DBMS Mini Project

Bloodbank Database Management

By

Diya Maria A (PES1UG21CS188)

Harsha R (PES1UG21CS222)

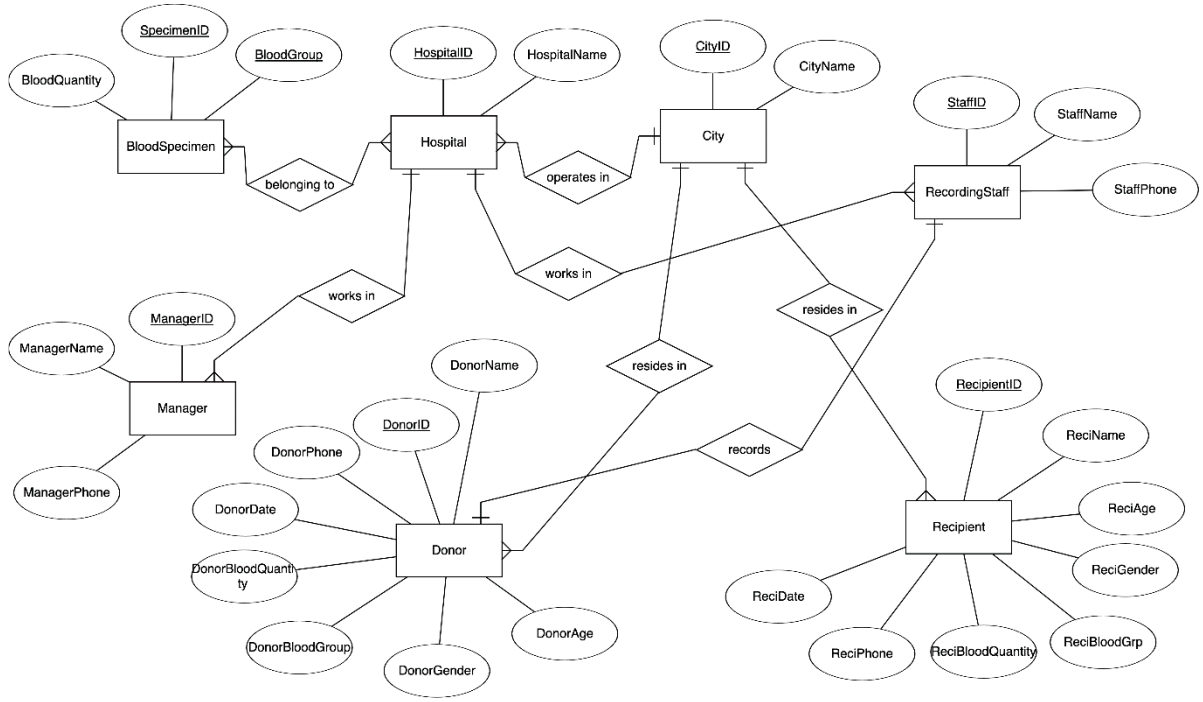
Abstract

The program handles the database for blood donation and it's concerned authorities . It provides a holistic approach to handle the donation process . It provides a platform for each end user to access the database based on their access rights as well as allows manipulation of the database on the same scale . The main end users are managers , recording staff and hospitals . We provide a special table for blood specimen , which keeps a tally on the remaining blood and it's details from the camp .

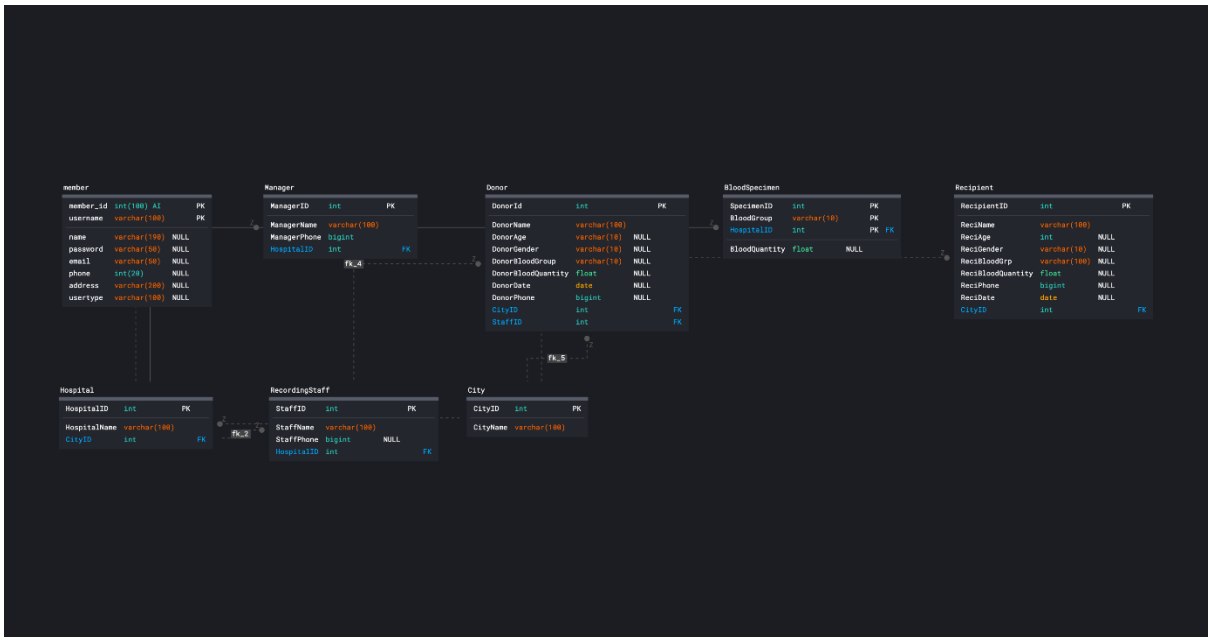
Software and Tools

- ➔ Php for the frontend
- ➔ Mysql for the database

ER Diagram



Relational Schema



The database design is in 3NF

- ➔ 1NF , as there are no multi variate attributes in the database
- ➔ 2NF , as there are no partial dependencies , all the functional dependencies are full dependencies
- ➔ 3NF , as there are no transitive dependencies with non prime attributes , but there are transitive dependencies with prime attributes which satisfy 3NF .

DDL Commands

Member Table:-

```
DROP TABLE IF EXISTS member;
CREATE TABLE member (
  member_id int(100) NOT NULL AUTO_INCREMENT,
  name varchar(190) DEFAULT NULL,
  username varchar(100) NOT NULL,
  password varchar(50) DEFAULT NULL,
  email varchar(50) DEFAULT NULL,
  phone int(20) DEFAULT NULL,
  address varchar(200) DEFAULT NULL,
  usertype varchar(100) DEFAULT NULL,
  PRIMARY KEY (member_id,username)
);
```

City Table:-

```
DROP TABLE IF EXISTS City;
CREATE TABLE City ( CityID int NOT NULL PRIMARY KEY,
                    CityName varchar(100) NOT NULL);
```

Hospital Table:-

```
DROP TABLE IF EXISTS Hospital;
CREATE TABLE Hospital (HospitalID int NOT NULL PRIMARY KEY,
                       HospitalName varchar(100) NOT NULL ,
                       CityID int NOT NULL,
                       FOREIGN KEY(CityID) REFERENCES City(CityID));
```

Recording Staff Table:-

```
create table RecordingStaff (StaffID int NOT NULL PRIMARY KEY,
                             StaffName varchar(100) NOT NULL,
                             StaffPhone bigint,
                             HospitalID int NOT NULL,
                             FOREIGN KEY(HospitalID) REFERENCES
Hospital(HospitalID));
```

Manager Table:-

```
DROP TABLE IF EXISTS Manager;
CREATE TABLE Manager(ManagerID int NOT NULL PRIMARY KEY,
                     ManagerName varchar(100) NOT NULL,
                     ManagerPhone bigint NOT NULL,
                     HospitalID int NOT NULL,
                     FOREIGN KEY(HospitalID) REFERENCES Hospital(HospitalID));
```

Donor Table:-

```
DROP TABLE IF EXISTS Donor;
create table Donor(DonorId int NOT NULL PRIMARY KEY,
                DonorName varchar(100)NOT NULL,
                DonorAge varchar(10),
                DonorGender varchar(10),
                DonorBloodGroup varchar(10),
                DonorBloodQuantity float,
                DonorDate date,
                DonorPhone bigint,
                CityID int NOT NULL,
                StaffID int NOT NULL ,
                FOREIGN KEY (StaffID) REFERENCES RecordingStaff(StaffID),
                FOREIGN KEY(CityID) REFERENCES City(CityId));
```

