESS, P_PH_NO, DOB, AGE)
3. **HOSPITAL** (Hospital_ID **PK**, Hospital_Name, Hos_Address, Hos_Ph_No)
4. **MEDICAL_PROFESSIONAL** (Professional_ID **PK**, Name, Specialization)
5. **ORGANS** (Organ_ID **PK**, Organ_Name)

2. Mapping of Weak Entities

There are no weak entities in the ER model, so no additional mapping is required.

3. Mapping of Binary 1:N Relationships

- DONOR AND HOSPITAL: Adding Donor_ID and Hospital_ID as foreign keys in DONATES_TO relation.
  DONATES_TO (Transaction_ID PK, Donor_ID FK, Hospital_ID FK, Date_of_Donation, Amount_Paid)
- DONOR AND ORGANS: Adding Donor_ID as a foreign key to ORGANS relation.
  ORGANS (Organ_ID PK, Organ_Name, Donor_ID FK)
- HOSPITAL AND ORGANS: Adding Hospital_ID as a foreign key to ORGANS relation.
  ORGANS (Organ_ID PK, Organ_Name, Donor_ID FK, Hospital_ID **FK`)
- PATIENT AND ORGANS: Adding Patient_ID as a foreign key to ORGANS relation.
  ORGANS (Organ_ID PK, Organ_Name, Donor_ID FK, Hospital_ID FK, Patient_ID **FK`)
- HOSPITAL AND PATIENT: Adding Hospital_ID and Patient_ID as foreign keys in ADMITTED_TO relation.
  ADMITTED_TO (Admission_ID PK, Hospital_ID FK, Patient_ID FK, Date_of_Admission)

- MEDICAL PROFESSIONAL AND PATIENT: Adding Professional_ID as a foreign key to PATIENT relation.
  PATIENT (Patient_ID PK, P_Fname, P_Lname, P_Address, P_PhNo, P_Dob, P_Age, P_ProfessionalId FK)

4. Mapping of Binary M:N Relationship

- HOSPITAL AND MEDICAL PROFESSIONAL: Creating a new table WORKS_IN to resolve the M:N relationship.
  WORKS_IN (Professional_ID FK, Hospital_ID FK, Composite Primary Key (Professional_ID, Hospital_ID))

5. Mapping of Multivalued Attribute

- DONOR_PHONE: Since D_PH_NO is a multivalued attribute, we create a separate relation DONOR_PHONE.
  DONOR_PHONE (Donor_ID FK, D_PH_NO)

6. Mapping of N-ary Relationships

- No N-ary relationships exist in the current model.

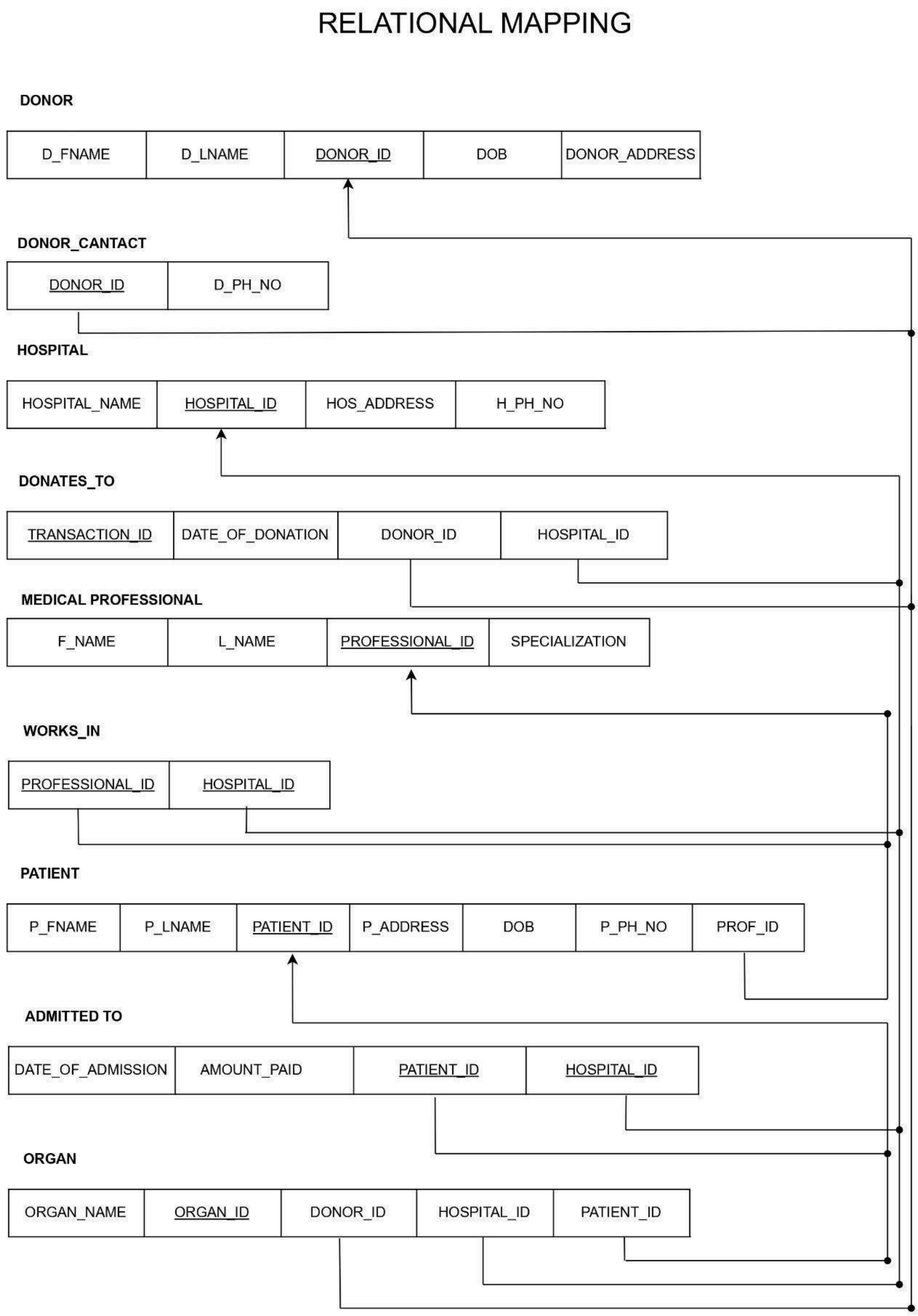

fig 3.2 : Relational Mapping

Chapter 4

Data Flow Diagram (DFD)

A Data Flow Diagram (DFD) is a visual representation that depicts the flow of information within a system, illustrating how data moves between entities, processes, and storage points. It provides a clear and structured understanding of system requirements and functionalities. DFDs can represent both manual and automated processes and serve as a crucial tool in system analysis and design. The primary objective of a DFD is to define the scope, interactions, and boundaries of a system.

