Readme

1. Requirements:

- The application runs on Python and MySQL Workbench which needs to be installed on the local host.
- Setting up a virtual environment for python libraries to be installed is optional.
- Once Python and MySQL Workbench are installed:
    a. execute the data dump. (mysql workbench)
    b. go into the project base directory through terminal/command prompt and run "pip install -r requirements.txt ". (This should install all the required libraries and dependencies.)
    c. The program can now be run using following command:
        - "**python app.py -u** <MySQLusername> **-p** <MySQL password>".
    d. You can alternatively refer to the 'run.txt' file attached as well to make sure that the required dependencies are installed and are up running as per your OS.
- Visit http://127.0.0.1:5000/ on your browser to see the running website.

2. Project Video Link

https://youtu.be/591L9ncY7Zs

3. Technical Specifications

We built a full stack website in order to make this project viable to implement in a real life scenario. Having a user friendly interactive design helps to not only ease the user engagement but could potentially also increase user footfall.
The front end of the website is built on HTML, CSS, and Bootstrap.
The middleware and backend is built on Python and hosted using the Flask framework.
Finally, the entire database is created using MySQL workbench.

3.1 Project Brief

It is estimated that every 2 seconds someone in the United States is in need of blood or platelets which sums up to a daily requirement of almost 36000 units of blood a day and a yearly demand of roughly 16 million blood components. It is safe to say that the entire healthcare system all around the world revolves around a scarce yet renewable resource that is present in significant quantities in every human body, i.e. blood.

"Save a Life" blood bank provides an online repository for citizens to register and donate blood and for patients to receive the required quantity of blood during their need hour.

3.2 Database Description

The database follows a relational data model represented in a columnar format. It has been built in MySQL Workbench. "Save a life" blood bank has on boarded volunteers called administrators in the database to assist new donors and facilitate hospitals for smooth functioning. Each admin has a unique volunteerID, username and a password.

The admin, after validating, assists a donor by registering him/her on the system on a particular date. An admin can register many donors. Each donor is assigned a unique donorID. Basic details of the donor like their name, address, phone, gender, and age are stored. Other medical remarks are recorded and mentioned in the remarks as well. On adding a donor to the system, an insert query is sent through a stored procedure including all the relevant details. As per the system flow, once a donor registers, they donate one unit of blood which is reflected and is added in the blood_bag table.

Each person has a blood group associated with them. A particular blood group can donate to other blood groups based on universal data available and mapped using the blood group receiver entity. It is a weak entity which is described using the donatedTo relationship. A particular blood group can be donated to many other blood groups and vice versa. A person can donate blood of only one blood group but a blood group can have multiple donors. We use this specifically when a patient is in need of blood. The database checks the blood type compatibility and when there are blood bags available of a compatible blood group, a request is generated for the admin to approve. If there are multiple blood bags available, we dispatch the blood bags with the earliest date of use.

When a donor donates blood, it is stored in a blood bag and each blood bag has associated with itself a unique bagID, a dateOfIssue(the date blood had been donated) and a dateOfUse to mark the validity of a blood bag.

An inventory can have many blood bags but a blood bag is stored in a single inventory. Each inventory holds an ID, and has an address(to track and deliver from the nearest available address). While maintaining stocks we also keep track of the current available quantity of blood bags in a particular inventory.

There are a group of hospitals, each being uniquely identified by their hospitalID, have a name, and an address associated with it. Hospitals are directly in touch with the inventories and request a blood bag when needed. A hospital may request for several blood bags based on need from several inventories depending upon availability, proximity, and time needed for the inventories to supply the required blood quantity.

A patient when admitted in a hospital gets registered with a unique patientID, his name, and the reason for admission along with the severity for admission. There can be multiple types of admission like emergency, critical, routine, etc. A person may be admitted for only one reason but there may be multiple patients in the hospital with similar types of admission. Based on this, a hospital will request the blood bank inventory to supply blood bags. Not every patient will be in need of blood, for instance, there may be certain patients who have come for general consulting or routine check up, while for emergency and critical cases, the hospital must contact the inventory requesting for available blood and gather the required quantity. The accident severity

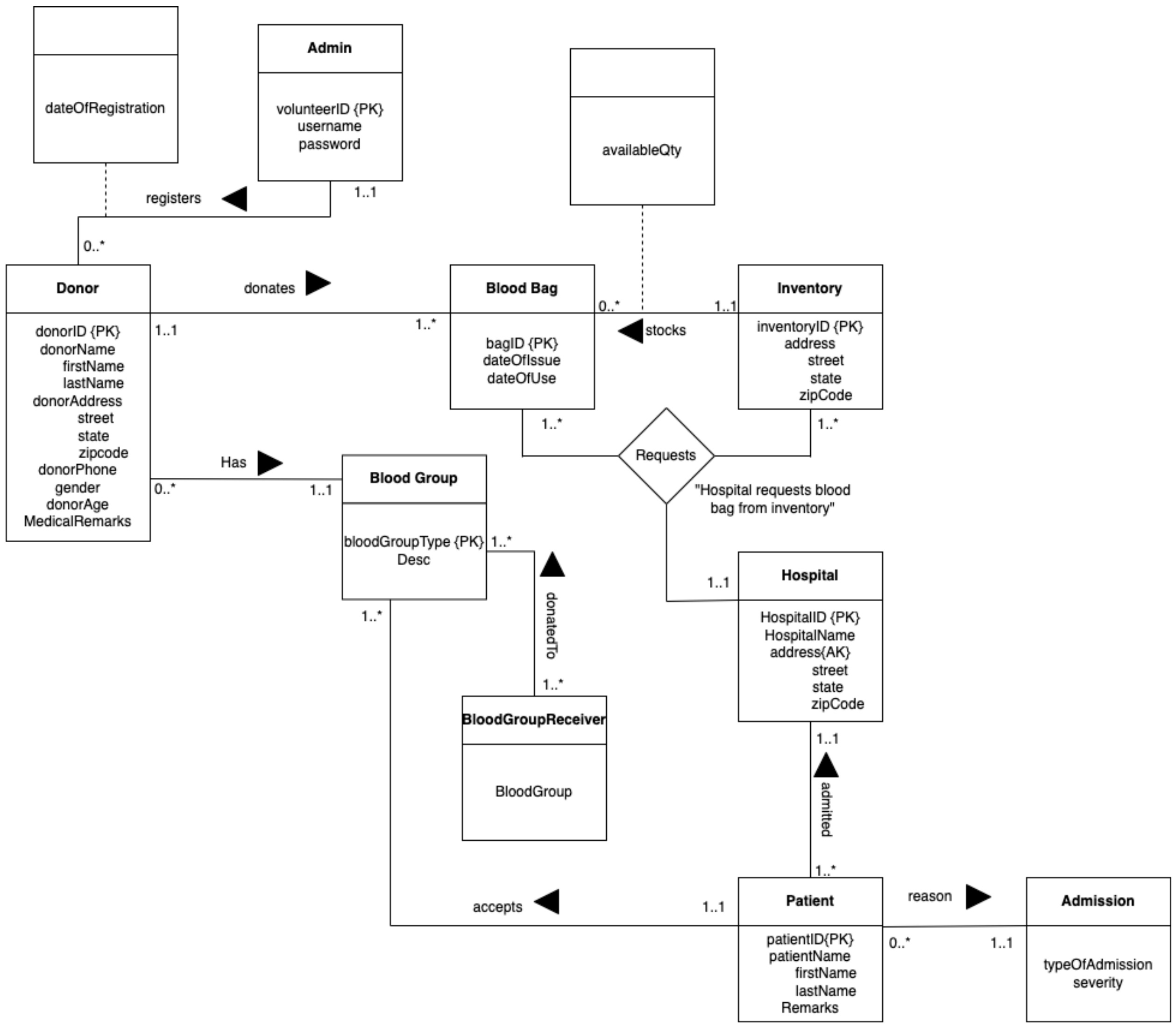
will define the quantity of blood needed for that patient. This has been created using a trigger, if a patient has been admitted to a hospital with severity < 5 denoting emergency cases, a request is generated by the hospital. The inventory records are checked and if compatible blood is available, it is displayed for the admin to approve.
Once it is requested to the admin, the available flag for blood bags is set to false and the available quantity at the inventory is also refreshed. The inventory is refreshed post every update or insert in the blood_bag table.

