

A Course Based Project Report on  
**BLOOD BANK MANAGEMENT SYSTEM**

Submitted to the

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING- (CyS, DS) AND AI&DS

in partial fulfilment of the requirements for the completion of course  
JAVA PROGRAMMING LABORATORY(A19PC2IT03)

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)**

Submitted by

K. NANDINI	21071A67A5
P. RAGHAVENDRA	21071A67B1
P. LOKESH	21071A67B2
B. SANDEEP	22075A6708

Under the guidance of

**Dr. N. Sunanda**

**(Course Instructor)**

Assistant Professor

Department of CSE- (CyS, DS) and AI&DS, VNRVJIET



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING- (CyS, DS) AND AI&DS  
**VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI**  
**INSTITUTE OF ENGINEERING & TECHNOLOGY**

An Autonomous Institute, NAAC Accredited with 'A++' Grade, NBA

Vignana Jyothi Nagar, Pragathi Nagar, Nizampet (S.O), Hyderabad – 500 090, TS,  
India

**AUGUST 2023**

**VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI  
INSTITUTE OF ENGINEERING AND TECHNOLOGY**

An Autonomous Institute, NAAC Accredited with 'A++' Grade, NBA Accredited for CE, EEE, ME, ECE, CSE, EIE, IT B. Tech Courses, Approved by AICTE, New Delhi, Affiliated to JNTUH, Recognized as "College with Potential for Excellence" by UGC, ISO 9001:2015 Certified, QS I GUAGE Diamond Rated Vignana Jyothi Nagar, Pragathi Nagar, Nizampet(SO), Hyderabad-500090, TS, India

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING- (CyS, DS) AND AI&DS**



**CERTIFICATE**

This is to certify that the project report entitled "**Blood Bank Management System**" is a bonafide work done under our supervision and is being submitted by team **Miss. Nandini(21071A67A5), Mr. Raghavendra(21071A67B1), Mr. Lokesh (21071A67B2), Mr. Sandeep (22075A6708)** in partial fulfilment for the award of the degree of **Bachelor of Technology** in Computer Science and Engineering Data Science, of the VNRVJIET, Hyderabad during the academic year 2022-2023.

  
Dr. N. Sumanda

Assistant Professor

Department of CSE- (CyS, DS) and AI&DS

VNRVJIET

  
Dr. M. Raja Sekar

Associate Professor & HOD

Department of CSE- (CyS, DS) and AI&DS

VNRVJIET

**Course based Projects Reviewer**

**VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI  
INSTITUTE OF ENGINEERING AND TECHNOLOGY**

An Autonomous Institute, NAAC Accredited with 'A++' Grade,  
Vignana Jyothi Nagar, Pragathi Nagar, Nizampet(SO), Hyderabad-500090, TS, India

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING- (CyS, DS) AND AI&DS**



**DECLARATION**

We declare that the course based project work entitled “**Blood Bank Management System**” submitted in the Department of Information Technology, Vallurupalli Nageswara Rao Vignana Jyothi Institute of Engineering and Technology, Hyderabad, in partial fulfilment of the requirement for the award of the degree of **Bachelor of Technology in Computer Science and Engineering Data Science** is a bonafide record of our own work carried out under the supervision of **Dr. N. Sunanda, Assistant Professor, Department of CSE- (CYS, DS) and AI&DS, VNRVJIET**. Also, we declare that the matter embodied in this thesis has not been submitted by us in full or in any part thereof for the award of any degree/diploma of any other institution or university previously.

Place: Hyderabad.

  
**K. Nandini**

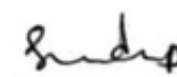
(21071A67A5)

  
**P. Raghavendra**

(21071A67B1)

  
**P. Lokesh**

(21071A67B2)

  
**B. Sandeep**

(22075A6708)

## ACKNOWLEDGEMENT

We express our deep sense of gratitude to our beloved President, Sri. D. Suresh Babu, VNR Vignana Jyothi Institute of Engineering & Technology for the valuable guidance and for permitting us to carry out this project.

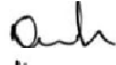

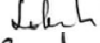

With immense pleasure, we record our deep sense of gratitude to our beloved Principal, Dr. C.D Naidu, for permitting us to carry out this project.

We express our deep sense of gratitude to our beloved Professor Dr. M. RAJA SEKAR, Professor and Head, Department of Computer Science & Engineering- (CyS, DS) and AI&DS, VNR Vignana Jyothi Institute of Engineering & Technology, Hyderabad- 500090 for the valuable guidance and suggestions, keen interest and through encouragement extended throughout the period of project work.

We take immense pleasure to express our deep sense of gratitude to our beloved Guide, **Dr. N. Sunanda**, Assistant Professor in CSE- (CyS, DS) and AI&DS, VNR Vignana Jyothi Institute of Engineering & Technology, Hyderabad, for his/her valuable suggestions and rare insights, for constant source of encouragement and inspiration throughout my project work.

We express our thanks to all those who contributed for the successful completion of our project work.

