

# **PROJECT**

for

## **Database Management Systems Laboratory (CS238)**

### **Bachelor of Technology (CSE)**

Second Year, Semester 4

*Course In-charge: Prof. Ujwala Petigara*

## **Title: Blood Bank Management System**

Group members:-

22000404 - Harsh Patel

22000409 – Ramoliya Kaushal



**NAVRACHANA  
UNIVERSITY**  
*a UGC recognized University*

Department of Computer Science and Engineering

School Engineering and Technology

Navrachana University, Vadodara

Spring Semester 2024

# Index

<b>Sr. No.</b>	<b>Topics</b>	<b>Page Number</b>
1	Description of E-R Diagram	3
2	Cardinalities	5
3	Entity Relationship (ER) Diagram	6
4	Reduction of ER diagram to Table	7
5	Creating Tables and Insert Data	8
6	Database Security	18
7	Linking Python and MySQL	19
8	GUI Using Tkinter	20
9	Conclusion	22

## **Description of E-R Diagram**

### **Entity:-**

1. Donor
2. Patient
3. Blood Bank
4. Registration\_Team
5. Hospital
6. Manager
7. Blood\_Bank\_Available

### **Attributes:-**

1. **Donor :-**  
d\_id,d\_name,age,gender,blood\_group,b\_id,r\_id,donation\_date,quantity,contact,city, pincode,state.
2. **Patient :-**  
p\_id,p\_name,age,gender,blood\_group,h\_id,r\_id,intake\_date,quantity,contact,city, pincode,state.
3. **Blood Bank :-** b\_id,b\_name,h\_id.
4. **Registration\_Team :-** r\_id,r\_name,h\_id.
5. **Hospital :-** h\_id,h\_name,city.
6. **Manager :-** m\_id,m\_name,b\_id.
7. **Blood\_Bank\_Available :-** b\_id,blood\_group,quantity.

## **Primary Key:-**

1. **Donor** :- d\_id.
2. **Patient** :- p\_id.
3. **Blood Bank** :- b\_id.
4. **Registration\_Team** :- r\_id.
5. **Hospital** :- h\_id.
6. **Manager** :- m\_id.
7. **Blood\_Bank\_Available** :- b\_id, blood\_group.

## **Foreign Key:-**

1. **Donor** :- b\_id - References Blood\_Bank(b\_id)  
              :- r\_id - References Registration\_Team(r\_id)
2. **Patient** :- h\_id - References Hospital(h\_id)  
              :- r\_id - References Registration\_Team(r\_id)
3. **Blood Bank** :- h\_id - References hospital(h\_id)
4. **Registration\_Team** :- h\_id References Hospital(h\_id)
5. **Manager** :- b\_id - References Blood\_Bank(b\_id)
6. **Blood\_Bank\_Available** :- b\_id - References Blood\_Bank(b\_id)

## **Cardinality**

### **1. Registration\_Team to Donor:- One-to-Many**

- Each registration team can handle registrations for many donors, but each donor is associated with only one registration team.

### **2. Registration\_Team to Patient:- One-to-Many**

- One registration team can be associated with many patients.,  
Each patient is registered by one registration team.

### **3. patient to hospital:- Many-to-One**

- A patient belongs to one hospital, But a hospital can have multiple patients.

### **4. registration\_team to hospital:- Many-to-One**

- A registration team belongs to one hospital, However, a hospital can have multiple registration teams.

### **5. donor to blood\_bank:- Many-to-One**

- A donor donates to one blood bank, But a blood bank can have multiple donors.

### **6. hospital to blood\_bank:- One-to-One**

- A hospital has one blood bank.

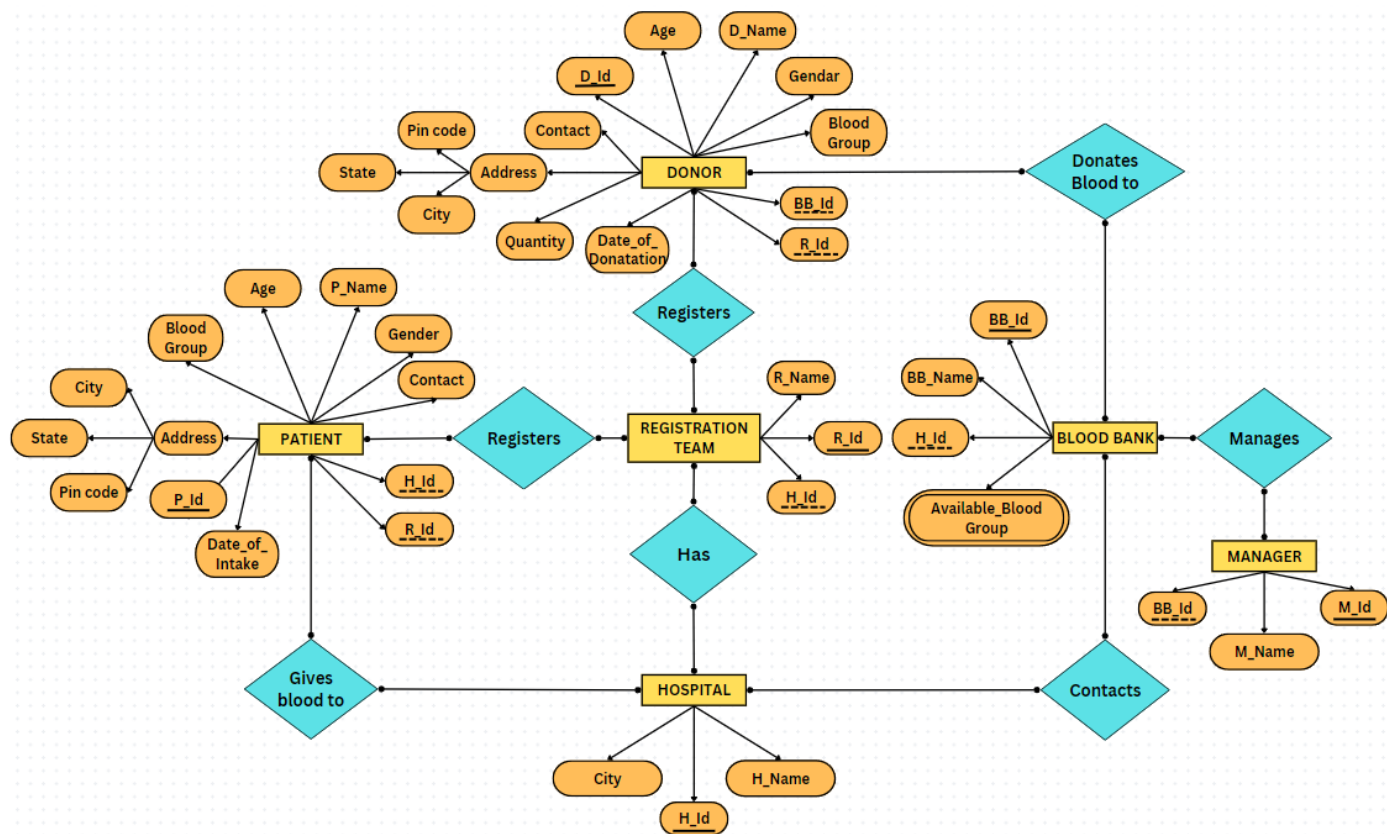
### **7. blood\_bank\_available to blood\_bank:- Many-to-One**