Bloodspecimen Table:-

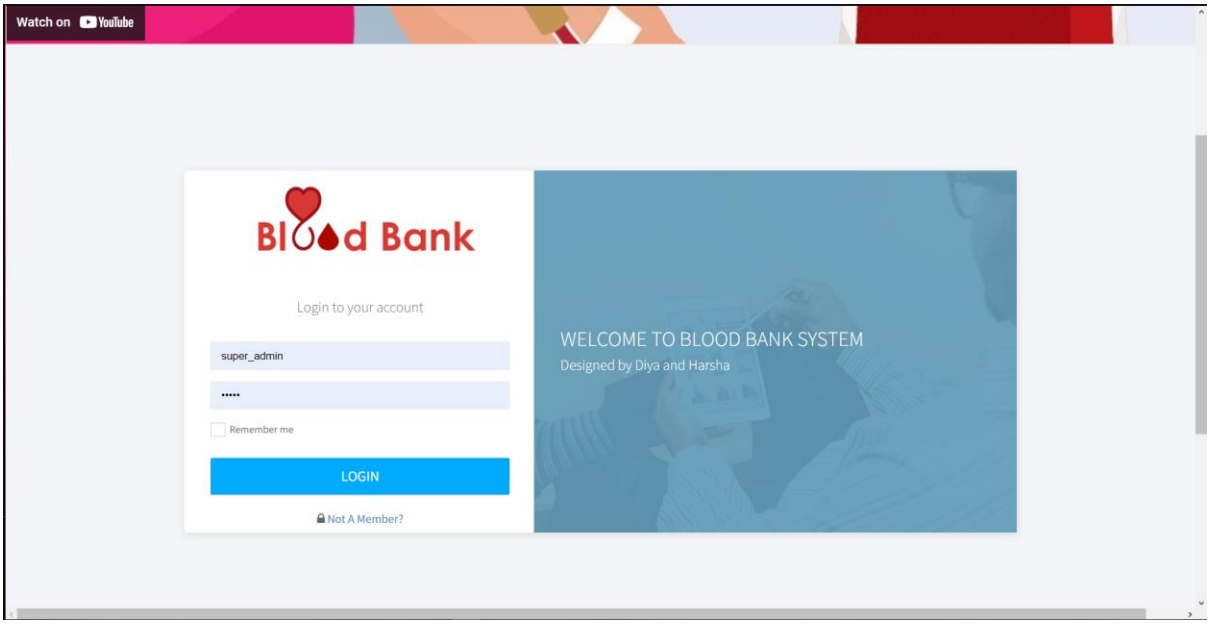
```
DROP TABLE IF EXISTS BloodSpecimen;
create table BloodSpecimen(SpecimenID int NOT NULL,
                           BloodGroup varchar(10) NOT NULL,
                           BloodQuantity float ,
                           HospitalID int NOT NULL,
                           FOREIGN KEY(HospitalID) REFERENCES
Hospital(HospitalID),
                           primary key(SpecimenID,BloodGroup,HospitalID));
```

Recipient Table:-

```
DROP TABLE IF EXISTS Recipeint;
create table Recipient(RecipientID int NOT NULL PRIMARY KEY,
                    ReciName varchar(100) NOT NULL,
                    ReciAge int ,
                    ReciGender varchar(10),
                    ReciBloodGrp varchar(100) ,
                    ReciBloodQuantity float,
                    ReciPhone bigint,
                    ReciDate date,
                    CityID int NOT NULL,
                    FOREIGN KEY (CityID) REFERENCES City(CityID));
```

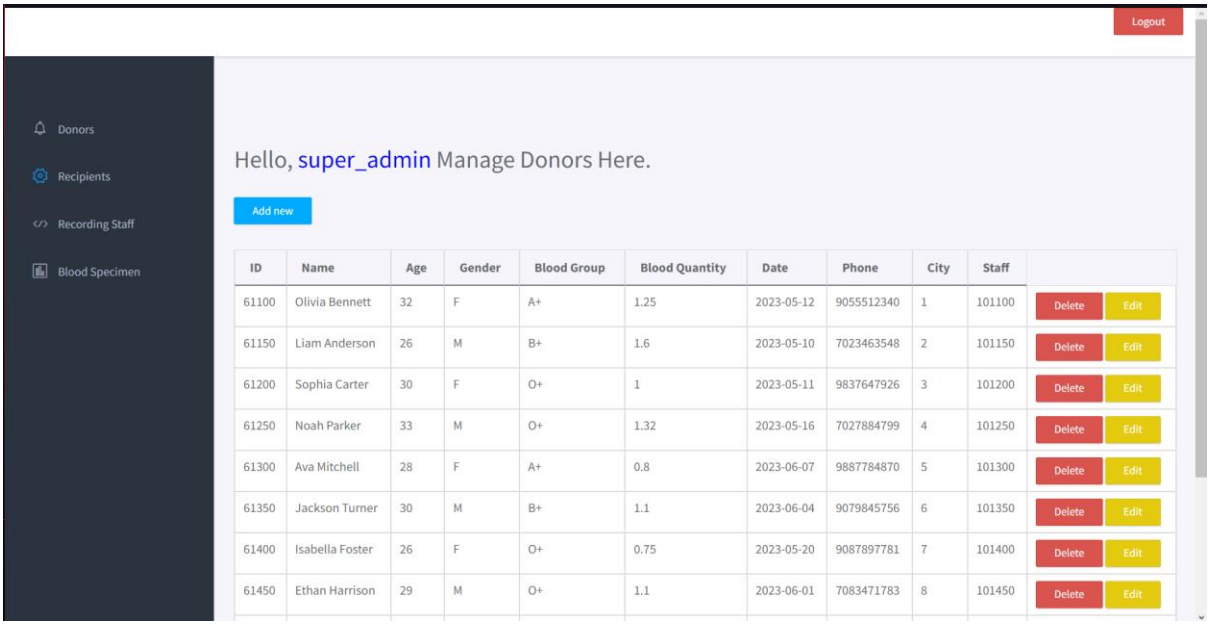
Functionalities And Screenshots

➔ Login Page

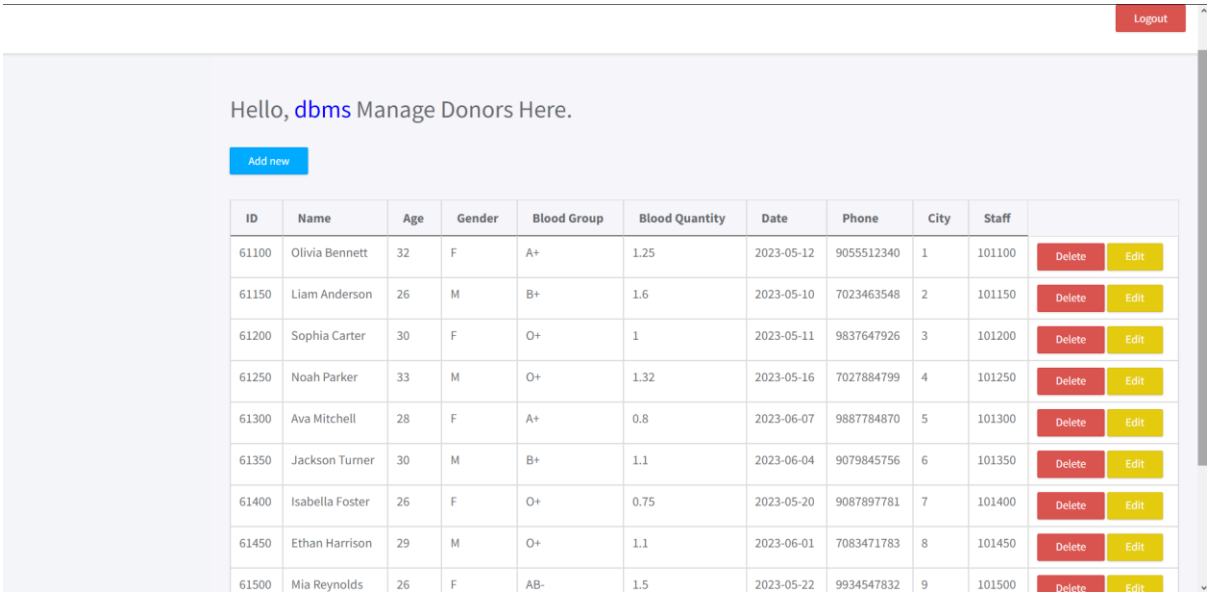


This login page verifies credentials based on the member table in the database and allows access as well as their respective *user roles* .