K. NANDINI  
P. RAGHAVENDRA  
P. LOKESH  
B.SANDEEP

21071A67A5   
21071A67B1   
21071A67B2   
22075A6708 

## TABLE OF CONTENTS

Details of Contents	Page. No.
Aim	06
Problem Statement	06
Chapter-1	
1.0 Introduction	07
1.1 Purpose	07
1.2 Scope	08
1.3 Acronyms	08
Chapter-2	
2.0 Overall Description	09
2.1 Software Interfaces	09
2.2 Hardware Interfaces	10
Chapter-3	
3.0 Functional Requirements	11
3.1 Non-Functional Requirements	11
3.2 Schema	12-13
3.3 Data	14-16
Chapter-4	
4.0 Use-Case Diagram	17
4.1 E-R Diagram	18
Chapter-5	
5.0 DDL and DML Queries	19-21
5.1 Queries	22-24
5.2 Implementation	25-30
Chapter-6	
6.0 Conclusion	31

## AIM

The aim of a Blood Bank Management System is to provide an efficient and organized platform for managing the collection, storage, testing, and distribution of blood and its components. This digital system streamlines the processes associated with blood donation, inventory management, donor and recipient information, and helps ensure the availability of safe and compatible blood products when needed.

## PROBLEM STATEMENT

The problem statement of a Blood Bank Management System revolves around addressing the challenges faced by blood banks and healthcare institutions in managing their blood inventory and donation processes. Key issues include:

- Scarcity of rare blood group
- Unavailability of blood during emergency.
- Less awareness among people about blood donation and blood transfusion.
- Deaths due to lack of blood during operations.
- **Inventory Management:** Manual tracking of blood donations, expiration dates, and blood types can lead to inefficiencies and errors in maintaining an adequate supply of blood.
- **Donor and Recipient Information:** Keeping accurate records of donor information, medical history, and recipient needs is crucial for safe and successful blood transfusions.
- **Blood Type Matching:** Ensuring that the right blood type is available for recipients and that cross-matching is accurate is vital to prevent adverse reactions.
- **Communication and Coordination:** Coordinating with hospitals, donors, and recipients for timely blood transfusions can be challenging without an organized system.
- **Reporting and Analytics:** Lack of proper reporting and analytics can hinder decision-making regarding blood supply and demand.

The Blood Bank Management System aims to address these challenges by providing a digital solution that optimizes blood donation processes, ensures accurate donor and recipient records, supports efficient inventory management, and facilitates effective communication and coordination.

# **CHAPTER-1**

## **1.0 INTRODUCTION**

A Blood Bank Management System is a digital platform that revolutionizes the way blood banks operate by providing a comprehensive solution for managing blood donation and distribution processes. It leverages technology to ensure the availability of safe blood products, streamline donor interactions, and enhance communication with healthcare institutions.

The population of the world is multiplying with each coming year and so are the diseases and health issues. With an increase in the population there is an increase in the need of blood. The growing population of the world results in a lot of potential blood donors. But in spite of this not more than 10% of the total world population participates in blood donation. With the growing population and the advancement in medical science the demand for blood has also increased. Due to the lack of communication between the blood donors and the blood recipients, most of the patients in need of blood do not get blood on time and hence lose their lives. There is a dire need of synchronization between the blood donors and hospitals and the blood banks. This improper management of blood leads to wastage of the available blood inventory. Improper communication and synchronization between the blood banks and hospitals leads to wastage of the blood available. These problems can be dealt with by automating the existing manual blood bank management system. A high-end, efficient, highly available and scalable system has to be developed to bridge the gap between the donors and the recipients and to reduce the efforts required to search for blood donors.

## **1.1 PURPOSE**

The main purpose of the project is to develop a web application for blood banks to manage information about their donors and blood stock. The main objectives of this website development can be defined as follows:

1. To develop a system that provides functions to support donors to view and manage their information conveniently.
2. To maintain records of blood donors, blood donation information and blood stocks in a centralized database system.
3. To inform donors of their blood result after their donation.
4. To support searching, matching and requesting for blood convenient for administrators.
5. To provide a function to send an e-mail directly to the donor for their user account and the hospital, the availability of the blood bag.

## 1.3 SCOPE

This system makes conveniently available good quality, safe blood and other blood components, which can be provided in a sound, ethical and acceptable manner, consistent with the long-term well-being of the community. It actively encourages voluntary blood donation, motivates and maintains a well-indexed record of blood donors and educates the community on the benefits of blood donation. This will also serve as the site for interaction of best practices in reducing unnecessary utilization of blood and help the state work more efficiently towards self-sufficiency in blood. The system will provide the user the option to look at the details of the existing Donor List, Blood Group and to add a new Donor. It also allows the user to modify the record. The administrator can alter all the system data. The system is able to generate a report to summarize all records including blood donation, blood requests and blood stock for the administrator.

## 1.4 ACRONYMS

**BBMS:** Blood Bank Management System (itself an acronym)

**ABO:** Blood group system (A, B, AB, O)

**CRM:** Customer Relationship Management

**EMR:** Electronic Medical Record

**API:** Application Programming Interface

**SMS:** Short Message Service

**GDPR:** General Data Protection Regulation

**HIPAA:** Health Insurance Portability and Accountability Act

These acronyms represent the various components and concepts within the Blood Bank Management System, contributing to its functionality and efficiency.



## CHAPTER-2

### 2.1 OVERALL DESCRIPTION

A Blood Bank Management System is a digital platform designed to streamline and optimize the various processes associated with blood donation, storage, testing, and distribution within blood banks and healthcare institutions. Through internet access and mobile applications, this system offers a user-friendly interface that simplifies blood management tasks, enhances communication, and ensures the availability of safe and compatible blood products.

The BBMS allows us to keep track of quality of blood and also keeps track of available blood when requested by the acceptor. The existing systems are Manual systems which are time consuming and not so effective. BBMS automates the distribution of blood. This database consists of thousands of records of each blood bank.