DFDs follow a hierarchical structure, typically organized into levels such as Level 0, Level 1, and Level 2, each offering progressively detailed insights into the system components and their interactions.

4.1 Level 0 Data Flow Diagram

The Level 0 DFD, also known as the context diagram, provides an overarching view of the system, encapsulating all its operations within a single process bubble. It illustrates how data enters and exits the system and the interactions between external entities. This high-level representation lays the groundwork for a deeper analysis in subsequent levels.

Context of Level 0 DFD for Organ Donation and Procurement System:

At Level 0, the Organ Donation and Procurement System is represented as a single entity handling the entire process of organ donation, availability, and procurement. The key external entities interacting with the system include:

- Donor: Initiates the donation process through Donor Registration.
- Hospital: Manages organ availability and oversees the admission of patients in need of organs.
- Patient: Submits requests for organ transplants.
- Organ Availability: Represents the database or record-keeping mechanism tracking the availability of organs.

Key Processes in Level 0 DFD:

1. Donor Registration: A donor expresses willingness to donate an organ and undergoes the necessary evaluation.
2. Request Organ: A patient or hospital requests a specific organ for transplantation.
3. Update Organ Availability: The system updates organ availability status when a donation occurs.
4. Admitted/Request Organ: Patients admitted to hospitals may request an organ transplant based on availability.

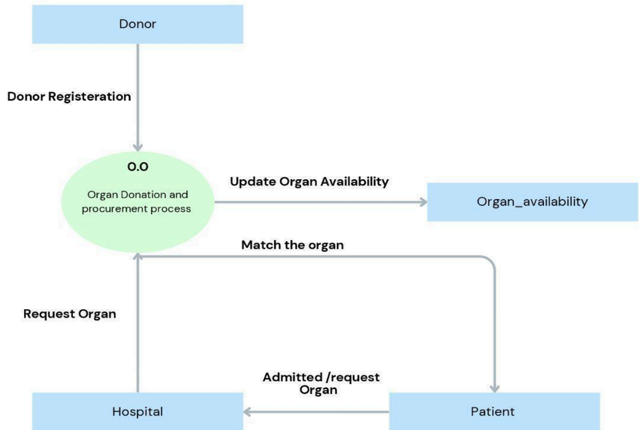

fig 4.1: Level-0 DFD

4.2 Level 1 Data Flow Diagram (DFD):

The Level 1 Data Flow Diagram (DFD) expands upon the Level 0 DFD by breaking down the high-level system processes into more detailed subprocesses. This level provides a more comprehensive view of how data moves between entities, storage units, and processes within the Organ Donation and Procurement System.

Main Processes :

- 0.1 Manages Donors: Collects and updates donor-related data.
- 0.2 Managing Hospital: Stores and updates hospital information.
- 0.3 Manages Patient: Records patient details and organ requests.
- 0.4 Manages Organ Availability: Tracks organ donations and availability.

- 0.5 Manages Doctor: Handles doctor information and storage.

Each of these processes interacts with a dedicated database for efficient information retrieval and management, ensuring a structured, organized, and streamlined organ donation process.

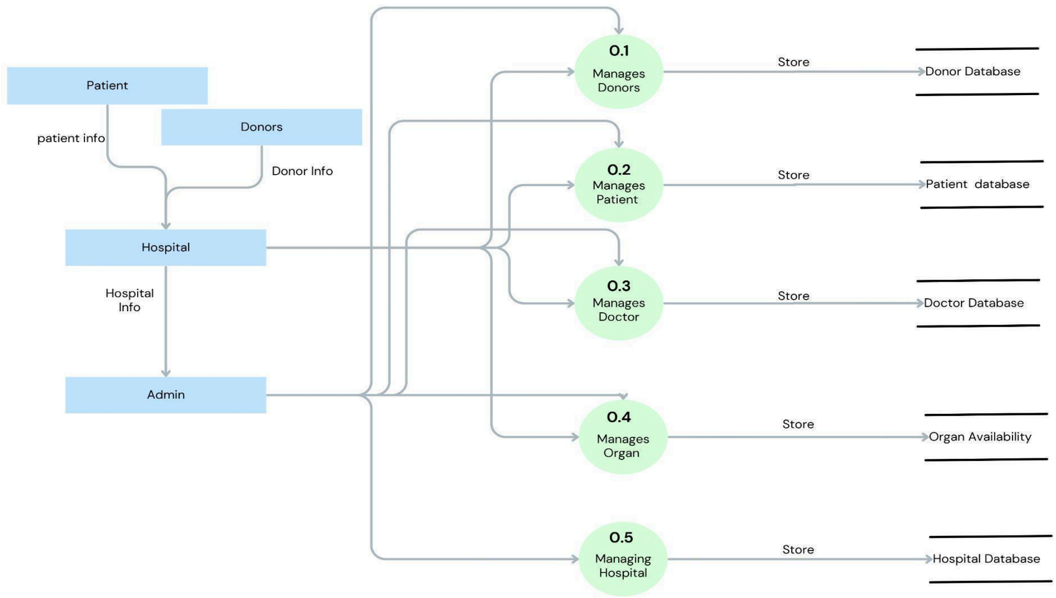

fig 4.2: Level-1 DFD

4.3 Level 2 Data Flow Diagram (DFD) :

A Level 2 Data Flow Diagram (DFD) further decomposes the Level 1 DFD into more granular subprocesses, detailing specific tasks within each module. It provides a deeper view of data interactions, such as how donor registration, organ matching, hospital management, and patient requests are processed within the system. This level enhances clarity in system operations and data flow.

