

Blood bank management system

Context:

- [W3H Analysis](#)
- [Uml](#)
- [ER Diagram](#)
- [Wire Frame](#)
- [Front end](#)
- [Bank end](#)
- [Sonar Report](#)
- [Testing Report](#)
- [CI / CD](#)

W3H ANALYSIS

Case Study 12: Blood Bank Management System

Blood bank management system automates the blood bank services. This interconnects blood banks, donors and users. It has the following features like

- Donor details management
- Blood stock management
- Blood collection management
- Blood request management
- Report generation

W3H Analysis

What	How
<p>1. What is the kind of user are there?</p> <ul style="list-style-type: none">- Admin (blood hub)- Blood Bank (bank)- User<ul style="list-style-type: none">- Donor- Request (For Hospital) <p>(Without Registration)</p> <ul style="list-style-type: none">- User<ul style="list-style-type: none">- Request (For individual)	<p>1. Registration for user, Blood Bank</p> <p>M1: Separate Page for all</p> <p>M2: Separate page for Blood bank, User as donor and hospital login</p> <p>2. Login for all</p> <p>M1: Separate page for login</p> <p>M2: Same page for login and enter to page based on their role</p> <p>3. Documentation upload</p> <p>M1: Documentation can upload as image</p> <p>M2: Documentation can upload as pdf</p> <p>M3: Documentation can upload as image and pdf</p>
<p>2. Is User have specific privilege for hospital and individual?</p>	

- Yes, Hospital can request from individual and bank
- Individual can request from bank

3. Is Requested User can track the request information

Yes

4. Is Request and Donor user need special field, if required what are the details?

- Request must give basic details for authentication
- Donor must give basic details and verification documentation for blood group

5. What are the basic details given by User, Blood Bank, Hospital

- User (as Donor, Hospital) => Name, Mobile Number, Aadhar number (if user), Address, Email, password
- Bank=> Bank Name, Location, Mobile Number, Address, Bank Official Id (Government Registered Id), Document for verify the Bank.

6. Any Specific details required for hospital login as user?

- Yes, Their Documentation regarding the government registration and Registration id.

7. Who can donate their blood?

- If they register as donor, they can donate their blood by call

8. What can Admin(hub) do?

- They can view the bank details, user details (donor, requester), Collection camp details conducted by bank, Blood stock details by individual bank and location.
- They approve or reject registration based on their details for user (donor, hospital) and bank registration details and approve or reject request for blood.

4. Camp Details

M1: Email sends to nearby user
 M2: Web notification to user
 M3: Camp Calendar shown in websites and sent email

5. Report Generation

M1: Report Generation is done by excel
 M2: Report is shown in pdf
 M3: Report is shown in websites

6. How is the appointment booked with the donor?

M1: Donor can choose date for appointment at the time of registration

M2: Bank can check the availability of donor when they needed and donor get called by bank

7. Document verification

M1: Done manually by admin
 M2: Partial verification is done by system.

8. Blood bank can view other bank stock

M1: View by location
 M2: View by blood Bank
 M3: View by location and blood bank

9. Blood bank can view the details of donor

M1: Necessary details like name, blood type, contact details and location
 M2: All details

10. Admin can approve and reject the user registration

M1: By verify the details and mail send to user
 M2: By verify the details and notification send to websites

11. Track Request of blood

M1:By Request Id and mobile number
 M2:By Request Id Only

<p>9. How can generate reports and what kind of report?</p> <ul style="list-style-type: none"> • Bank must only view their blood donated details of their own bank and blood stock details of their own and nearby bank • Admin can view blood stock of all banks <p>10. what can the bank do?</p> <ul style="list-style-type: none"> • Bank can view the report of blood stock and donor details • They can donate blood to those who requested. • They can add, update, delete the blood stock • They can conduct camp • They can view local donor • They can see blood stock of another nearby bank <p>11. What are the details required for blood?</p> <ul style="list-style-type: none"> • Blood=> Type, Units, Location, Bank Details 	
<p>Why</p> <p>1. Registration for user, Blood Bank M2: Separate page for Blood bank, User as donor and hospital login R: It is easy for register and easy to maintain the page</p> <p>2. Login for all M2: Same page for login and enter to page based on their role R: It is easy to maintain the website and easy for user</p> <p>3. Documentation upload M1: Documentation can upload as image R: It is easy to view by admin</p> <p>4. Camp Details</p>	<p>Why not?</p> <p>1. Registration for user, Blood Bank M1: Separate Page for all R: It is hard for user and hard maintain the page</p> <p>2. Login for all M1: Separate page for login R: It is easy but hard for user to login</p> <p>3. Documentation upload M2: Documentation can upload as pdf M3: Documentation can upload as pdf and image R: It is hard to display</p> <p>4. Camp Details M1: Email sends to nearby user M2: Web notification to user</p>

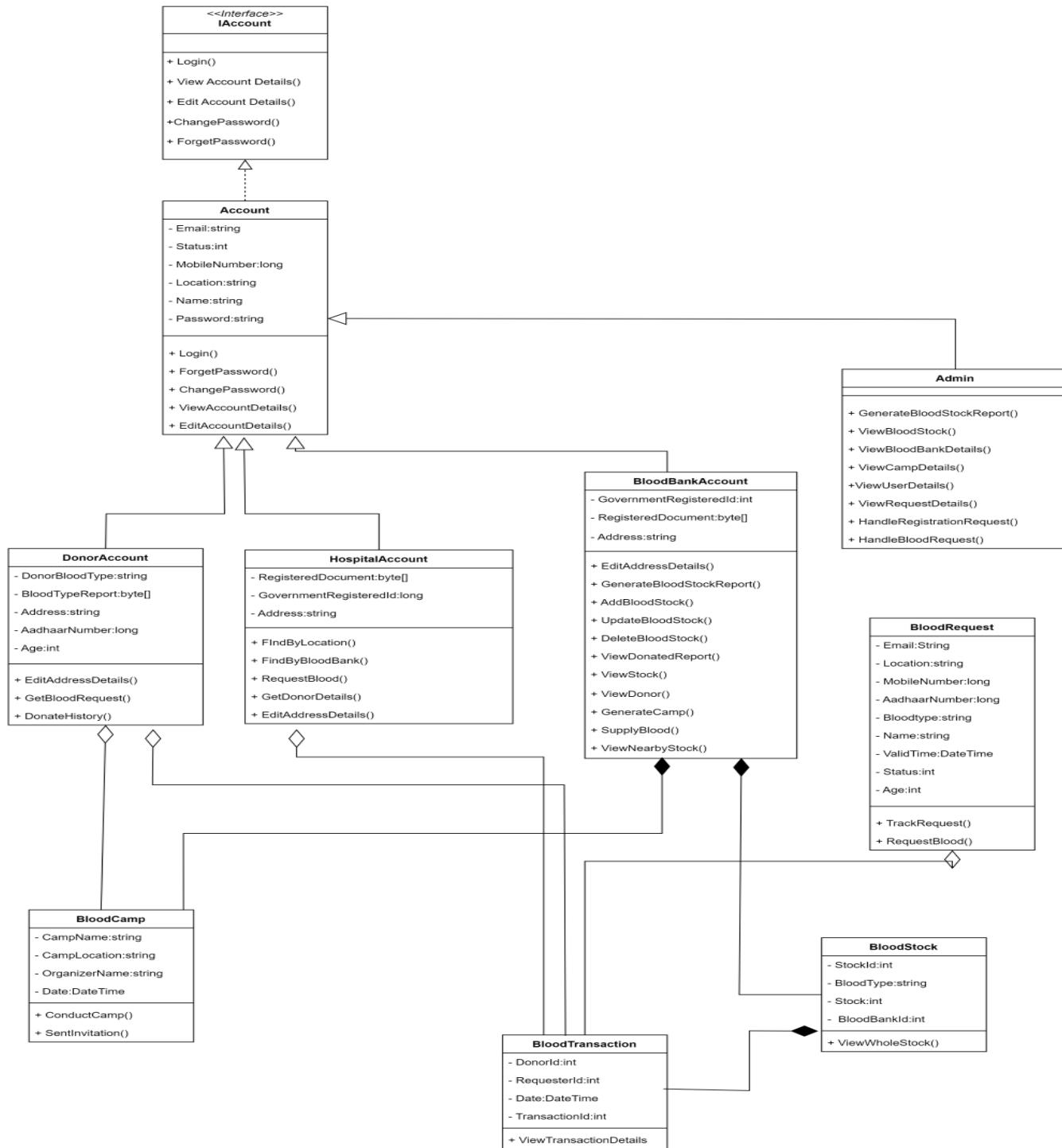
<p>M3: Camp Calendar shown in websites and sent email</p> <p>R: It can increase the donor and pass the information to large people</p> <p>5. Report Generation</p> <p>M3: Report is shown in websites</p> <p>R: It is efficient way to view the report</p> <p>6. How is the appointment booked with the donor?</p> <p>M2: Bank can check the availability of donor when they needed and donor fix appointment after getting the request from bank</p> <p>R: It is more usable case</p> <p>7. Document verification</p> <p>M1: Done manually by admin</p> <p>R: More trustable</p> <p>8. Blood bank can view other bank stock</p> <p>M3: View by location and blood bank</p> <p>R: It is useful to search and find needed blood</p> <p>9. Blood bank can view the details of donor</p> <p>M1: Necessary details like name, blood type, contact details and location</p> <p>R: It helps to hide confidently details</p> <p>10. Admin can approve and reject the user registration</p> <p>M1: By verify the details and mail send to user</p> <p>R: It is reachable to user</p> <p>11. Track Request of blood</p> <p>M1:By Request Id and mobile number</p> <p>R: It is easy and secure</p>	<p>R: It is not properly received by large people</p> <p>5. Report Generation</p> <p>M1: Report Generation is done by excel</p> <p>M2: Report is shown in pdf</p> <p>R: It is not an efficient way to view the report.</p> <p>6. How is the appointment booked with the donor?</p> <p>M1: Donor can choose date for appointment at the time of registration</p> <p>R: It is less usable case</p> <p>7. Document verification</p> <p>M2: Partial verification is done by system.</p> <p>R: Less Trustable</p> <p>8. Blood bank can view other bank stock</p> <p>M1: View by location</p> <p>M2: View by blood Bank</p> <p>R: It is not more useful to find blood</p> <p>9. Blood bank can view the details of donor</p> <p>M2: All details</p> <p>R: It not hide confidently details</p> <p>10. Admin can approve and reject the user registration</p> <p>M2: By verify the details and notification send to websites</p> <p>R: It is not reachable to user as it take time</p> <p>11. Track Request of blood</p> <p>M2:By Request Id Only</p> <p>R: It is easy and not secure</p>
---	--



UML

UML

Class Diagram



Use Case Diagram

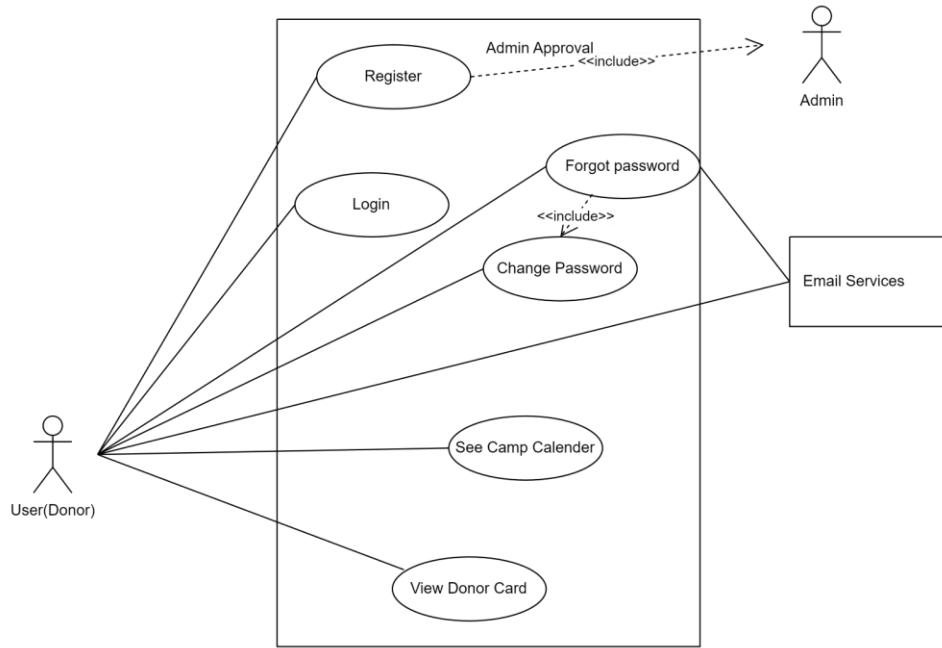
Admin



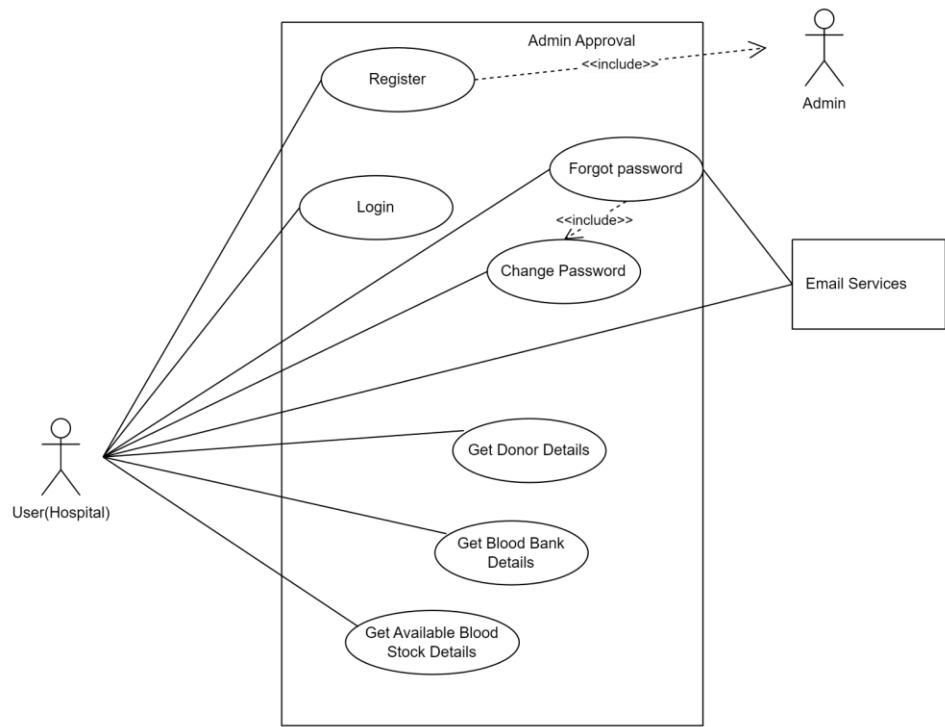
Blood Bank



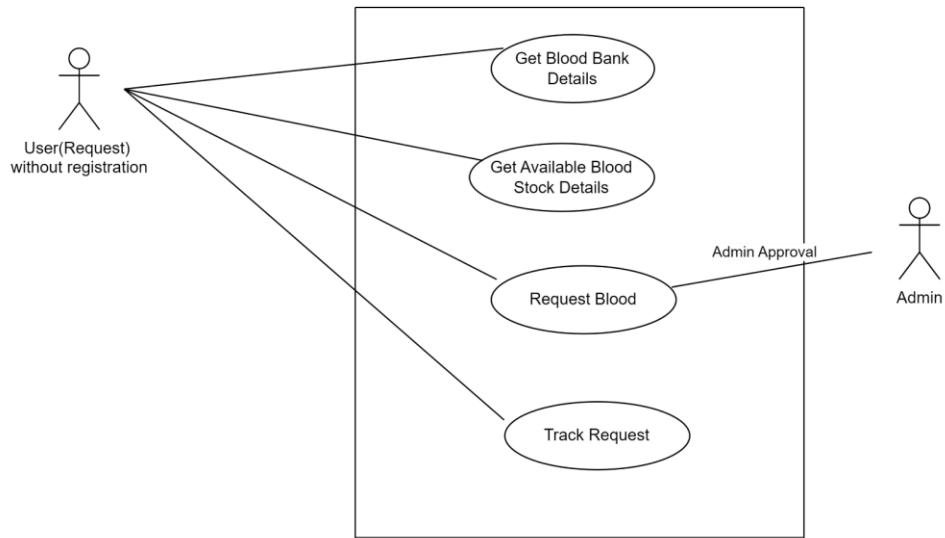
User(Donor)



User(Hospital)



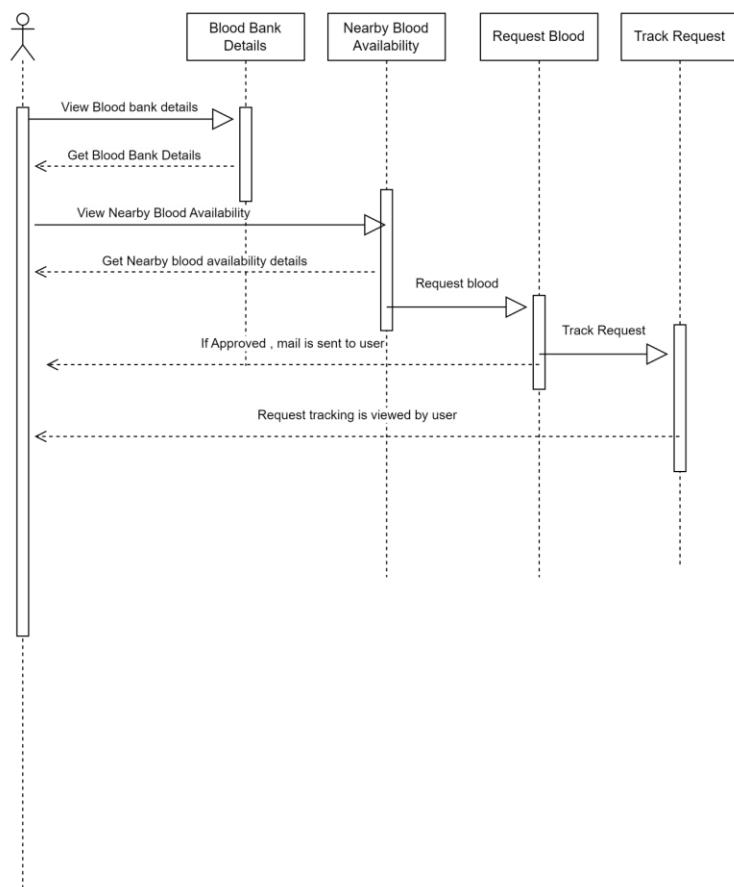
User (Blood Requester)