By using this system searching the available blood becomes easy and saves lot of time than the manual system. It will hoard, operate, recover and analyze information concerned with the administrative and inventory management within a blood bank. This system is developed in a manner that it is manageable, time effective, cost effective, flexible and much man power is not required. This System revolutionizes the blood banking process by digitizing and optimizing its various components. This digital platform empowers stakeholders with tools to ensure blood safety, efficient operations, and improved patient care, marking a significant advancement in the healthcare sector.

### 2.2 SOFTWARE INTERFACES

The software interfaces of a Blood Bank Management System encompass crucial components that facilitate efficient blood management. These interfaces ensure seamless interactions and effective management for various stakeholders involved.

**Database Interface:** The BBMS software needs to interact with a database management system (DBMS) to store and retrieve data related to donors, recipients, blood units, inventory, and transactions.

**APIs (Application Programming Interfaces):**

- **External Hospital System APIs:** If the BBMS needs to integrate with hospital systems for patient data or blood requests, APIs would facilitate this data exchange.
- **SMS/Email Gateway APIs:** To send notifications and alerts to donors, recipients, and staff members via SMS or email.
- **Authentication and Authorization Interfaces:** Interfaces for integrating authentication services, such as LDAP or OAuth, for secure user access and role-based permissions.
- **Reporting Interfaces:** Integration with reporting tools or libraries to generate various types of reports for donors, recipients, inventory, and other data.
- **Notification Interfaces:** Integration with email, SMS, or push notification services for sending reminders, alerts, and updates to users.

## 2.3 HARDWARE INTERFACES

The hardware interfaces of a Blood Bank Management System cater to the devices used by various stakeholders involved.

- **Donor Devices:** Donors primarily use personal devices such as smartphones or computers to access the donor interface for registration and appointment scheduling.
- **Blood Bank Devices:** Blood bank staff use computers or tablets with internet connectivity to access the inventory management, records, and communication interfaces.
- **Medical Devices:** Blood testing and processing equipment may be integrated with the system for real-time tracking of blood units during these stages.
- **Delivery Partner Devices:** If external services are involved, delivery partners utilize smartphones or specialized devices with GPS capabilities to access delivery information.
- **Healthcare Provider Devices:** Hospital staff use computers or tablets to interact with the communication interface, accessing real-time blood availability and expected delivery times.

Overall, the hardware interfaces of the BBMS are designed to accommodate the devices used by donors, blood banks, healthcare providers, and delivery partners, ensuring smooth communication, accurate records, and safe blood transfusion processes.

## CHAPTER-3

### 3.0 FUNCTIONAL REQUIREMENTS

#### **Donor facility:**

1. A donor can register himself as a donor
2. Adding his ability of donating the blood
3. Can see the request for blood
4. Change personal, contact details by the donors himself.

#### **User facility:**

1. Can see all the donors and blood bank
2. Can filter donor and blood bank according to the blood group and city
3. Sending the request for the blood to donor and blood bank
4. Send blood donation details to the relevant donors.

#### **Admin facilities:**

1. Can see donors
2. Update, delete and verify donor

### 3.1 NON-FUNCTIONAL REQUIREMENTS

- **Performance:** The system should be able to handle a high volume of concurrent users during peak times (e.g., blood donation events) without significant performance degradation.
- **Reliability:** The system should have a high level of availability, aiming for minimal downtime for essential operations.
- **Usability:** The user interface should be intuitive and user-friendly, requiring minimal training for users to navigate and perform tasks.
- **Compatibility:** The system should be compatible with various web browsers, devices (desktops, tablets, smartphones), and operating systems to ensure broad user accessibility.
- **Backup and Recovery:** Regular data backups should be performed to prevent data loss in case of system failures.
- **Security:** The system should ensure the privacy and confidentiality of donor and recipient information, adhering to relevant data protection regulations

### 3.2 SCHEMA

**Patient Table -**

Column name	Datatype	width	Constraint
id	INT	10	Primary Key
name	VARCHAR	50	NOT NULL
blood_type	VARCHAR	3	NOT NULL
last_transfusion_date	DATE	7	NOT NULL

**Blood\_bank Table-**

Col Name	datatype	Width	Constraint
blood_bank_id	INT	10	Primary Key
name	VARCHAR	50	NOT NULL
address	VARCHAR	100	NOT NULL

**Donor Table –**

<b>Column Name</b>	<b>Datatype</b>	<b>Width</b>	<b>Constraint</b>
id	INT	10	Primary Key
name	VARCHAR	50	NOT NULL
age	INT	10	NOT NULL
blood_type	VARCHAR	3	NOT NULL
phone_number	VARCHAR	15	NOT NULL
is_eligible	BOOLEAN	1	NOT NULL
donation_count	INT	10	NOT NULL

### Donation Table

Col Name	Datatype	Width	Constraint
id	INT	10	Primary Key
donor_id	INT	10	Foreign key
blood_bank_id	INT	10	Foreign key
date	DATE	7	NOT NULL

### Transfusion Table

Col Name	Datatype	Width	Constraint
id	INT	10	Primary Key
patient_id	INT	10	Foreign key
donation_id	INT	10	Foreign key
date	DATE	7	NOT NULL

### 3.3 DATA

**Patient Table:**

id	name	blood_type	last_transfusion_date
1	Alice	A+	2022-12-01
2	Bob	B+	2023-01-15
3	Charlie	AB-	2022-11-23
4	Dave	O-	2023-02-18
5	Emily	A-	2022-12-30

**Blood bank TABLE :**

Blood_bank_id	name	address
1	Blood Bank A	123 Main St, Anytown
2	Blood Bank B	456 Maple Ave, Otherville
3	Blood Bank C	789 Oak Rd, Smalltown