- Each entry in the Blood Bank Available table is associated with one blood bank, A blood bank can have multiple entries in the Blood Bank Available table for different blood groups.

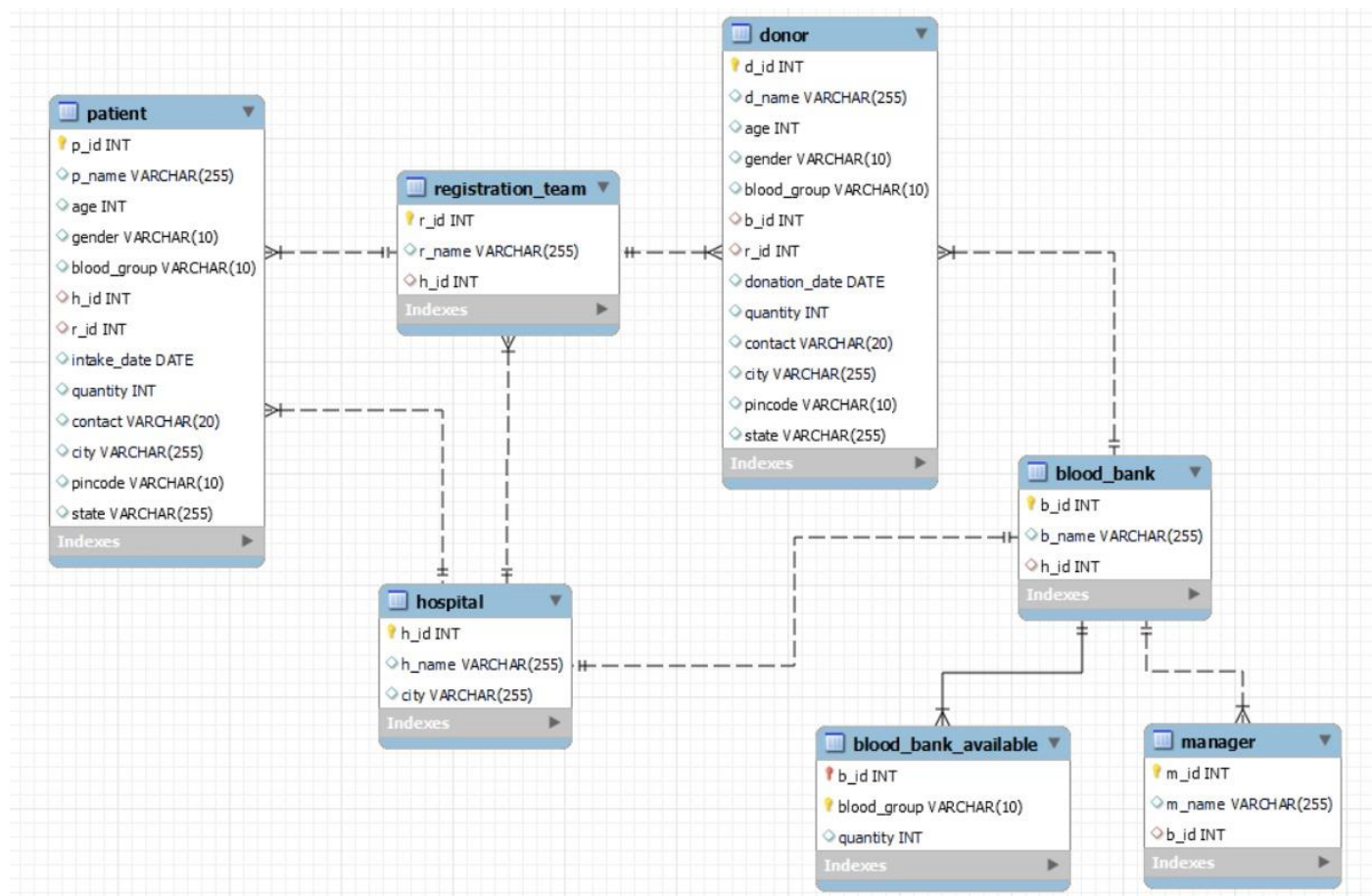
### **8. manager to blood\_bank:- One-to-Many**

- A manager manages one blood bank, However, a blood bank can have Many manager.