Sequence Diagram

User (Blood Requester)

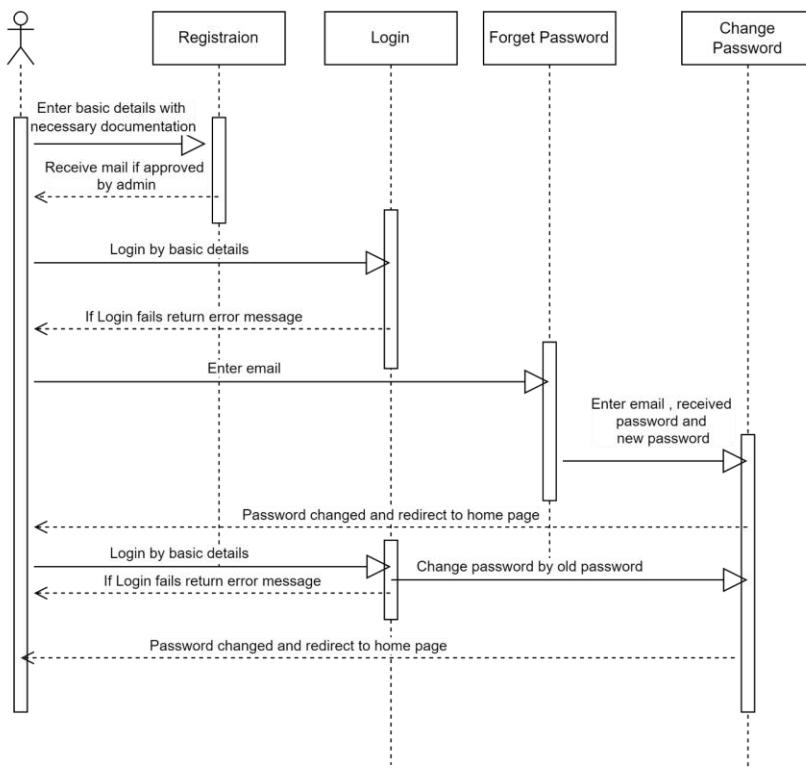
User without registration



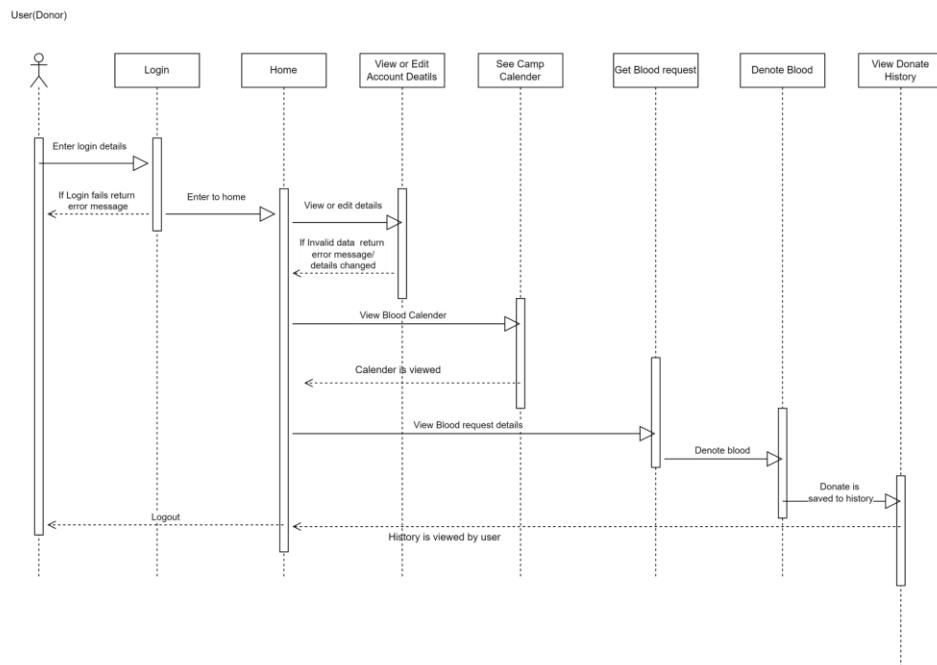
Registration and login for User (Donor,Hospital), Blood Bank

Registration and login for User(Donor,Hospital), Blood Bank

User(Donor,Hospital) , Blood Bank

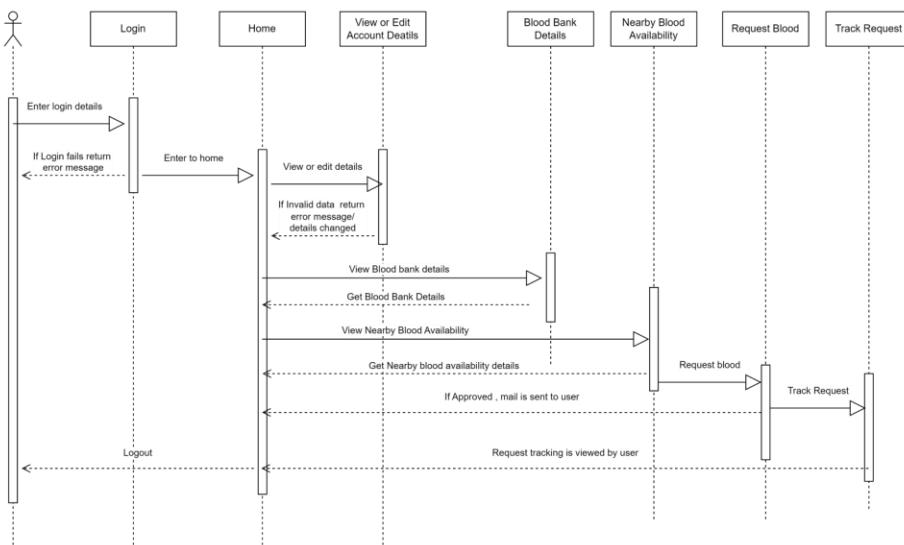


User(Donor)



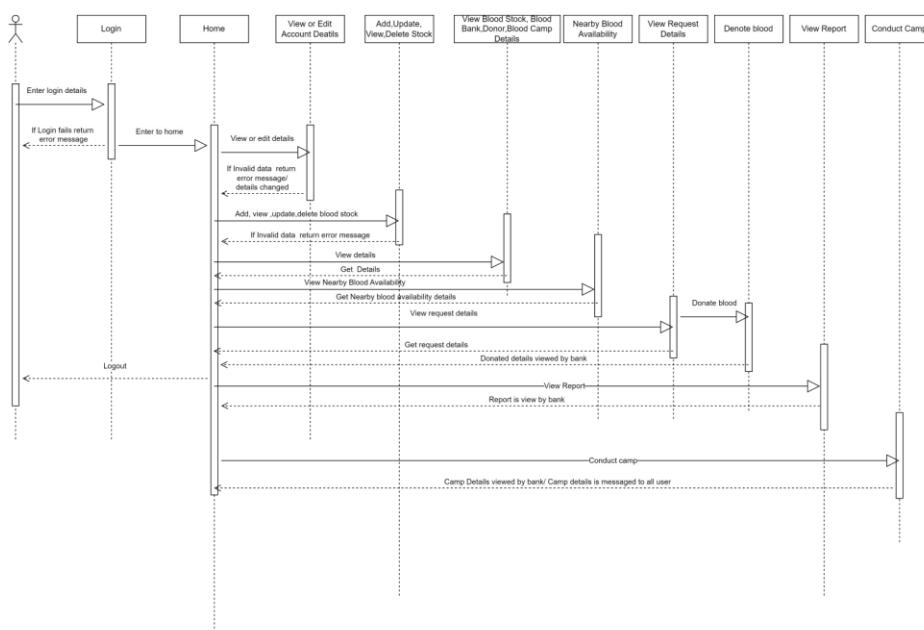
User(Hospital)

User(Hospital)



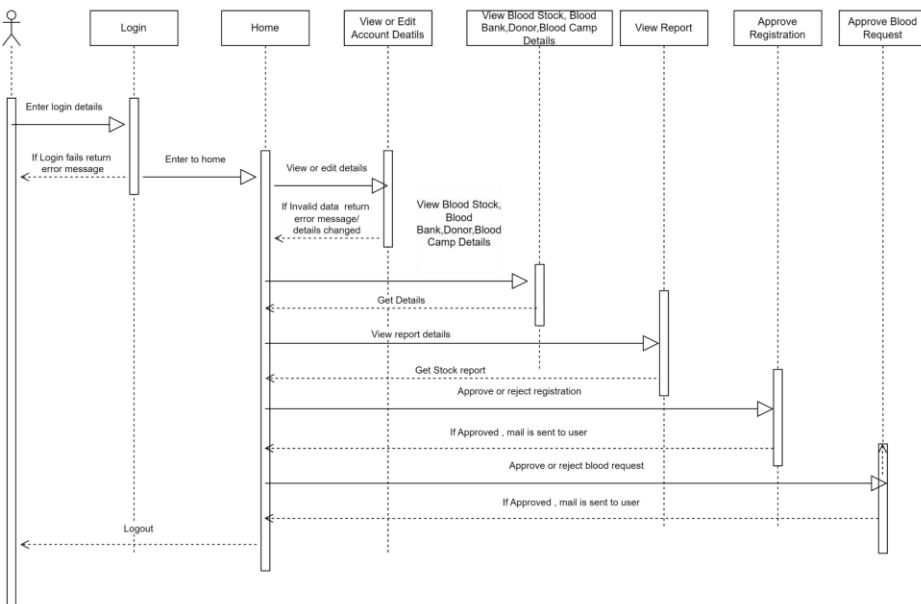
Blood Bank

Blood Bank



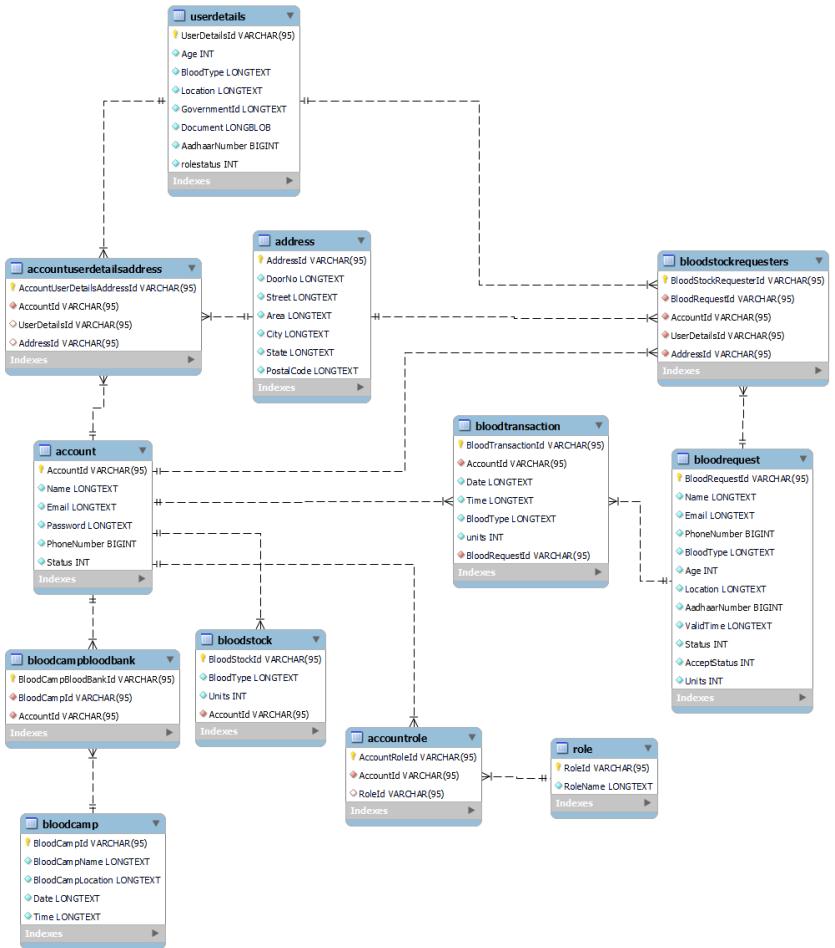
Admin

Admin



ER DIAGRAM

ER Diagram



WIRE FRAME

Wire Frame

Welcome Page

Donate	Login		
Donate Blood Save Life			
Donor	Hospital	Blood Request	Blood Bank
Location	Blood Bank		
About Us	Contact Us		
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.			

Login Page

Donate	Login
<div style="border: 1px solid black; padding: 10px; text-align: center;"><p>Login</p><p>Enter a email</p><input type="text"/><p>Enter a password</p><input type="password"/><p><input type="button" value="Login"/></p></div>	

Register Page (Getting Common details)

Donate	Login
<div style="border: 1px solid black; padding: 10px; text-align: center;"><p>Register</p><p>Enter a Name</p><input type="text"/><p>Enter a Email</p><input type="text"/><p>Enter a Mobile Number</p><input type="text"/><p>Enter a Password</p><input type="text"/><p><input type="button" value="Register"/></p></div>	

Blood Request page

Donate	Login
<div style="border: 1px solid black; padding: 10px; text-align: center;"><p>Blood Request</p><p>Enter a name</p><input type="text"/><p>Enter a email</p><input type="text"/><p>Enter a mobile number</p><input type="text"/><p>Age of required person</p><input type="text"/><p>Enter a location(city)</p><input type="text"/><p>Enter a aadhaar number</p><input type="text"/><p>Enter a blood type</p><input type="text"/><p><input type="button" value="Request"/></p></div>	

Checking the Blood Availability

Donate Username

Blood Availability

Enter a blood type

Blood	Unit	Blood Bank
B+	10	Qlab Bank, Palalyamkottai
B+	2	Zlab Bank, Palalyamkottai

Incoming Donor Request (In Admin)

Display Upcoming Blood Camp

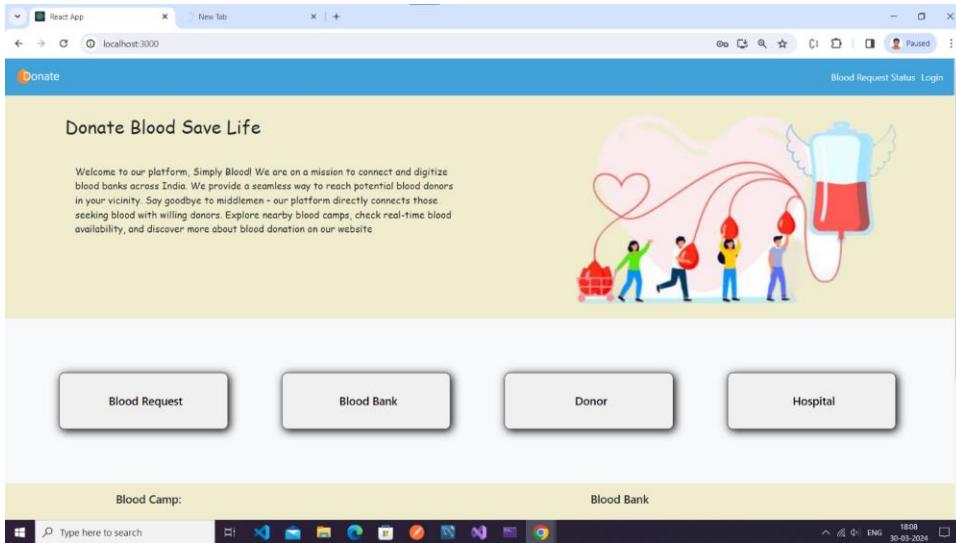
Donate		<input type="text"/> Username		
Upcoming Blood Camp				
Camp Name	Location	Conduct by	Date	Time
Bharat Sevai	Tirunelveli	Qlab Bank	20-05-2002	10:00-12:00
Tamil sevai	Chennai	Zlab Bank	20-05-2002	12:00-2:00