**Donor TABLE :**

id	name	age	blood_type	is_eligible	donation_count	phone_number
1	Frank	23	A+	1	3	555-1234
2	Gina	30	O+	1	8	555-5678
3	Henry	42	B+	1	2	555-1111
4	Irene	29	AB-	0	5	555-2222
5	Jack	37	O-	1	6	555-3333

**Donation TABLE :**

id	donor_id	blood_bank_id	date
1	1	1	2022-11-01
2	2	1	2023-01-01
3	3	2	2022-12-01'
4	4	3	2023-02-01
5	5	2	2022-10-01
1	1	3	2022-11-15
2	2	3	2023-01-15
3	3	1	2022-12-15
4	4	2	2023-02-15
5	5	1	2022-10-15

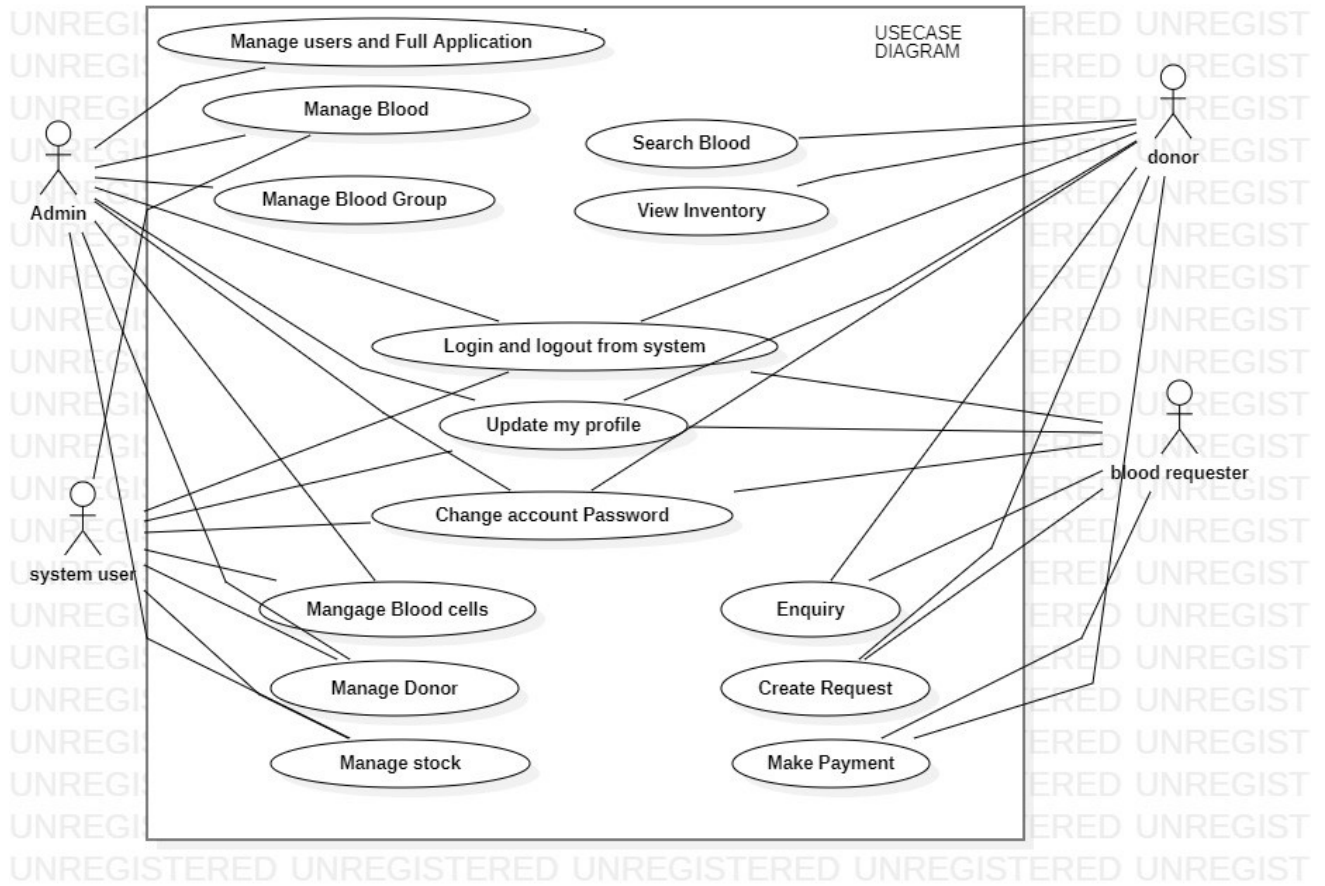
**Transfusion Table:**

id	patient_id	donation_id	transfusion_date
1	1	2	2023-01-16
2	2	1	2022-11-16
3	3	3	2022-12-16
4	4	4	2023-02-16
5	5	5	2022-10-16

