DBMS Project Synopsis (Go + SQLite)

1. Title Page Project Title: Blood Bank Management System

Course Name & Code: UCS310 – Database Management Systems

Degree & Year: B.Tech (2nd Year)

Department: [To Be Filled]

Institute Name: [To Be Filled]

Group Members: [To Be Filled]

Lab Instructor: [To Be Filled]

Academic Year: 2025–26

1. Introduction The Blood Bank Management System is a database-driven web application designed to manage blood donors, recipients, blood requests, and inventory efficiently. The system stores and manages donor and recipient records, donation logs, request status, and blood stock information.

Traditional manual or file-based systems suffer from data redundancy, inconsistency, and delayed updates. A Database Management System (DBMS) provides structured data storage, integrity constraints, and efficient query processing.

This project emphasizes backend implementation using SQL and relational database concepts with SQLite as the database.

1. Problem Statement In many blood banks, donor records and blood requests are handled manually or using spreadsheets. This leads to:

• Duplicate or outdated donor/recipient records • Inaccurate inventory and expiry tracking • Difficulty in matching requests to available stock • Slow data retrieval and reporting • Lack of data integrity

The proposed Blood Bank Management System provides a structured relational database solution to manage donors, donations, requests, and inventory efficiently.

1. Objectives of the Project • To design a relational database for blood bank management • To implement primary and foreign key constraints • To normalize the database up to Third Normal Form (3NF) • To implement DDL and DML SQL commands • To ensure referential integrity and consistent inventory updates • To manage donation and request workflows effectively

2. Scope of the Project Users:

• Admin/Staff

Modules:

• Donor Registration Module • Recipient Registration Module • Donation Recording Module • Blood Request Module • Inventory Tracking Module

The system focuses primarily on backend database operations.

1. Proposed System Description The system is developed using:

• Go (Golang) for backend logic • SQLite for database • HTML/CSS for frontend

Working of the System:

1. Admin/staff registers donors and recipients.
2. Donations are recorded and linked to donors.
3. Blood requests are created and tracked by status.
4. Inventory is updated based on donations and fulfilled requests.
5. Staff can view and manage all records through the web interface.

The system improves efficiency, data consistency, and traceability of blood units.

1. Database Design 7.1 Entities Identified

2. Donor

3. Donation
4. Recipient
5. Request
6. Inventory

7.2 ER Diagram

Entities and Relationships:

• Donor submits Donation (1-M) • Recipient submits Request (1-M) • Inventory summarizes stock by blood type

7.3 Relational Schema

DONORS id (PK), name, blood_type, phone, city, created_at, deleted_at

DONATIONS id (PK), donor_id (FK), blood_type, units, donation_date, expiry_date, deleted_at

RECIPIENTS id (PK), name, blood_type, phone, hospital, created_at, deleted_at

REQUESTS id (PK), recipient_id (FK), blood_type, units, status, request_date, deleted_at

INVENTORY id (PK), blood_type (UNIQUE), units, deleted_at

1. Normalization Functional Dependencies

Donors: id → name, blood_type, phone, city, created_at

Donations: id → donor_id, blood_type, units, donation_date, expiry_date

Recipients: id → name, blood_type, phone, hospital, created_at

Requests: id → recipient_id, blood_type, units, status, request_date

Inventory: blood_type → units

Third Normal Form (3NF)

• No transitive dependency. • All non-key attributes depend only on the primary key.

Therefore, the database is normalized up to 3NF.

1. Database Implementation 9.1 SQL Implementation

DDL and DML Commands Used

UPDATE Query

9.2 Backend Logic Implementation

Since SQLite does not support PL/SQL, application logic is implemented using Go programming language.

• Donor/recipient creation • Donation insertion and inventory update • Request creation, status update, and fulfillment • Database connectivity

Database connection is handled in main.go.

1. Tools & Technologies Used • SQLite Database • SQL • Go (Golang) • HTML • CSS

2. Expected Outcomes • Properly structured and normalized database • Efficient blood inventory management • Fast retrieval of donor and request data • Improved data consistency and integrity • Reduced manual errors