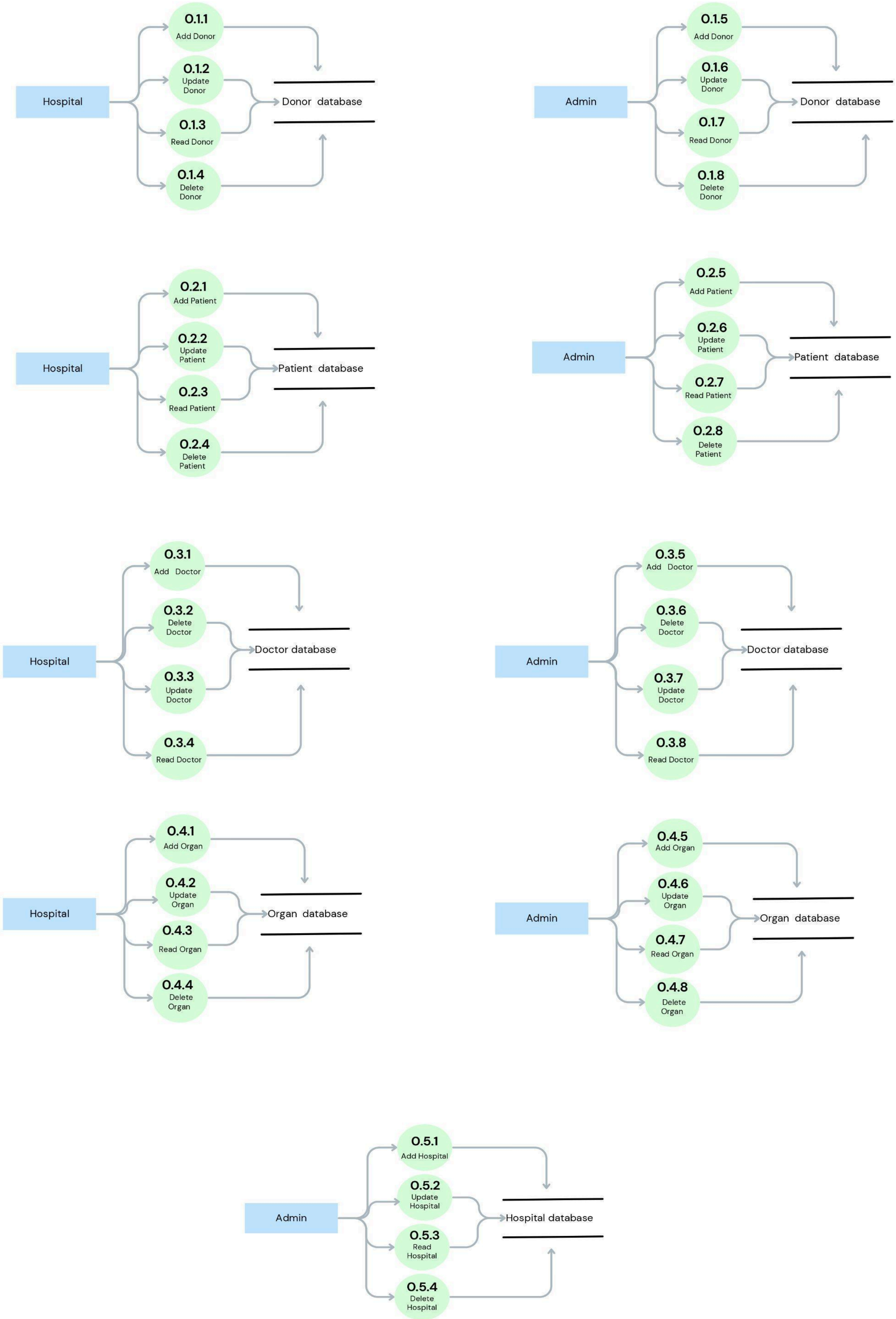

fig 4.3 : Level-2 DFD

Chapter 5

Normalization

Normalization is a process used to eliminate **redundancy** and **inconsistencies** in a relational database by organizing data into multiple related tables. Each **normal form (NF)** addresses specific problems.

5.1 First Normal Form:

A relation to be in 1NF it should satisfies 2 condition:

1. Each column should contain only atomic (indivisible) values
2. there should be no duplicate rows or tuples.

In our schema all the attributes for all the entities are atomic in nature. So it is already in 1NF and need not be reduced.

PATIENT Table

PATIENT_ID	P_FNAME	P_LNAME	P_ADDRESS	DOB	P_PH_NO	PROF_ID
P1001	John	Doe	New York	1985-06-15	9876543210	PR2001
P1002	Alice	Smith	Los Angeles	1992-09-25	8765432109	PR2002

Table 5.1 Sample patient Data

HOSPITAL Table

HOSPITAL_ID	HOSPITAL_NAME	HOS_ADDRESS	H_PH_NO
H1001	City Care Hospital	New York	2123456789
H1002	Green Valley Medical Center	Los Angeles	3109876543

Table 5.2 Sample Hospital Data

5.2 Second Normal Form:

A relation to be in 2NF it should satisfies 2 condition:

1. The table must be in **1NF**.
2. And **all non-key attributes must be fully dependent on the entire primary key**

In our relations There is **no** 2NF violation

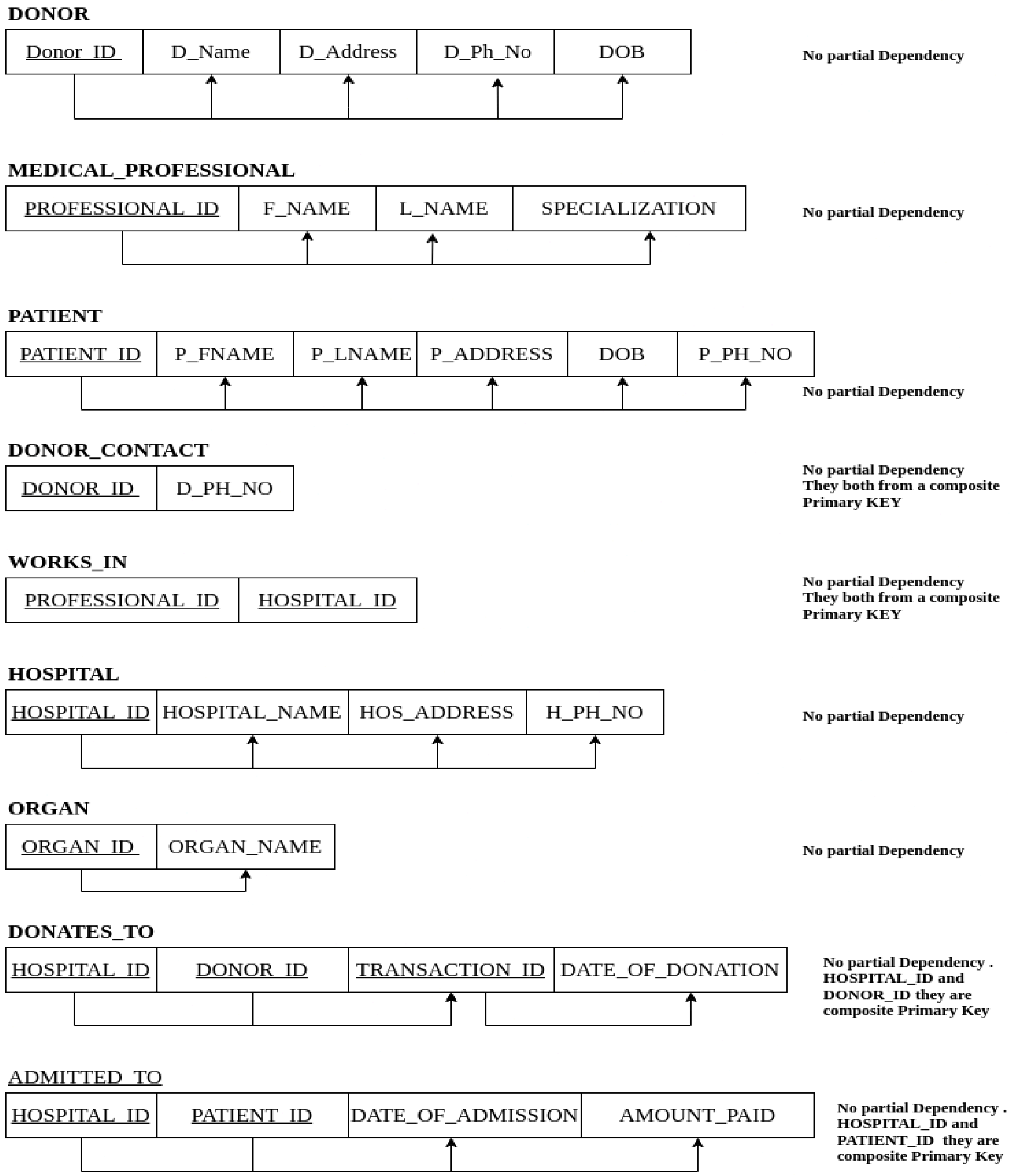

fig 5.1 : 1'st and 2'nd Normal Form

5.3 Third Normal Form:

A relation to be in 3NF it should satisfies 2 condition:

1. The table must be in **2NF**.
2. **No non-key attribute should depend on another non-key attribute.**

In our relations **DONATES_TO** is violating the 3NF form. So We are creating another Relation called TRANSACTION .

- While this decomposition we have also ensured that it's a **lossless decomposition** by taking Super key(TRANSACTION_ID) form R1 as the common attribute.
- And also we have maintained **dependency** Properly.

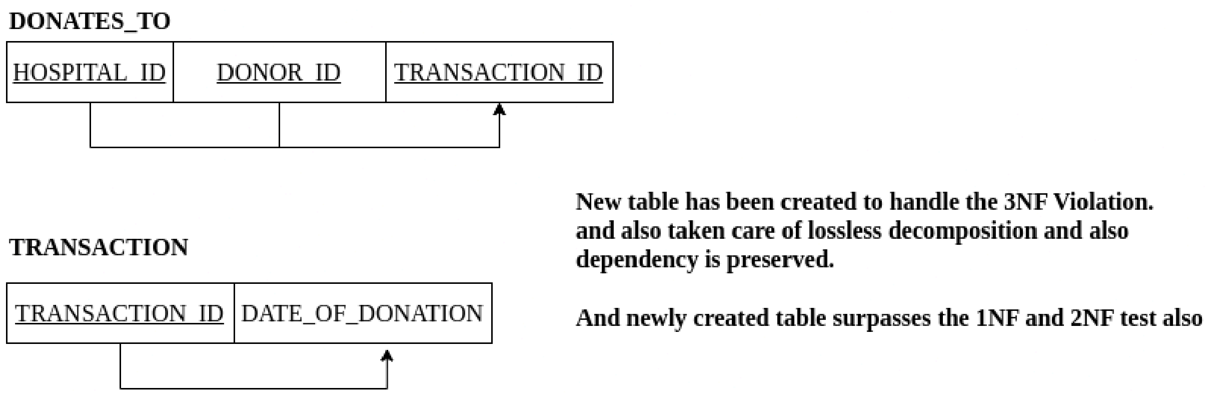

fig 5.2: 2'nd Normal Form

BCNF:

A relation to be in BCNF it should satisfies 2 condition:

1. The table must be in 3**NF**.
2. And **every determinant must be a candidate key**

There is **NO violation** of BCNF in our relations.

Chapter 6

Integration of NoSQL (MongoDB)

The integration of MongoDB as a NoSQL database in our Blood Bank Management System enhances the handling of diverse and unstructured medical records for both patients and donors. Recognizing the limitations of traditional relational databases in managing dynamic medical data, our system adopts a hybrid approach that leverages the strengths of both SQL (MySQL) and NoSQL (MongoDB).