FRONT END

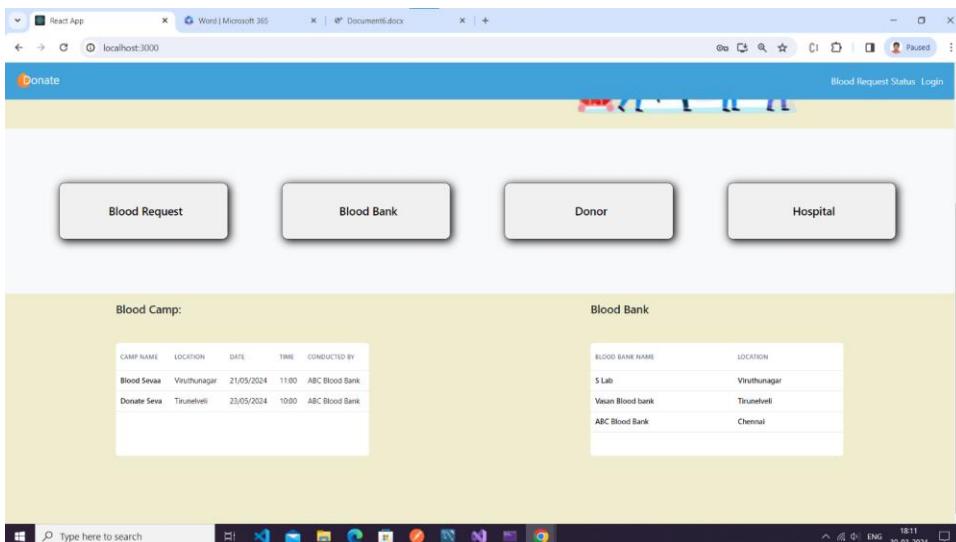
Front End

Git Link <https://github.com/smano-20052002/MicropjjectReactapp.git>

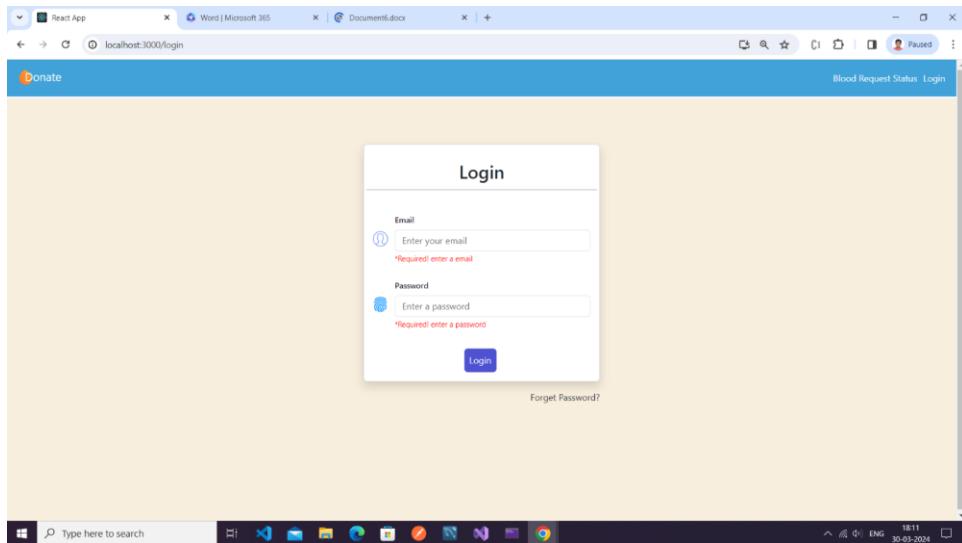
Welcome Page



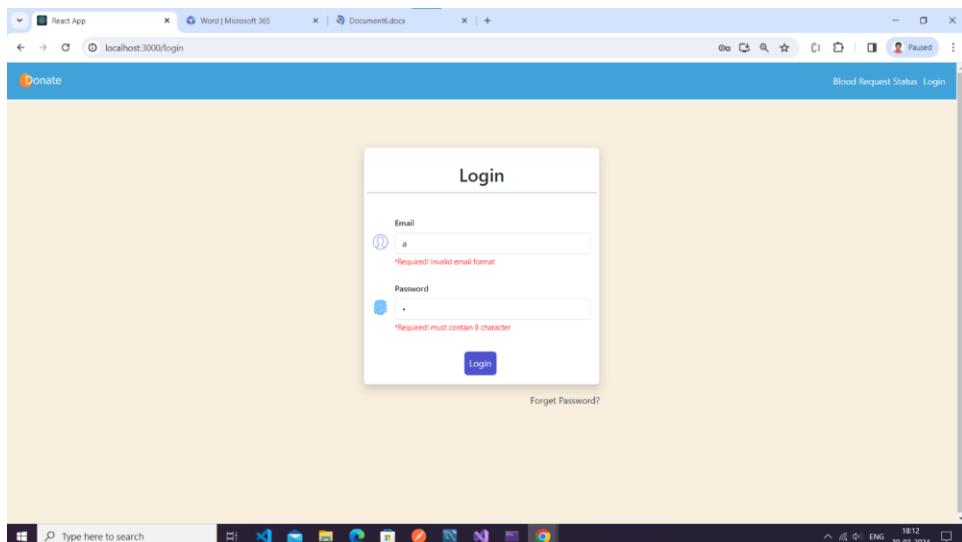
Welcome page contain blood camp and blood bank details



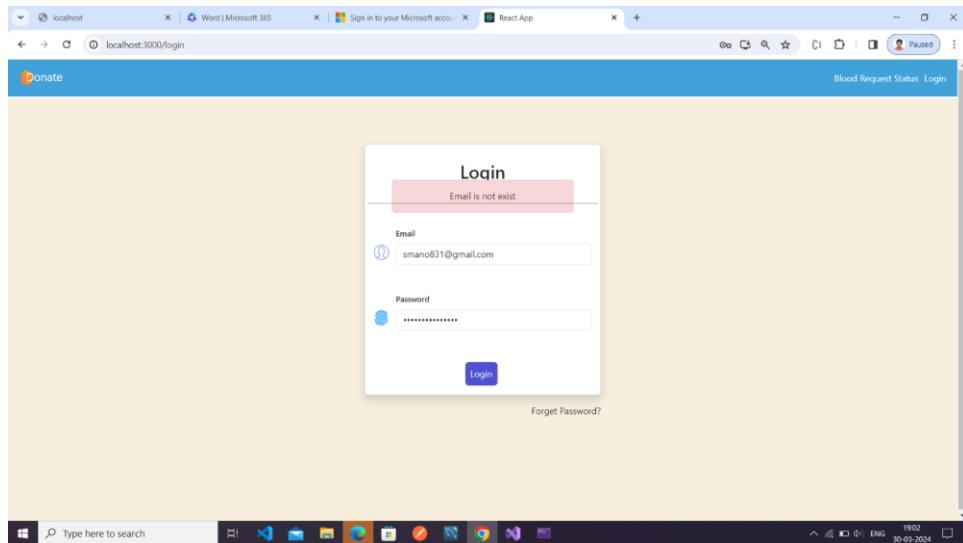
Login show error without entering data after login is clicked



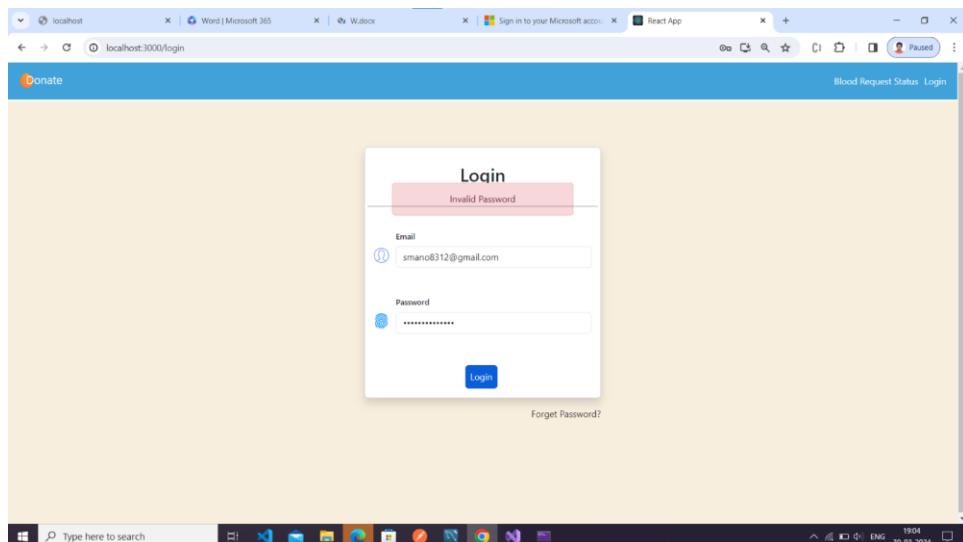
Invalid email and password with less than 8 character show error



Invalid email error message is shown



Incorrect password error message is shown



Admin Dashboard

The screenshot shows the Admin Dashboard with a sidebar on the left containing links for Dashboard, View Hospital, View Donor, View Blood Bank, View Blood Camp, Hospital Request, Donor Request, Blood Bank Request, Blood Request, All Blood Stock, and Logout. The main area is titled 'Dashboard' and contains several summary boxes and a blood type distribution chart.

Category	Count
Hospital	1
Blood Bank	3
Donor	1
Blood Camp	2
Blood Request	5
Pending Blood Request	2
Blood Bank Request	0
Donor Request	0
Hospital Request	0
A+ve	1
B+ve	2
O+ve	0
AB+ve	1
B-ve	0
A-ve	96
O-ve	0
AB-ve	0

View Hospital details by admin

The screenshot shows the 'View Hospital' page with a sidebar on the left containing the same set of links as the dashboard. The main area is titled 'Hospital' and displays a table with the following data:

HOSPITAL NAME	EMAIL	MOBILE NUMBER	GOVERNMENT ID	DOCUMENT	ADDRESS DETAILS	LOCATION	APPROVAL
ABC Hospital	smans4570@gmail.com	6382202586	1245433		21/2, Salai Street, Ktc Nagar, Ktc Nagar, Tirunelveli, Tamil Nadu-627001	Chennai	Approved

View Donor Details by admin

The screenshot shows a web application interface. On the left, a sidebar titled "Donate" contains a "Dashboard" section and links for "View Hospital", "View Donor", "View Blood Bank", "View Blood Camp", "Hospital Request", "Donor Request", "Blood Bank Request", "Blood Request", "All Blood Stock", and "Logout". The main content area is titled "Donor" and displays a single record:

DONOR NAME	BLOOD TYPE	AGE	CONTACT	LOCATION	ADDRESS	APPROVAL STATUS
Keerthi Vasan	B+ve	21	keerthivasan28092001@gmail.com 9089678776	Chennai	21, Vt Street, Narayana nagar, Periyar Nagar, Parambalam, Tamil Nadu-627001	Approved

View blood bank details by admin

The screenshot shows a web application interface. On the left, a sidebar titled "Donate" contains a "Dashboard" section and links for "View Hospital", "View Donor", "View Blood Bank", "View Blood Camp", "Hospital Request", "Donor Request", "Blood Bank Request", "Blood Request", "All Blood Stock", and "Logout". The main content area is titled "Blood Bank" and displays three records. The third record, "Vasan Blood bank", has a detailed view of its registration certificate displayed on the right side:

Vasan Blood bank

BLOOD BANK NAME	EMAIL	MOBILE NUMBER	GOVERNMENT ID	DOCUMENT	ADDRESS DETAILS	LOCATION	APPROVAL
S Lab	arunisrinivasan33@gmail.com	9878675432	TN00932233		Chennai	Approved	
Vasan Blood bank	keerthivasan28092001@gmail.com	9790953230	TN00932222		Vellore	Approved	
ABC Blood Bank	sanjalhar2002@gmail.com	638202487	1245433		Chennai	Approved	

View all camp details by admin

A screenshot of a web browser window titled "Blood Camp". The left sidebar, titled "Donate", contains a list of administrative options: Dashboard, View Hospital, View Donor, View Blood Bank, View Blood Camp (which is selected and highlighted in blue), Hospital Request, Donor Request, Blood Bank Request, Blood Request, All Blood Stock, and Logout. The main content area displays two rows of camp details:

CAMP NAME	LOCATION	DATE	TIME	CONDUCTED BY
Blood Seva	Viruthunagar	21/05/2024	11:00	ABC Blood Bank
Donate Seva	Tirunelveli	23/05/2024	10:00	ABC Blood Bank

View Incoming Hospital request by admin

A screenshot of a web browser window titled "Hospital Request". The left sidebar, titled "Donate", contains a list of administrative options: Dashboard, View Hospital, View Donor, View Blood Bank, View Blood Camp, Hospital Request (which is selected and highlighted in blue), Donor Request, Blood Bank Request, Blood Request, All Blood Stock, and Logout. The main content area displays a single hospital request row:

HOSPITAL NAME	EMAIL	MOBILE NUMBER	GOVERNMENT ID	DOCUMENT	ADDRESS DETAILS	LOCATION	APPROVAL
ABC Hospital	shano4570@gmail.com	6382202586	1245433		21/2, Salar Street, Kic Nagar Kic Nagar Thoothukudi Tamil Nadu - 627011	Chennai	<button>Approve</button> <button>Reject</button>

View Incoming Donor request by admin

DONOR NAME	BLOOD TYPE	AGE	CONTACT	LOCATION	ADDRESS	APPROVAL STATUS
Keerthi Vasan	B+ve	21	keerthivasan280901@gmail.com 9089078776	Chennai	21, Vt Street, Narayana Nagar, Narayana Nagar, Perambalur, Tamil Nadu-627001	Approve Reject

View Incoming Blood bank request by admin

BLOOD BANK NAME	EMAIL	MOBILE NUMBER	GOVERNMENT ID	DOCUMENT	ADDRESS DETAILS	LOCATION	APPROVAL
Vasan Blood bank	keerthivasan28092001@gmail.com	9790953230	TN00132232		28, Kott College, Vennerpetal, Vennerpetal, Tirunelveli, Tamil nadu-687300	Tirunelveli	Approve Reject

View Incoming Blood Request request by admin

REQUEST ID	NAME	EMAIL	AGE	BLOOD TYPE	MOBILE NUMBER	AADHAAR NUMBER	VALID TIME	LOCATION	ACTION
4uOCW9W	Mano	keerthivasan289091@gmail.com	19	B+ve	9790953230	90909909079	03/09/2024 16:43:07	Tirunelveli	Rejected
OcityY9	Mano	smano4570@gmail.com	1	AB+ve	9790953230	23453432987		Chennai	Assign Reject
SP39NEvE	Vasan	senathipathik@gmail.com	35	A-ve	9876543210	34543243222	04/01/2024 08:40:20	Vinuthnagar	Rejected
T2937hCU	Sanjai	sanjairocks@gmail.com	20	B+ve	9887541243	23548998762	03/09/2024 17:11:20	Chennai	Rejected
WfRkBxCa	string	string	0	string	0	0		string	Assign Reject

View all blood bank blood stock details by admin

BLOOD TYPE	UNIT	BLOOD BANK	LOCATION	MOBILE NUMBER
A+ve	1	S Lab	Vrithnagar	6382202487
AB+ve	1	S Lab	Vrithnagar	6382202487
B+ve	2	S Lab	Vrithnagar	6382202487
A-ve	96	S Lab	Vrithnagar	6382202487
A+ve	0	S Lab	Vrithnagar	9790953230
A+ve	1	ABC Blood Bank	Chennai	6382202487
AB+ve	1	ABC Blood Bank	Chennai	6382202487
B+ve	2	ABC Blood Bank	Chennai	6382202487
A-ve	96	ABC Blood Bank	Chennai	6382202487
A+ve	0	ABC Blood Bank	Chennai	9790953230

Search by blood type

The screenshot shows a web browser window with a sidebar menu on the left and a main content area on the right.

Left Sidebar (Donate):

- Dashboard
- View Hospital
- View Donor
- View Blood Bank
- View Blood Camp
- Hospital Request
- Donor Request
- Blood Bank Request
- Blood Request
- All Blood Stock
- Logout

Main Content Area (Blood Bank):

A search bar at the top has the letter 'A' typed into it. Below the search bar is a table with the following columns: BLOOD TYPE, UNIT, BLOOD BANK, LOCATION, and MOBILE NUMBER.

BLOOD TYPE	UNIT	BLOOD BANK	LOCATION	MOBILE NUMBER
A+ve	1	S Lab	Vinthanagar	6382202487
AB+ve	1	S Lab	Vinthanagar	6382202487
A-ve	96	S Lab	Vinthanagar	6382202487
A+ve	0	S Lab	Vinthanagar	9790953230
A-ve	1	ABC Blood Bank	Chennai	6382202487
AB+ve	1	ABC Blood Bank	Chennai	6382202487
A-ve	96	ABC Blood Bank	Chennai	6382202487
A+ve	0	ABC Blood Bank	Chennai	9790953230

Login with pending request of account creation

The screenshot shows a web browser window with a sidebar menu on the left and a main content area on the right.

Left Sidebar (Donate):

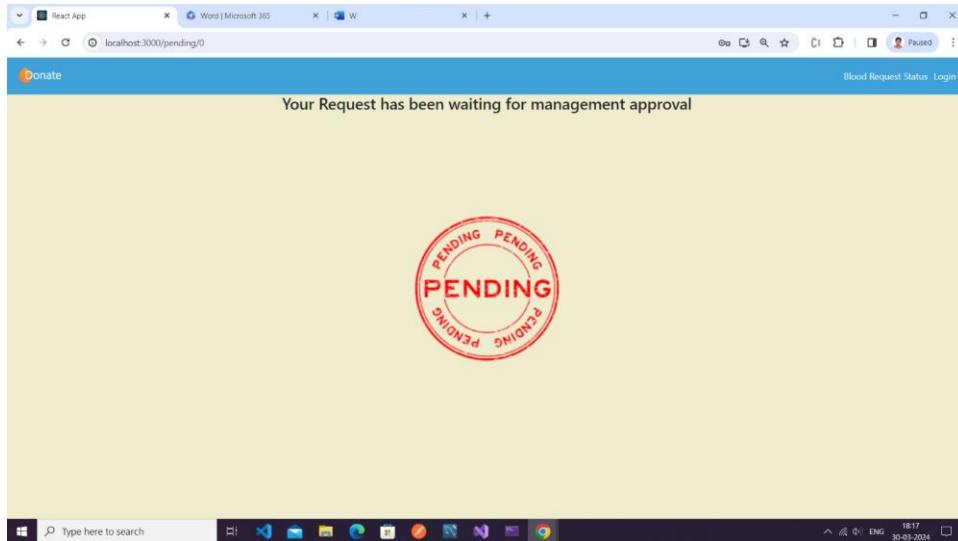
- Dashboard
- View Hospital
- View Donor
- View Blood Bank
- View Blood Camp
- Hospital Request
- Donor Request
- Blood Bank Request
- Blood Request
- All Blood Stock
- Logout

Main Content Area:

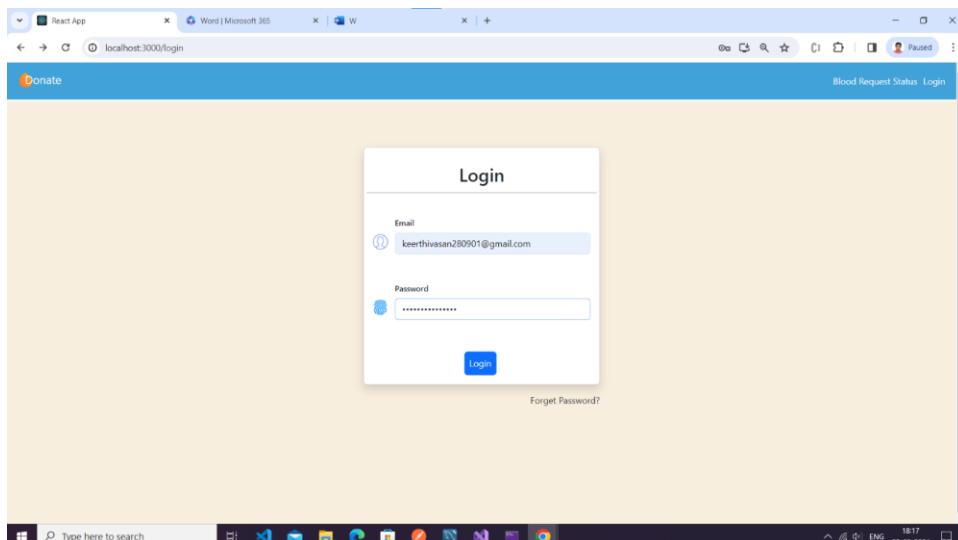
The main content area features a "Login" form. The "Email" field contains "keerthivasan280901@gmail.com". The "Password" field contains a masked password. A "Login" button is below the fields. At the bottom of the form, there is a link "Forgot Password?"