## E-R Diagram



## Reduction of ER diagram to Table



## Creating Tables and Insert Data

### Creating Tables:-

**Step 1:** Create Database:

**Query:** create database dbms\_project;

**Output:**

```
mysql> create database dbms_project;  
Query OK, 1 row affected (0.07 sec)
```

**Step 2:** use Database:

**Query:** use dbms\_project;

**Output:**

```
mysql> use dbms_project;  
Database changed
```

**Step 3:** Create Table Hospital:

**Query:** CREATE TABLE Hospital (

h\_id INT PRIMARY KEY,

h\_name VARCHAR(255),

city VARCHAR(255)

);

**Output:**

```
mysql> CREATE TABLE Hospital (  
-> h_id INT PRIMARY KEY,  
-> h_name VARCHAR(255),  
-> city VARCHAR(255)  
-> );  
Query OK, 0 rows affected (0.07 sec)
```



**Step4:** Create Table Blood\_Bank:**Query:** CREATE TABLE Blood\_Bank (

b\_id INT PRIMARY KEY,

b\_name VARCHAR(255),

h\_id INT,

FOREIGN KEY (h\_id) REFERENCES Hospital(h\_id)

);

**Output:**

```
mysql> CREATE TABLE Blood_Bank (
->   b_id INT PRIMARY KEY,
->   b_name VARCHAR(255),
->   h_id INT,
->   FOREIGN KEY (h_id) REFERENCES Hospital(h_id)
-> );
Query OK, 0 rows affected (0.06 sec)
```

**Step 5:** Create Table Blood\_Bank\_Available:**Query:** CREATE TABLE Blood\_Bank\_Available (

b\_id INT,

blood\_group VARCHAR(10),

quantity INT,

FOREIGN KEY (b\_id) REFERENCES Blood\_Bank(b\_id),

PRIMARY KEY (b\_id, blood\_group)

);

**Output:**

```
mysql> CREATE TABLE Blood_Bank_Available (
->   b_id INT,
->   blood_group VARCHAR(10),
->   quantity INT,
->   FOREIGN KEY (b_id) REFERENCES Blood_Bank(b_id),
->   PRIMARY KEY (b_id, blood_group)
-> );
Query OK, 0 rows affected (0.03 sec)
```

**Step6: Create Table Manager:****Query:** CREATE TABLE Manager (

```
m_id INT PRIMARY KEY,  
m_name VARCHAR(255),  
b_id INT,  
FOREIGN KEY (b_id) REFERENCES Blood_Bank(b_id)
```

);

**Output:**

```
mysql> CREATE TABLE Manager (  
->     m_id INT PRIMARY KEY,  
->     m_name VARCHAR(255),  
->     b_id INT,  
->     FOREIGN KEY (b_id) REFERENCES Blood_Bank(b_id)  
-> );  
Query OK, 0 rows affected (0.08 sec)
```

**Step 7: Create Table Registration\_Team:****Query:** CREATE TABLE Registration\_Team (

```
r_id INT PRIMARY KEY,  
r_name VARCHAR(255),  
h_id INT,  
FOREIGN KEY (h_id) REFERENCES Hospital(h_id)
```

);

**Output:**

```
mysql> CREATE TABLE Registration_Team (  
->     r_id INT PRIMARY KEY,  
->     r_name VARCHAR(255),  
->     h_id INT,  
->     FOREIGN KEY (h_id) REFERENCES Hospital(h_id)  
-> );  
Query OK, 0 rows affected (0.08 sec)
```

**Step 8: Create Table Donor:****Query:** CREATE TABLE Donor (

```
d_id INT PRIMARY KEY,  
d_name VARCHAR(255),  
age INT CHECK (age BETWEEN 18 AND 65),  
gender VARCHAR(10),  
blood_group VARCHAR(10),  
b_id INT,  
r_id INT,  
donation_date DATE,  
quantity INT,  
contact VARCHAR(20),  
city VARCHAR(255),  
pincode VARCHAR(10),  
state VARCHAR(255),  
FOREIGN KEY (b_id) REFERENCES Blood_Bank(b_id),  
FOREIGN KEY (r_id) REFERENCES Registration_Team(r_id)  
);
```

**Output:**

```
mysql> CREATE TABLE Donor (  
-> d_id INT PRIMARY KEY,  
-> d_name VARCHAR(255),  
-> age INT CHECK (age BETWEEN 18 AND 65),  
-> gender VARCHAR(10),  
-> blood_group VARCHAR(10),  
-> b_id INT,  
-> r_id INT,  
-> donation_date DATE,  
-> quantity INT,  
-> contact VARCHAR(20),  
-> city VARCHAR(255),  
-> pincode VARCHAR(10),  
-> state VARCHAR(255),  
-> FOREIGN KEY (b_id) REFERENCES Blood_Bank(b_id),  
-> FOREIGN KEY (r_id) REFERENCES Registration_Team(r_id)  
-> );  
Query OK, 0 rows affected (0.04 sec)
```

**Step 9: Create Table Patient:****Query:** CREATE TABLE Patient (

```
p_id INT PRIMARY KEY,  
p_name VARCHAR(255),  
age INT,  
gender VARCHAR(10),  
blood_group VARCHAR(10),  
h_id INT,  
r_id INT,  
intake_date DATE,  
quantity INT,  
contact VARCHAR(20),  
city VARCHAR(255),  
pincode VARCHAR(10),  
state VARCHAR(255),  
FOREIGN KEY (h_id) REFERENCES Hospital(h_id),  
FOREIGN KEY (r_id) REFERENCES Registration_Team(r_id)  
);
```