System Architecture

Our system ensures seamless integration between MySQL and MongoDB through a shared unique identifier:

- MySQL: Stores structured data such as personal details of patients and donors.
- MongoDB: Stores dynamic medical records, including diagnoses, treatment history, and reports for both patients and donors.

The primary key, `PATIENT_ID` for patients and `DONOR_ID` for donors, is used to establish a link between the two databases. This approach ensures that each medical record in MongoDB corresponds accurately to its respective entry in MySQL.

Benefits of the Hybrid Approach

- Scalability: MongoDB efficiently handles expanding medical data.
- Flexibility: Supports unstructured and evolving patient and donor records.
- Data Integrity: Maintains a structured relationship through SQL while enabling flexible medical data storage in NoSQL.

This integration fosters a robust and adaptive system, ensuring efficient management of comprehensive donor and patient profiles in the evolving healthcare and blood donation landscape.

Chapter 7

Security and Validation:

1. SQL Injection Prevention

Parameterized queries ensure that user input is treated strictly as data and not executable code. By separating SQL commands from user-supplied data, this prevents attackers from altering query logic, such as injecting malicious SQL commands to bypass authentication or extract sensitive information.

2. Password Hashing

Passwords are securely stored as cryptographic hashes using algorithms like bcrypt or PBKDF2. This makes it computationally infeasible for attackers to reverse-engineer passwords even if the database is compromised.

3. Logging

Both successful and failed login attempts are logged with details such as username and IP address. This enables administrators to monitor for suspicious activity, including brute-force attacks or unauthorized access attempts. These logs serve as an audit trail for forensic analysis and help in implementing proactive security measures.

4. Session Management

Secure session handling ensures that once a user logs in, their authenticated state is maintained securely throughout the session. This prevents issues like session hijacking or fixation. Session variables (e.g., session['login'], session['role']) are used to enforce access controls and verify the user's identity on subsequent requests.

5. Role-Based Access Control (RBAC)

RBAC ensures that users can only access resources and perform actions appropriate to their assigned roles (e.g., admin vs. regular user). By storing role information in the session (e.g., session['isAdmin']), the application enforces fine-grained access control, minimizing the risk of unauthorized access to sensitive functionality or data.

Chapter 8

Conclusion and Future Enhancement

In conclusion, our model represents a significant advancement in organ donation and management systems, offering robust features for efficiently managing donor records, organ availability, and patient requests. Through the integration of MySQL and MongoDB, we have achieved a balanced approach, leveraging the relational integrity of MySQL for structured data while harnessing MongoDB's flexibility and scalability for unstructured and evolving medical records. The system's capabilities extend beyond mere data storage, with real-time data retrieval, efficient record management, and high availability mechanisms ensuring reliable and insightful organ donation processes. Our model effectively demonstrates the synergy between traditional relational databases and modern NoSQL solutions, providing a comprehensive and scalable platform for organ donation management.

Looking towards future enhancements, there are several opportunities for refinement and expansion. One potential improvement lies in enhancing the user experience, incorporating features such as personalized dashboards, mobile optimization, and intuitive data visualization tools. Additionally, while we have not automated the matching of organs to patient requirements, a key enhancement could be the integration of AI-driven compatibility matching to streamline the process. Another major advancement is the inclusion of an AI chatbot, which we have implemented to answer common queries based on the database, making information retrieval more efficient for patients, donors, and healthcare providers. Further improvements in data security, compliance with medical regulations, and encryption techniques will also be prioritized to maintain the integrity and confidentiality of sensitive medical information.

By fostering innovation and adaptability, we aim to continuously improve our organ donation and management system, empowering healthcare providers with cutting-edge tools to ensure the efficient and timely allocation of life-saving organs to those in need.

References

[1] P. S. L. Bhavana and S. Gaddada, "Organ Donation Management and Allocation System," International Refereed Journal of Engineering and Science, vol. 13, no. 2, pp. 267-275, 2024.
https://www.irjes.com/Papers/vol13-issue2/1302267275.pdf

[2] "Online Organ Donation Management System," International Journal of Food and Nutritional Sciences, vol. 13, no. 3, pp. 254-260, 2024.

[3] M. Rychlik, B. Tanriover, and Y. Han, "Large-scale data extraction from the UNOS organ donor documents," arXiv preprint arXiv:2308.15752, 2023.
https://arxiv.org/abs/2308.15752

[4] "Deceased Organ Donor Management and Organ Distribution From Organ Recovery Facilities," Progress in Transplantation, vol. 33, no. 4, pp. 294-301, 2023.

[5] K. Wangsa, S. Karim, E. Gide, and M. Elkhodr, "A Systematic Review and Comprehensive Analysis of Pioneering AI Chatbot Models from Education to Healthcare: ChatGPT, Bard, Llama, Ernie, and Grok," Future Internet, vol. 16, no. 7, p. 219, 2024.
https://www.mdpi.com/1999-5903/16/7/219

Appendix

1. **Login Page**

This login page serves as the entry point for both administrators and hospitals in the Organ Donation and Procurement Management System. It provides secure access to authorized users, ensuring proper management and coordination of organ donation and transplantation.

Admin Access: The administrator has full control over the system, managing hospital registrations, approving organ donation requests, and overseeing the entire procurement process.

Hospital Access: Hospitals receive login credentials from the admin. Using their assigned username and password, they can access the system to register donors, update recipient details, and monitor the availability of organs.

The interface is designed for ease of use, featuring a clean and professional look. The login form ensures secure authentication, allowing only authorized personnel to access critical information related to organ donation and transplantation.