All the above database operations(CRUD) are offered to the user through an interactive UI. The web page launches through the localhost and establishes a connection with the database. The video link provided with the submission demonstrates few of the operations offered by the program.

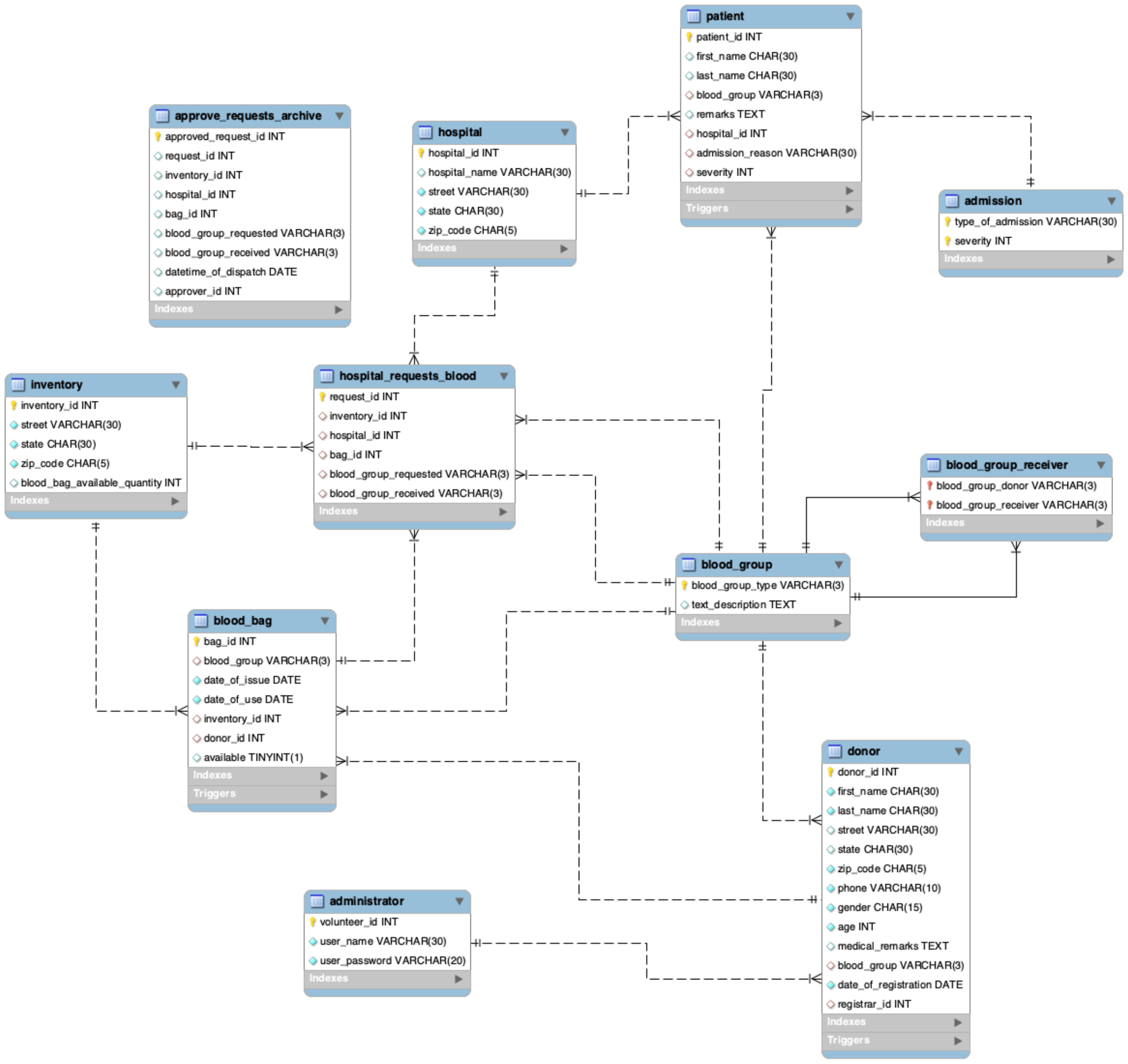
3.3 User Interaction

The donor donates blood bag(s) of a particular blood group to the bank when registering. Blood can be donated later on as well, using the patient’s phone number which is unique to identify. The blood bag can be used by patients eligible to receive that particular blood group. A patient who reports at a hospital and is in need of blood, directly contacts the blood bank’s inventory to request blood and get it delivered in near real time based on availability.