**Output:**

```
mysql> CREATE TABLE Patient (  
-> p_id INT PRIMARY KEY,  
-> p_name VARCHAR(255),  
-> age INT,  
-> gender VARCHAR(10),  
-> blood_group VARCHAR(10),  
-> h_id INT,  
-> r_id INT,  
-> intake_date DATE,  
-> quantity INT,  
-> contact VARCHAR(20),  
-> city VARCHAR(255),  
-> pincode VARCHAR(10),  
-> state VARCHAR(255),  
-> FOREIGN KEY (h_id) REFERENCES Hospital(h_id),  
-> FOREIGN KEY (r_id) REFERENCES Registration_Team(r_id)  
-> );  
Query OK, 0 rows affected (0.06 sec)
```

## Insert Data:-

### 1. Hospital Table:

#### Query:

```
INSERT INTO Hospital (h_id, h_name, city) VALUES
(1, 'City Hospital', 'New York'),
(2, 'Central Hospital', 'London'),
(3, 'Community Hospital', 'Paris'),
(4, 'General Hospital', 'Berlin'),
(5, 'Regional Hospital', 'Tokyo');
```

#### Output:

```
mysql> INSERT INTO Hospital (h_id, h_name, city) VALUES
-> (1, 'City Hospital', 'New York'),
-> (2, 'Central Hospital', 'London'),
-> (3, 'Community Hospital', 'Paris'),
-> (4, 'General Hospital', 'Berlin'),
-> (5, 'Regional Hospital', 'Tokyo');
Query OK, 5 rows affected (0.02 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

#### Show Table:

```
mysql> select * from Hospital;
+-----+-----+-----+
| h_id | h_name          | city   |
+-----+-----+-----+
| 1    | City Hospital   | New York |
| 2    | Central Hospital | London  |
| 3    | Community Hospital | Paris  |
| 4    | General Hospital | Berlin  |
| 5    | Regional Hospital | Tokyo   |
+-----+-----+-----+
5 rows in set (0.00 sec)
```

### 2. Blood\_Bank Table:

#### Query:

```
INSERT INTO Blood_Bank (b_id, b_name, h_id) VALUES
(101, 'Blood Bank A', 1),
(102, 'Blood Bank B', 2),
(103, 'Blood Bank C', 3),
(104, 'Blood Bank D', 4),
(105, 'Blood Bank E', 5);
```

**Output:**

```
mysql> INSERT INTO Blood_Bank (b_id, b_name, h_id) VALUES
-> (101, 'Blood Bank A', 1),
-> (102, 'Blood Bank B', 2),
-> (103, 'Blood Bank C', 3),
-> (104, 'Blood Bank D', 4),
-> (105, 'Blood Bank E', 5);
Query OK, 5 rows affected (0.01 sec)
Records: 5 Duplicates: 0 Warnings: 0
```

**Show Table:**

```
mysql> select * from Blood_Bank;
+-----+-----+-----+
| b_id | b_name      | h_id |
+-----+-----+-----+
| 101  | Blood Bank A | 1    |
| 102  | Blood Bank B | 2    |
| 103  | Blood Bank C | 3    |
| 104  | Blood Bank D | 4    |
| 105  | Blood Bank E | 5    |
+-----+-----+-----+
5 rows in set (0.00 sec)
```

**3. Blood\_Bank\_Available Table:****Query:**

```
INSERT INTO Blood_Bank_Available (b_id, blood_group, quantity) VALUES
(101, 'A+', 100),
(101, 'B+', 150),
(101, 'O+', 200),
(102, 'A-', 120),
(102, 'AB+', 180),
(103, 'B-', 90);
```

**Output:**

```
mysql> INSERT INTO Blood_Bank_Available (b_id, blood_group, quantity) VALUES
-> (101, 'A+', 100),
-> (101, 'B+', 150),
-> (101, 'O+', 200),
-> (102, 'A-', 120),
-> (102, 'AB+', 180),
-> (103, 'B-', 90);
Query OK, 6 rows affected (0.01 sec)
Records: 6 Duplicates: 0 Warnings: 0
```

**Show Table:**

```
mysql> select * from Blood_Bank_Available;
+-----+-----+-----+
| b_id | blood_group | quantity |
+-----+-----+-----+
| 101  | A+          | 100      |
| 101  | B+          | 150      |
| 101  | O+          | 200      |
| 102  | A-          | 120      |
| 102  | AB+         | 180      |
| 103  | B-          | 90       |
+-----+-----+-----+
6 rows in set (0.00 sec)
```

**4. Manager Table:****Query:**

```
INSERT INTO Manager (m_id, m_name, b_id) VALUES
(201, 'John Doe', 101),
(202, 'Jane Smith', 102),
(203, 'Michael Johnson', 103),
(204, 'Emma Williams', 104),
(205, 'David Brown', 105);
```

**Output:**

```
mysql> INSERT INTO Manager (m_id, m_name, b_id) VALUES
-> (201, 'John Doe', 101),
-> (202, 'Jane Smith', 102),
-> (203, 'Michael Johnson', 103),
-> (204, 'Emma Williams', 104),
-> (205, 'David Brown', 105);
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

**Show Table:**

```
mysql> select * from Manager;
+-----+-----+-----+
| m_id | m_name          | b_id |
+-----+-----+-----+
| 201  | John Doe        | 101  |
| 202  | Jane Smith      | 102  |
| 203  | Michael Johnson | 103  |
| 204  | Emma Williams   | 104  |
| 205  | David Brown     | 105  |
+-----+-----+-----+
5 rows in set (0.00 sec)
```

## 5. Registration\_Team Table:

### Query:

```
INSERT INTO Registration_Team (r_id, r_name, h_id) VALUES
(301, 'Team A', 1),
(302, 'Team B', 2),
(303, 'Team C', 3),
(304, 'Team D', 4),
(305, 'Team E', 5);
```

### Output:

```
mysql> INSERT INTO Registration_Team (r_id, r_name, h_id) VALUES
-> (301, 'Team A', 1),
-> (302, 'Team B', 2),
-> (303, 'Team C', 3),
-> (304, 'Team D', 4),
-> (305, 'Team E', 5);
Query OK, 5 rows affected (0.01 sec)
Records: 5 Duplicates: 0 Warnings: 0
```