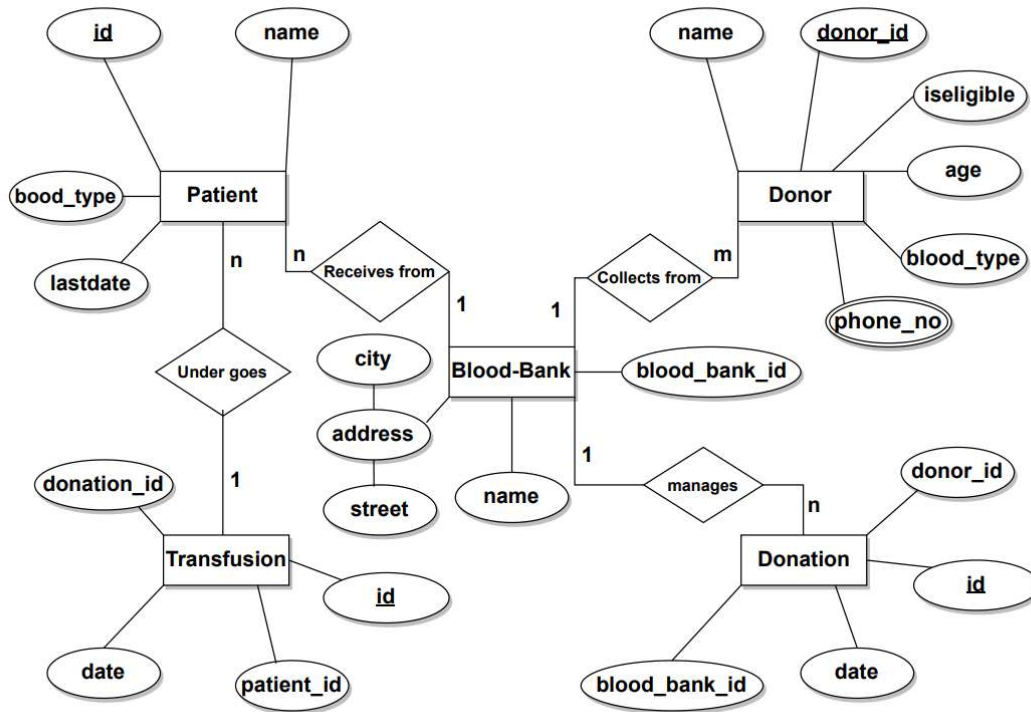


## CHAPTER-4

### 4.0 USECASE DIAGRAM



## 4.1 E-R DIAGRAM



## CHAPTER-5

### 5.0 DDL AND DML QUERIES

#### Patients Table:

DDL-

```
1 CREATE TABLE patient (  
2     id INT PRIMARY KEY,  
3     name VARCHAR(50),  
4     age INT,  
5     blood_type VARCHAR(5),  
6     last_transfusion_date DATE  
7 );  
-
```

Table created.

DML-

```
-- Insert sample data into patient table  
INSERT INTO patient (name, blood_type, last_transfusion_date) VALUES  
(  
'Alice', 'A+', '2022-12-01'),  
(  
'Bob', 'B+', '2023-01-15'),  
(  
'Charlie', 'AB-', '2022-11-23'),  
(  
'Dave', 'O-', '2023-02-18'),  
(  
'Emily', 'A-', '2022-12-30');  
  
row created.
```

---

#### Donor Table:

DDL-

```
CREATE TABLE donor (  
    id INT PRIMARY KEY,  
    name VARCHAR(50),  
    age INT,  
    blood_type VARCHAR(5),  
    is_eligible BOOLEAN,  
    donation_count INT,  
    phone_number VARCHAR(15)  
);
```

Table created.

DML-

```
-- Insert sample data into donor table  
INSERT INTO donor (name, age, blood_type, is_eligible, phone_number, donation_count) VALUES  
(  
'Frank', 23, 'A+', true, '555-1234', 3),  
(  
'Gina', 30, 'O+', true, '555-5678', 8),  
(  
'Henry', 42, 'B+', true, '555-1111', 2),  
(  
'Irene', 29, 'AB-', false, '555-2222', 5),  
(  
'Jack', 37, 'O-', true, '555-3333', 6);  
  
row created.
```

## Blood bank Table-

DDL-

```
CREATE TABLE blood_bank (  
    id INT PRIMARY KEY,  
    name VARCHAR(50),  
    address VARCHAR(100)  
);
```

Table created.

DML-

```
INSERT INTO blood_bank (name, address) VALUES  
( 'Blood Bank A', '123 Main St, Anytown'),  
( 'Blood Bank B', '456 Maple Ave, Otherville'),  
( 'Blood Bank C', '789 Oak Rd, Smalltown');
```

row created.

## Donations Table-

DDL-

```
CREATE TABLE donation (  
    id INT PRIMARY KEY,  
    donor_id INT,  
    blood_bank_id INT,  
    date DATE,  
    FOREIGN KEY (donor_id) REFERENCES donor(id),  
    FOREIGN KEY (blood_bank_id) REFERENCES blood_bank(id)  
);
```

Table created.

DML-

```
INSERT INTO donation (donor_id, blood_bank_id, date) VALUES  
(1, 1, '2022-11-01'),  
(2, 1, '2023-01-01'),  
(3, 2, '2022-12-01'),  
(4, 3, '2023-02-01'),  
(5, 2, '2022-10-01'),  
(1, 3, '2022-11-15'),  
(2, 3, '2023-01-15'),  
(3, 1, '2022-12-15'),  
(4, 2, '2023-02-15'),  
(5, 1, '2022-10-15');
```

row created.

## Transfusion Table-

DDL-

```
CREATE TABLE transfusion (  
    id INT PRIMARY KEY,  
    patient_id INT,  
    donation_id INT,  
    transfusion_date DATE,  
    FOREIGN KEY (patient_id) REFERENCES patient(id),  
    FOREIGN KEY (donation_id) REFERENCES donation(id)  
);
```

Table created.

DML-

```
INSERT INTO transfusion (patient_id, donation_id,transfusion_date) VALUES  
(1, 2, '2023-01-16'),  
(2, 1, '2022-11-16'),  
(3, 3, '2022-12-16'),  
(4, 4, '2023-02-16'),  
(5, 5, '2022-10-16');
```

row created.