➔ Manager View



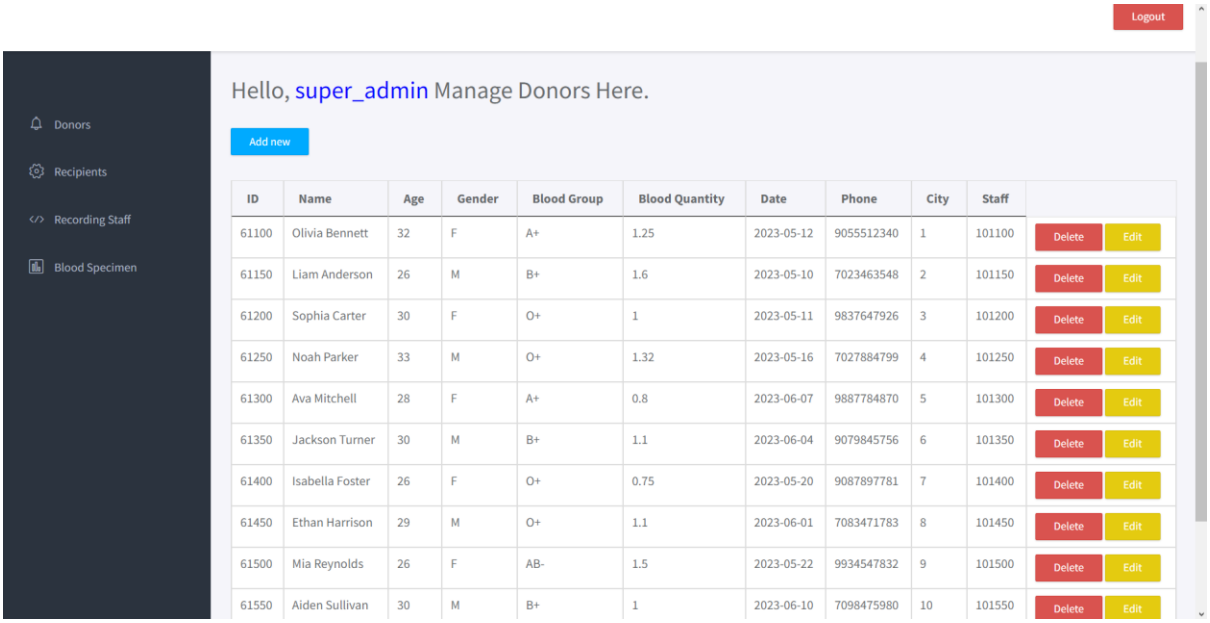
➔ Staff View



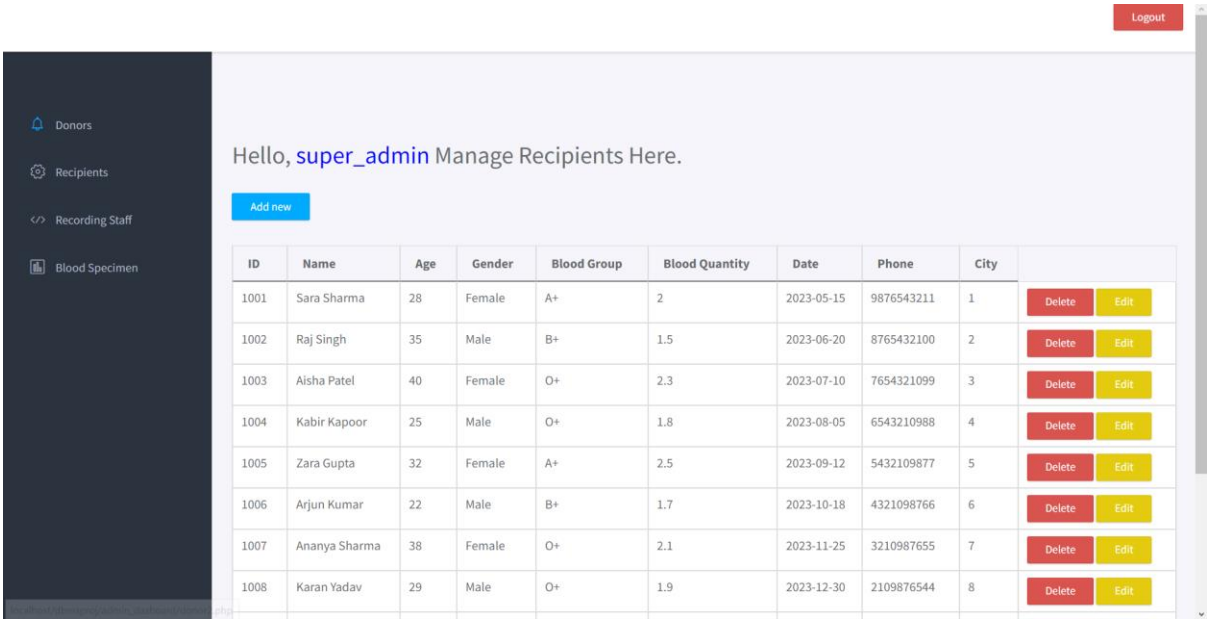
These two views are created with the help of the member table as well as the user roles concept in sql . This way the website maintains the hierarchy through the bloodbank.