### Show Table:

```
mysql> select * from Registration_Team;
+-----+-----+-----+
| r_id | r_name | h_id |
+-----+-----+-----+
| 301 | Team A | 1 |
| 302 | Team B | 2 |
| 303 | Team C | 3 |
| 304 | Team D | 4 |
| 305 | Team E | 5 |
+-----+-----+-----+
5 rows in set (0.00 sec)
```

## 6. Donor Table:

### Query:

```
INSERT INTO Donor (d_id, d_name, age, gender, blood_group, b_id, r_id, donation_date, quantity,
contact, city, pincode, state) VALUES
(401, 'Alice Johnson', 30, 'Female', 'A+', 101, 301, '2024-04-01', 2, '1234567890', 'New York', '10001',
'NY'),
(402, 'Bob Smith', 45, 'Male', 'O-', 102, 302, '2024-04-02', 1, '9876543210', 'London', 'SW1A 1AA',
'London'),
(403, 'Charlie Brown', 25, 'Male', 'B+', 103, 303, '2024-04-03', 3, '5678901234', 'Paris', '75001', 'Paris'),
(404, 'Diana Williams', 35, 'Female', 'AB-', 104, 304, '2024-04-04', 2, '3456789012', 'Berlin', '10115',
```



'Berlin'),

(405, 'Eva Garcia', 28, 'Female', 'A-', 105, 305, '2024-04-05', 1, '7890123456', 'Tokyo', '100-0001', 'Tokyo');

### Output:

```
mysql> INSERT INTO Donor (d_id, d_name, age, gender, blood_group, b_id, r_id, donation_date, quantity, contact, city, pincode, state) VALUES
-> (401, 'Alice Johnson', 30, 'Female', 'A+', 101, 301, '2024-04-01', 2, '1234567890', 'New York', '10001', 'NY'),
-> (402, 'Bob Smith', 45, 'Male', 'O-', 102, 302, '2024-04-02', 1, '9876543210', 'London', 'SW1A 1AA', 'London'),
-> (403, 'Charlie Brown', 25, 'Male', 'B+', 103, 303, '2024-04-03', 3, '5678901234', 'Paris', '75001', 'Paris'),
-> (404, 'Diana Williams', 35, 'Female', 'AB-', 104, 304, '2024-04-04', 2, '3456789012', 'Berlin', '10115', 'Berlin'),
-> (405, 'Eva Garcia', 28, 'Female', 'A-', 105, 305, '2024-04-05', 1, '7890123456', 'Tokyo', '100-0001', 'Tokyo');
Query OK, 5 rows affected (0.01 sec)
Records: 5 Duplicates: 0 Warnings: 0
```

### Show Table:

```
mysql> select * from Donor;
```

d_id	d_name	age	gender	blood_group	b_id	r_id	donation_date	quantity	contact	city	pincode	state
401	Alice Johnson	30	Female	A+	101	301	2024-04-01	2	1234567890	New York	10001	NY
402	Bob Smith	45	Male	O-	102	302	2024-04-02	1	9876543210	London	SW1A 1AA	London
403	Charlie Brown	25	Male	B+	103	303	2024-04-03	3	5678901234	Paris	75001	Paris
404	Diana Williams	35	Female	AB-	104	304	2024-04-04	2	3456789012	Berlin	10115	Berlin
405	Eva Garcia	28	Female	A-	105	305	2024-04-05	1	7890123456	Tokyo	100-0001	Tokyo

5 rows in set (0.00 sec)

## 7. Patient Table:

### Query:

INSERT INTO Patient (p\_id, p\_name, age, gender, blood\_group, h\_id, r\_id, intake\_date, quantity, contact, city, pincode, state) VALUES

(501, 'John Green', 50, 'Male', 'B+', 1, 301, '2024-04-01', 2, '1234567890', 'New York', '10001', 'NY'),

(502, 'Emily White', 35, 'Female', 'O-', 2, 302, '2024-04-02', 1, '9876543210', 'London', 'SW1A 1AA', 'London'),

(503, 'Michael Lee', 40, 'Male', 'A+', 3, 303, '2024-04-03', 3, '5678901234', 'Paris', '75001', 'Paris'),

(504, 'Sophia Lopez', 20, 'Female', 'AB-', 4, 304, '2024-04-04', 2, '3456789012', 'Berlin', '10115', 'Berlin'),

(505, 'Daniel Martinez', 55, 'Male', 'A-', 5, 305, '2024-04-05', 1, '7890123456', 'Tokyo', '100-0001', 'Tokyo');

### Output:

```
mysql> INSERT INTO Patient (p_id, p_name, age, gender, blood_group, h_id, r_id, intake_date, quantity, contact, city, pincode, state) VALUES
-> (501, 'John Green', 50, 'Male', 'B+', 1, 301, '2024-04-01', 2, '1234567890', 'New Ydonorregistration_teamr_idr_idblood_bankork', '10001', 'NY'),
-> (502, 'Emily White', 35, 'Female', 'O-', 2, 302, '2024-04-02', 1, '9876543210', 'London', 'SW1A 1AA', 'London'),
-> (503, 'Michael Lee', 40, 'Male', 'A+', 3, 303, '2024-04-03', 3, '5678901234', 'Paris', '75001', 'Paris'),
-> (504, 'Sophia Lopez', 20, 'Female', 'AB-', 4, 304, '2024-04-04', 2, '3456789012', 'Berlin', '10115', 'Berlin'),
-> (505, 'Daniel Martinez', 55, 'Male', 'A-', 5, 305, '2024-04-05', 1, '7890123456', 'Tokyo', '100-0001', 'Tokyo');
Query OK, 5 rows affected (0.01 sec)
Records: 5 Duplicates: 0 Warnings: 0
```