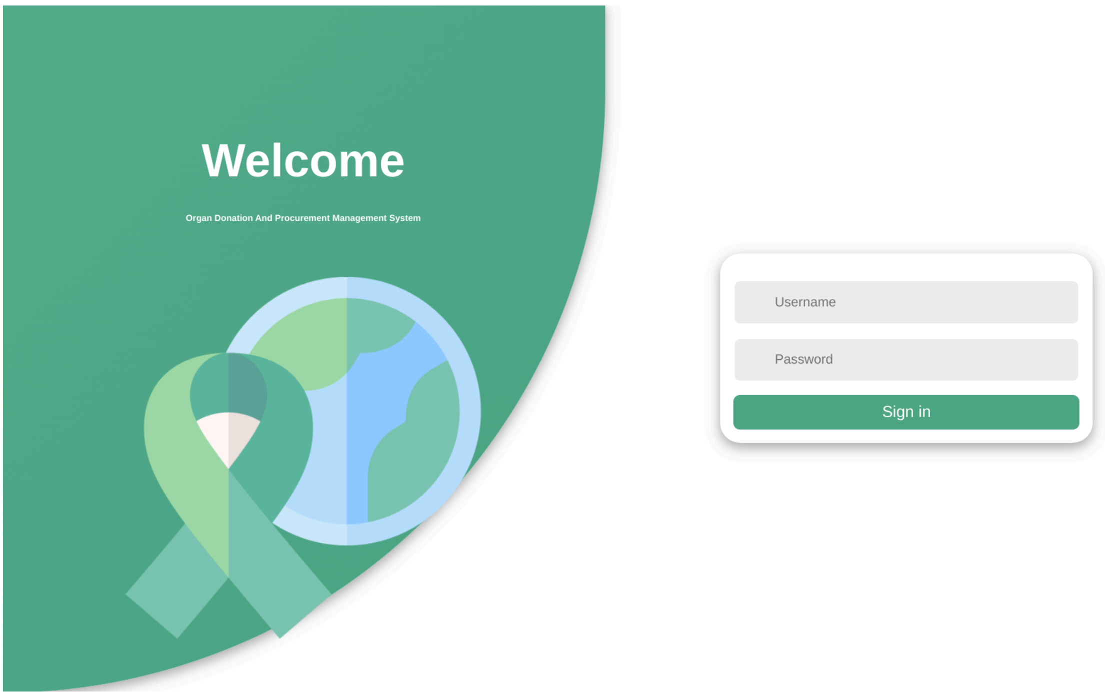

fig A.1: Login page

2. Home Page

This is the main dashboard of the Organ Donation and Procurement Management System

Key Features & Operations:

1. Search Records
2. Add Records
3. Update Records
4. Remove Records
5. User-Specific Pages

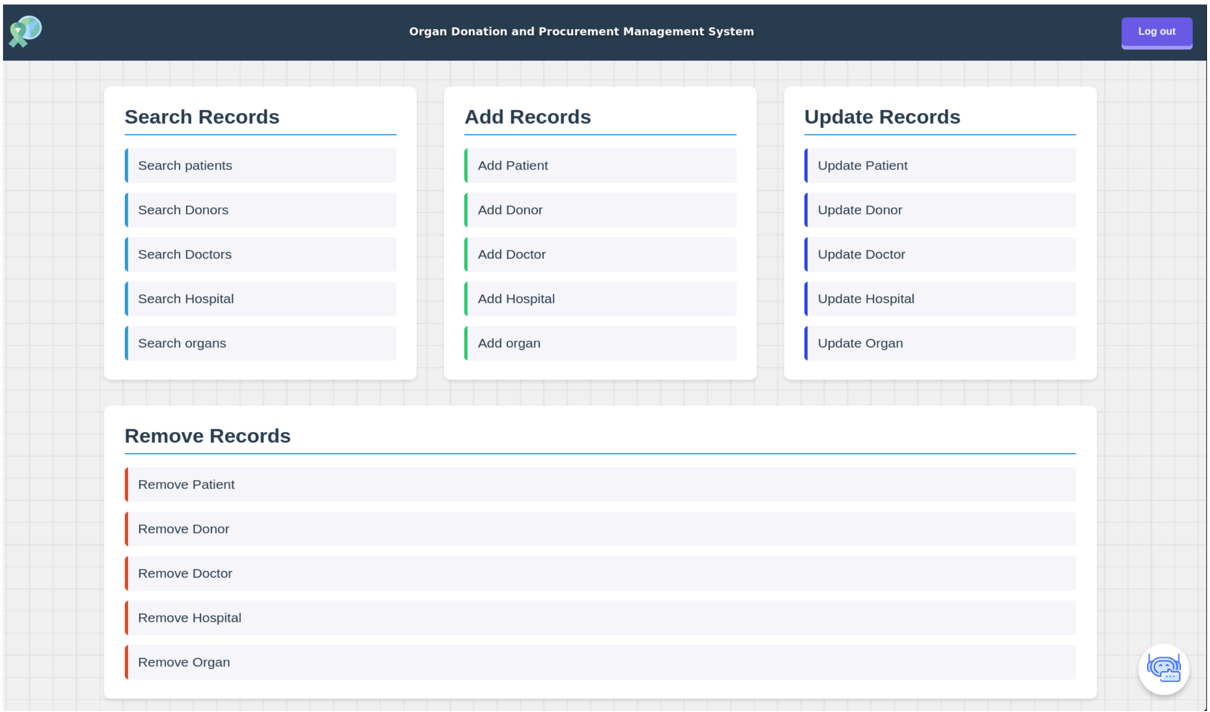

fig A.2: Home page

3. **List of Patients**

This page displays a list of registered patients in a table format, providing essential details such as name, address, date of birth, blood group, phone number, and hospital ID.

Key Features:

- Search Functionality: Users can search for patients by Patient ID using the input field and search button.
- Patient Details: The table includes information such as blood group, hospital ID, and admission details.