➔ The Tables

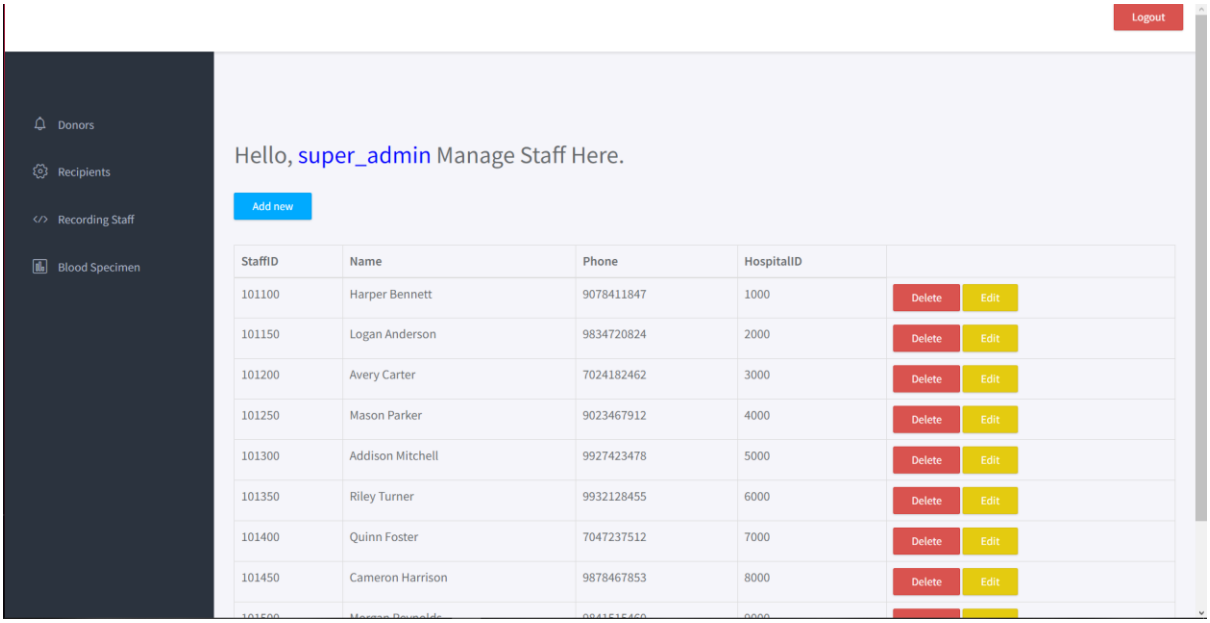
Donor



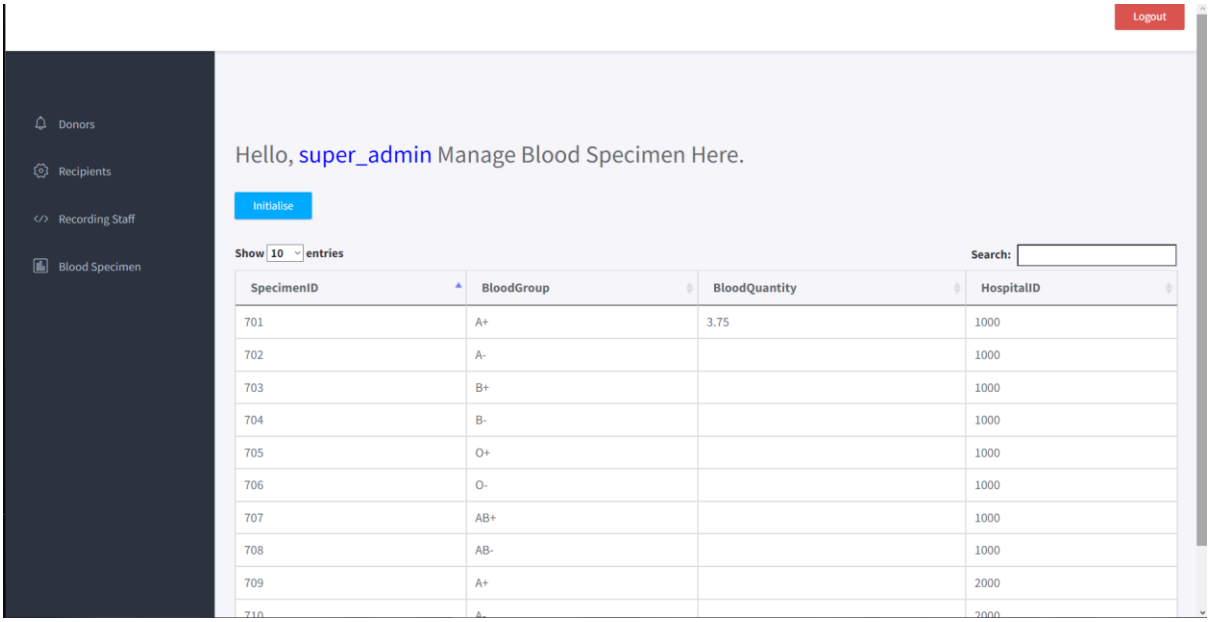
Recipient



Recording Staff



Bloodspecimen



➔ Initialization code snippets

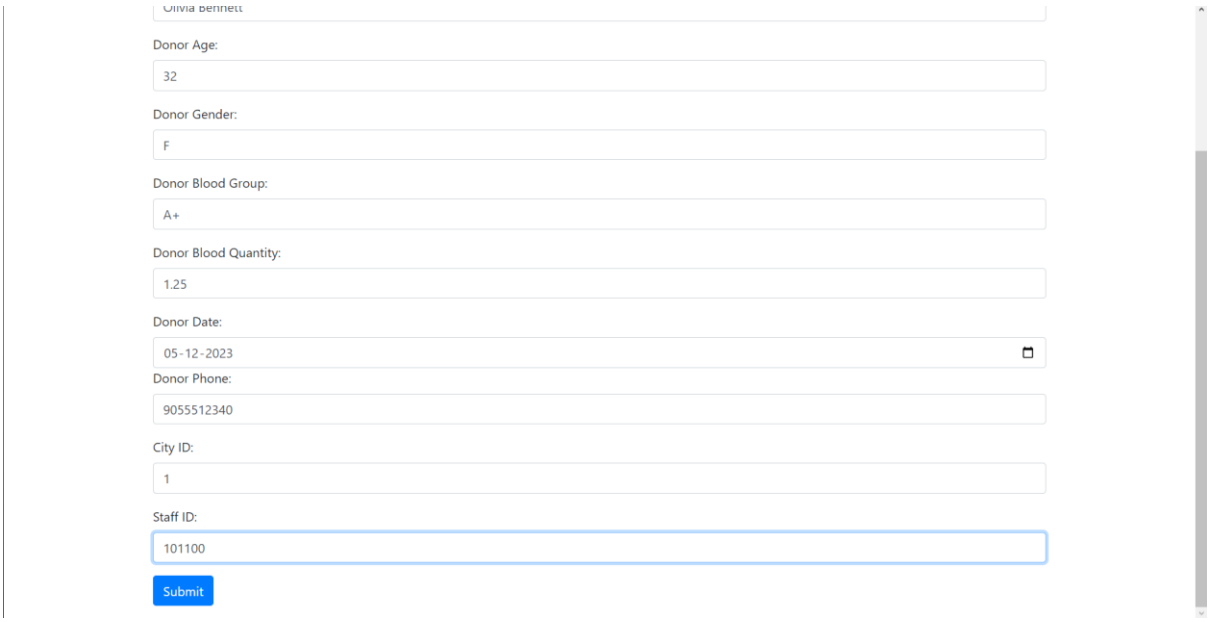
```
➔ INSERT INTO member values
➔             (1,'admin','super_admin','admin',NULL,NULL,NULL,'admin'),
➔             (2,'diya','diyamaria','trial',NULL,NULL,NULL,'staff');
➔
```

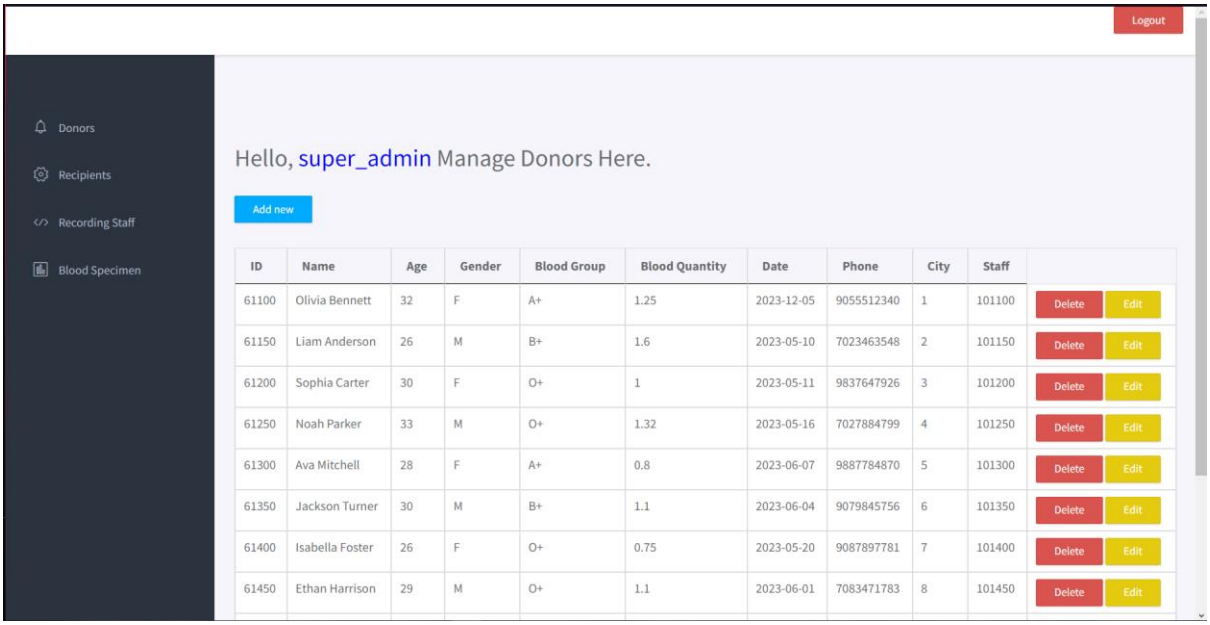
```
INSERT INTO City values
               (001,'Delhi'),
               (002,'Mumbai'),
               (003,'Kochi'),
               (004,'Chennai'),
               (005,'Pune'),
               (006,'Jaipur'),
               (007,'Kolkata'),
               (008,'Hyderabad'),
               (009,'Ahmedabad'),
               (010,'Bangalore');
INSERT INTO Hospital values
               (1000,'Grace Mercy General Hospital',001),
               (2000,'Hope Springs Medical Center',002),
               (3000,'Sunrise Health and Wellness Hospital',003),
               (4000,'Harmony Care Hospital',004),
               (5000,'Evergreen Community Medical Center',005),
               (6000,'Serenity Memorial Hospital',006),
               (7000,'Beacon Hill Medical Center',007),
               (8000,'Unity Regional Hospital',008),
               (9000,'Oasis Medical Plaza',009),
               (10000,'Tranquil Grove Healthcare Center',010);
INSERT INTO RecordingStaff values
               (101100,'Harper Bennett',9078411847,1000),
               (101150,'Logan Anderson',9834720824,2000),
```

```
                (101200,'Avery Carter',7024182462,3000),
                (101250,'Mason Parker',9023467912,4000),
                (101300,'Addison Mitchell',9927423478,5000),
                (101350,'Riley Turner',9932128455,6000),
                (101400,'Quinn Foster',7047237512,7000),
                (101450,'Cameron Harrison',9878467853,8000),
                (101500,'Morgan Reynolds',9841515460,9000),
                (101550,'Jordan Sullivan',9024792313,10000);
INSERT INTO Manager values
                (1100,'Alice Johnson',5551234,1000),
                (1150,'Bob Smith',5555678,2000),
                (1200,'Carol Davis',5559876,3000),
                (1250,'David Miller',5554321,4000),
                (1300,'Emily Wilson',5558765,5000),
                (1350,'Frank Taylor',5552345,6000),
                (1400,'Grace Anderson',5556789,7000),
                (1450,'Henry Parker',5553456,8000),
                (1500,'Isabel Brown',5557890,9000),
                (1550,'Jack Harris',5556543,10000);
INSERT INTO Donor values
                (61100,'Olivia Bennett','32','F','A+',1.25,'2023-05-
12',9055512340,1,101100),
                (61150,'Liam Anderson','26','M','B+',1.6,'2023-05-
10',7023463548,2,101150),
                (61200,'Sophia Carter','30','F','O+',1.0,'2023-05-
11',9837647926,3,101200),
                (61250,'Noah Parker','33','M','O+',1.32,'2023-05-
16',7027884799,4,101250),
                (61300,'Ava Mitchell','28','F','A+',0.8,'2023-06-
07',9887784870,5,101300),
                (61350,'Jackson Turner','30','M','B+',1.1,'2023-06-
04',9079845756,6,101350),
                (61400,'Isabella Foster','26','F','O+',0.75,'2023-05-
20',9087897781,7,101400),
                (61450,'Ethan Harrison','29','M','O+',1.1,'2023-06-
01',7083471783,8,101450),
                (61500,'Mia Reynolds','26','F','AB-',1.5,'2023-05-
22',9934547832,9,101500),
                (61550,'Aiden Sullivan','30','M','B+',1.0,'2023-06-
10',7098475980,10,101550);
INSERT INTO Recipient VALUES
                (1001, 'Sara Sharma', 28, 'Female', 'A+', 2.0, 9876543211,
'2023-05-15', 1),
                (1002, 'Raj Singh', 35, 'Male', 'B+', 1.5, 8765432100, '2023-
06-20', 2),
                (1003, 'Aisha Patel', 40, 'Female', 'O+', 2.3, 7654321099,
'2023-07-10', 3),
```

```
                (1004, 'Kabir Kapoor', 25, 'Male', 'O+', 1.8, 6543210988,
'2023-08-05', 4),
                (1005, 'Zara Gupta', 32, 'Female', 'A+', 2.5, 5432109877,
'2023-09-12', 5),
                (1006, 'Arjun Kumar', 22, 'Male', 'B+', 1.7, 4321098766,
'2023-10-18', 6),
                (1007, 'Ananya Sharma', 38, 'Female', 'O+', 2.1, 3210987655,
'2023-11-25', 7),
                (1008, 'Karan Yadav', 29, 'Male', 'O+', 1.9, 2109876544,
'2023-12-30', 8),
                (1009, 'Pooja Verma', 33, 'Female', 'AB-', 1.6, 1098765433,
'2024-01-14', 9),
                (1010, 'Rohan Kumar', 27, 'Male', 'B+', 2.2, 9876543210,
'2024-02-22', 10);
```