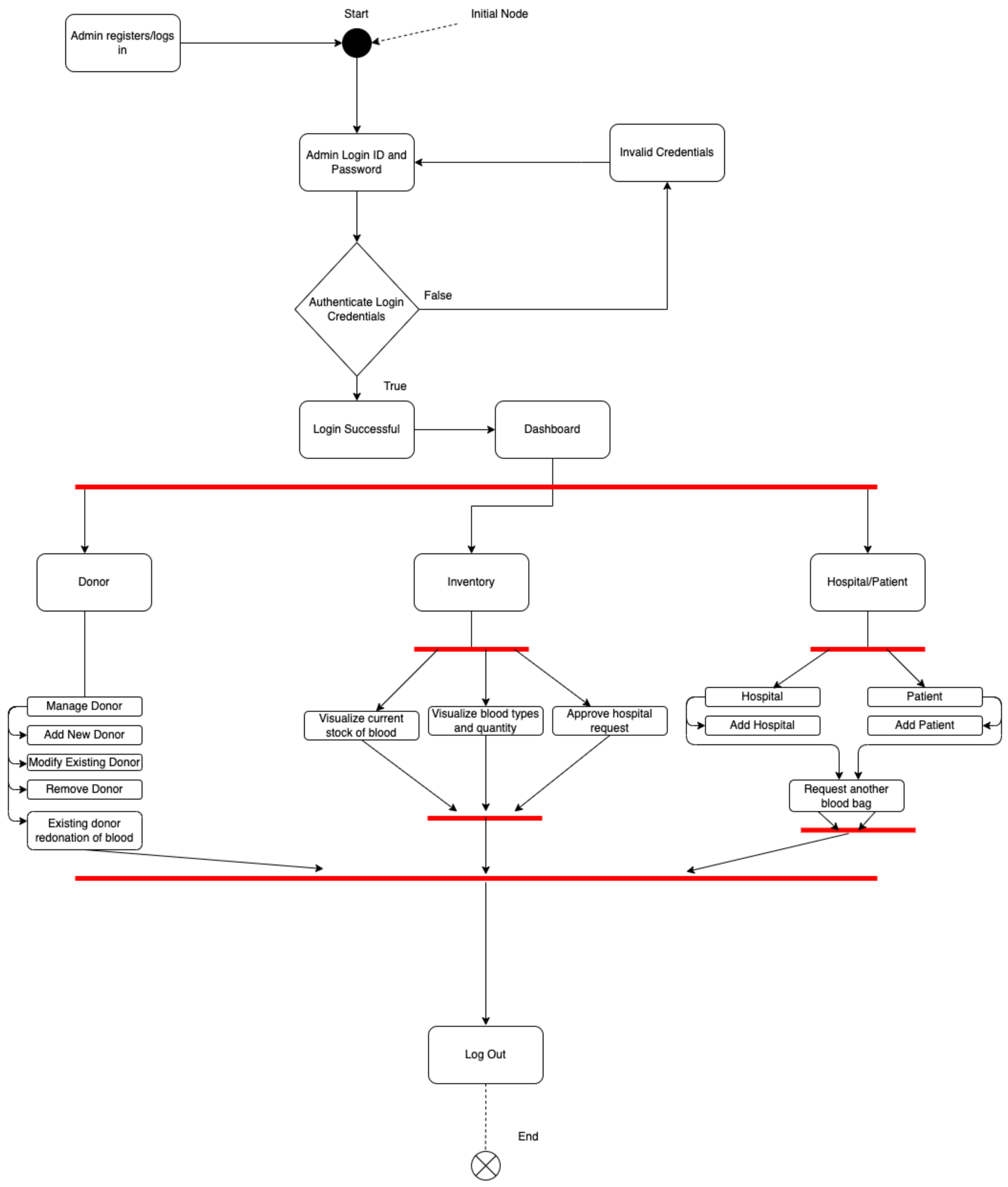
4. Conceptual Design


dateOfRegistration
Admin
volunteerID {PK}
username
password
registers
1..1
0..*
availableQty
Donor
donorID {PK}
donorName
firstName
lastName
donorAddress
street
state
zipcode
donorPhone
gender
donorAge
MedicalRemarks
donates
1..1
1..*
Blood Bag
bagID {PK}
dateOfIssue
dateOfUse
0..*
stocks
1..1
Inventory
inventoryID {PK}
address
street
state
zipCode
1..*
Requests
1..*
"Hospital requests blood bag from inventory"
Has
0..*
1..1
Blood Group
bloodGroupType {PK}
Desc
1..*
donatedTo
1..*
BloodGroupReceiver
BloodGroup
1..*
1..1
Hospital
HospitalID {PK}
HospitalName
address{AK}
street
state
zipCode
1..1
admitted
1..*
accepts
1..1
Patient
patientID{PK}
patientName
firstName
lastName
Remarks
reason
0..*
1..1
Admission
typeOfAdmission
severity

5. Logical Design



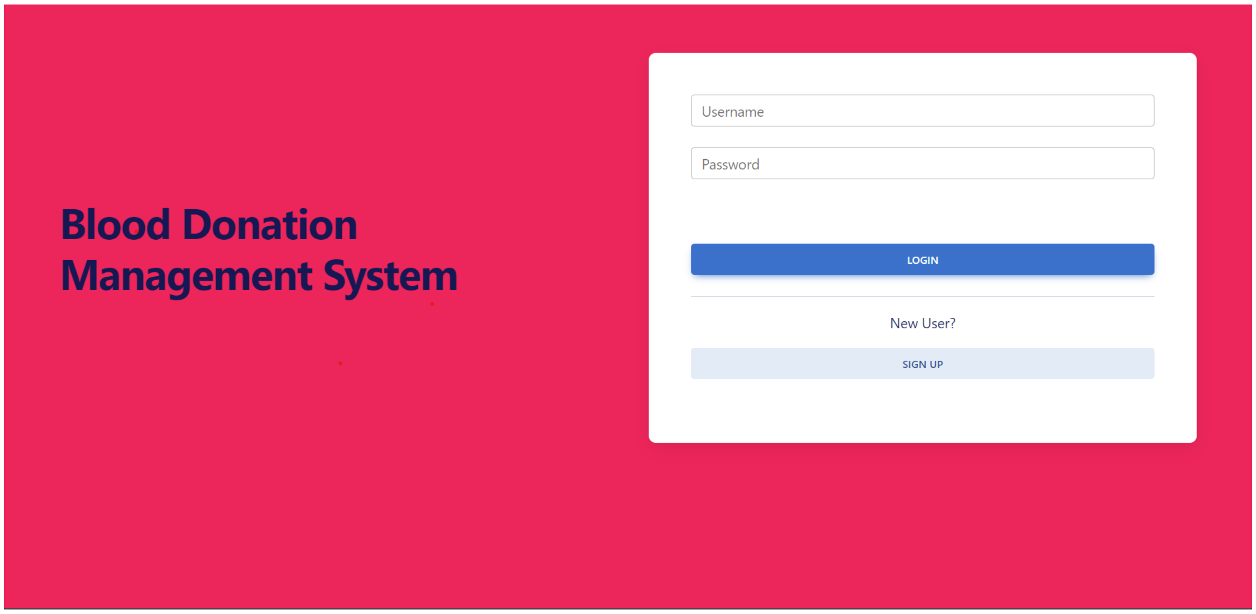
6. User Flow



The user interface provides a navigation guide for all the actions supported by the program. Below is a brief description of the operations which can be performed with each button click. After the hospital's request has been approved by an admin, records are stored in the archive with the blood bag date of dispatch and the admin id who has approved this request. This has been done in order to back track the blood bag for future auditing.

6.1 Login Page

On this page, a new volunteer can be registered or an existing one can login using their credentials. To create a new admin, the user can click on sign up, select their unique username and enter their password. Once registered, they can enter their user credentials and click the login button to view the dashboard.