At the top right of the main content area, there are links for "Blood Request Status" and "Login".

Pending Page



Login donor account



Donor Home Page

The screenshot shows a web browser window titled "React App" with the URL "localhost:3000/donorhome". The page has a blue header bar with the word "Donate" and a "Logout" button. Below the header, there is a section titled "Blood Camp" containing a table with two rows of data. The columns are labeled "CAMP NAME", "LOCATION", "DATE", "TIME", and "CONDUCTED BY". The first row shows "Blood Seva" at "Vishnunagar" on "21/05/2024" at "1100" by "ABC Blood Bank". The second row shows "Donate Seva" at "Tirunelveli" on "23/05/2024" at "1000" by "ABC Blood Bank". At the bottom right of the table, there is a "VIEW CARD" link.

View Donor Card

The screenshot shows a web browser window titled "React App" with the URL "localhost:3000/donorhome". The page has a blue header bar with the word "Donate" and a "Logout" button. Below the header, there is a section titled "Blood Camp" containing a table with two rows of data. The columns are labeled "CAMP NAME", "LOCATION", "DATE", "TIME", and "CONDUCTED BY". The first row shows "Blood Seva" at "Vishnunagar" on "21/05/2024" at "1100" by "ABC Blood Bank". The second row shows "Donate Seva" at "Tirunelveli" on "23/05/2024" at "1000" by "ABC Blood Bank". In the center of the page, there is a large, semi-transparent red rectangular overlay containing a "Blood Donor" card. The card displays the donor's name "Koorthi Vasan", blood type "B+ve", and location "Chennai". It also features a small blood drop icon and a barcode. At the bottom of the card, the ID "477b7d70-299c-4428-a130-7b5bf5176586" is printed.

Blood Bank Dashboard

The screenshot shows a dashboard with a sidebar menu titled "Donate". The main area displays summary statistics: Hospital (1), Blood Bank (3), Donor (1), and Blood Camp (2). Below these are detailed counts for blood types: A+ve (1), B+ve (2), O+ve (0), AB+ve (1), B-ve (0), A-ve (96), O-ve (0), and AB-ve (0).

Blood Type	Count
A+ve	1
B+ve	2
O+ve	0
AB+ve	1
B-ve	0
A-ve	96
O-ve	0
AB-ve	0

View Donor details by blood bank

The screenshot shows a "Donor" details page. The sidebar menu is identical to the dashboard. The main table displays one donor record:

DONOR NAME	BLOOD TYPE	AGE	CONTACT	LOCATION	ADDRESS	APPROVAL STATUS
Keerthi Vasan	B+ve	21	keerthivasan200901@gmail.com 9089678778	Chennai	21, VK Street, Narayana nagar, Paramba nagar, Paramba, Tamil Nadu-627001	Approved

View stock details of that bank by blood bank

BLOOD TYPE	UNIT
A+ve	1
AB+ve	1
B+ve	2
A-ve	96

Add stock by bank shows error with empty data entry

Blood Type: A+ve
Unit: 0

Successful message after added

A screenshot of a web browser window titled "React App". The URL is "localhost:3000/addbloodstock". The main content area displays a success message: "Added Successfully". Below it, there are two input fields: "Blood Type" (set to "A+ve") and "Unit" (set to "0"). A blue button labeled "Add Blood Stock" is visible. On the left side, there is a sidebar with a "Donate" heading and a list of navigation items: Dashboard, View Donor, View Blood Stock, Add Blood Stock, Add Blood Transfer, View Blood Transfer, Add Blood Camp, View Blood Camp, Incoming Request, All Blood Stock, and Logout.

Show error msg with empty data entry

A screenshot of a web browser window titled "React App". The URL is "localhost:3000/updatebloodtransaction". The main content area shows an "Add Blood Transaction" form. It has three input fields: "Blood Requester Id" (highlighted in red with an error message "*Required field"), "Blood Type" (highlighted in red with an error message "*Required field"), and "Blood Units" (highlighted in red with an error message "*Required field! Minimum value is 1"). A blue button labeled "Donate Blood" is visible. On the left side, there is a sidebar with a "Donate" heading and a list of navigation items: Dashboard, View Donor, View Blood Stock, Add Blood Stock, Add Blood Transfer, View Blood Transfer, Add Blood Camp, View Blood Camp, Incoming Request, All Blood Stock, and Logout.

View Blood Transaction Details of that bank by blood bank

The screenshot shows a web application interface titled "Blood Transaction". On the left, there is a sidebar menu under "Donate" with various options: Dashboard, View Donor, View Blood Stock, Add Blood Stock, Add Blood Transfer, View Blood Transfer, Add Blood Camp, View Blood Camp, Incoming Request, All Blood Stock, and Logout. The main content area displays two rows of blood transaction details. The first row shows an A+ve unit requested by Mano (mano.kirthivasan200901@gmail.com, 9790953230) on 2024-03-21 at 10:00 Am. The second row shows a B+ve unit requested by Sanjal (sanjalrock85@gmail.com, 9867541243) on 2024-01-21 at 1:00 Pm.

Show error while entry the empty data in add blood camp

The screenshot shows a web application interface titled "Blood Camp". On the left, there is a sidebar menu under "Donate" with various options: Dashboard, View Donor, View Blood Stock, Add Blood Stock, Add Blood Transfer, View Blood Transfer, Add Blood Camp, View Blood Camp, Incoming Request, All Blood Stock, and Logout. The main content area displays a form for adding a blood camp. The "Blood Camp Name" field has a red error message "Required field" above it. The "Blood Camp Location" field also has a red error message "Required field" above it. The "Date" field has a red error message "Required field" above it. The "Time" field has a red error message "Required field" above it. A blue "Add Blood Camp" button is visible at the bottom of the form.

View Blood camp details of that blood bank by blood bank

The screenshot shows a web application interface. On the left, there is a sidebar titled "Donate" with various menu items: Dashboard, View Donor, View Blood Stock, Add Blood Stock, Add Blood Transfer, View Blood Transfer, Add Blood Camp, View Blood Camp, Incoming Request, All Blood Stock, and Logout. The main content area is titled "Blood Camp". It displays two rows of camp details:

CAMP NAME	LOCATION	DATE	TIME	CONDUCTED BY	ACTION
Blood Seva	Viruthunagar	21/05/2024	11:00	ABC Blood Bank	<button>Remove</button>
Donate Seva	Tirunelveli	23/05/2024	10:00	ABC Blood Bank	<button>Remove</button>

Blood stock details of all bank

The screenshot shows a web application interface. On the left, there is a sidebar titled "Donate" with various menu items: Dashboard, View Donor, View Blood Stock, Add Blood Stock, Add Blood Transfer, View Blood Transfer, Add Blood Camp, View Blood Camp, Incoming Request, All Blood Stock, and Logout. The main content area is titled "Blood Bank". It displays a table of blood stock details. At the top right of the table is a search bar labeled "Search By Blood Type".

BLOOD TYPE	UNIT	BLOOD BANK	LOCATION	MOBILE NUMBER
A+ve	1	S Lab	Viruthunagar	6382202487
AB+ve	2	S Lab	Viruthunagar	6382202487
B+ve	2	S Lab	Viruthunagar	6382202487
A-ve	96	S Lab	Viruthunagar	6382202487
A+ve	0	S Lab	Viruthunagar	9790953230
A+ve	1	Vasan Blood bank	Tirunelveli	6382202487
AB+ve	2	Vasan Blood bank	Tirunelveli	6382202487
B+ve	2	Vasan Blood bank	Tirunelveli	6382202487
A-ve	96	Vasan Blood bank	Tirunelveli	6382202487
A+ve	0	Vasan Blood bank	Tirunelveli	9790953230
A+ve	1	ABC Blood Bank	Chennai	6382202487
AB+ve	2	ABC Blood Bank	Chennai	6382202487
B+ve	2	ABC Blood Bank	Chennai	6382202487

View Blood request by bank they can accept or reject that request

REQUEST ID	NAME	EMAIL	AGE	BLOOD TYPE	UNITS	MOBILE NUMBER	AADHAAR NUMBER	LOCATION	ACTION
Qcttyf9	Mano	smano4570@gmail.com	1	AB+ve	1	9790053230	23454320987	Chennai	Accept

Login with hospital details

Login

Email

Password

[Forgot Password?](#)

Hospital Dashboard

The screenshot shows a web browser window titled "React App" with the URL "localhost:3000/hospitalhome". The interface has a sidebar on the left labeled "Donate" with icons for Dashboard, View Donor, View Blood Bank, View Blood Camp, All Blood Stock, and Logout. The main area is titled "Dashboard" and contains four cards: "Hospital" (1), "Blood Bank" (3), "Donor" (1), and "Blood Camp" (2).

View Donor details by hospital

The screenshot shows a web browser window titled "React App" with the URL "localhost:3000/viewdonor". The sidebar "Donate" includes "View Donor" which is currently selected. The main content displays donor details for "Keerthi Vasan":
DONOR NAME: Keerthi Vasan
BLOOD TYPE: B+ve
AGE: 21
CONTACT: keerthivasan200901@gmail.com
9089678778
LOCATION: Chennai
ADDRESS: 21, VK Street,
Narayana nagar,
Parampara nagar,
Parampara nagar,
Tamil Nadu-627001
APPROVAL STATUS: Approved

View Blood bank details by hospital

BLOOD BANK NAME	EMAIL	MOBILE NUMBER	ADDRESS DETAILS	LOCATION	APPROVAL
S Lab	anumrlnivasan33@gmail.com	9878675432	34, K Street, Vanarpettai, Vanarpettai, Vizhunagar, TamilNadu-687300	Viruthunagar	Approved
Vasan Blood bank	keerthivasan28092001@gmail.com	9790953230	28, Kot College, Vanarpettai, Vanarpettai, Tirunelveli, Tamil nadu-687300	Tirunelveli	Approved
ABC Blood Bank	sanjalhart2002@gmail.com	6382202487	24, VK Street, Kk Nagar, Kk Nagar, Veerthalangudi, Tamil Nadu-627001	Chennai	Approved

View blood camp details by hospital

CAMP NAME	LOCATION	DATE	TIME	CONDUCTED BY
Blood Seva	Viruthunagar	21/05/2024	11:00	ABC Blood Bank
Donate Seva	Tirunelveli	23/05/2024	10:00	ABC Blood Bank

View Blood Bank with stock by hospital

The screenshot shows a web application interface titled "Blood Bank". On the left, there is a sidebar with navigation links: "Dashboard", "View Donor", "View Blood Bank" (which is currently selected), "View Blood Camp", "All Blood Stock", and "Logout". The main content area displays a table with columns: BLOOD TYPE, UNIT, BLOOD BANK, LOCATION, and MOBILE NUMBER. The data in the table is as follows:

BLOOD TYPE	UNIT	BLOOD BANK	LOCATION	MOBILE NUMBER
A+ve	1	S Lab	Vrathnagar	6382202487
AB+ve	2	S Lab	Vrathnagar	6382202487
B+ve	2	S Lab	Vrathnagar	6382202487
A+ve	96	S Lab	Vrathnagar	6382202487
A+ve	0	S Lab	Vrathnagar	9790953230
A+ve	1	Vasan Blood bank	Tirunelveli	6382202487
AB+ve	2	Vasan Blood bank	Tirunelveli	6382202487
B+ve	2	Vasan Blood bank	Tirunelveli	6382202487
A+ve	96	Vasan Blood bank	Tirunelveli	6382202487
A+ve	0	Vasan Blood bank	Tirunelveli	9790953230
A+ve	1	ABC Blood Bank	Chennai	6382202487
AB+ve	2	ABC Blood Bank	Chennai	6382202487
B+ve	2	ABC Blood Bank	Chennai	6382202487

View pending status of pending blood request

The screenshot shows a web application interface titled "Check Blood Request Status". On the left, there is a sidebar with a single link: "donate". The main content area displays a form with a text input field containing "4iaOCW9V" and a green "Check" button. Below the button, the word "Pending" is displayed in bold. The status bar at the bottom of the browser window shows the date and time as "30-03-2024 18:27".

Show invalid id msg with invalid id

A screenshot of a web browser window titled "Check Blood Request Status". The URL bar shows "localhost:3000/checkrequest". The main content area displays a form with a text input field containing "4aOCW9" and a green "Check" button. Below the input field, the text "Invalid Id" is displayed in red.

Show bank and accepted bank details with approved request id

A screenshot of a web browser window titled "Check Blood Request Status". The URL bar shows "localhost:3000/checkrequest". The main content area displays a form with a text input field containing "T2937kCU" and a green "Check" button. Below the input field, the text "Accepted Bank" is displayed in green, followed by the instruction "Please call the bank to collect your blood". A separate section titled "Blood Bank" provides detailed information: ABC Blood Bank, Location: Chennai, and Unit: 2.

Blood request form

A screenshot of a web browser window titled "Request Blood". The page has a light blue header with the "Donate" logo and navigation links for "Blood Request Status" and "Login". The main content area is titled "Request Blood". It contains several input fields:

- Requester name: "Blood Requester Name" (placeholder)
- Email: "Blood Requester Email" (placeholder)
- Mobile Number: "0" (placeholder)
- Blood Type: "A+ve" (placeholder)
- Unit: "0" (placeholder)
- Blood Requester Age: "1" (placeholder)

The browser's address bar shows "localhost:3000/addbloodrequest". The taskbar at the bottom includes icons for File, Home, Mail, Microsoft Edge, and Google Chrome.

Show error message with no data entry

A screenshot of a web browser window titled "Request Blood". The page has a light blue header with the "Donate" logo and navigation links for "Blood Request Status" and "Login". The main content area is titled "Request Blood". It contains several input fields, each with a red error message indicating it is required:

- Requester name: "0" (placeholder) - "Required!"
- Email: "A+ve" (placeholder) - "Required!"
- Mobile Number: "0" (placeholder) - "Required!"
- Blood Type: "1" (placeholder) - "Required!"
- Unit: "Location" (placeholder) - "Required!"
- Blood Requester Age: "0" (placeholder) - "Required!"
- Aadhaar Number: "0" (placeholder) - "Required!"

The browser's address bar shows "localhost:3000/addbloodrequest". The taskbar at the bottom includes icons for File, Home, Mail, Microsoft Edge, and Google Chrome.

Registration page for blood bank and hospital

A screenshot of a web browser window showing a registration form. The title bar indicates the page is at `localhost:3000/register`. The form has a blue header with the word "Donate". The main section is titled "Registration". It contains fields for "Organization Name" (with placeholder "Name"), "Email" (placeholder "Email"), "Mobile Number" (placeholder "0"), "Government Id" (placeholder "Government Id"), "Upload Document" (button "Choose File" with message "No file chosen"), and "Location" (placeholder "Location"). The browser's address bar shows "Type here to search" and the taskbar at the bottom includes icons for various applications like File Explorer, Microsoft Word, and Google Chrome.

Show error message with no data entry

A screenshot of a web browser window showing a registration form with validation errors. The title bar indicates the page is at `localhost:3000/register`. The form has a blue header with the word "Donate". The main section is titled "Registration". It contains fields for "Area" (placeholder "Area", error message "Required!"), "District" (placeholder "District", error message "Required!"), "State" (placeholder "State", error message "Required!"), "Postal Code" (placeholder "Postal Code", error message "Required!"), and "Password" (placeholder "Password", error message "Required!"). A "Register" button is at the bottom. The browser's address bar shows "Type here to search" and the taskbar at the bottom includes icons for various applications like File Explorer, Microsoft Word, and Google Chrome.

Registration page for donor

A screenshot of a web browser window titled "Donor". The main content area is titled "Donor Registration". It contains the following form fields:

- Donor name: Name
- Email: Email
- Mobile Number: 0
- Blood Type: A+ve
- Age: 15
- Location: Location

The browser's address bar shows "localhost:3000/newdonorrequest". The system status bar at the bottom right indicates "1829 ENG 30-03-2024".

Show error message with no data entry

A screenshot of a web browser window titled "Donor". The main content area is titled "Donor Registration". It contains the following form fields, each with a red "Required" error message:

- Street: Street
**Required!
- Area: Area
**Required!
- District: District
**Required!
- State: State
**Required!
- Postal Code: Postal Code
**Required!
- Password: Password
**Required!

The browser's address bar shows "localhost:3000/newdonorrequest". The system status bar at the bottom right indicates "1829 ENG 30-03-2024".