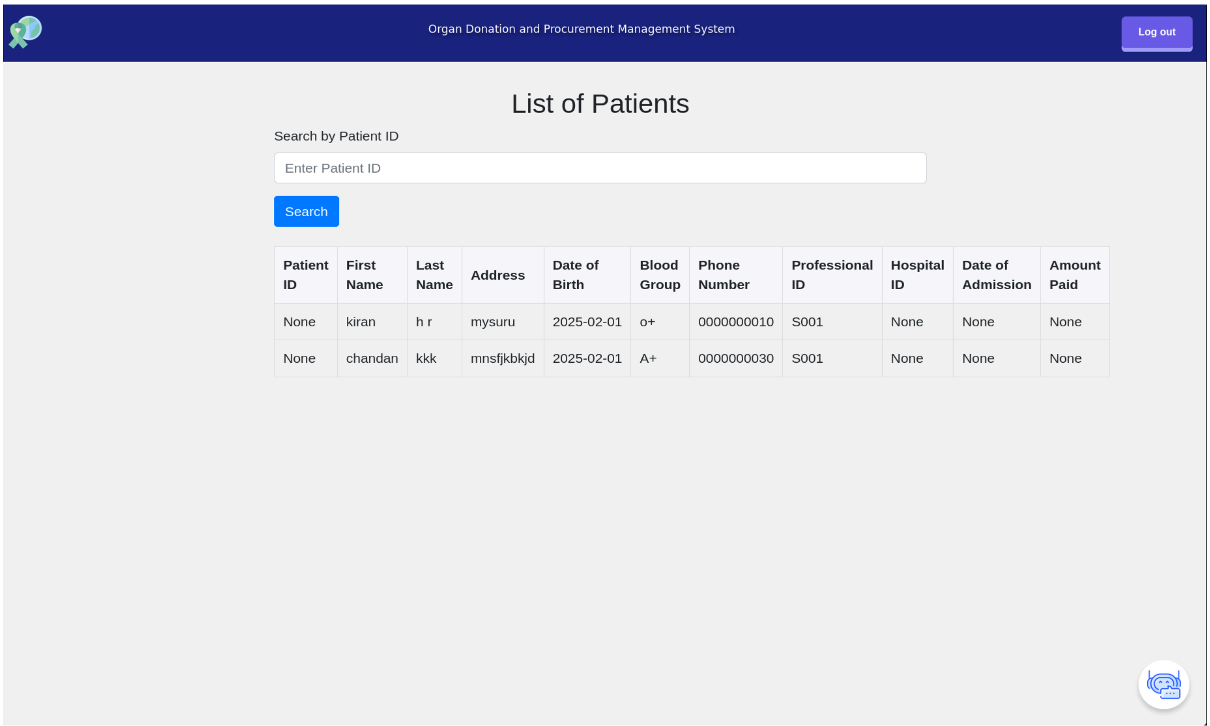

fig 9.3: List Patient Page

4. **Add Patients**

Purpose of the Page:

- It allows hospitals or administrators to register a new patient in the system.
- The PDF upload option suggests that patient-related documents (like medical history or consent forms) can be attached.
- The form likely connects to a database where patient records are stored for organ donation and transplantation tracking.

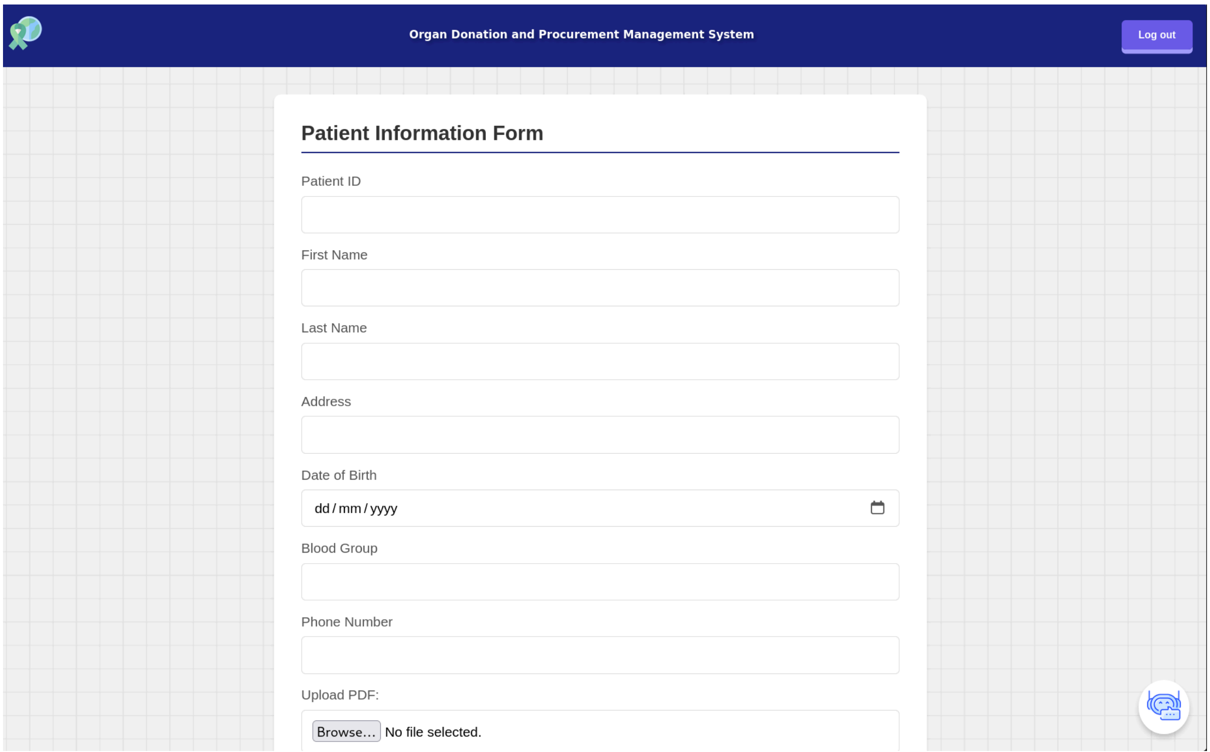

fig 9.4: Add Patient Page

5. **Remove Patients**

The "Remove Patient" page in the Organ Donation and Procurement Management System allows administrators to delete a patient record from the database. The interface consists of a single input field where the user enters the Patient ID and a Submit button to confirm the deletion.