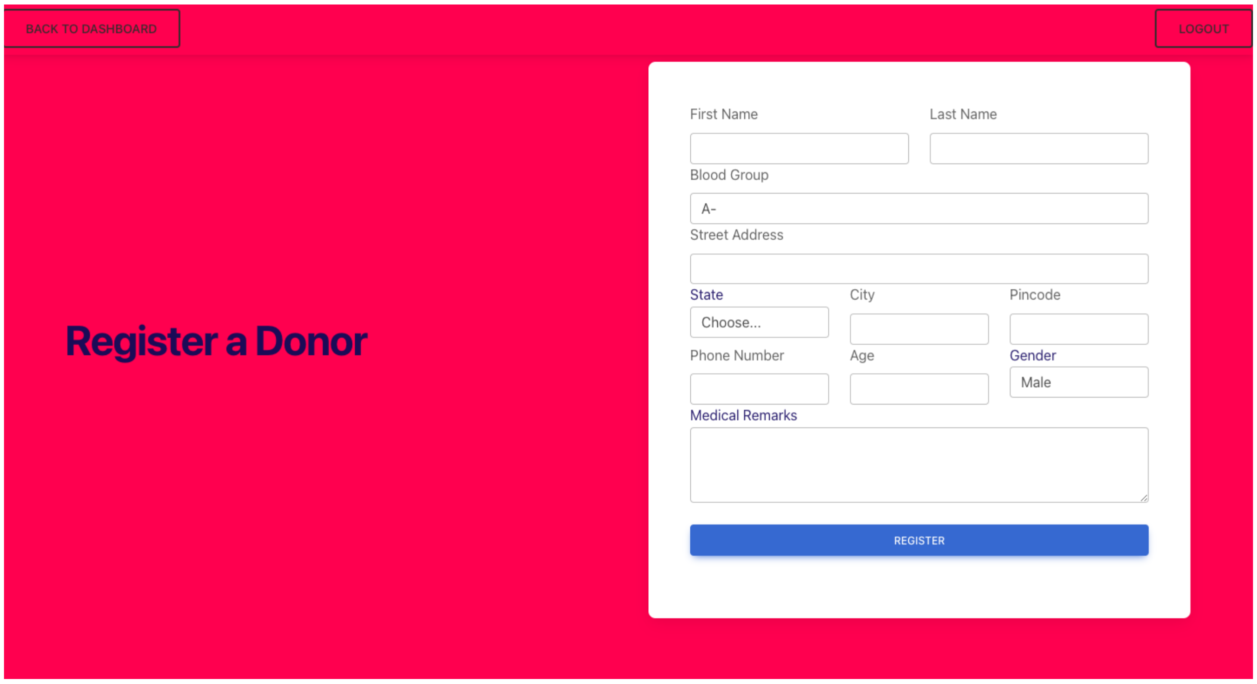


6.2 Dashboard When a valid admin/volunteer logs in

Once the volunteer logs in, the user views the below dashboard. The dashboard is segregated into components as visible below. These include updating and managing the donor, issuing a unit of blood, viewing the current inventory status, and adding a patient/hospital. Click on appropriate buttons to perform the next sequence of operations.

6.3 Add Donor

When the administrator clicks on add a donor from the dashboard, they are navigated to the below screen. The blood group dropdown is populated by reading all blood groups available in the database. The administrator can fill in all the required details and click register to add the user in the database. Once added, the database administrator can verify this using a select query on the donor table. The registration is based on the fact that every user has a unique phone number to register. If there is a phone which already exists in the database, the same user cannot be added and they have to register with a different number. On adding a new donor, the system adds a blood bag denoting that on first-time registration, a donor donates at least one bag of blood. This can be verified by checking the inventory where the count of blood bags for the recently added user's blood group gets incremented by 1. In the blood bag table, one *can also see that this newly added blood bag has its 'is available' value as true. This is further explained in* ***section 6.12.***



First Name

Last Name

Blood Group: A-

Street Address

State: Choose...

City

Pincode

Phone Number

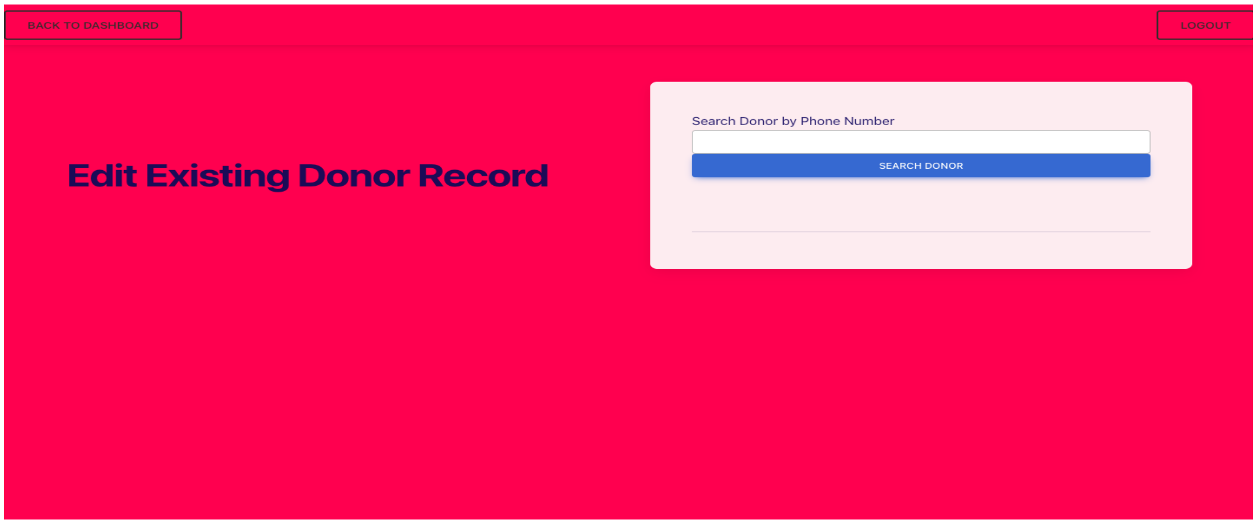
Age

Gender: Male

Medical Remarks

6.4 Modify/edit Donor

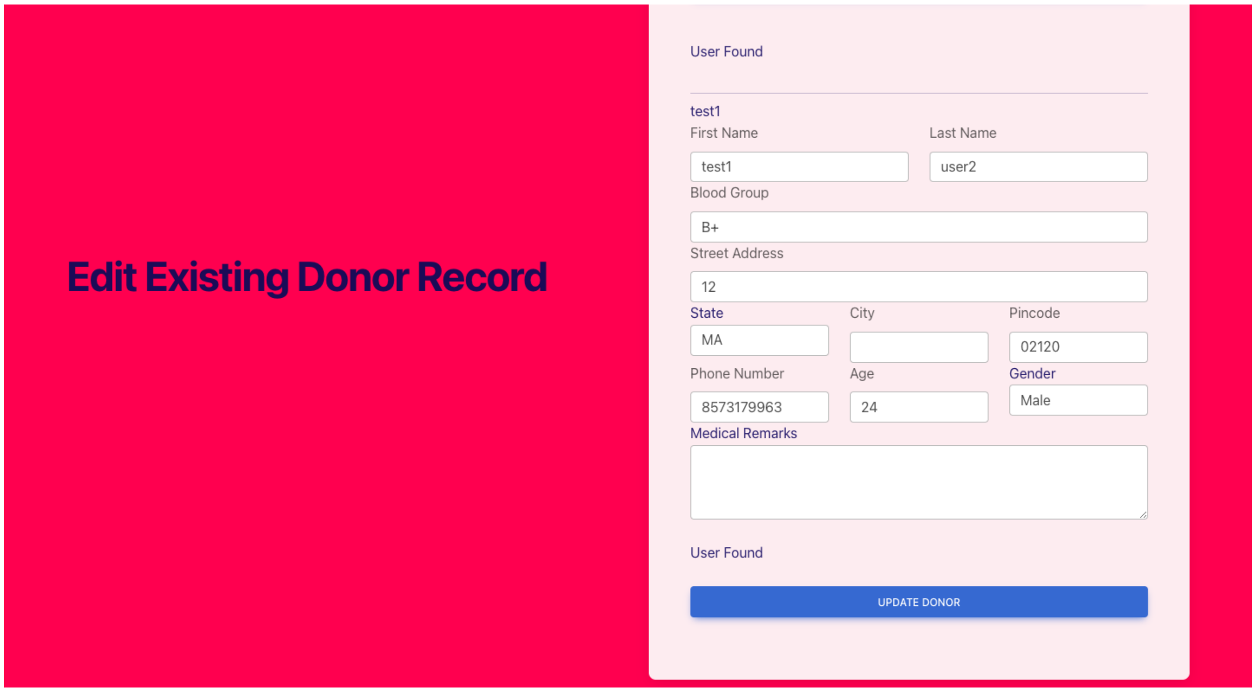
To update details of an existing donor, we first look up our donor database using the unique phone number the user has registered with. Below screenshot, provides an overview of the same. After the phone number has been verified and matched in the database, the admin can view user details populated as seen in screenshot (b). All fields in the form are pre populated with the existing data in the database. The administrator can update only the required fields and submit. Modifications can be verified by running a simple select query in the donor table.