Database

Account

AccountId	Name	Email	Password	PhoneNumber	Status
10bab100-d6b3-41d0-9d3b-46491cda7ca0	ABC Blood Bank	sanjalhari2002@gmail.com	ceb200a6cea4095b86c8bc4d1e801aff9812eaa...	6382202487	1
26e36c6a-0c9a-48e2-8eac-0296706b2604	ABC Hospital	smano4570@gmail.com	ceb200a6cea4095b86c8bc4d1e801aff9812eaa...	6382202586	1
3cb987c-841f-4cd2-9c6b-46b45fe22ea6	Vasan Blood bank	keerthivasan2809201@gmail.com	ceb200a6cea4095b86c8bc4d1e801aff9812eaa...	9790953230	1
477b7d70-299c-4428-a130-7b5bf5176586	Keerthi Vasan	keerthivasan280901@gmail.com	ceb200a6cea4095b86c8bc4d1e801aff9812eaa...	9089678776	1
5f20561c-8175-4826-81a8-3f9ea108c2aa	S Lab	arunsrinivasan33@gmail.com	ceb200a6cea4095b86c8bc4d1e801aff9812eaa...	9878675432	1

Role

	RoleId	RoleName
►	08e882cc-d50f-41c8-97e9-9ddfe1a8375d	ADMIN
	1cc06485-e084-447a-a138-cd5af7751a14	DONOR
	1ee71642-e3a4-4605-b53c-873624279e7b	HOSPITAL
	d9157f05-ab6e-49de-b404-ab9b53b1e4e2	BLOODBANK

Account role

AccountRoleId	AccountId	RoleId
1766ba8d-742b-41d7-8e65-a3e3f6e40fe	10bab100-d6b3-41d0-9d3b-46491cda7ca0	d9157f05-ab6e-49de-b404-ab9b53b1e4e2
1a690a83-a006-4ad8-ae14-127fa682b63a	26e36c6a-0c9a-48e2-8eac-0296706b2604	1ee71642-e3a4-4605-b53c-873624279e7b
5091cb80-d24d-48bf-86f0-daaa2e78d347	62a5ec56-1022-43a8-b125-e9ada4a7ba30	08e882cc-d50f-41c8-97e9-9ddfe1a8375d
9e21a3c9-68d6-4c66-94ef-afb78846043	477b7d70-299c-4428-a130-7b5bf5176586	1cc06485-e084-447a-a138-cd5af7751a14

Address

AddressId	DoorNo	Street	Area	City	State	PostalCode
1248b1df-df9d-4ebf-8044-73728397f1bf	28	Kct College	Vanarpettai	Tirunelveli	Tamil nadu	687300
3ab9743d-1900-4a1c-b575-cd5e19d6bdc4						
4b52532d-8e26-4d93-b6c1-aa5793b33cec	21	D street	D area	Dharmapuri	Tamilnadu	647001
6926a1fb-8e34-4ef4-b7e2-a948a60c4a5e	21/2	Salai Street,	Ktc Nagar	Tirunelveli	Tamil Nadu	627011
76ce5ff9-35d5-40ee-96de-4342a78dee5	34	K Street	Vanarpettai	Viruthunagar	TamilNadu	687300

UserDetails

UserDetailsId	Age	BloodType	Location	GovernmentId	Document	AadhaarNumber	rolestatus
09eaed17-49bc-47d3-9795-5671d3ae18b9	0		Chennai	BLOB	234598762345	0	
1b60eba4-66f9-496f-a764-7afb10d7ee87	21	B+ve	Chennai	BLOB	345623458900	3	
7032d920-e6bb-45e3-af07-8b40ea7575b3	0		Chennai	1245433	BLOB	0	2
75762986-e018-48c5-a001-0f3627a1fc19	0		Viruthunagar	TN00132233	BLOB	298756784566	1
a150cf2-f3d5-4752-a391-6173ffb7fd2	0		Chennai	1245433	BLOB	0	1

Link between userdetails , address , account

AccountUserDetailsAddressId	AccountId	UserDetailsId	AddressId
3056ac14-9ab6-4dda-abb9-41317b5b7d40	5f20561c-8175-4826-81a8-3f9ea108c2aa	75762986-e018-48c5-a001-0f3627a1fc19	76ce5ff9-35d5-40ee-96de-4342a78dec5
3d052f0d-2b70-4030-ab6e-c9c79cd2fbff	3cb987fc-841f-4cd2-9cb6-46b45fe22ea6	ed3d72fc-c081-443e-adc1-45c3c02ca6a4	1248b1df-df9d-4ebf-8044-737283971bf
4c3b4f11-3c80-452f-91af-6a45ed029035	26e36c6a-0c9a-48c2-8eac-0296706b2604	7032d920-e6bb-45e3-a0f7-8b40ea7575b3	6926a1fb-e834-4ef4-b7e2-a948a60c4a5e
50fa9fae-bfb2-4e5c-a219-2ebdc1fb4f47	477b7d70-299c-4428-a130-7b5bf5176586	1b60eba4-66f9-496f-a764-7afb10d7ee87	b6872d86-1eb1-4fc4-b2ad-3f2c798440b7
c5c7a9e-db93-4266-8137-a3042498872e	10bab100-d6b3-41d0-9d3b-46491cda7ca0	a150cf2-f3d5-4752-a391-6173fb7fd2	db99c12e-3241-468d-b5d9-977d38b7c702

Blood camp

	BloodCampId	BloodCampName	BloodCampLocation	Date	Time
▶	4bd17a67-2eb9-43e7-b7fb-c76a51f8d237	Blood Sevaa	Viruthunagar	21/05/2024	11:00
▶	791d6770-e7bd-4192-8bd2-94787b9b7507	Donate Seva	Tirunelveli	23/05/2024	10:00
*	NULL	NULL	NULL	NULL	NULL

Link blood camp and blood bank

Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content:		
BloodCampBloodBankId	BloodCampId	AccountId
4bb41fea-362f-4c49-95fd-0ba570e9d858	4bd17a67-2eb9-43e7-b7fb-c76a51f8d237	10bab100-d6b3-41d0-9d3b-46491cda7ca0
b8ab560b-03d4-4f00-a593-72d66f64510b	791d6770-e7bd-4192-8bd2-94787b9b7507	10bab100-d6b3-41d0-9d3b-46491cda7ca0
NULL	NULL	NULL

Blood Request

	BloodRequestId	Name	Email	PhoneNumber	BloodType	Age	Location	AadhaarNumber	ValidTime	Status	AcceptStatus	U
▶	4aOCW9V	Mano	keerthivasan280901@gmail.com	9790953230	B+ve	19	Tirunelveli	909090909079	03/29/2024 16:43:07	4	1	2
QcLtyYFJ	Mano	smano4570@gmail.com		9790953230	AB+ve	1	Chennai	234554320987		1	0	1
SP39NVE	Vasan	senathipathik@gmail.com		9876543210	A-ve	35	Viruthunagar	345432433222	04/01/2024 08:40:20	2	1	3
T2937kCU	Sanjai	sanjairock85@gmail.com		9867541243	B+ve	20	Chennai	235489098762	03/29/2024 17:11:20	1	1	2

Blood transaction

	BloodTransactionId	AccountId	Date	Time	BloodType	units	BloodRequestId
▶	8bd10343-2d3c-4675-9f74-daebe35c432c	10bab100-d6b3-41d0-9d3b-46491cda7ca0	2024-02-21	10:00 Am	A-ve	1	4aOCW9V
cbbf7fec-2ebf-45b3-af88-12729cb4fa41	10bab100-d6b3-41d0-9d3b-46491cda7ca0	NULL	2024-01-21	1:00 Pm	B+ve	2	T2937kCU
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Blood stock

	BloodStockId	BloodType	Units	AccountId
▶	2e219639-4946-465b-a58e-59b62e2eb46f	A+ve	1	10bab100-d6b3-41d0-9d3b-46491cda7ca0
411c93d3-2c48-47ce-86eb-51af420ebbce	AB+ve	2	10bab100-d6b3-41d0-9d3b-46491cda7ca0	
52143f5a-e264-4a0e-82cd-3f7d8f12cdd2	B+ve	2	10bab100-d6b3-41d0-9d3b-46491cda7ca0	
d4c1e8fd-7acd-41a7-a2c9-1a40c21849c6	A-ve	96	10bab100-d6b3-41d0-9d3b-46491cda7ca0	

BANK END

Backend report

1.Coding

2. Output in swagger (screenshot)

Git Link - <https://github.com/smano-20052002/MicroProjectWebapi.git>

Coding :

DataContext :

AppDbContext.cs

```
using Microsoft.EntityFrameworkCore;
using BloodBankManagementWebapi.Model;
namespace BloodBankManagementWebapi.DataContext
{
    public class AppDbContext:DbContext
    {
        public AppDbContext(DbContextOptions<AppDbContext> options) : base(options) { }

        public DbSet<Account> Account{ get; set; }

        public DbSet<Role> Role { get; set; }

        public DbSet<AccountRole> AccountRole { get; set; }

        public DbSet<Address> Address { get; set; }

        public DbSet<UserDetails> UserDetails { get; set; }

        public DbSet<BloodCamp> BloodCamp { get; set; }
```

```
public DbSet<BloodStock> BloodStock { get; set; }

public DbSet<BloodTransaction> BloodTransaction { get; set; }

public DbSet<BloodRequest> BloodRequest { get; set; }

public DbSet<BloodStockRequester> bloodStockRequesters { get; set; }

public DbSet<AccountUserDetailsAddress> AccountUserDetailsAddress { get; set; }

public DbSet<BloodCampBloodBank> BloodCampBloodBank { get; set; }

}

}

}
```

Appsettings.json

```
{

    "Logging": {

        "LogLevel": {

            "Default": "Information",

            "Microsoft.AspNetCore": "Warning"

        }

    },

    "AllowedHosts": "*",

    "ConnectionStrings": {

        "DefaultConnection": "server=localhost;user=root;password=admin123;database=BloodBankManagementSystem1.0;port=3306"

    },

    "JWT": {

        "ValidIssuer": "*",

        "ValidAudience": "*",

        "SecretKey": "bd1a1ccAS09R37f361a4Dd351e7c0de65f0776bfc278ea8d312c763bb6c#3ac!a=="

    }

}
```

Program.cs

```
using BloodBankManagementWebapi.DataContext;
using Microsoft.AspNetCore.Authentication.JwtBearer;
using Microsoft.EntityFrameworkCore;
using Microsoft.IdentityModel.Tokens;
using System.Text;

var builder = WebApplication.CreateBuilder(args);

var ConnectionString = builder.Configuration.GetConnectionString("DefaultConnection");

builder.Services.AddDbContext<AppDbContext>(options => options.UseMySql(ConnectionString, new MySqlServerVersion(new Version())));
builder.Services.AddControllers();
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen();

#region CORS setting for API

builder.Services.AddCors(options =>
{
    options.AddPolicy(name: "_myAllowSpecificOrigins",
        policy =>
    {
        policy.AllowAnyOrigin().AllowAnyHeader().AllowAnyMethod().AllowAnyMethod();
    }
);
});

#endregion

//start jwt configuration

builder.Services.AddAuthentication(options =>
{
    options.DefaultScheme = JwtBearerDefaults.AuthenticationScheme;
    options.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;
```

```
options.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;

})

.AddJwtBearer(options =>
{
    options.SaveToken = true;
    options.RequireHttpsMetadata = false;
    options.TokenValidationParameters = new TokenValidationParameters()
    {
        ValidateIssuer = true,
        ValidateAudience = true,
        ValidateLifetime = true,
        ValidIssuer = builder.Configuration["JWT:ValidIssuer"],
        ValidAudience = builder.Configuration["JWT:ValidAudience"],
        IssuerSigningKey = new
            SymmetricSecurityKey(Encoding.UTF8.GetBytes(builder.Configuration["JWT:SecretKey"]!))
    };
});

//end

var app = builder.Build();

app.UseSwagger();
app.UseSwaggerUI();

app.UseCors("_myAllowSpecificOrigins");

app.UseHttpsRedirection();
app.UseAuthentication();
app.UseRouting();
app.UseAuthorization();
app.MapControllers();

app.Run();
```

OtherOperation

SHA256Encrypt.cs

```
using System.Text;
using System.Security.Cryptography;
namespace BloodBankManagementWebapi.OtherOperation
{
    public static class SHA256Encrypt
    {
        public static string ComputePasswordToSha256Hash(string plainText)
        {
            using (SHA256 sha256Hash = SHA256.Create())
            {
                byte[] bytes = sha256Hash.ComputeHash(Encoding.UTF8.GetBytes(plainText));
                StringBuilder stringBuilder = new StringBuilder();
                for (int i = 0; i < bytes.Length; i++)
                {
                    stringBuilder.Append(bytes[i].ToString("x2"));
                }
                return stringBuilder.ToString();
            }
        }
    }
}
```

EmailGenerator.cs

```
using BloodBankManagementWebapi.Model;
using Google.Protobuf;
using Microsoft.AspNetCore.Http.HttpResults;
using Org.BouncyCastle.Asn1.Ocsp;
```

```
using System.Net;
using System.Net.Mail;
using System.Xml.Linq;

namespace BloodBankManagementWebapi.OtherOperation
{
    public static class EmailGenerator
    {
        public static bool SendEmail(string Username, string Mail, string Password)
        {

            string fromMail = "smano8312@gmail.com";
            string senderPass = "mktt mzmx pasy gdgl";
            MailMessage message = new MailMessage();
            message.From = new MailAddress(fromMail);
            message.To.Add(Mail);
            message.Subject = $"Confidential! New Password for Your {Mail} Accounts";

            message.Body = $"Dear {Username},\r\n\r\nThis is a notification from the management. Your new password for {Mail} accounts has been generated. Please take a moment to update your password.\r\n\r\nNew Password: {Password}\r\n\r\nRemember to change your password promptly for security reasons.\r\n\r\nThank you, Management Team";

            SmtpClient smtpClient = new SmtpClient("smtp.gmail.com");
            smtpClient.Port = 587;
            smtpClient.UseDefaultCredentials = false;
            smtpClient.Credentials = new NetworkCredential(fromMail, senderPass);
            smtpClient.EnableSsl = true;
            try
            {
                smtpClient.Send(message);
            }
        }
    }
}
```

```
        return true;

    }

    catch (Exception ex)

    {

        return false;

    }

}

public static bool SendEmail(string Username, string Mail, string Message, string RequestId, string ValidDate)

{

    string fromMail = "smano8312@gmail.com";

    string senderPass = "mktt mzmx pasy gdgl";

    MailMessage message = new MailMessage();

    message.From = new MailAddress(fromMail);

    message.To.Add(Mail);

    if (Message == "BloodRequestApprove")

    {

        message.Subject = $"Important! Blood Request is approved by management";

        message.Body = $"Dear {Username},\r\n\r\nThis is a notification from the management. Your Blood Request for Id {RequestId} is approved by management. Please check the website and it is valid for date {ValidDate}.\r\n\r\nThank you, Management Team";

    }

    else if (Message == "BloodRequestReject")

    {

        message.Subject = $"Important! Blood Request is rejected by management";

        message.Body = $"Dear {Username},\r\n\r\nThis is a notification from the management. Your Blood Request for Id {RequestId} is rejected by management. Please apply again with correct information.\r\n\r\nThank you, Management Team";

    }

}
```

```
SmtpClient smtpClient = new SmtpClient("smtp.gmail.com");
smtpClient.Port = 587;
smtpClient.UseDefaultCredentials = false;
smtpClient.Credentials = new NetworkCredential(fromMail, senderPass);
smtpClient.EnableSsl = true;

try
{
    smtpClient.Send(message);
    return true;
}

catch (Exception ex)
{
    return false;
}

}

public static bool SendEmailForAccountApproval(string Name, string Email, string ApprovalStatus)
{
    string fromMail = "smano8312@gmail.com";
    string senderPass = "mktt mzmx pasy gdgl";
    MailMessage message = new MailMessage();
    message.From = new MailAddress(fromMail);
    message.To.Add(Email);
    if (ApprovalStatus == "AccountApprove")
    {
        message.Subject = $"Important! Account is approved by management";
        message.Body = $"Dear {Name},\r\n\r\nThis is a notification from the management. Your Account is approved by management. Please check the website .\r\n\r\nThank you, Management Team";
    }
}
```

```
}

else if (ApprovalStatus == "AccountReject")

{

    message.Subject = $"Important! Account is rejected by management";

    message.Body = $"Dear {Name},\r\n\r\nThis is a notification from the management. Your Account is rejected by management.Please check the website .\r\n\r\nThank you, Management Team";


}

SmtpClient smtpClient = new SmtpClient("smtp.gmail.com");

smtpClient.Port = 587;

smtpClient.UseDefaultCredentials = false;

smtpClient.Credentials = new NetworkCredential(fromMail, senderPass);

smtpClient.EnableSsl = true;

try

{

    smtpClient.Send(message);

    return true;

}

catch (Exception ex)

{

    return false;

}

}
```

```
public static void SendEmailForAcceptRequest(string Email, string BloodRequestId, Account  
account, Address address, UserDetails userdetails)  
{  
    string fromMail = "smano8312@gmail.com";  
    string senderPass = "mktt mzmxt pasy gdgl";  
    MailMessage message = new MailMessage();  
    message.From = new MailAddress(fromMail);  
    message.To.Add(Email);  
    message.Subject = $"Important! Your Blood Request is accept by {account.Name}";  
  
    message.Body = $"Dear {account.Name},\r\n\r\nYour request is accepted by {account.Name}-  
{userdetails.Location}.Please check the bank and collect the blood.  
\nAddress :\n{address.DoorNo},\n{address.Street},\n{address.Area},\n{address.City},\n{address.State}-  
{address.PostalCode}.\r\n\r\nThank you, Management Team";
```

```
SmtpClient smtpClient = new SmtpClient("smtp.gmail.com");  
smtpClient.Port = 587;  
smtpClient.UseDefaultCredentials = false;  
smtpClient.Credentials = new NetworkCredential(fromMail, senderPass);  
smtpClient.EnableSsl = true;  
try  
{  
    smtpClient.Send(message);  
    return;  
}  
catch (Exception ex)  
{  
    return;  
}
```

```
    }  
}  
}
```