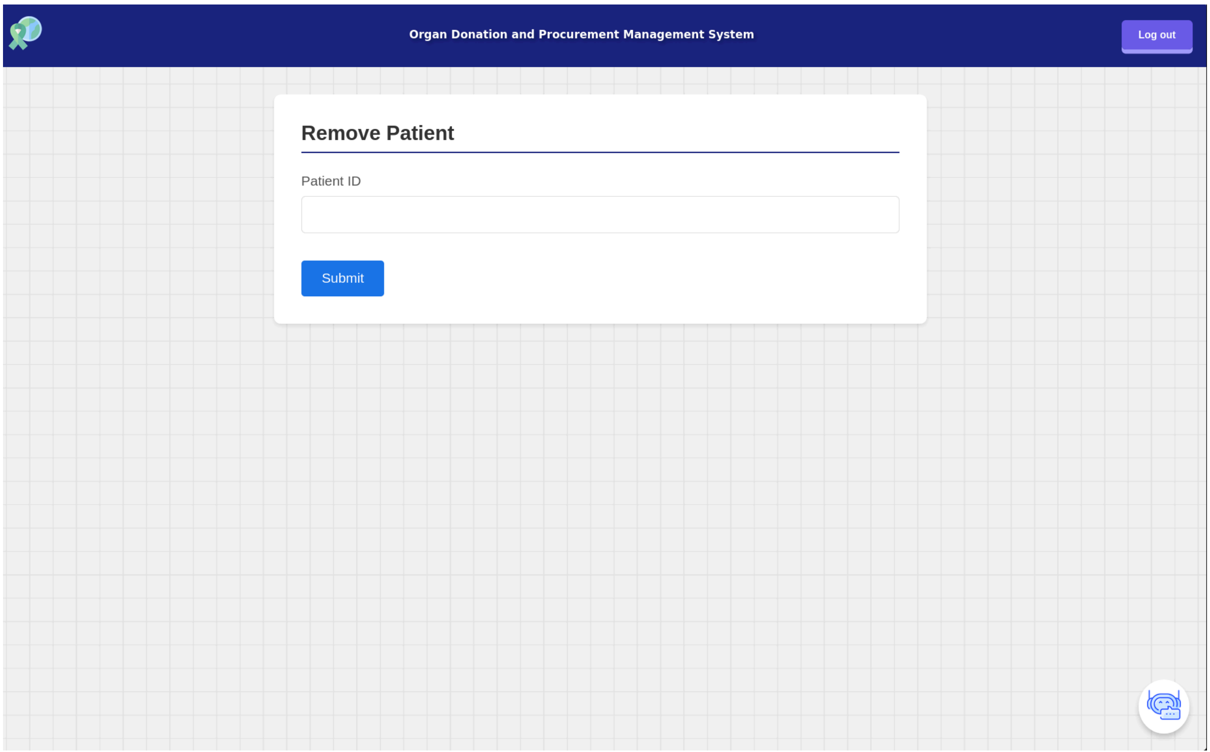

fig 9.5: Removing Patient Page

5. Database Interaction Chatbot

The Database Interaction Chatbot is a user-friendly web application that allows users to interact with a MySQL database through a simple chat interface. By providing host, port, user credentials, and database name, users can establish a connection and send SQL queries in natural language.

This chatbot is designed to assist users in retrieving database records without needing deep SQL knowledge

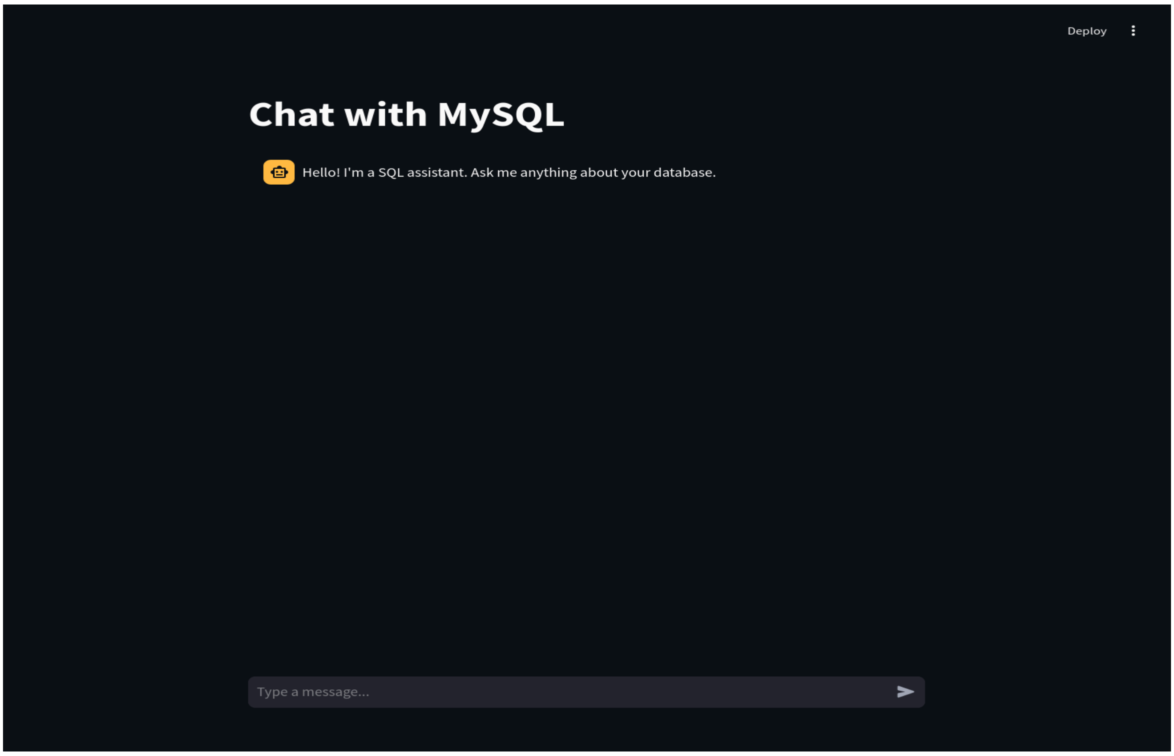

fig 9.6: Chatbot