Search Donor by Phone Number

a. Phone number lookup

In order to edit details of an existing donor, we need to search the donor using their phone number. If a donor is found, the page populates the current details of the donor using the 'select_donor_details' and whichever updates are done are updated inside of the donor table as well with the help of the 'update_donor_details' stored procedure.

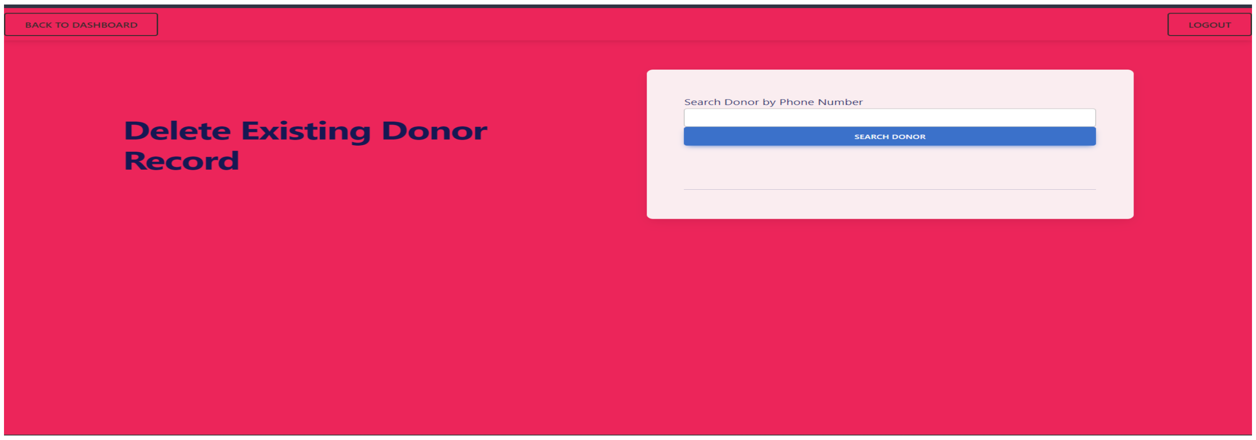


First Name: test1	Last Name: user2	
Blood Group: B+		
Street Address: 12		
State: MA	City:	Pincode: 02120
Phone Number: 8573179963	Age: 24	Gender: Male
Medical Remarks:		

b. User details populated with the matched record.

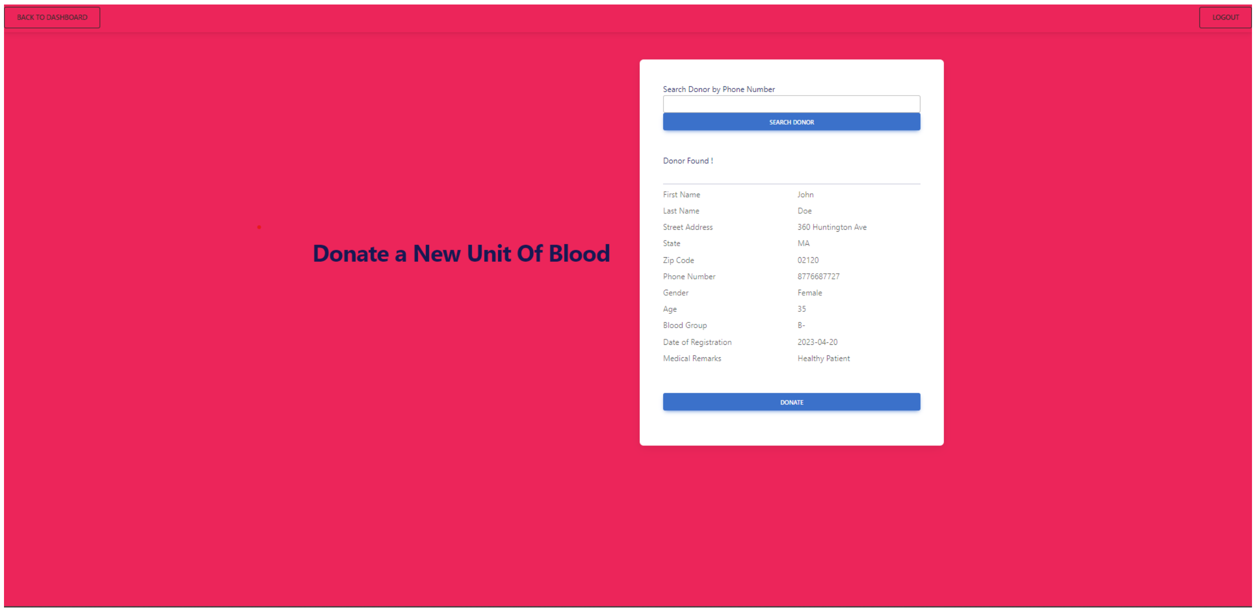
6.5 Delete Donor

This page is used to perform the delete operation of a donor. We use the phone number to search for the donor. If the donor with the entered phone number exists, the 'delete_donor' stored procedure is called and the donor is deleted from the database. This operation can be verified by doing a select query on the donor table.



6.6 Donor Donates Blood:

To donate a unit of blood, the user needs to be first registered on the system with a few basic details. Once a registered Donor enters their phone number, we search the existence of the donor in the donor table. If the donor exists, we display their details and give them an option to 'DONATE'. Once the user clicks on the Donate Button, a stored procedure 'add_blood_bag' is called. Essentially adds a blood bag into the inventory and is linked to the donor. This addition can be verified by checking the inventory from the dashboard. A Select query on the blood_bags table can also be used to verify the addition of a new blood bag.