RandomPasswordGenerator.cs

```
using System.Text;  
  
namespace BloodBankManagementWebapi.OtherOperation  
{  
    public static class RandomPasswordGeneration  
    {  
        public static string RandomPasswordGenerator()  
        {  
            Random random = new Random();  
            int passwordLength = random.Next(8, 12);  
            const string validCharacters =  
"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890@#$&!?";  
            StringBuilder password = new StringBuilder();  
            for (int i = 0; i < passwordLength; i++)  
            {  
                int randomcode = random.Next(0, validCharacters.Length-1);  
                password.Append(validCharacters[randomcode]);  
            }  
            return password.ToString();  
        }  
    }  
}
```

Model

Account.cs

```
using System.ComponentModel;

namespace BloodBankManagementWebapi.Model

{

    public class Account

    {

        public string? AccountId { get; set; }

        public string? Name { get; set; }

        public string? Email { get; set; }

        public string? Password { get; set; }

        public long PhoneNumber { get; set; }

        [DefaultValue(0)]

        public int Status { get; set; } = 0;

        public BloodTransaction? BloodTransaction { get; }

        public BloodCampBloodBank? BloodCampBloodBank { get; }

        public BloodBankBloodStock? BloodBankBloodStock { get; }

    }

}
```

AccountRole.cs

```
namespace BloodBankManagementWebapi.Model

{

    public class AccountRole

    {

        public string? AccountRoleId { get; set; }

    }

}
```

```
    public Account? Account { get; set; }

    public Role? Role { get; set; }

}
```

```
}
```

AccountUserDetailsAddress.cs

```
namespace BloodBankManagementWebapi.Model
```

```
{
    public class AccountUserDetailsAddress
    {
        public string? AccountUserDetailsAddressId { get; set; }

        public Account? Account { get; set; }

        public UserDetails? UserDetails { get; set; }

        public Address? Address { get; set; }

    }
}
```

Address.cs

```
namespace BloodBankManagementWebapi.Model
```

```
{
    public class Address
    {
        public string? AddressId { get; set; }

        public string? DoorNo { get; set; }

        public string? Street { get; set; }

        public string? Area { get; set; }

        public string? City { get; set; }

    }
}
```

```
    public string? State { get; set; }

    public string? PostalCode { get; set; }

}

}
```

BloodCamp.cs

```
namespace BloodBankManagementWebapi.Model

{

    public class BloodCamp

    {

        public string? BloodCampId { get; set; }

        public string? BloodCampName { get; set; }

        public string? BloodCampLocation { get; set; }

        public string? Date { get; set; }

        public string? Time { get; set; }

        public BloodCampBloodBank? BloodCampBloodBank { get; }

    }

}
```

BloodCampBloodBank.cs

```
namespace BloodBankManagementWebapi.Model

{

    public class BloodCampBloodBank

    {

        public string? BloodCampBloodBankId { get; set; }

        public BloodCamp? BloodCamp { get; set; }

        public Account? Account { get; set; }

    }

}
```

```
}
```

BloodRequest.cs

```
using System.ComponentModel;  
namespace BloodBankManagementWebapi.Model  
{  
    public class BloodRequest  
    {  
        public string? BloodRequestId { get; set; }  
        public string? Name { get; set; }  
        public string? Email { get; set; }  
        public int Units { get; set; }  
        public long PhoneNumber { get; set; }  
        public string? BloodType { get; set; }  
        public int Age { get; set; }  
        public string? Location { get; set; }  
        public long AadhaarNumber { get; set; }  
        [DefaultValue(null)]  
        public string ValidTime { get; set; }  
        [DefaultValue(0)]  
        public int Status { get; set; }  
        public int AcceptStatus { get; set; }  
        public BloodTransaction BloodTransaction { get; }= null;  
    }  
}
```

BloodStock.cs

```
namespace BloodBankManagementWebapi.Model  
{
```

```
public class BloodStock
{
    public string? BloodStockId { get; set; }

    public string? BloodType { get; set; }

    public int Units { get; set; }

    public Account? Account { get; set; }
}
```

BloodStockRequester.cs

```
namespace BloodBankManagementWebapi.Model
{
    public class BloodStockRequester
    {
        public string? BloodStockRequestId { get; set; }

        public BloodRequest BloodRequest { get; set; }

        public Account Account { get; set; }

        public UserDetails UserDetails { get; set; }

        public Address Address { get; set; }
    }
}
```

BloodTransaction.cs

```
namespace BloodBankManagementWebapi.Model
{
    public class BloodTransaction
    {
        public string? BloodTransactionId { get; set; }
```

```
public string? AccountId { get; set; }

public Account Account { get; set; }

public string? BloodRequestId { get; set; }

public BloodRequest BloodRequest { get; set; }

public string? BloodType { get; set; }

public int units { get; set; }

public string? Date { get; set; }

public string? Time { get; set; }

}

}
```

Role.cs

```
namespace BloodBankManagementWebapi.Model

{

    public class Role

    {

        public string? RoleId { get; set; }

        public string? RoleName { get; set; }

    }

}
```

UserDetails.cs

```
namespace BloodBankManagementWebapi.Model

{

    public class UserDetails

    {

        public string? UserDetailsId { get; set; }

        public int Age { get; set; }

        public string? BloodType { get; set; }

    }

}
```

```
    public string? Location { get; set; }

    public string? GovernmentId { get; set; }

    public byte[] Document { get; set; }

    public long AadhaarNumber { get; set; }

    public int rolestatus { get; set; }

}

}
```

Controller

AcceptRequesterByBankController.cs

```
using BloodBankManagementWebapi.ApiModel;

using BloodBankManagementWebapi.DataContext;

using BloodBankManagementWebapi.Model;

using BloodBankManagementWebapi.OtherOperation;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

namespace BloodBankManagementWebapi.Controllers

{

    [Route("api/[controller]")]

    [ApiController]

    public class AcceptRequesterByBankController : ControllerBase

    {

        private readonly AppDbContext _context;

        public AcceptRequesterByBankController(AppDbContext context)

        {

            _context = context;

        }

    }
```

```
[HttpPost]
```

```
public IActionResult Post([FromBody] AcceptRequester acceptRequester)
```

```
{
```

```
    var requester = _context.BloodRequest.Find(acceptRequester.RequesterId);
```

```
    requester.AcceptStatus = 1;
```

```
    _context.BloodRequest.Update(requester);
```

```
    _context.SaveChanges();
```

```
    var account = _context.Account.Find(acceptRequester.AccountId);
```

```
    var accountuserdetailsaddress = _context.AccountUserDetailsAddress.Include(x => x.Account).Include(x => x.Address).Include(x => x.UserDetails).Where(x => x.Account == account).FirstOrDefault();
```

```
    var userdetails = accountuserdetailsaddress.UserDetails;
```

```
    var address = accountuserdetailsaddress.Address;
```

```
    BloodStockRequester bloodStockRequester = new BloodStockRequester()
```

```
{
```

```
        BloodStockRequestId=Guid.NewGuid().ToString(),
```

```
        Account=account,
```

```
        UserDetails=userdetails,
```

```
        Address=address,
```

```
        BloodRequest= requester
```

```
};
```

```
    _context.bloodStockRequesters.Add(bloodStockRequester);
```

```
    _context.SaveChanges();
```

```
    EmailGenerator.SendEmailForAcceptRequest(requester.Email, requester.BloodRequestId, account, address, userdetails);
```

```
    return Ok();
```

```
}
```

```
 }  
 }
```

AdminDashboardController.cs

```
using BloodBankManagementWebapi.DataContext;  
using Microsoft.AspNetCore.Http;  
using Microsoft.AspNetCore.Mvc;  
using Microsoft.EntityFrameworkCore;  
using BloodBankManagementWebapi.ApiModel;  
namespace BloodBankManagementWebapi.Controllers  
{  
    [Route("api/[controller]/[action]")]  
    [ApiController]  
    public class AdminDashboardController : ControllerBase  
    {  
        private readonly AppDbContext _context;  
        public AdminDashboardController(AppDbContext context)  
        {  
            _context = context;  
        }  
        [HttpGet]  
        public ActionResult GetBasicDetails()  
        {  
            var Hospital = _context.AccountUserDetailsAddress.Include(x => x.UserDetails).Include(a =>  
                a.Account).Include(s => s.Address).Where(x => x.UserDetails.rolestatus == 2).Where(x =>  
                x.Account.Status != 0).Count();  
  
            var Donor = _context.AccountUserDetailsAddress.Include(x => x.UserDetails).Include(a =>  
                a.Account).Include(s => s.Address).ToList().Where(x => x.UserDetails.rolestatus == 3).Where(x =>  
                x.Account.Status != 0).Count();  
  
            var Bloodcamp = _context.BloodCampBloodBank.ToList().Count();  
        }  
}
```

```

var BloodBank = _context.AccountUserDetailsAddress.Include(x => x.UserDetails).Include(a =>
a.Account).Include(s => s.Address).Where(x => x.UserDetails.rolestatus == 1).Where(x =>
x.Account.Status != 0).Count();

var BloodRequest = _context.BloodRequest.ToList().Count();

var PendingBloodRequest = _context.BloodRequest.Where(x=>x.Status==0).ToList().Count();

var HospitalRequest = _context.AccountUserDetailsAddress.Include(x => x.UserDetails).Include(a
=> a.Account).Include(s => s.Address).Where(x => x.UserDetails.rolestatus == 2).Where(x =>
x.Account.Status == 0).Count();

var BloodBankPendingRequest = _context.AccountUserDetailsAddress.Include(x =>
x.UserDetails).Include(a => a.Account).Include(s => s.Address).Where(x => x.UserDetails.rolestatus ==
1).Where(x => x.Account.Status == 0).Count();

var DonorPendingRequest = _context.AccountUserDetailsAddress.Include(x =>
x.UserDetails).Include(a => a.Account).Include(s => s.Address).ToList().Where(x =>
x.UserDetails.rolestatus == 3).Where(x => x.Account.Status == 0).Count();

var HospitalPendingRequest = _context.AccountUserDetailsAddress.Include(x =>
x.UserDetails).Include(a => a.Account).Include(s => s.Address).Where(x => x.UserDetails.rolestatus ==
2).Where(x => x.Account.Status == 0).Count();

var Result=new Dashboard()

{

    Hospital=Hospital,
    Donor=Donor,
    BloodBank=BloodBank,
    Bloodcamp=Bloodcamp,
    BloodRequest=BloodRequest,
    HospitalRequest=HospitalRequest,
    BloodBankPendingRequest=BloodBankPendingRequest,
    DonorPendingRequest=DonorPendingRequest,
    HospitalPendingRequest =HospitalPendingRequest,
    PendingBloodRequest= PendingBloodRequest
};

return Ok(Result);
}

```

```

[HttpGet]
public IActionResult GetBloodStockDetails()
{
    var Apositive = _context.BloodStock.Where(x=>x.BloodType=="A+ve").Sum(x=>x.Units);
    var Bpositive = _context.BloodStock.Where(x => x.BloodType == "B+ve").Sum(x => x.Units);
    var Opositive = _context.BloodStock.Where(x => x.BloodType == "O+ve").Sum(x => x.Units);
    var ABpositive = _context.BloodStock.Where(x => x.BloodType == "AB+ve").Sum(x => x.Units);
    var Anegative = _context.BloodStock.Where(x => x.BloodType == "A-ve").Sum(x => x.Units);
    var Bnegative = _context.BloodStock.Where(x => x.BloodType == "B-ve").Sum(x => x.Units);
    var Onegative = _context.BloodStock.Where(x => x.BloodType == "O-ve").Sum(x => x.Units);
    var ABnegative = _context.BloodStock.Where(x => x.BloodType == "AB-ve").Sum(x => x.Units);

    BloodDashboard bloodDashboard = new BloodDashboard
    {
        ABnegative = ABnegative,
        Bnegative = Bnegative,
        Anegative = Anegative,
        Onegative = Onegative,
        Opositive = Opositive,
        ABpositive = ABpositive,
        Apositive = Apositive,
        Bpositive = Bpositive,
    };
    return Ok(bloodDashboard);
}

[HttpGet]
public IActionResult GetBloodStockDetailsByIndividualBank(string Id)
{
    var account= _context.Account.Find(Id);
}

```

```

    var Apositive = _context.BloodStock.Include(x => x.Account).Where(x => x.Account == account).Where(x => x.BloodType == "A+ve").Sum(x => x.Units);

    var Bpositive = _context.BloodStock.Include(x => x.Account).Where(x => x.Account == account).Where(x => x.BloodType == "B+ve").Sum(x => x.Units);

    var Opositive = _context.BloodStock.Include(x => x.Account).Where(x => x.Account == account).Where(x => x.BloodType == "O+ve").Sum(x => x.Units);

    var ABpositive = _context.BloodStock.Include(x => x.Account).Where(x => x.Account == account).Where(x => x.BloodType == "AB+ve").Sum(x => x.Units);

    var Anegative = _context.BloodStock.Include(x => x.Account).Where(x => x.Account == account).Where(x => x.BloodType == "A-ve").Sum(x => x.Units);

    var ABnegative = _context.BloodStock.Include(x => x.Account).Where(x => x.Account == account).Where(x => x.BloodType == "AB-ve").Sum(x => x.Units);

    var Onegative = _context.BloodStock.Include(x => x.Account).Where(x => x.Account == account).Where(x => x.BloodType == "O-ve").Sum(x => x.Units);

    var Bnegative = _context.BloodStock.Include(x => x.Account).Where(x => x.Account == account).Where(x => x.BloodType == "B-ve").Sum(x => x.Units);

    BloodDashboard bloodDashboard = new BloodDashboard

    {
        ABnegative = ABnegative,
        Bnegative = Bnegative,
        Anegative = Anegative,
        Onegative = Onegative,
        Opositive = Opositive,
        ABpositive = ABpositive,
        Apositive = Apositive,
        Bpositive = Bpositive,
    };
    return Ok(bloodDashboard);
}
}
}

```

ApproveBloodRequestByAdmin.cs

```
using BloodBankManagementWebapi.ApiModel;
using BloodBankManagementWebapi.Model;
using BloodBankManagementWebapi.DataContext;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using BloodBankManagementWebapi.OtherOperation;
using Microsoft.VisualStudio.Web.CodeGenerators.Mvc.Templates.BazorIdentity.Pages.Manage;
using Org.BouncyCastle.Asn1.Ocsp;

namespace BloodBankManagementWebapi.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class ApproveBloodRequestByAdminController : ControllerBase
    {
        private readonly AppDbContext _context;
        public ApproveBloodRequestByAdminController(AppDbContext context)
        {
            _context = context;
        }
        [HttpPost]
        public IActionResult ApproveOrRequestBloodRequest([FromBody] ApproveOrReject
approveOrReject)
        {
            ApproveOrRejectReturnMessage message = null;
            if(approveOrReject.Id==null
|| !_context.BloodRequest.Any(s=>s.BloodRequestId==approveOrReject.Id))
```

```

{
    message = new ApproveOrRejectReturnMessage()

    {
        ValidEmail= false,
        IdExists= false,
        ChangeStatus =false
    };

    return Ok(message);
}

BloodRequest oldRequest=_context.BloodRequest.Where(s => s.BloodRequestId ==
approveOrReject.Id).FirstOrDefault()!;

if(approveOrReject.Status == true)

{
    oldRequest.Status = 1;
    oldRequest.ValidTime=DateTime.Now.AddDays(2).ToString();
    _context.BloodRequest.Update(oldRequest);

    if>EmailGenerator.SendEmail(oldRequest.Name, oldRequest.Email, "BloodRequestApprove",
oldRequest.BloodRequestId, oldRequest.ValidTime))

    {
        if (_context.SaveChanges()>0)

        {
            message = new ApproveOrRejectReturnMessage()

            {
                ValidEmail = true,
                IdExists = true,
                ChangeStatus = true
            };

            return Ok(message);
        }
    }
}

```

```
        else

    {

        message = new ApproveOrRejectReturnMessage()

        {

            ValidEmail = true,

            IdExists = true,

            ChangeStatus = false

        };

        return Ok(message);

    }

}

else

{

    message = new ApproveOrRejectReturnMessage()

    {

        ValidEmail = false,

        IdExists = true,

        ChangeStatus = false

    };

    return Ok(message);

}

}

else

{

    oldRequest.Status = 0;

    _context.BloodRequest.Update(oldRequest);

    if (EmailGenerator.SendEmail(oldRequest.Name, oldRequest.Email, "BloodRequestReject",

oldRequest.BloodRequestId, oldRequest.ValidTime))
```

```
{  
    if (_context.SaveChanges() > 0)  
    {  
        message = new ApproveOrRejectReturnMessage()  
        {  
            ValidEmail = true,  
            IdExists = true,  
            ChangeStatus = true  
        };  
        return Ok(message);  
  
    }  
    else  
    {  
        message = new ApproveOrRejectReturnMessage()  
        {  
            ValidEmail = true,  
            IdExists = true,  
            ChangeStatus = false  
        };  
        return Ok(message);  
    }  
}  
else  
{  
    message = new ApproveOrRejectReturnMessage()  
    {  
        ValidEmail = false,  
        IdExists = true,  
    }  
}
```

```
        ChangeStatus = false  
    };  
  
    return Ok(message);  
}  
}  
}  
}
```