### Show Table:

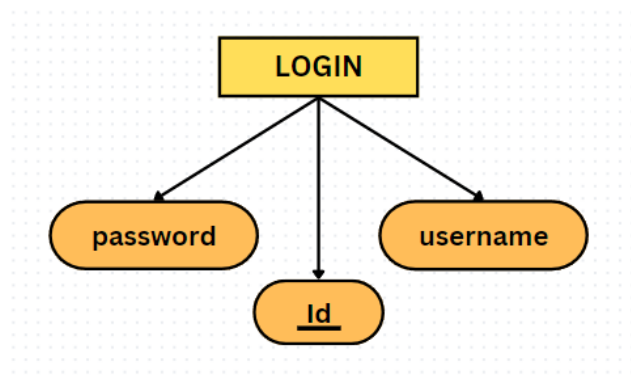
```
mysql> select * from Patient;
```

p_id	p_name	age	gender	blood_group	h_id	r_id	intake_date	quantity	contact	city	pincode	state
501	John Green	50	Male	B+	1	301	2024-04-01	2	1234567890	New York	10001	NY
502	Emily White	35	Female	O-	2	302	2024-04-02	1	9876543210	London	SW1A 1AA	London
503	Michael Lee	40	Male	A+	3	303	2024-04-03	3	5678901234	Paris	75001	Paris
504	Sophia Lopez	20	Female	AB-	4	304	2024-04-04	2	3456789012	Berlin	10115	Berlin
505	Daniel Martinez	55	Male	A-	5	305	2024-04-05	1	7890123456	Tokyo	100-0001	Tokyo

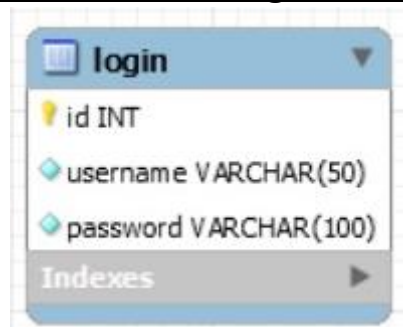
5 rows in set (0.00 sec)

## Database Security

### ER-Diagram



### Reduction of ER diagram to Table



#### Query:

```

CREATE TABLE login (
    id INT PRIMARY KEY,
    username VARCHAR(50) NOT NULL UNIQUE,
    password VARCHAR(100) NOT NULL
);
    
```

#### Output:

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	
username	varchar(50)	NO	UNI	NULL	
password	varchar(100)	NO		NULL	

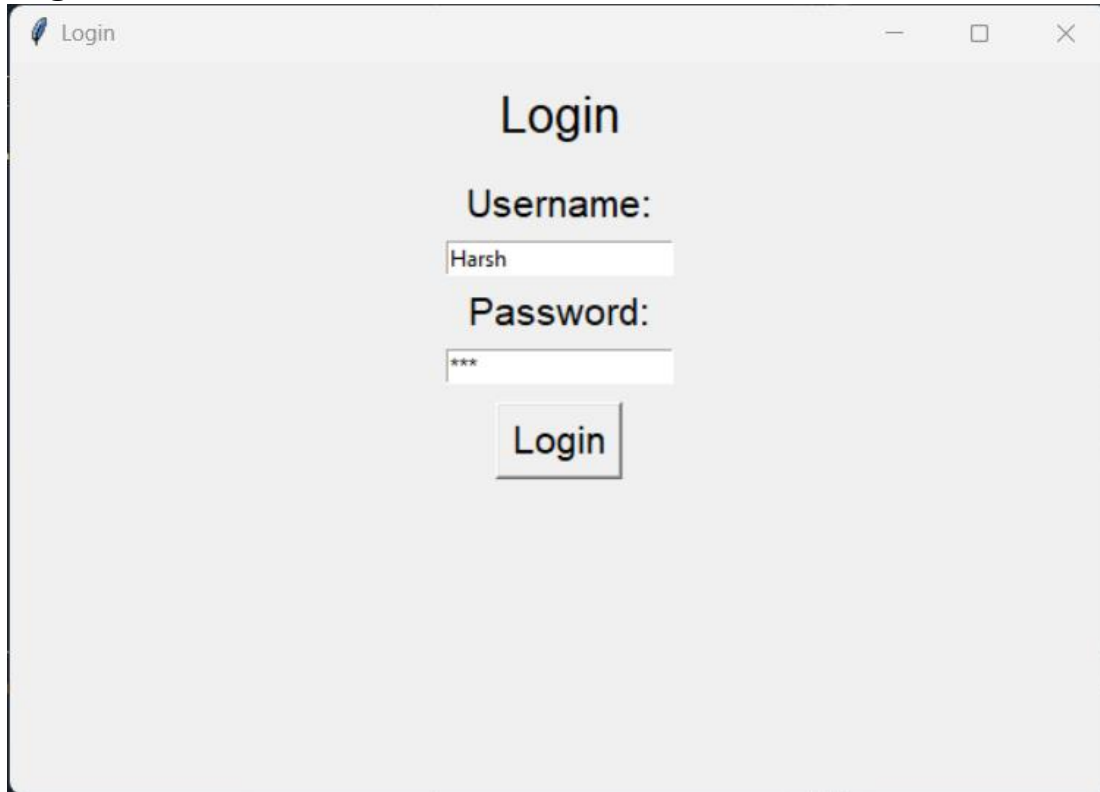
3 rows in set (0.01 sec)

## Linking Python and MySQL

1. Import necessary libraries:
  - `import mysql.connector`
2. Establish connection:
  - Replace 'username', 'password', 'host', and 'database\_name' with your MySQL credentials.
  - Syntax:
    - `mydb = mysql.connector.connect(  
host="localhost",  
user="username",  
password="password",  
database="database_name"  
)`
3. Create a cursor object:
  - Syntax:
    - `mycursor = mydb.cursor()`
4. Perform database operations:
  - Execute SQL queries using the cursor object.
  - Example:
    - `mycursor.execute("SELECT * FROM table_name")  
result = mycursor.fetchall()`
5. Close the connection:
  - Always remember to close the connection once the operations are completed.
  - Syntax:
    - `mydb.close()`
6. Now you're connected! Start querying and manipulating your MySQL database with Python.