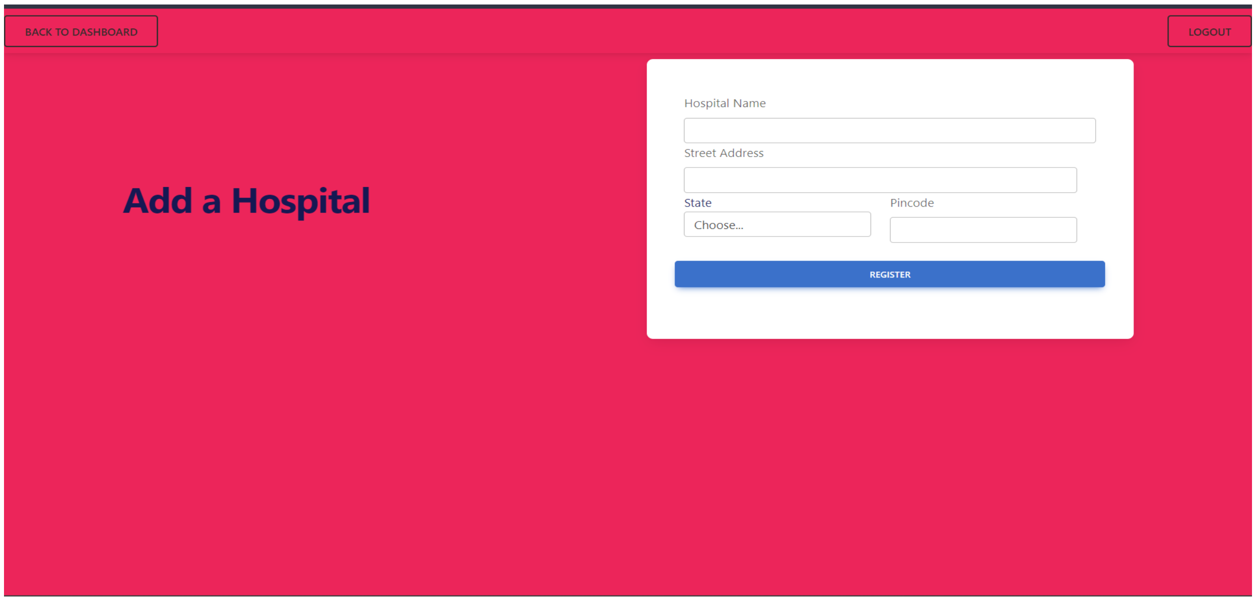


Donor Found !

First Name	John
Last Name	Doe
Street Address	360 Huntington Ave
State	MA
Zip Code	02120
Phone Number	8776687727
Gender	Female
Age	35
Blood Group	B-
Date of Registration	2023-04-20
Medical Remarks	Healthy Patient

6.7 Hospital Registers

The administrator can request the addition of a new hospital to this system. This signifies that a hospital registers with the blood bank to request blood bags from one of its inventories. To add a hospital, we ask the user to fill a simple below form with a unique street address for each of the hospitals. Once a user clicks register, the hospital is added to the 'hospital' table in the database and can be verified by running a simple select query on the same.



Add a Hospital

Hospital Name

Street Address

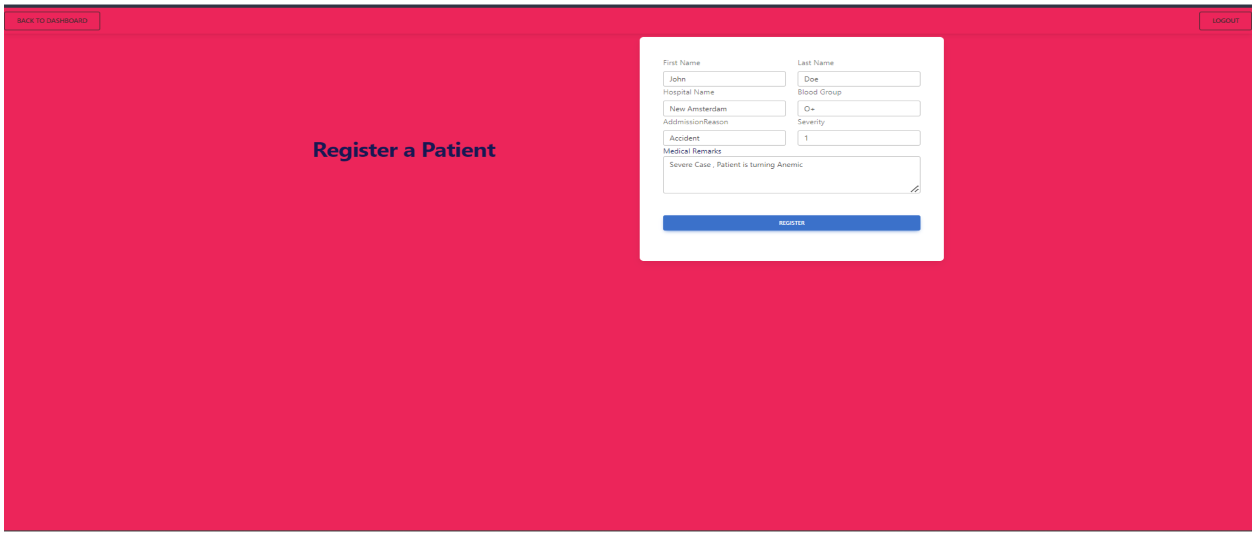
State

Choose...

Pincode

6.8 Patient Registers

The administrator in our system has been granted privilege to add a patient to a hospital. They have to fill in the form below which includes the patient details of the hospital that they are being admitted to along with their blood group, reason for admission and the severity of this reason. Based on the severity the patient is being registered with, the system decides if a unit of blood is needed on a critical basis by the patient. If the severity of the incident is less than 5, the system assumes it to be a critical case and looks up the inventory to check for available units of blood. If there are blood bags of compatible type available, a request is generated and can be viewed in the 'Issue a Unit' section of the dashboard. Section 5.7 gives a brief overview of this. If no blood bags of any compatible type are available in the inventory a message is displayed to the user.