ApproveOrRejectAccountByAdminController.cs

```
using BloodBankManagementWebapi.ApiModel;  
using BloodBankManagementWebapi.DataContext;  
using BloodBankManagementWebapi.Model;  
using BloodBankManagementWebapi.OtherOperation;  
using Microsoft.AspNetCore.Http;  
using Microsoft.AspNetCore.Mvc;  
namespace BloodBankManagementWebapi.Controllers  
{  
    [Route("api/[controller]")]  
    [ApiController]  
    public class ApproveOrRejectAccountByAdminController : ControllerBase  
    {  
        private readonly AppDbContext _context;  
        public ApproveOrRejectAccountByAdminController(AppDbContext context)  
        {  
            _context = context;  
        }  
  
        [HttpPost]  
        [Route("ApproveOrRejectAccount/{id}/{status}")]  
        public IActionResult ApproveOrRejectAccount(int id, string status)  
        {  
            var account = _context.Accounts.Find(id);  
            if (account != null)  
            {  
                account.Status = status == "Approved" ? true : false;  
                _context.SaveChanges();  
                return Ok("Account status updated successfully");  
            }  
            else  
            {  
                return NotFound("Account not found");  
            }  
        }  
    }  
}
```

```

public IActionResult ApproveOrRequestAccount([FromBody] ApproveOrReject approveOrReject)
{
    ApproveOrRejectReturnMessage message = null;
    if (approveOrReject.Id == null || !_context.Account.Any(s => s.AccountId == approveOrReject.Id))
    {
        message = new ApproveOrRejectReturnMessage()
        {
            ValidEmail = false,
            IdExists = false,
            ChangeStatus = false
        };
        return Ok(message);
    }

    Account oldRequest = _context.Account.Where(s => s.AccountId ==
approveOrReject.Id).FirstOrDefault()!;
    if (approveOrReject.Status == true)
    {
        oldRequest.Status = 1;
        _context.Account.Update(oldRequest);
        if (EmailGenerator.SendEmailForAccountApproval(oldRequest.Name, oldRequest.Email,
"AccountApprove"))
        {
            if (_context.SaveChanges() > 0)
            {
                message = new ApproveOrRejectReturnMessage()
                {
                    ValidEmail = true,
                    IdExists = true,
                    ChangeStatus = true
                };
            }
        }
    }
}

```

```
};

return Ok(message);

}

else

{

    message = new ApproveOrRejectReturnMessage()

    {

        ValidEmail = true,

        IdExists = true,

        ChangeStatus = false

    };

    return Ok(message);

}

}

else

{

    message = new ApproveOrRejectReturnMessage()

    {

        ValidEmail = false,

        IdExists = true,

        ChangeStatus = false

    };

    return Ok(message);

}

}

else

{



    oldRequest.Status = 2;

    _context.Account.Update(oldRequest);
```

```
    if (EmailGenerator.SendEmailForAccountApproval(oldRequest.Name, oldRequest.Email,
"AccountReject"))

    {

        if (_context.SaveChanges() > 0)

        {

            message = new ApproveOrRejectReturnMessage()

            {

                ValidEmail = true,

                IdExists = true,

                ChangeStatus = true

            };

            return Ok(message);

        }

        else

        {

            message = new ApproveOrRejectReturnMessage()

            {

                ValidEmail = true,

                IdExists = true,

                ChangeStatus = false

           };

            return Ok(message);

        }

    }

    else

    {

        message = new ApproveOrRejectReturnMessage()

        {

            ValidEmail = false,
```

```

        IdExists = true,
        ChangeStatus = false
    };
    return Ok(message);
}
}
}
}

```

BloodCampController.cs

```

using BloodBankManagementWebapi.DataContext;
using BloodBankManagementWebapi.ApiModel;
using BloodBankManagementWebapi.Model;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
namespace BloodBankManagementWebapi.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class BloodCampController : ControllerBase
    {
        private readonly AppDbContext _context;
        public BloodCampController(AppDbContext context)
        {
            _context = context;
        }
        [HttpPost]
        public IActionResult AddBloodCamp([FromBody]BloodCampModel bloodCampModel)

```

```

{
    BloodCamp bloodCamp = new BloodCamp()
    {
        BloodCampId= Guid.NewGuid().ToString(),
        BloodCampName= bloodCampModel.BloodCampName,
        BloodCampLocation= bloodCampModel.BloodCampLocation,
        Date= bloodCampModel.Date,
        Time = bloodCampModel.Time
    };
    Account account = _context.Account.Find(bloodCampModel.AccountId);
    BloodCampBloodBank bloodCampBloodBank = new BloodCampBloodBank()
    {
        BloodCampBloodBankId = Guid.NewGuid().ToString(),
        BloodCamp = bloodCamp,
        Account = account
    };
    _context.BloodCamp.Add(bloodCamp);
    _context.BloodCampBloodBank.Add(bloodCampBloodBank);
    _context.SaveChanges();
    return Ok();
}
}
}

```

BloodRequestController.cs

```

using BloodBankManagementWebapi.DataContext;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using BloodBankManagementWebapi.Model;

```

```
using BloodBankManagementWebapi.ApiModel;
using Microsoft.EntityFrameworkCore;
namespace BloodBankManagementWebapi.Controllers
{
    [Route("api/[controller]/[action]")]
    [ApiController]
    public class BloodRequestController : ControllerBase
    {
        private readonly AppDbContext _context;
        public BloodRequestController(AppDbContext context)
        {
            _context = context;
        }
        [HttpPost]
        public IActionResult RequestBlood([FromBody] BloodRequestDto bloodRequestDto)
        {
            Random rand = new Random();
            if (bloodRequestDto == null)
            {
                return BadRequest();
            }
            BloodRequest bloodRequest= new BloodRequest()
            {
                BloodRequestId = Convert.ToBase64String(Guid.NewGuid().ToByteArray()).Replace("/", "-").Replace("+", "-").Substring(rand.Next(0, 10), 8),
                Name=bloodRequestDto.Name,
                Email=bloodRequestDto.Email,
                PhoneNumber=bloodRequestDto.PhoneNumber,
                BloodType=bloodRequestDto.BloodType,
            }
        }
    }
}
```

```

Age=bloodRequestDto.Age,
Location=bloodRequestDto.Location,
Units=bloodRequestDto.Units,
AcceptStatus=0,
AadhaarNumber=bloodRequestDto.AadhaarNumber,
ValidTime=bloodRequestDto.ValidTime,
Status = bloodRequestDto.Status
};

_context.BloodRequest.Add(bloodRequest);
if (_context.SaveChanges()>0)
{
    return Ok(bloodRequest.BloodRequestId);
}
return BadRequest();
}

[HttpPost]
public IActionResult CheckStatus([FromBody]CheckRequest Request)
{
    var bloodrequest = _context.BloodRequest.Find(Request.Id);
    CheckBloodRequestStatusMessage message = null;
    List<BloodBankStock> bloodstock = new List<BloodBankStock>();
    if (Request == null || !_context.BloodRequest.Any(s=>s.BloodRequestId==Request.Id))
    {
        message = new CheckBloodRequestStatusMessage()
        {
            IdExists = false,
            Status = null,
            bloodRequestBloodBank= bloodstock
        };
    }
}

```

```

        return Ok(message);

    }

    BloodRequest bloodRequest=_context.BloodRequest.Find(Request.Id);

    var BloodBank = _context.AccountUserDetailsAddress.Include(x => x.UserDetails).Include(a =>
a.Account).Include(s => s.Address).Where(x => x.UserDetails.rolestatus == 1).Where(x =>
x.Account.Status != 0).Where(x => x.UserDetails.Location == bloodrequest!.Location).Select(l => new
BloodRequestBloodBank

    {

        Id = l.Account.AccountId,
        Name = l.Account.Name,
        Location = l.UserDetails.Location
    }).ToList();

    if (bloodRequest.Status == 1)

    {

        foreach (var bank in BloodBank)

        {

            var o = _context.BloodStock.Include(c => c.Account).Where(c => c.BloodType ==
bloodrequest.BloodType).Where(d => d.Account.AccountId == bank.Id).ToList().Select(p => new
BloodBankStock

            {

                BloodBank = p.Account.Name,
                Location = bank.Location,
                Units = p.Units
            }).FirstOrDefault();

            bloodstock.Add(o);
        };
    }

    message = new CheckBloodRequestStatusMessage()

    {

        IdExists = true,
        Status="approved",
    }

```

```
        bloodRequestBloodBank= bloodstock

    };

    return Ok(message);

}

else if(bloodRequest.Status == 2)

{

    message = new CheckBloodRequestStatusMessage()

    {

        IdExists = true,

        Status = "rejected",

        bloodRequestBloodBank = bloodstock

    };

    return Ok(message);

}

else

{

    message = new CheckBloodRequestStatusMessage()

    {

        IdExists = true,

        Status = "pending",

        bloodRequestBloodBank = bloodstock

    };

    return Ok(message);

}

}
```

```
using BloodBankManagementWebapi.ApiModel;
using BloodBankManagementWebapi.DataContext;
using BloodBankManagementWebapi.Model;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
namespace BloodBankManagementWebapi.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class BloodStockController : ControllerBase
    {
        private readonly AppDbContext _context;
        public BloodStockController(AppDbContext context)
        {
            _context = context;
        }
        [HttpPost]
        public IActionResult AddBloodStock([FromBody] BloodStockDto bloodStockDto)
        {
            var bloodstockcount = _context.BloodStock.Include(z => z.Account).Where(x => x.BloodType == bloodStockDto.BloodType).Where(i => i.Account.AccountId == bloodStockDto.AccountId).Count();
            if (bloodstockcount == 0)
            {
                Account account = _context.Account.Find(bloodStockDto.AccountId)!;
                BloodStock bloodStock = new BloodStock()
                {
                    BloodStockId = Guid.NewGuid().ToString(),
                    BloodType = bloodStockDto.BloodType,
                    AccountId = account.Id
                };
                _context.BloodStock.Add(bloodStock);
                _context.SaveChanges();
                return Ok();
            }
            else
            {
                return Conflict();
            }
        }
    }
}
```

```

        Units = bloodStockDto.Units,
        Account= account
    };

    _context.BloodStock.Add(bloodStock);
    _context.SaveChanges();
    return Ok();
}

else
{
    var bloodstock = _context.BloodStock.Include(z => z.Account).Where(x => x.BloodType ==
bloodStockDto.BloodType).Where(i => i.Account.AccountId ==
bloodStockDto.AccountId).FirstOrDefault();

    var blood = _context.BloodStock.Where(x => x.BloodStockId ==
bloodstock.BloodStockId).FirstOrDefault();

    blood.Units += bloodStockDto.Units;
    _context.BloodStock.Update(blood);
    _context.SaveChanges();
    return Ok();
}
}

[Route("GetStockByBank/{id}/{type}")]
[HttpGet]
public IActionResult Get(string id,string type)
{
    var account = _context.Account.Find(id);
    var bloodstock = 0;

    var blood = _context.BloodStock.Include(a => a.Account).Where(e => e.Account ==
account).Where(d => d.BloodType == type).FirstOrDefault();

    if (blood == null)

```

```
{  
    bloodstock = 0;  
}  
  
else  
{  
    bloodstock=blood.Units;  
}  
  
return Ok(bloodstock);  
}  
}  
}
```

BloodTransactionController.cs

```
using BloodBankManagementWebapi.DataContext;  
using Microsoft.AspNetCore.Http;  
using Microsoft.AspNetCore.Mvc;  
using BloodBankManagementWebapi.ApiModel;  
using BloodBankManagementWebapi.Model;  
using Microsoft.EntityFrameworkCore;  
namespace BloodBankManagementWebapi.Controllers  
{  
    [Route("api/[controller]/[action]")]  
    [ApiController]  
    public class BloodTransactionController : ControllerBase  
    {  
        private readonly AppDbContext _context;  
        public BloodTransactionController(AppDbContext appDbContext)  
        {  
            _context = appDbContext;
```

```
}
```

```
[HttpPost]
```

```
public IActionResult AddBloodTransaction([FromBody]BloodTransactionModel  
bloodTransactionModel)
```

```
{
```

```
if(_context.BloodRequest.Any(x=>x.BloodRequestId==bloodTransactionModel.BloodRequestId)) {
```

```
    var account = _context.Account.Find(bloodTransactionModel.AccountId);
```

```
    var bloodRequest = _context.BloodRequest.Find(bloodTransactionModel.BloodRequestId);
```

```
    bloodRequest.Status = 4;
```

```
    _context.BloodRequest.Update(bloodRequest);
```

```
    _context.SaveChanges();
```

```
    BloodTransaction bloodTransaction = new BloodTransaction()
```

```
{
```

```
    BloodTransactionId = Guid.NewGuid().ToString(),
```

```
    BloodType = bloodTransactionModel.BloodType,
```

```
    units = bloodTransactionModel.units,
```

```
    Date = bloodTransactionModel.Date,
```

```
    Time = bloodTransactionModel.Time,
```

```
    Account = account,
```

```
    BloodRequest = bloodRequest
```

```
};
```

```
    var bloodStock = _context.BloodStock.Include(x => x.Account).Where(x => x.Account ==  
account).Where(x => x.BloodType == bloodTransactionModel.BloodType).Select(x => new BloodStock
```

```
{
```

```
    BloodStockId = x.BloodStockId,
```

```
    BloodType = x.BloodType,
```

```
    Units = x.Units
```

```
}).FirstOrDefault();
```

```
    bloodStock.Units = bloodStock.Units - bloodTransactionModel.units;
```

```

        _context.BloodStock.Update(bloodStock);

        _context.SaveChanges();

        _context.BloodTransaction.Add(bloodTransaction);

        _context.SaveChanges();

        return Ok();
    }

    return BadRequest();
}

[HttpPost]

public IActionResult GetBloodTransaction([FromBody]GetBloodTransaction getBloodTransaction)
{
    var bloodTransaction = _context.BloodTransaction.Include(x => x.Account).Include(x =>
x.BloodRequest).Where(x => x.Account.AccountId == getBloodTransaction.Id).ToList();

    return Ok(bloodTransaction);
}
}
}
}

```

ChangePasswordController.cs

```

using BloodBankManagementWebapi.DataContext;

using BloodBankManagementWebapi.Model;

using BloodBankManagementWebapi.OtherOperation;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using BloodBankManagementWebapi.ApiModel;

```

```

namespace BloodBankManagementWebapi.Controllers
{

```

```
[Route("api/[controller]")]
[ApiController]
public class ChangePasswordController : ControllerBase
{
    private readonly AppDbContext _context;
    public ChangePasswordController(AppDbContext context)
    {
        _context = context;
    }
    [HttpPost]
    public IActionResult ChangePassword([FromBody] ChangePassword changePassword)
    {
        ChangePasswordMessage changePasswordMessage;
        if (_context.Account.Any(user => user.Email == changePassword.Email))
        {
            Account olduser = _context.Account.Where(s => s.Email ==
changePassword.Email).FirstOrDefault()!;
            if (olduser.Password ==
SHA256Encrypt.ComputePasswordToSha256Hash(changePassword.OldPassword))
            {
                olduser.Password =
SHA256Encrypt.ComputePasswordToSha256Hash(changePassword.NewPassword);
                _context.Account.Update(olduser);
                _context.SaveChanges();
                changePasswordMessage = new ChangePasswordMessage()
                {
                    EmailExists = true,
                    Passcheck = true
                };
                return Ok(changePasswordMessage);
            }
        }
    }
}
```

```
    }

    else

    {

        changePasswordMessage = new ChangePasswordMessage()

        {

            EmailExists = true,

            Passcheck = false

        };

        return Ok(changePasswordMessage);

    }

}

else

{

    changePasswordMessage = new ChangePasswordMessage()

    {

        EmailExists = false,

        Passcheck = false

    };

    return Ok(changePasswordMessage);

}

}

}
```

DeleteBloodCampController.cs

```
using BloodBankManagementWebapi.DataContext;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;
```

```

namespace BloodBankManagementWebapi.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class DeleteBloodCampController : ControllerBase
    {
        private readonly AppDbContext _context;

        public DeleteBloodCampController(AppDbContext context)
        {
            _context = context;
        }

        [Route("/BloodCamp/{id}")]
        [HttpDelete]
        public IActionResult Delete(string id)
        {
            var bloodCampBank = _context.BloodCampBloodBank.Where(s => s.BloodCamp.BloodCampId == id).FirstOrDefault();

            var bloodCamp=_context.BloodCamp.Find(id);

            _context.BloodCampBloodBank.Remove(bloodCampBank);
            _context.BloodCamp.Remove(bloodCamp);
            _context.SaveChanges();

            return Ok();
        }
    }
}

```

ForgetPasswordController.cs

```
using BloodBankManagementWebapi.DataContext;
```

```
using BloodBankManagementWebapi.OtherOperation;
using Microsoft.AspNetCore.Http;
using BloodBankManagementWebapi.ApiModel;
using BloodBankManagementWebapi.Model;
using Microsoft.AspNetCore.Mvc;

namespace BloodBankManagementWebapi.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class ForgetPasswordController : ControllerBase
    {
        private readonly AppDbContext _context;
        public ForgetPasswordController(AppDbContext context)
        {
            _context= context;
        }
        [HttpPost]
        public IActionResult SendMailForPassword([FromBody] EmailClass Mail)
        {
            ForgetPasswordMessage message = null;
            if (!_context.Account.Any(s => s.Email == Mail.Email))
            {
                message = new ForgetPasswordMessage
                {
                    EmailExists = false,
                    SendMail = false
                };
                return Ok(message);
            }
        }
    }
}
```

```
}

string Password = RandomPasswordGeneration.RandomPasswordGenerator();

Account olduser = _context.Account.Where(s => s.Email == Mail.Email).FirstOrDefault()!;

olduser.Password = SHA256Encrypt.ComputePasswordToSha256Hash(Password);

//_context.Entry(olduser).State = EntityState.Detached;

_context.Account.Update(olduser);

_context.SaveChangesAsync();

if (EmailGenerator.SendEmail(olduser.Name, olduser.Email, Password))

{

    message = new ForgetPasswordMessage

    {

        EmailExists = true,

        SendMail = true

    };

    return Ok(message);

}

else

{

    message = new ForgetPasswordMessage

    {

        EmailExists = true,

        SendMail = false

    };

    return Ok(message);

}

}
```