## 5.1 QUERIES

1. Retrieve the names of all patients along with their blood types.

```
SELECT patient.name, patient.blood_type FROM patient;
```

Output

```
Alice|A+
Bob|B+
Charlie|AB-
Dave|O-
Emily|A-
```

2. Retrieve the names and phone numbers of all donors who have A+ blood type.

```
SELECT donor.name, donor.phone_number FROM donor WHERE donor.blood_type = 'A+';
```

Output

```
Frank|555-1234
```

3. Retrieve the names of all donors who have donated blood more than 5 times.

```
SELECT donor.name FROM donor WHERE donor.donation_count > 5;
```

Output

```
Gina
Jack
```

4. Retrieve the names and blood types of all donors who are eligible to donate blood.

```
SELECT donor.name, donor.blood_type FROM donor WHERE donor.is_eligible = 1;
```

Output

```
Frank|A+
Gina|O+
Henry|B+
Jack|O-
```

5. Retrieve the names and addresses of all blood banks.

```
SELECT blood_bank.name, blood_bank.address FROM blood_bank;
```

#### Output

```
Blood Bank A|123 Main St, Anytown  
Blood Bank B|456 Maple Ave, Otherville  
Blood Bank C|789 Oak Rd, Smalltown
```

6. Retrieve the names and phone numbers of all donors who have donated blood to a specific blood bank.

```
SELECT donor.name, donor.phone_number FROM donor JOIN  
donation ON donor.id = donation.donor_id JOIN blood_bank  
ON donation.blood_bank_id = blood_bank.id WHERE blood_bank.name = 'Blood Bank B';
```

#### Output

```
Gina|555-5678
```

```
Jack|555-3333
```

7. Retrieve the total number of donations made by each donor.

```
SELECT donor_id, COUNT(*) as num_donations  
FROM donation  
GROUP BY donor_id;
```

#### Output

```
1|2  
2|2  
3|2  
4|2  
5|2
```

8. Retrieve the total number of transfusions received by each patient.

```
SELECT patient_id, COUNT(*) as num_transfusions
FROM transfusion
GROUP BY patient_id;
```

#### Output

```
1|1
2|1
3|1
4|1
5|1
```

9. Retrieve all donations made on a specific date of day 15.

```
SELECT * FROM donation WHERE date like '%15';
```

#### Output

```
|1|3|2022-11-15
|2|3|2023-01-15
|3|1|2022-12-15
|4|2|2023-02-15
|5|1|2022-10-15
```

10. Show the total number of donations made by each donor, sorted in descending order.

```
SELECT distinct name, COUNT(donation.donor_id) AS total_donations
FROM donor
JOIN donation ON donor_id = donation.donor_id
GROUP BY donor_id
ORDER BY total_donations DESC;
```

#### Output

```
Frank|10
```

```
[Execution complete with exit code 0]
```



## IMPLEMENTATION

1. Retrieve the names of all patients along with their blood types.

```
Q1.java x
1 package JavaCbp;
2 import java.sql.*;
3 public class Q1 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         try{
8             Class.forName("com.mysql.cj.jdbc.Driver");
9             Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sys","root","root");
10            //here sys is database name, root is username and password
11            Statement stmt=con.createStatement();
12            ResultSet rs=stmt.executeQuery("SELECT patient.name, patient.blood_type FROM patient");
13            while(rs.next())
14                System.out.println(rs.getString(1)+" "+rs.getString(2)+" ");
15            con.close();
16        }
17        catch(Exception e) {
18            e.printStackTrace();
19        }
20    }
21 }
22 }
23 }
```

Problems Javadoc Declaration Console x

<terminated> Q1 [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (02-Aug-2023, 12:38:00 pm – 12:38:01 pm) [pid: 32380]

Alice A+  
Bob B+  
Charlie AB-  
Dave O-  
Emily A-

2. Retrieve the names and phone numbers of all donors who have A+ blood type.

```
1 package JavaCbp;
2 import java.sql.*;
3 public class Q1 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         try{
8             Class.forName("com.mysql.cj.jdbc.Driver");
9             Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sys","root","root");
10            //here sys is database name, root is username and password
11            Statement stmt=con.createStatement();
12            ResultSet rs=stmt.executeQuery("SELECT donor.name, donor.phone_number FROM donor WHERE "
13            + "donor.blood_type = 'A+'");
14            while(rs.next())
15                System.out.println(rs.getString(1)+" "+rs.getString(2)+" ");
16            con.close();
17        }
18        catch(Exception e) {
19            e.printStackTrace();
20        }
21    }
22 }
23 }
```

Problems Javadoc Declaration Console x

<terminated> Q1 [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (02-Aug-2023, 12:46:45 pm – 12:46:47 pm) [pid: 27008]

Frank 555-1234

3. Retrieve the names of all donors who have donated blood more than 5 times.

```
1 package JavaCbp;
2 import java.sql.*;
3 public class Q1 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         try{
8             Class.forName("com.mysql.cj.jdbc.Driver");
9             Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sys","root","root");
10            //here sys is database name, root is username and password
11            Statement stmt=con.createStatement();
12            ResultSet rs=stmt.executeQuery("SELECT donor.name FROM donor WHERE donor.donation_count>5");
13            while(rs.next())
14                System.out.println(rs.getString(1));
15            con.close();
16        }
17        catch(Exception e) {
18            e.printStackTrace();
19        }
20    }
21
22 }
```