Register a Patient

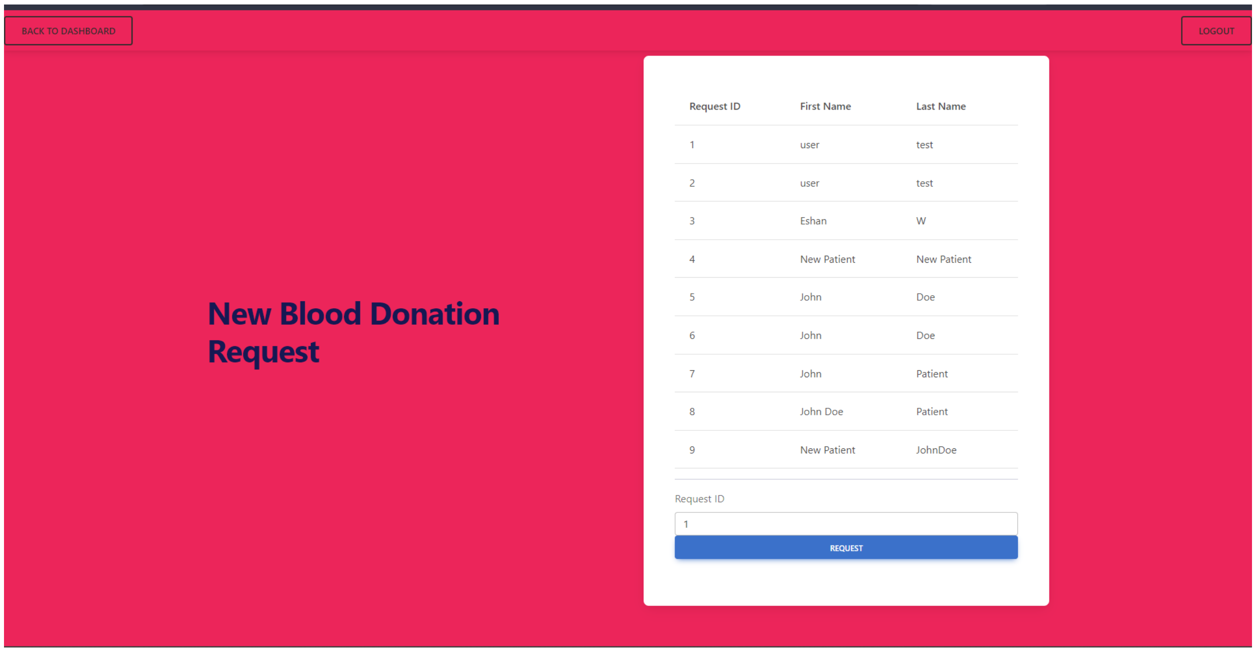
First Name	Last Name
John	Doe
Hospital Name	Blood Group
New Amsterdam	O+
AddmissionReason	Severity
Accident	1

Medical Remarks

Severe Case , Patient is turning Anemic

6.9 Patient requests additional units

This page is used to create additional requests for existing patients. Just like the donate blood webpage, we display all current patients using a select query on the patients table. Using the dropdown menu , the user can choose which patient is requesting the blood unit and click on the request button. The 'add_additional_blood_bag' query is executed on the selected patient ID and a new request is triggered. This request can be seen on the requests page of the application. And can be verified using the select_hospital_requests procedure.

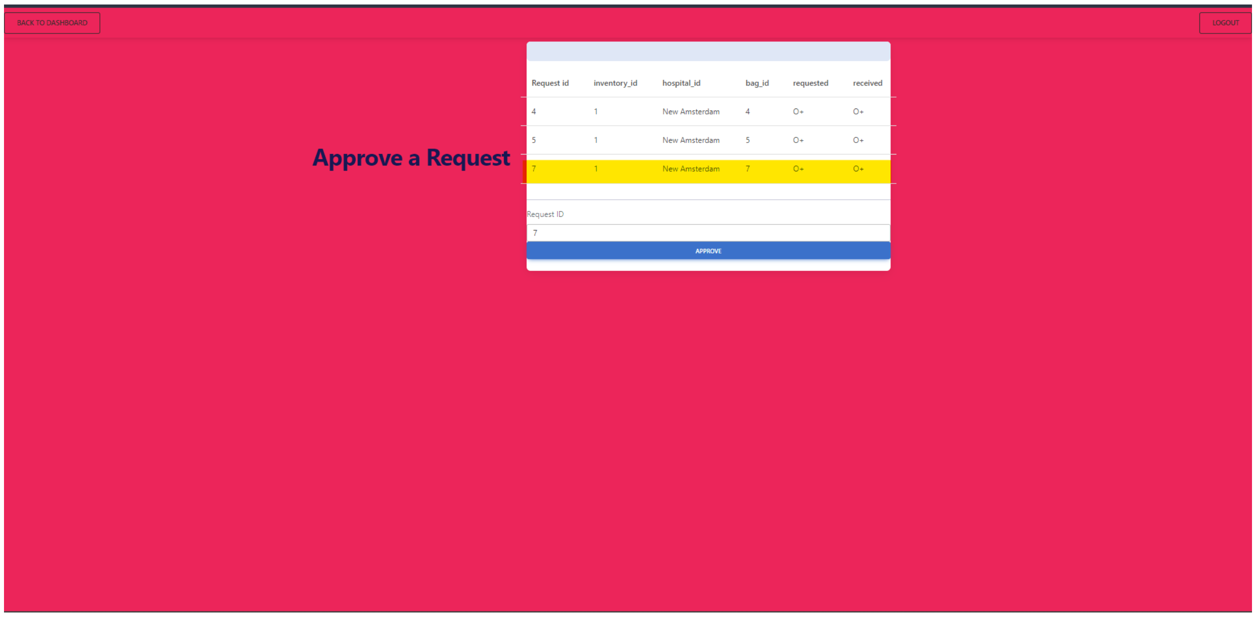


Request ID	First Name	Last Name
1	user	test
2	user	test
3	Eshan	W
4	New Patient	New Patient
5	John	Doe
6	John	Doe
7	John	Patient
8	John Doe	Patient
9	New Patient	JohnDoe

Request ID: 1

6.10 Request is received

Once a Donation request is received, It is displayed on the requests page of the website. The data on this page is generated by calling the select_hospital_requests stored procedure. The user can select the request which needs to be approved by using the dropdown menu available. On clicking the 'Approve' Button, a stored procedure called 'approve_hospital_request' is called. This procedure updates the inventory by setting the 'IS_available' value of the assigned bloodbag to 0.



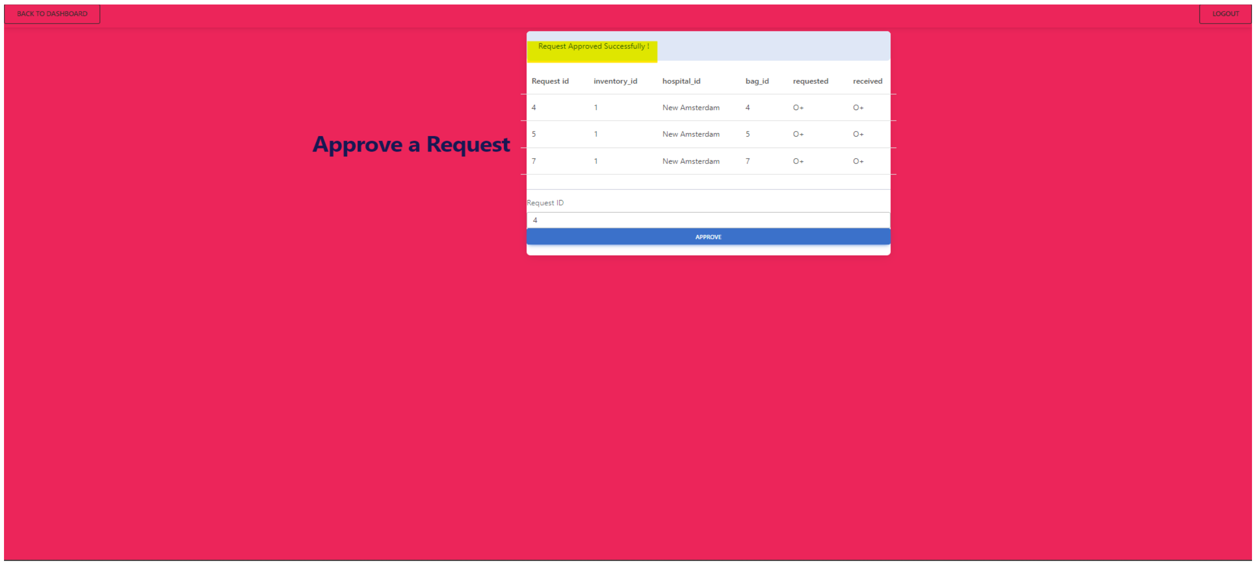
Approve a Request

Request id	inventory_id	hospital_id	bag_id	requested	received
4	1	New Amsterdam	4	O+	O+
5	1	New Amsterdam	5	O+	O+
7	1	New Amsterdam	7	O+	O+