➔ Insertion



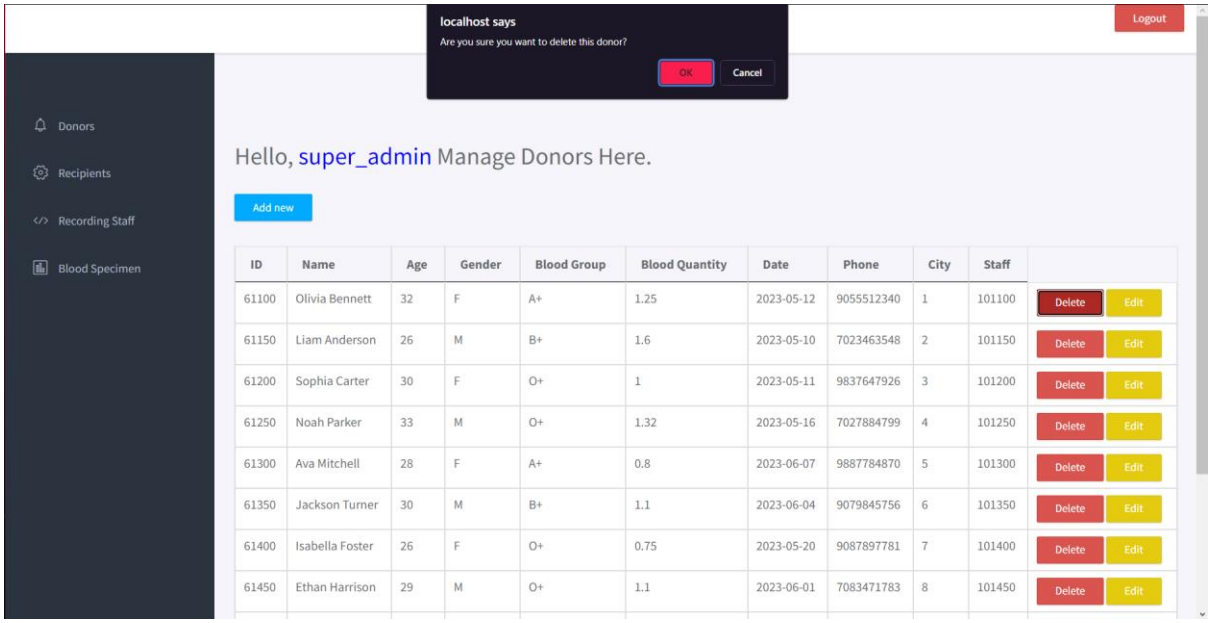


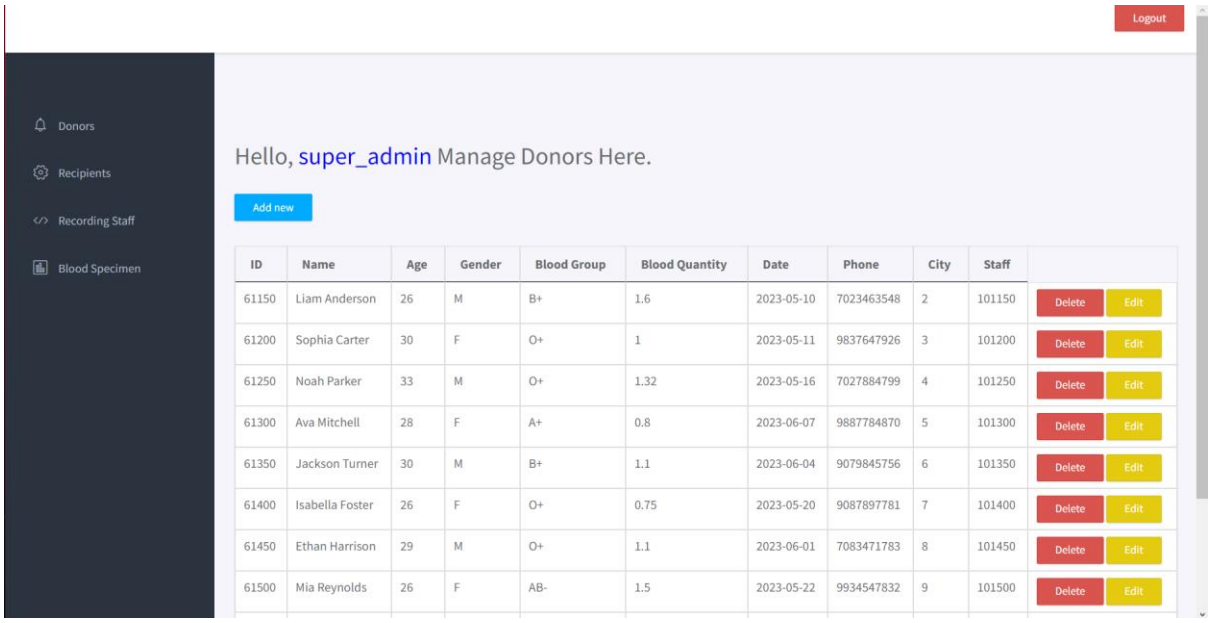
Here , we can see the insertion operation being used . This operation is been shown for donor in the above screenshots but we can also add entries through the front end for the recipients as well the . The staff user can only access the donor table , while the manager can access all tables .


Add Recipient
Recipient ID:
Recipient Name:
Recipient Age:
Recipient Gender:
Recipient Blood Group:
Recipient Blood Quantity:
Recipient Date:
dd-mm-yyyy
Recipient Phone:


Add Recording Staff
Staff ID:
Staff Name:
Staff Phone:
Hospital ID:
Submit

➔ Deletion

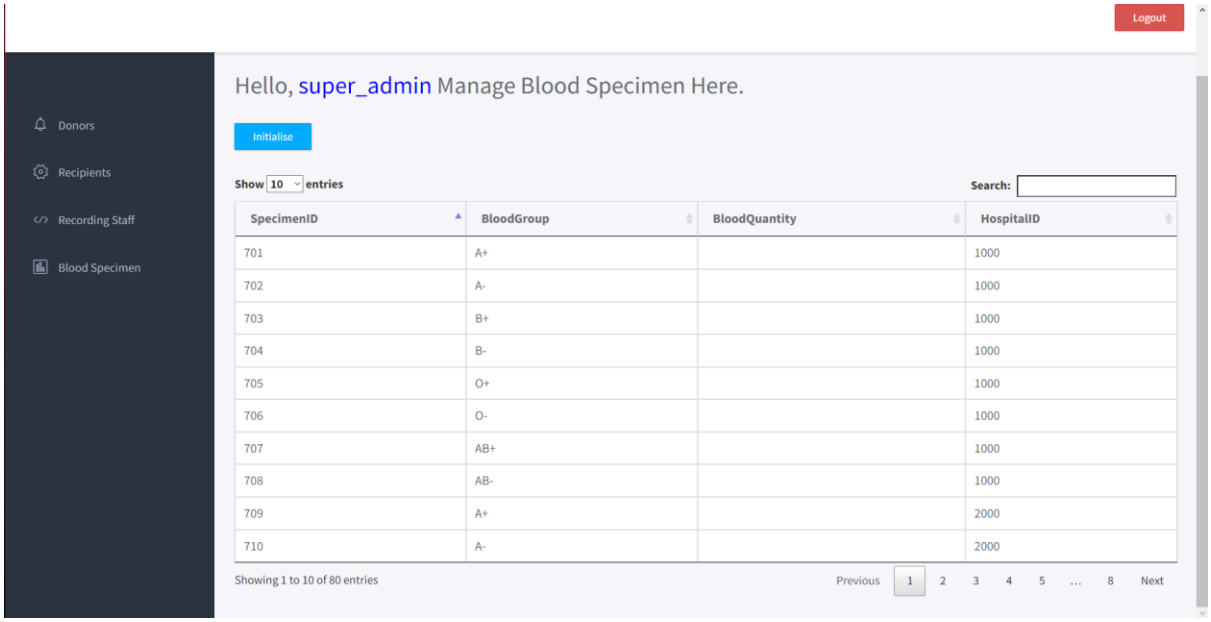




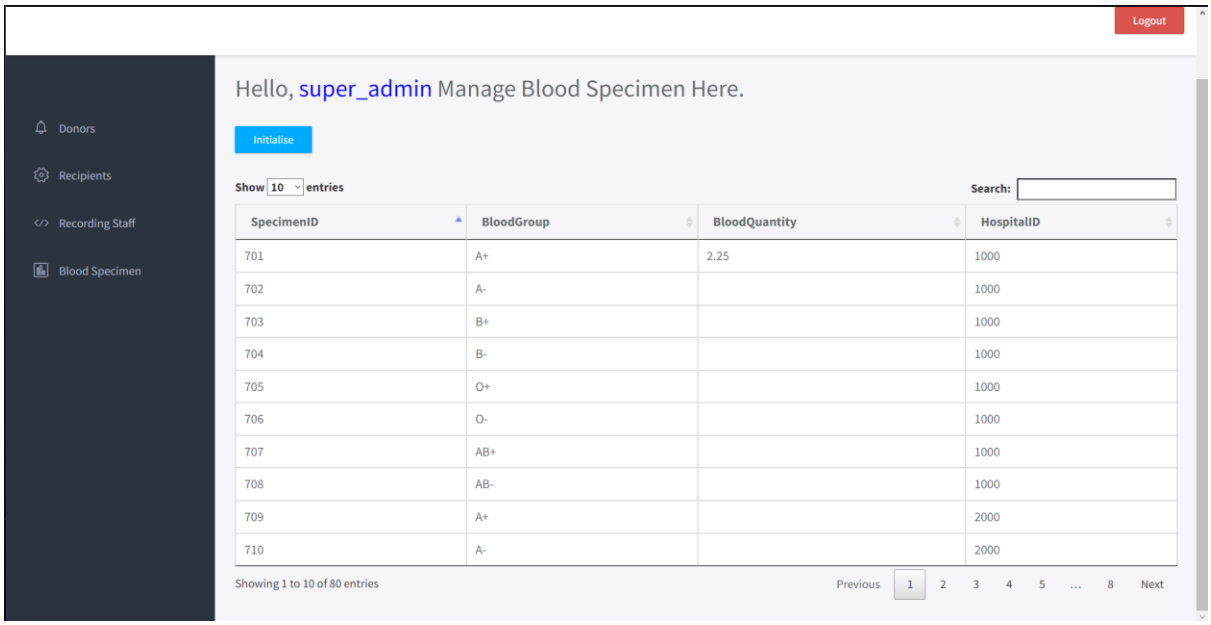
➔ Initialize

This functionality initializes the bloodspecimen table according to the respective blood quantities from the donor and recipient table.

The function uses update , join , nested queries , aggregate function (SUM) as well as procedures



Notice how the blood quantity attribute for this table is empty at the moment
After we click on the initialize button it update on the procedure call



We can see the first row being updated with the sum of blood quantities based from the donor table , mentioned above.

Code snippet:

```
DELIMITER $$
CREATE PROCEDURE update_on_bloodspecimen()
```

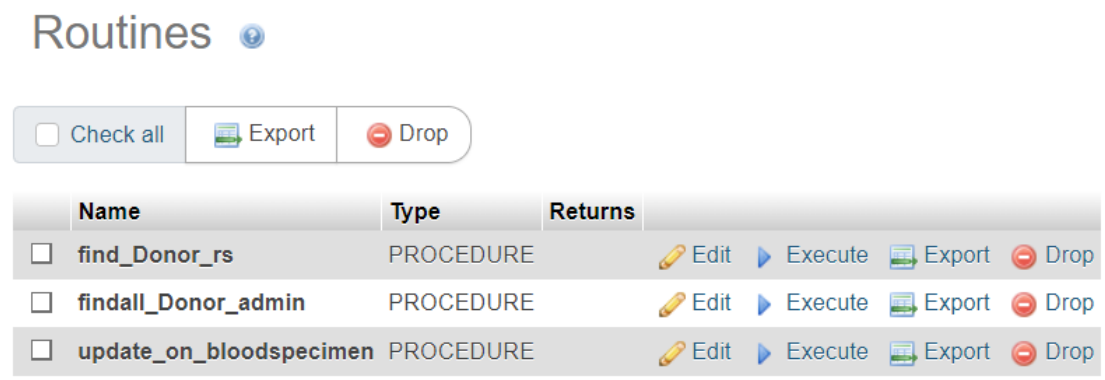
```
BEGIN
  UPDATE BloodSpecimen BS
  SET BS.BloodQuantity = (
     SELECT SUM(D.DonorBloodQuantity)
     FROM Donor D
     INNER JOIN RecordingStaff RS ON D.StaffID = RS.StaffID
     WHERE D.DonorBloodGroup = BS.BloodGroup AND RS.HospitalID = BS.HospitalID);
END $$
DELIMITER  ;
```

➔ Other procedures and their code snippets

```
DELIMITER $$
CREATE PROCEDURE find_Donor_rs(IN id INT)
BEGIN
  SELECT
DonorName,DonorAge,DonorGender,DonorBloodGroup,DonorBloodQuantity,DonorDate,Do
norPhone
  FROM Donor
  where StaffID=id;
END $$
DELIMITER;
```

```
DELIMITER $$
CREATE PROCEDURE find_Donor_admin(IN id INT)
BEGIN
  SELECT *
  FROM Donor
  WHERE DonorId=id;
END $$
DELIMITER;
```

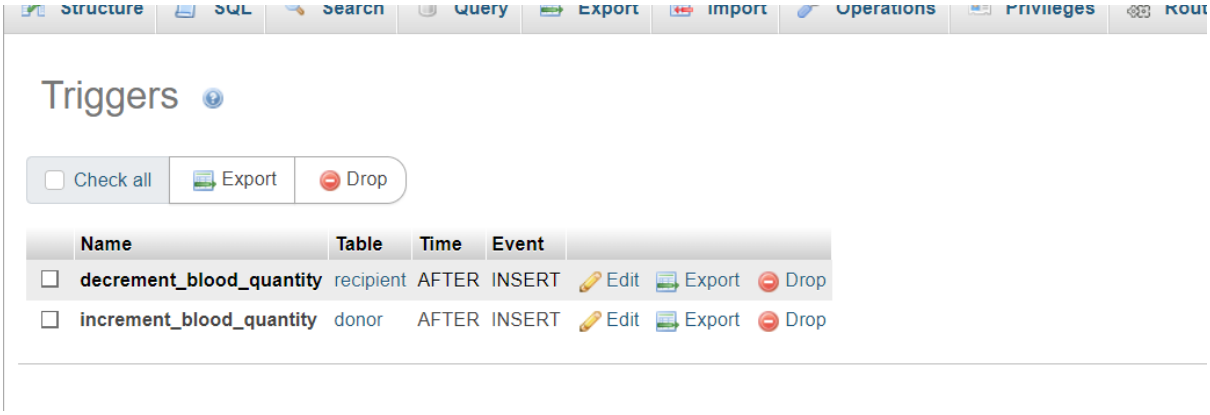
```
DELIMITER $$
CREATE PROCEDURE findall_Donor_admin()
BEGIN
  SELECT *
  FROM Donor;
END $$
DELIMITER;
```