## GUI Using Tkinter

### Login:-



Login

Username:

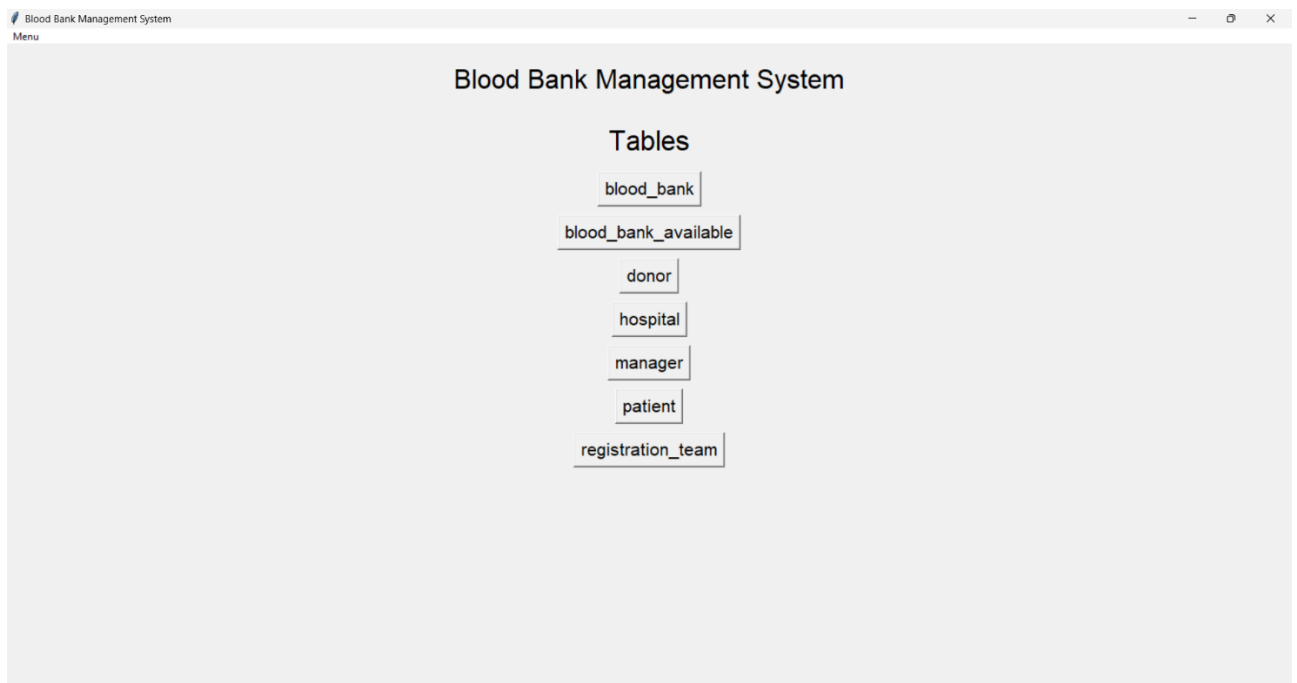
Harsh

Password:

\*\*\*

Login

### Tables:-



Blood Bank Management System

Tables

blood\_bank

blood\_bank\_available

donor

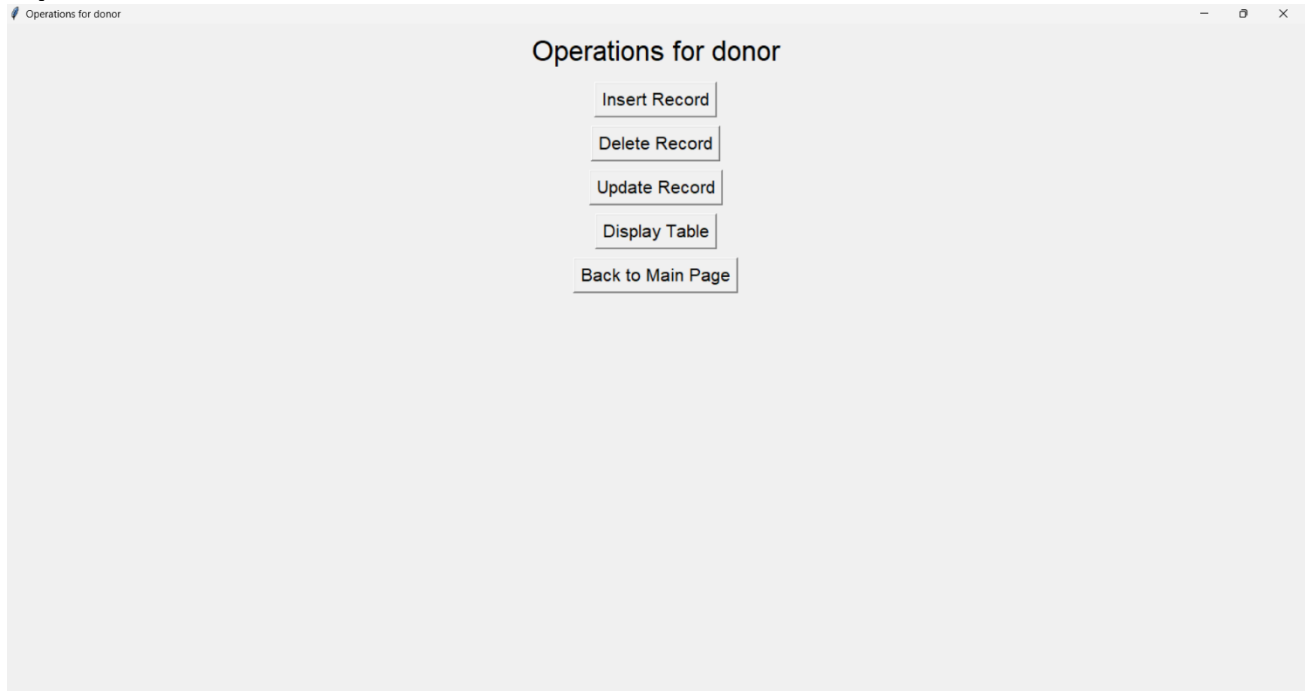
hospital

manager

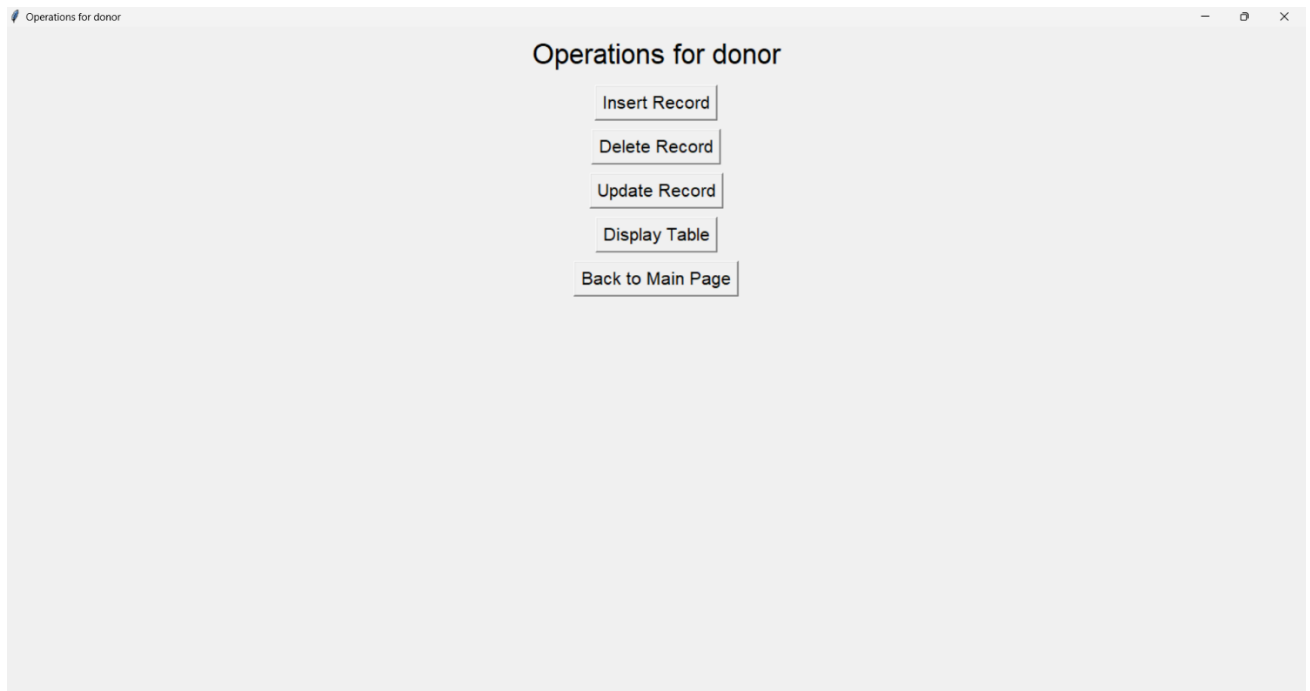
patient

registration\_team

## Operations:-



## Insert Record:-



## **Conclusion**

In this project, we have developed a Blood Bank Management System using Python and Tkinter GUI toolkit. The system allows users to perform various operations such as inserting, deleting, updating records, and displaying tables in a MySQL database. It provides a user-friendly interface for managing donor, patient, registration team, blood bank, and blood bank available information.

### **Here's how the implementation works:**

- **Login System:** Users need to authenticate themselves using a username and password to access the system. This ensures security and restricts unauthorized access.
- **Main Page:** After logging in, users are presented with a main page displaying options to perform operations on different tables such as donors, patients, etc.
- **Operations:** Users can perform CRUD (Create, Read, Update, Delete) operations on the tables. They can insert new records, delete existing ones, update records, and display tables to view information.
- **Integration with Blood Bank Availability:** We have integrated functionality to update the Blood Bank Availability table whenever new records are inserted into the donor or patient tables. This ensures that the blood bank availability is automatically updated based on incoming donations or patient requirements.

### **Impact on Blood Bank Management System:**

- **Efficiency:** The system automates many manual tasks involved in managing donor and patient records, making the process more efficient.
- **Accuracy:** By automating updates to the Blood Bank Availability table, the system ensures that the blood bank inventory is always up-to-date, reducing the chances of errors due to manual entry.
- **Accessibility:** The GUI interface makes it easy for blood bank staff to interact with the database, even if they have limited technical knowledge.
- **Data Management:** Centralized storage and management of donor, patient, and blood bank information improve data organization and accessibility.
- **Enhanced Decision Making:** Access to accurate and real-time data enables blood bank administrators to make informed decisions regarding blood inventory management and allocation.

### **Learning Experience:**

- **Database Connectivity:** We learned how to connect Python applications with MySQL databases using the `mysql.connector` module.
- **GUI Development:** Developing GUI applications using Tkinter taught us how to create interactive interfaces for users to interact with.
- **CRUD Operations:** Implementing CRUD operations helped us understand the fundamentals of database management and data manipulation.
- **Integration:** Integrating different components of the system, such as the login system and data management functionalities, enhanced our understanding of software design and

development.

Overall, the Blood Bank Management System provides a robust solution for managing blood bank operations efficiently, ensuring timely access to blood products, and ultimately contributing to the effective management of blood supplies for patient care.