Request ID: 7

6.11 Request Is Approved

A success message is seen when the transaction has been approved.



Request Approved Successfully !

Approve a Request

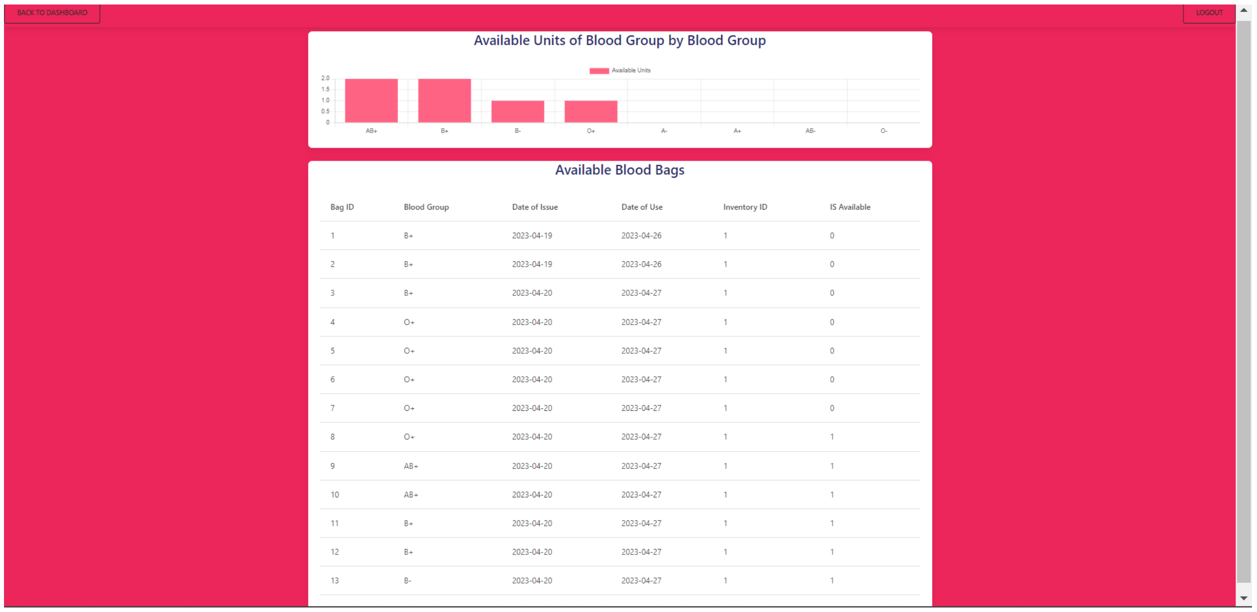
Request id	inventory_id	hospital_id	bag_id	requested	received
4	1	New Amsterdam	4	O+	O+
5	1	New Amsterdam	5	O+	O+
7	1	New Amsterdam	7	O+	O+

Request ID: 4

6.12 Check Inventory

The check inventory option on the dashboard allows the administrator to have a quick glance on all the units of blood currently available in the inventory. Since the current implementation assumes that only a single inventory exists for the blood bank, we can look over at the visualization for the current inventory. Whenever a new donor gets added or an existing donor donates blood, we see an increase in the count of the blood bag for that particular blood group.

We have a query in the get_current_blood_stock_at_inventory() procedure to refresh and get the latest count of available blood bag units for each of the blood groups. Whenever a patient is added or an existing patient requests for an additional unit of blood bag, the add_additional_blood_bag() procedure checks for compatible blood available in the inventory using the blood_group_receiver table. If there is any blood bag that matches the requested type, we order the request by the date of use of matched blood units and request for a dispatch of that unit.



Available Blood Bags

Bag ID	Blood Group	Date of Issue	Date of Use	Inventory ID	IS Available
1	B+	2023-04-19	2023-04-26	1	0
2	B+	2023-04-19	2023-04-26	1	0
3	B+	2023-04-20	2023-04-27	1	0
4	O+	2023-04-20	2023-04-27	1	0
5	O+	2023-04-20	2023-04-27	1	0
6	O+	2023-04-20	2023-04-27	1	0
7	O+	2023-04-20	2023-04-27	1	0
8	O+	2023-04-20	2023-04-27	1	1
9	AB+	2023-04-20	2023-04-27	1	1
10	AB+	2023-04-20	2023-04-27	1	1
11	B+	2023-04-20	2023-04-27	1	1
12	B+	2023-04-20	2023-04-27	1	1
13	B-	2023-04-20	2023-04-27	1	1

7. Lessons Learned

During the course of this project, the two of us learnt a lot about databases and designing a database given a problem statement. We learnt building a logical database from the conceptual design, determining the data types, relationship between different entities and their representation within a particular database. The project also helped us understand integration of databases with external applications for I/O operations. Throughout the journey, we understood, realized and reimplemented design changes within our database in order to provide better user experience with ease of operation. Since all data is being handled and manipulated within the database system, we learnt ways to encapsulate and protect our data from variation by external systems.

The project also helped us learn more about the healthcare domain and understand the need for data privacy and protection. As a future scope of reference we look towards protecting this data using encryption while storing it in the database.

All the features as mentioned in the activity flow diagram work as expected. Please follow the video link for project demo and logical flow.

8. Future Work

Listed below are a few areas where we feel our application can grow on

- As a part of future implementation, we plan on encrypting our data and then storing/manipulating it in the database.
- We look towards extending this platform for multiple users and inventories. As a pilot, we have the application up and running through the administrator perspective. In the future, we can have role segregation with each user gaining access to only parts of the database relevant to them. For example, a volunteer shall have access to only add/edit/modify a donor while an internal staff of the blood bank can communicate with hospitals and patients to receive and work upon their requests.
- We can also scale the process by adding my admission reasons and their severity codes to provide visibility to cover all hospitals in need of blood bags for patients with provided severity.
- We also look towards scaling the project where the patient receives multiple blood bags and not just one.
- To include a list of all codes for reasons of admission to hospital
- Adding more Data Visualization components for the inventory module.
- Integrate a text based alert system to alert patients if they have been matched with a compatible unit of blood.