Problems Javadoc Declaration Console ×

<terminated> Q1 [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (02-Aug-2023, 12:48:15 pm – 12:48:17 pm) [pid: 30380]

Gina  
Jack

4. Retrieve the names and blood types of all donors who are eligible to donate blood.

```
1 package JavaCbp;
2 import java.sql.*;
3 public class Q1 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         try{
8             Class.forName("com.mysql.cj.jdbc.Driver");
9             Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sys","root","root");
10            //here sys is database name, root is username and password
11            Statement stmt=con.createStatement();
12            ResultSet rs=stmt.executeQuery("SELECT donor.name, donor.blood_type FROM donor WHERE "
13            + "donor.is_eligible = 1");
14            while(rs.next())
15                System.out.println(rs.getString(1)+" "+rs.getString(2));
16            con.close();
17        }
18        catch(Exception e) {
19            e.printStackTrace();
20        }
21    }
22
23 }
```

Problems Javadoc Declaration Console ×

<terminated> Q1 [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (02-Aug-2023, 12:49:47 pm – 12:49:50 pm) [pid: 24220]

Frank A+  
Gina O+  
Henry B+  
Jack O-

5. Retrieve the names and phone numbers of all donors who have donated blood to a specific blood bank.

```
1 package JavaCbp;
2 import java.sql.*;
3 public class Q1 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         try{
8             Class.forName("com.mysql.cj.jdbc.Driver");
9             Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sys","root","root");
10            //here sys is database name, root is username and password
11            Statement stmt=con.createStatement();
12            ResultSet rs=stmt.executeQuery("SELECT donor.name, donor.phone_number FROM donor JOIN "
13                + "donation ON donor.id = donation.donor_id JOIN blood_bank ON "
14                + "donation.bb_id = blood_bank.id WHERE blood_bank.name = 'Blood Bank B' ");
15            while(rs.next())
16                System.out.println(rs.getString(1)+" "+rs.getString(2));
17            con.close();
18        }
19        catch(Exception e) {
20            e.printStackTrace();
21        }
22    }
23 }
24 }
```

Problems Javadoc Declaration Console X

<terminated> Q1 [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (02-Aug-2023, 12:52:57 pm - 12:52:59 pm) [pid: 17312]

Henry 555-1111  
Jack 555-3333  
Irene 555-2222

6. Retrieve all donations made on a specific date of day 15.

```
1 package JavaCbp;
2 import java.sql.*;
3 public class Q1 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         try{
8             Class.forName("com.mysql.cj.jdbc.Driver");
9             Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sys","root","root");
10            //here sys is database name, root is username and password
11            Statement stmt=con.createStatement();
12            ResultSet rs=stmt.executeQuery("SELECT * from donation where date like '%15%'");
13            while(rs.next())
14                System.out.println(rs.getInt(1)+" "+rs.getInt(2)+" "+rs.getInt(3)+" "+rs.getDate(4));
15            con.close();
16        }
17        catch(Exception e) {
18            e.printStackTrace();
19        }
20    }
21 }
22 }
```

Problems Javadoc Declaration Console X

<terminated> Q1 [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (02-Aug-2023, 12:57:47 pm - 12:57:49 pm) [pid: 25984]

6 1 3 2022-11-15  
7 2 3 2023-01-15  
8 3 1 2022-12-15  
9 4 2 2023-02-15  
10 5 1 2022-10-15

7. Show the total number of donations made by each donor, sorted in descending order.

```
1 package JavaCbp;
2 import java.sql.*;
3 public class Q1 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         try{
8             Class.forName("com.mysql.cj.jdbc.Driver");
9             Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sys","root","root");
10            //here sys is database name, root is username and password
11            Statement stmt=con.createStatement();
12            ResultSet rs=stmt.executeQuery("SELECT donor_id,COUNT(*) as num_donations "
13                + "FROM donation GROUP BY donor_id");
14            while(rs.next())
15                System.out.println(rs.getInt(1)+" "+rs.getInt(2));
16            con.close();
17        }
18        catch(Exception e) {
19            e.printStackTrace();
20        }
21    }
22 }
23 }
24 }
```

Problems Javadoc Declaration Console X

<terminated> Q1 [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (02-Aug-2023, 12:55:01 pm – 12:55:04 pm) [pid: 33580]

```
1 2
2 2
3 2
4 2
5 2
```

8. Show the total number of donations made by each blood bank, sorted in ascending order of banks id

```
Q1.java X
1 package JavaCbp;
2 import java.sql.*;
3 public class Q1 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         try{
8             Class.forName("com.mysql.cj.jdbc.Driver");
9             Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sys","root","root");
10            //here sys is database name, root is username and password
11            Statement stmt=con.createStatement();
12            ResultSet rs=stmt.executeQuery("SELECT blood_bank.id, COUNT(donation.id) AS total_donations\r\n"
13                + "FROM blood_bank\r\n"
14                + "JOIN donation ON blood_bank.id = donation.id\r\n"
15                + "GROUP BY blood_bank.id\r\n"
16                + "ORDER BY blood_bank.id;");
17            while(rs.next())
18                System.out.println(rs.getInt(1)+" "+rs.getInt(2));
19            con.close();
20        }
21        catch(Exception e) {
22            e.printStackTrace();
23        }
24    }
25 }
26 }
27 }
```

Problems Javadoc Declaration Console X

<terminated> Q1 [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (02-Aug-2023, 1:10:26 pm – 1:10:29 pm) [pid: 3612]

```
1 1
2 1
3 1
```