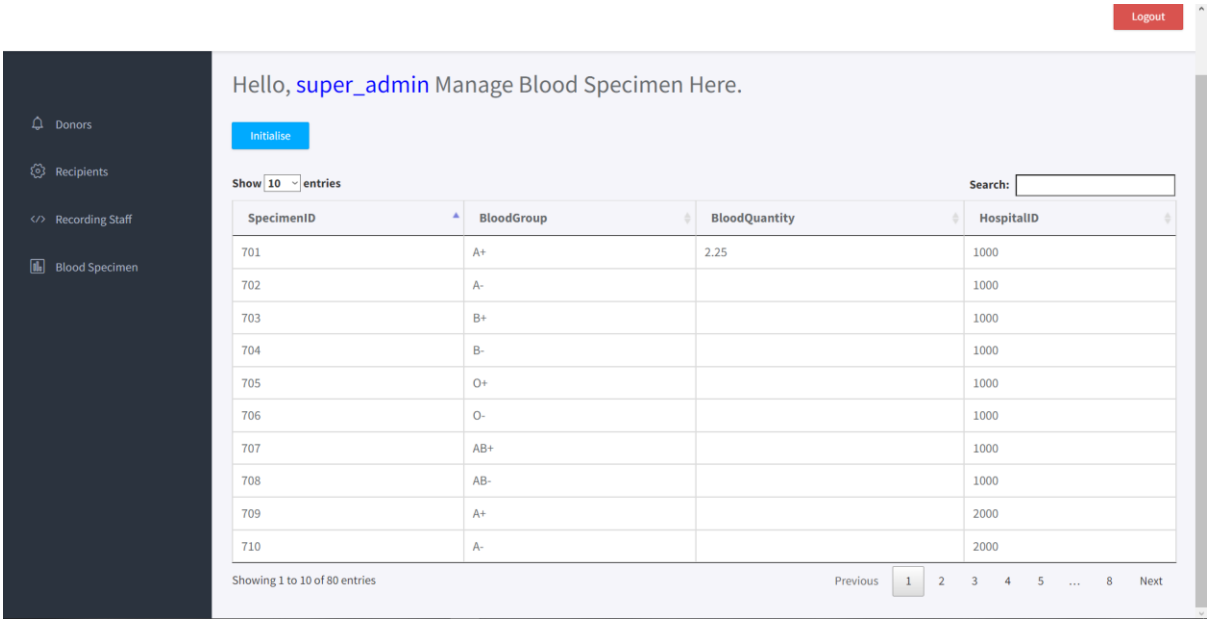
The above procedures are also used to differentiate between the users .

➔ Triggers

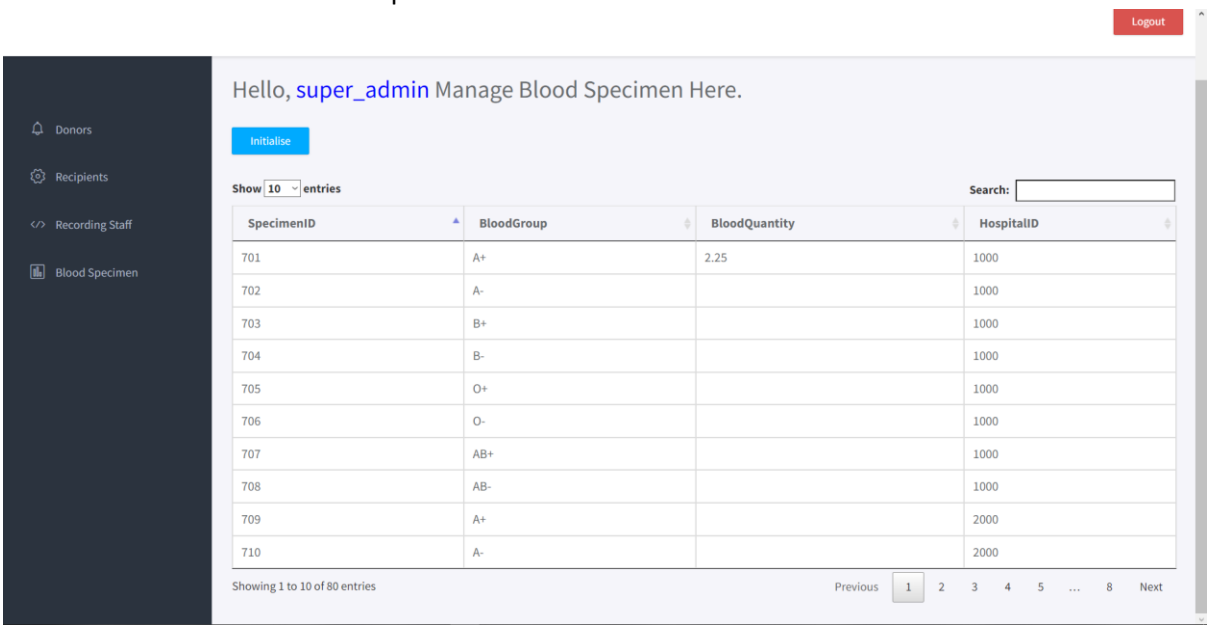
Now we only have triggers left



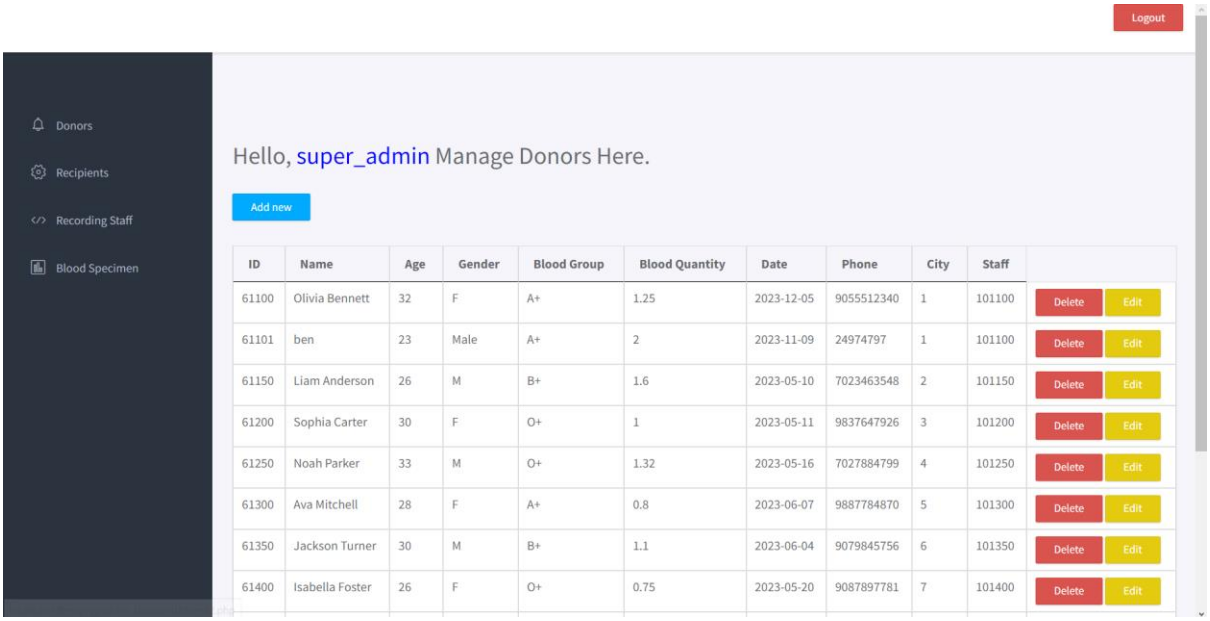
we have two triggers that help with the blood quantity attribute of the bloodspecimen table. These triggers dynamically update the blood quantity on the bloodspecimen table , with every insert to the donor table as well as the recipient table . It increments the quantity on insert into donor and decrements on insert into recipient.