9. Display the max amount of blood donated by each donor sorted in descending order

```
Q1.java X
1 package JavaCbp;
2 import java.sql.*;
3 public class Q1 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         try{
8             Class.forName("com.mysql.cj.jdbc.Driver");
9             Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sys","root","root");
10            //here sys is database name, root is username and password
11            Statement stmt=con.createStatement();
12            ResultSet rs=stmt.executeQuery("SELECT donor.id,donor.name, MAX(donor.donation_count) AS max_donatio
13            + "FROM donor\r\n"
14            + "JOIN donation ON donor.id = donation.donor_id\r\n"
15            + "GROUP BY donor.id\r\n"
16            + "ORDER BY max_donation DESC;");
17            while(rs.next())
18                System.out.println(rs.getInt(1)+" "+rs.getString(2)+" "+rs.getInt(3));
19            con.close();
20        }
21        catch(Exception e) {
22            e.printStackTrace();
23        }
24    }
25 }
26 }
27 }
```

Problems Javadoc Declaration Console X

<terminated> Q1 [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (02-Aug-2023, 1:24:57 pm – 1:24:59 pm) [pid: 29092]

2 Gina 8  
5 Jack 6  
4 Irene 5  
1 Frank 3  
3 Henry 2

10. Retrieve the names of all patients who have received a blood from donor with specific name

```
Q1.java X
1 package JavaCbp;
2 import java.sql.*;
3 public class Q1 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         try{
8             Class.forName("com.mysql.cj.jdbc.Driver");
9             Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sys","root","root");
10            //here sys is database name, root is username and password
11            Statement stmt=con.createStatement();
12            ResultSet rs=stmt.executeQuery("SELECT patient.name FROM patient JOIN transfusion "
13            + "ON patient.id = transfusion.patient_id JOIN donation ON "
14            + "transfusion.donation_id = donation.id JOIN donor ON donation.donor_id = donor.id "
15            + "WHERE donor.name = 'Gina';\r\n");
16            while(rs.next())
17                System.out.println(rs.getString(1));
18            con.close();
19        }
20        catch(Exception e) {
21            e.printStackTrace();
22        }
23    }
24 }
25 }
```

Problems Javadoc Declaration Console X

<terminated> Q1 [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (02-Aug-2023, 1:32:34 pm – 1:32:36 pm) [pid: 13208]

Alice



11. Display the average age of donors who have made the donation sorted by age

```
*Q1.java ×
1 package JavaCbp;
2 import java.sql.*;
3 public class Q1 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         try{
8             Class.forName("com.mysql.cj.jdbc.Driver");
9             Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sys","root","root");
10            //here sys is database name, root is username and password
11            Statement stmt=con.createStatement();
12            ResultSet rs=stmt.executeQuery("SELECT ROUND(AVG(donor.age), 2) AS avg_age,"
13            + "COUNT(donation.id) AS total_donations\r\n"
14            + "FROM donor\r\n"
15            + "JOIN donation ON donor.id = donation.donor_id\r\n"
16            + "GROUP BY donor.age\r\n"
17            + "ORDER BY donor.age;");
18            while(rs.next())
19                System.out.println(rs.getInt(1)+" "+rs.getInt(2));
20            con.close();
21        }
22        catch(Exception e) {
23            e.printStackTrace();
24        }
25    }
26
27 }
28
```

Problems Javadoc Declaration Console ×

<terminated> Q1 [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (02-Aug-2023, 1:15:24 pm – 1:15:26 pm) [pid: 10680]

```
23 2
29 2
30 2
37 2
42 2
```

## **CHAPTER-6**

### **6.0 CONCLUSION**

Prior to this project, a general study of blood bank management system was conducted from recent researches of various authors and facts were gathered in which helped to uncover the misfits that the system was facing. After proper analyzation of these problems, a solution was then developed in order to meet up the needs of a more advanced system. This system is known as the centralized blood bank repository which helped in eliminating all the problems that the previous systems were facing. With this system, Blood banks/ Centers, Hospitals, Patients and Blood donors will be brought together to enjoy a large number of functionalities and access a vast amount of information, thereby making blood donation and reception a lot easier and faster.

Before implementing the database, in the design phase, We have explored various features, operations of a blood bank to figure out required entities, attributes and the relationship among entities to make an efficient Entity Relationship Diagram(ERD). After analyzing all the requirements, we have created our ERD and then converted the ERD to relational model and normalized the tables.

In our project Blood bank management system we have stored all the information about the donors, patients, blood banks etc. This data base is helpful for the applications which facilitate users to check the details of different blood groups available and their donors from their place itself it avoids inconvenience of going to blood banks for each and every query they get.

We had considered the most important requirements only, many more features and details can be added to our project in order to obtain even more user friendly applications. These applications are already in progress and in future they can be upgraded and may become part of amazing technology.

# REFERENCES

- [1]. IEEE Software Requirement Specification format.
- [2]. <https://photograph.Slidesharecdn.Com/reportsbb-180225201600/95/file-on-smart-blood-financial-institution-venture-24-638.Jpg?Cb=1519589905>
- [3]. <https://image.Slidesharecdn.Com/reportsbb-180225201600/95/document-on-smart-blood-bank-undertaking-24-638.Jpg?Cb=1519589905>
- [4]. <https://photo.Slidesharecdn.Com/reportsbb-180225201600/ninetyfive/file-on-smart-blood-financialinstitution-challenge-24-638.Jpg?Cb=519589905>
- [5]. [www.project-management-basics.com](http://www.project-management-basics.com)
- [6]. <https://www.javatpoint.com/uml-diagrams>