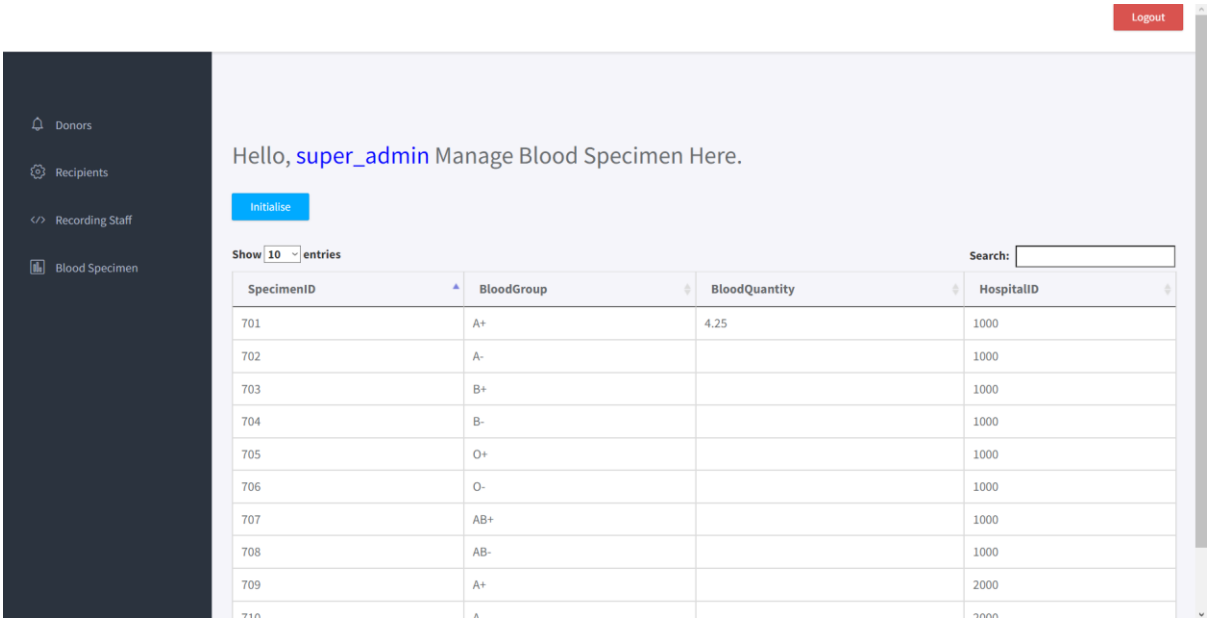
Before insert into donor and recipient



After insert into donor

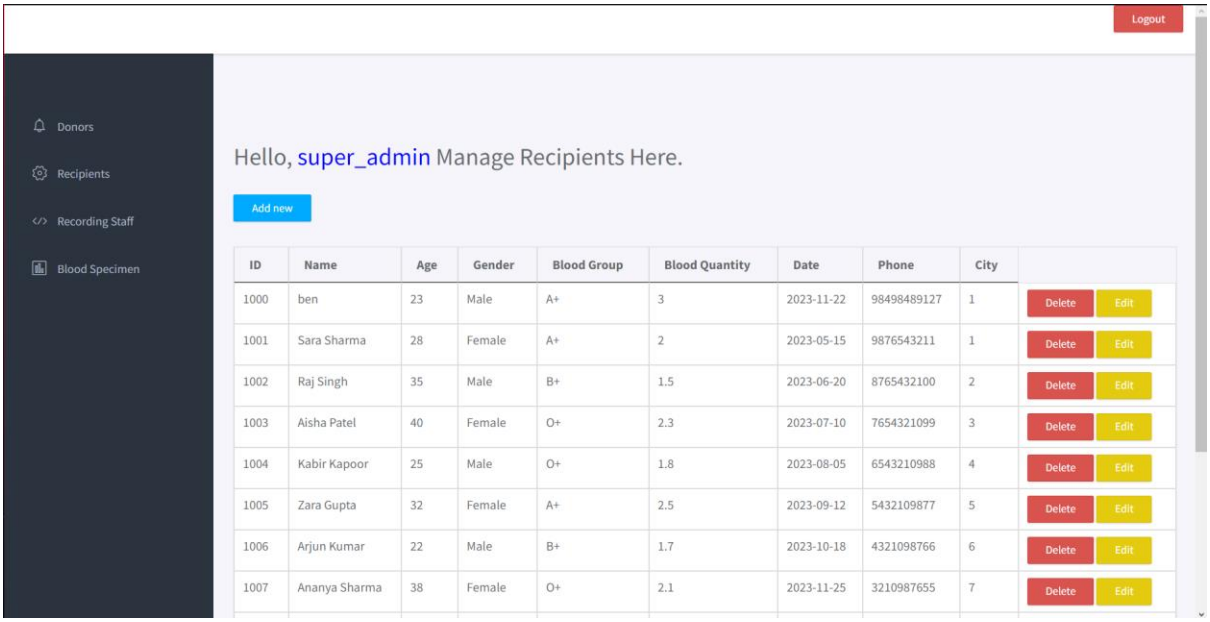


we have inserted “ben”



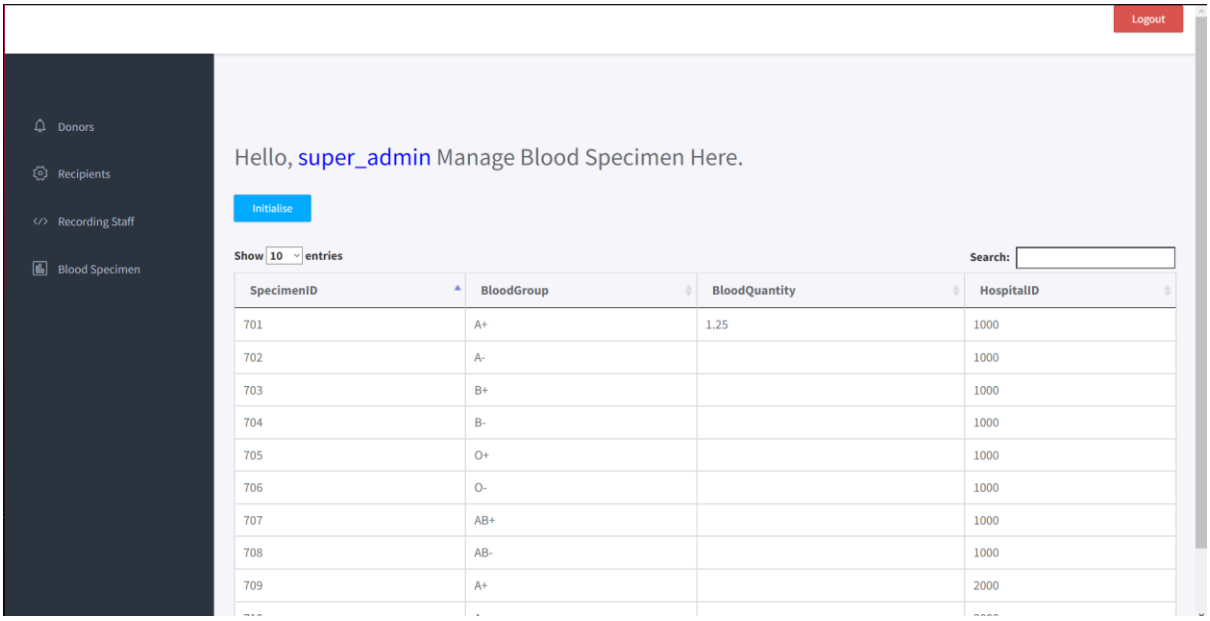
and accordingly the quantity for the A+ blood group for that particular hospital has incremented itself.

Now to decrement it by insert on recipient:



now we have added a “ben” to our recipient table

Let’s check the bloodspecimen table



the quantity has been decreased accordingly

Code snippet

Trigger 1

```
DELIMITER //

CREATE TRIGGER increment_blood_quantity
AFTER INSERT ON Donor
FOR EACH ROW
BEGIN
   UPDATE BloodSpecimen
```

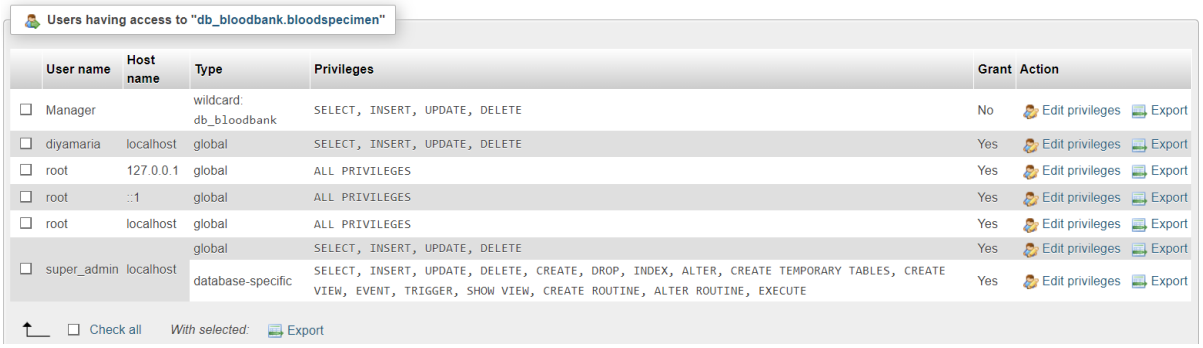
```
    SET BloodQuantity = BloodQuantity + NEW.DonorBloodQuantity
    WHERE HospitalID = (
        SELECT HospitalID
        FROM RecordingStaff
        WHERE StaffID = NEW.StaffID
    ) AND BloodGroup = NEW.DonorBloodGroup;
END //

DELIMITER ;
```

Trigger 2

```
BEGIN
    UPDATE BloodSpecimen BS
    INNER JOIN City C ON BS.BloodGroup = NEW.ReciBloodGrp
    SET BS.BloodQuantity = GREATEST(0, BS.BloodQuantity - NEW.ReciBloodQuantity)
    WHERE C.CityID = NEW.CityID AND BS.BloodGroup = NEW.ReciBloodGrp;
END
```

➔ Users and Roles



Code

```
CREATE ROLE staff;

-- Grant privileges to the Staff Role (only on the Donor table)
GRANT SELECT, INSERT, UPDATE, DELETE ON db_bloodbank.Donor TO staff;

-- Create Manager Role
CREATE ROLE manager;

-- Grant privileges to the Manager Role (on all tables)
GRANT SELECT, INSERT, UPDATE, DELETE ON db_bloodbank.* TO manager;
```