LoginController.cs

```
using BloodBankManagementWebapi.ApiModel;
using BloodBankManagementWebapi.Model;
using BloodBankManagementWebapi.DataContext;
using BloodBankManagementWebapi.OtherOperation;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using System.Security.Claims;

namespace BloodBankManagementWebapi.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class LoginController : ControllerBase
    {
        private readonly AppDbContext _context;
        private readonly IConfiguration _configuration;
        public LoginController(AppDbContext context, IConfiguration configuration)
        {
            _context = context;
            _configuration = configuration;
        }
        [HttpPost]
        public IActionResult Login([FromBody] LoginModel loginModel)
        {
            AuthMessageModel authmessage;
            List<Claim> authClaim;
            if (loginModel != null)
            {

```

```

if (_context.Account.Any(user => user.Email == loginModel.Email))

{

    Account user = _context.Account.Where(user => user.Email ==
loginModel.Email).FirstOrDefault(); 

    if (user.Status==1)

    {

        if (user.Password ==

SHA256Encrypt.ComputePasswordToSha256Hash(loginModel.Password))

        {

            var accountroleid = _context.AccountRole.Where(userrole => userrole.Account == user)

                .Select(user => user.AccountRoleId)

                .FirstOrDefault()!;

            var roleid = _context.AccountRole.Where(ur => ur.AccountRoleId == accountroleid) // Filter

by userrole id

                .Select(ur => ur.Role.RoleId) // Select the role id

                .FirstOrDefault();

            authClaim = new List<Claim>()

            {

                new Claim("Id",user.AccountId),

                new Claim("Role",roleid!)

            };

            var token = Jwt.GenerateJWTToken(authClaim, _configuration);

            authmessage = new AuthMessageModel

            {

                AccountApproval = "approve",

                AccountExists = true,

                PasswordStatus = true,

                Token = token

           };

            return Ok(authmessage);
}

```

```

        }

        authmessage = new AuthMessageModel
        {

            AccountApproval = "approve",
            AccountExists = true,
            PasswordStatus = false,
            Token = null
        };

        return Ok(authmessage);
    }

    else if(user.Status==2)
    {
        if (user.Password ==
SHA256Encrypt.ComputePasswordToSha256Hash(loginModel.Password))

        {
            var accountroleid = _context.AccountRole.Where(userrole => userrole.Account == user)
                .Select(user => user.AccountRoleId)
                .FirstOrDefault();

            var roleid = _context.AccountRole.Where(ur => ur.AccountRoleId == accountroleid) // Filter
by userrole id
                .Select(ur => ur.Role.RoleId) // Select the role id
                .FirstOrDefault();

            authClaim = new List<Claim>()

            {
                new Claim("Id",user.AccountId),
                new Claim("Role",roleid!)
            };
        }

        var token = Jwt.GenerateJWTToken(authClaim, _configuration);
    }
}

```

```

authmessage = new AuthMessageModel

{
    AccountApproval = "reject",
    AccountExists = true,
    PasswordStatus = true,
    Token = token
};

return Ok(authmessage);
}

authmessage = new AuthMessageModel

{
    AccountApproval = "reject",
    AccountExists = true,
    PasswordStatus = false,
    Token = null
};

return Ok(authmessage);
}

else
{
    if (user.Password ==
SHA256Encrypt.ComputePasswordToSha256Hash(loginModel.Password))

    {
        var accountroleid = _context.AccountRole.Where(userrole => userrole.Account == user)
            .Select(user => user.AccountRoleId)
            .FirstOrDefault()!;

        var roleid = _context.AccountRole.Where(ur => ur.AccountRoleId == accountroleid) // Filter
by userrole id
            .Select(ur => ur.Role.RoleId) // Select the role id
    }
}

```

```
        .FirstOrDefault();

    authClaim = new List<Claim>()

    {
        new Claim("Id",user.AccountId),
        new Claim("Role",roleid!)
    };

    var token = Jwt.GenerateJWTToken(authClaim, _configuration);

    authmessage = new AuthMessageModel

    {

        AccountApproval = "pending",
        AccountExists = true,
        PasswordStatus = true,
        Token = token
    };

    return Ok(authmessage);
}

authmessage = new AuthMessageModel

{

    AccountApproval = "pending",
    AccountExists = true,
    PasswordStatus = false,
    Token = null
};

return Ok(authmessage);

}

authmessage = new AuthMessageModel

{
```

```

        AccountApproval = "pending",
        AccountExists = false,
        PasswordStatus = false,
        Token = null
    );
    return Ok(authmessage);
}

return BadRequest();

}

}

}

```

RegisterController.cs

```

using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using BloodBankManagementWebapi.DataContext;
using BloodBankManagementWebapi.OtherOperation;
using System.Security.Claims;
using BloodBankManagementWebapi.ApiModel;
using BloodBankManagementWebapi.Model;
using static System.Net.Mime.MediaTypeNames;
using System.Text;
namespace BloodBankManagementWebapi.Controllers
{
    [Route("api/[controller]/[action]")]
    [ApiController]
    public class RegisterController : ControllerBase
    {
        ...
    }
}

```

```
{  
    private readonly AppDbContext _context;  
    //private readonly IConfiguration _configuration;  
    public RegisterController(AppDbContext context)  
    {  
        _context = context;  
        // _configuration = configuration;  
    }  
    [HttpPost]  
  
    public IActionResult PostUser(RegisterModel registerUser)  
    {  
        SignUpMessageModel message = null;  
        if (_context.Account.Any(s => s.Email == registerUser.Email) || _context.Account.Any(s =>  
            s.PhoneNumber == registerUser.PhoneNumber))  
        {  
            if (_context.Account.Any(s => s.Email == registerUser.Email) && _context.Account.Any(s =>  
                s.PhoneNumber == registerUser.PhoneNumber))  
            {  
                message = new SignUpMessageModel()  
                {  
                    EmailExists = true,  
                    MobileNumberExists = true  
                };  
                return Ok(message);  
            }  
            if (_context.Account.Any(s => s.PhoneNumber == registerUser.PhoneNumber))  
            {  
                message = new SignUpMessageModel()  
            }  
        }  
    }  
}
```

```
{  
    EmailExists = false,  
    MobileNumberExists = true  
};  
  
return Ok(message);  
}  
  
message = new SignUpMessageModel()  
{  
    EmailExists = true,  
    MobileNumberExists = false  
};  
  
return Ok(message);  
}  
  
else  
{  
    Account account = new Account()  
{  
        AccountId = Guid.NewGuid().ToString(),  
        Name = registerUser.Name,  
        Email = registerUser.Email,  
        Password = SHA256Encrypt.ComputePasswordToSha256Hash(registerUser.Password),  
        PhoneNumber = registerUser.PhoneNumber,  
        Status = 0,  
    };  
  
    UserDetails userDetails = new UserDetails()  
{  
        UserDetailsId = Guid.NewGuid().ToString(),  
        Age = registerUser.Age,  
    };  
}
```

```
Location = registerUser.Location,  
GovernmentId = registerUser.GovernmentId,  
Document = Encoding.ASCII.GetBytes(registerUser.Document),  
AadhaarNumber = registerUser.AadhaarNumber,  
BloodType = registerUser.BloodType,  
  
};  
  
if (registerUser.Role=="ADMIN")  
{  
    userDetails.rolestatus = 0;  
}  
  
else if(registerUser.Role == "HOSPITAL")  
{  
    userDetails.rolestatus = 2;  
  
}  
  
else if (registerUser.Role == "DONOR")  
{  
    userDetails.rolestatus = 3;  
  
}  
  
else  
{  
    userDetails.rolestatus = 1;  
  
}  
  
Address address = new Address()
```

```
{  
    AddressId = Guid.NewGuid().ToString(),  
    City = registerUser.City,  
    DoorNo = registerUser.DoorNo,  
    Street = registerUser.Street,  
    State = registerUser.State,  
    PostalCode = registerUser.PostalCode,  
    Area = registerUser.Area,  
  
};  
  
AccountUserDetailsAddress accountUserDetailsAddress = new AccountUserDetailsAddress()  
{  
    AccountUserDetailsAddressId=Guid.NewGuid().ToString(),  
    Address = address,  
    Account=account,  
    UserDetails=userDetails  
};  
  
  
  
Role role = _context.Role.Where(x => x.RoleName == registerUser.Role).FirstOrDefault();  
AccountRole accountRole = new AccountRole()  
{  
    AccountRoleId = Guid.NewGuid().ToString(),  
    Role = role,  
    Account = account  
};  
_context.Account.Add(account);  
_context.UserDetails.Add(userDetails);  
_context.Address.Add(address);
```

```

_context.AccountUserDetailsAddress.Add(accountUserDetailsAddress);

_context.AccountRole.Add(accountRole);

_context.SaveChanges();

message = new SignUpMessageModel()

{

    EmailExists = false,

    MobileNumberExists = false

};

return Ok(message);

}

}

}

```

RoleController.cs

```

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using BloodBankManagementWebapi.DataContext;

using BloodBankManagementWebapi.Model;

namespace BloodBankManagementWebapi.Controllers

{

    [Route("api/[controller]")]

    [ApiController]

    public class RoleController : ControllerBase

    {

        private readonly AppDbContext _context;

        public RoleController(AppDbContext context)

```

```

{
    _context = context;
}

[HttpPost]

public IActionResult PostRole([FromBody] Role roles)
{
    Role roles1 = new Role()
    {
        RoleId = Guid.NewGuid().ToString(),
        RoleName = roles.RoleName.ToUpper()
    };
    _context.Role.Add(roles1);
    _context.SaveChanges();
    return Ok();
}

}

```

ViewAcceptedBankController.cs

```

using BloodBankManagementWebapi.DataContext;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using BloodBankManagementWebapi.Model;
using Microsoft.EntityFrameworkCore;

namespace BloodBankManagementWebapi.Controllers
{
    [Route("api/[controller]")]

```

```
[ApiController]

public class ViewAcceptedBankController : ControllerBase
{
    private readonly AppDbContext _context;

    public ViewAcceptedBankController(AppDbContext context)
    {
        _context = context;
    }

    [Route("/GetBank/{id}")]
    [HttpGet]
    public IActionResult GetBankbyId(string id)
    {
        if (_context.BloodRequest.Any(x=>x.AcceptStatus==1))
        {
            var requester = _context.BloodRequest.Find(id);

            var bank = _context.bloodStockRequesters.Include(x => x.UserDetails).Include(x => x.Account).Include(x => x.BloodRequest).Include(x => x.Address).Where(x => x.BloodRequest == requester).ToList();

            return Ok(bank);
        }
        else
        {
            return Ok(null);
        }
    }

}
}
```

ViewAccountRequestController.cs

```
using BloodBankManagementWebapi.DataContext;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using System.Text.Json.Serialization;
using System.Text.Json;
using BloodBankManagementWebapi.Model;
using Microsoft.IdentityModel.Tokens.Configuration;
using System.Text;
using BloodBankManagementWebapi.ApiModel;
```

```
namespace BloodBankManagementWebapi.Controllers
```

```
{
    [Route("api/[controller]/[action]")]
    [ApiController]
    public class ViewAccountRequestController : ControllerBase
    {
        private readonly AppDbContext _context;
        public ViewAccountRequestController(AppDbContext context)
        {
            _context = context;
        }
        [HttpGet]
        public IActionResult GetAccountDetails()
        {
```

```

    var accountDetails =
_context.AccountUserDetailsAddress.Include(x=>x.UserDetails).Include(a=>a.Account).Include(s=>s.Address).ToList();

    return Ok(accountDetails);
}

[HttpGet]

public IActionResult GetAccountDetailsById(string id)
{
    var accountDetails = _context.UserDetails.Select(l => new UserDetailsDto
    {
        Document = Encoding.ASCII.GetString(l.Document),
    });

    return Ok(accountDetails);
}

[HttpGet]

public IActionResult GetDonorById(string id)
{
    var account = _context.Account.Find(id);

    var accountDetails = _context.AccountUserDetailsAddress.Include(x => x.UserDetails).Include(a => a.Account).Include(s => s.Address).ToList().Where(x => x.UserDetails.rolestatus == 3).Where(x => x.Account.Status != 0).Where(x=>x.Account==account);

    return Ok(accountDetails);
}

//Approved Donor

[HttpGet]

public IActionResult GetAccountDetailsDonor()
{

```

```

        var accountDetails = _context.AccountUserDetailsAddress.Include(x => x.UserDetails).Include(a => a.Account).Include(s => s.Address).ToList().Where(x => x.UserDetails.rolestatus == 3).Where(x => x.Account.Status != 0);

        return Ok(accountDetails);
    }

    [HttpGet]
    public IActionResult GetAccountDetailsHospital()
    {

        var accountDetails = _context.AccountUserDetailsAddress.Include(x => x.UserDetails).Include(a => a.Account).Include(s => s.Address).Where(x => x.UserDetails.rolestatus == 2).Where(x => x.Account.Status != 0).Select(l => new HospitalBloodBankDetails
        {

            Id=l.Account.AccountId,
            Name = l.Account.Name,
            Email = l.Account.Email,
            PhoneNumber = l.Account.PhoneNumber,
            Location = l.UserDetails.Location,
            GovernmentId = l.UserDetails.GovernmentId,
            Document = Encoding.ASCII.GetString(l.UserDetails.Document),
            DoorNo = l.Address.DoorNo,
            Street =l.Address.Street,
            Area =l.Address.Area,
            City =l.Address.City,
            State =l.Address.State,
            PostalCode =l.Address.PostalCode,
            Status=l.Account.Status
        });
    }
}
```

```
});  
  
return Ok(accountDetails);  
}  
  
[HttpGet]  
public IActionResult GetAccountDetailsBloodBank()  
{  
  
    var accountDetails = _context.AccountUserDetailsAddress.Include(x => x.UserDetails).Include(a => a.Account).Include(s => s.Address).Where(x => x.UserDetails.rolestatus == 1).Where(x => x.Account.Status != 0).Select(l => new HospitalBloodBankDetails  
{  
  
        Id=l.Account.AccountId,  
        Name = l.Account.Name,  
        Email = l.Account.Email,  
        PhoneNumber = l.Account.PhoneNumber,  
        Location = l.UserDetails.Location,  
        GovernmentId = l.UserDetails.GovernmentId,  
  
        Document = Encoding.ASCII.GetString(l.UserDetails.Document),  
        DoorNo = l.Address.DoorNo,  
        Street = l.Address.Street,  
        Area = l.Address.Area,  
        City = l.Address.City,  
        State = l.Address.State,  
        PostalCode = l.Address.PostalCode,  
        Status = l.Account.Status  
});
```

```

        return Ok(accountDetails);
    }

    [HttpGet]
    public IActionResult GetAccountDetailsPendingBloodBank()
    {

        var accountDetails = _context.AccountUserDetailsAddress.Include(x => x.UserDetails).Include(a => a.Account).Include(s => s.Address).Where(x => x.UserDetails.rolestatus == 1).Where(x => x.Account.Status == 0).Select(l => new HospitalBloodBankDetails
        {
            Id = l.Account.AccountId,
            Name = l.Account.Name,
            Email = l.Account.Email,
            PhoneNumber = l.Account.PhoneNumber,
            Location = l.UserDetails.Location,
            GovernmentId = l.UserDetails.GovernmentId,
            Document = Encoding.ASCII.GetString(l.UserDetails.Document),
            DoorNo = l.Address.DoorNo,
            Street = l.Address.Street,
            Area = l.Address.Area,
            City = l.Address.City,
            State = l.Address.State,
            PostalCode = l.Address.PostalCode,
            Status = l.Account.Status
        });
        return Ok(accountDetails);
    }
}

```

```

[HttpGet]
public IActionResult GetAccountDetailsPendingDonor()
{
    var accountDetails = _context.AccountUserDetailsAddress.Include(x => x.UserDetails).Include(a => a.Account).Include(s => s.Address).ToList().Where(x => x.UserDetails.rolestatus == 3).Where(x => x.Account.Status == 0);

    return Ok(accountDetails);
}

[HttpGet]
public IActionResult GetAccountDetailsPendingHospital()
{
    var accountDetails = _context.AccountUserDetailsAddress.Include(x => x.UserDetails).Include(a => a.Account).Include(s => s.Address).Where(x => x.UserDetails.rolestatus == 2).Where(x => x.Account.Status == 0).Select(l => new HospitalBloodBankDetails
    {
        Id= l.Account.AccountId,
        Name = l.Account.Name,
        Email = l.Account.Email,
        PhoneNumber = l.Account.PhoneNumber,
        Location = l.UserDetails.Location,
        GovernmentId = l.UserDetails.GovernmentId,
        Document = Encoding.ASCII.GetString(l.UserDetails.Document),
        DoorNo = l.Address.DoorNo,
        Street = l.Address.Street,
        Area = l.Address.Area,
        City = l.Address.City,
    });
}

```

```
        State = l.Address.State,  
        PostalCode = l.Address.PostalCode,  
        Status = l.Account.Status  
    });  
  
    return Ok(accountDetails);  
}  
}  
  
}
```

ViewBloodCampController.cs

```
using BloodBankManagementWebapi.DataContext;  
using Microsoft.AspNetCore.Http;  
using Microsoft.AspNetCore.Mvc;  
using Microsoft.EntityFrameworkCore;  
  
namespace BloodBankManagementWebapi.Controllers  
{  
    [Route("api/[controller]/[action]")]  
    [ApiController]  
    public class ViewBloodCampController : ControllerBase  
    {  
        private readonly AppDbContext _context;  
        public ViewBloodCampController(AppDbContext context)  
        {  
            _context = context;
```

```

    }

    [HttpGet]
    public IActionResult Get()
    {
        var BloodCamp = _context.BloodCampBloodBank.Include(x =>
x.BloodCamp).Include(x =>x.Account).ToList();

        return Ok(BloodCamp);

    }

    [HttpGet]
    public IActionResult GetByIndividual(string id)
    {
        var account = _context.Account.Find(id);

        var BloodCamp = _context.BloodCampBloodBank.Include(x => x.BloodCamp).Include(x =>
x.Account).Where(x=>x.Account==account);

        return Ok(BloodCamp);

    }

}

```

ViewBloodRequestByBankController.cs

```

using BloodBankManagementWebapi.DataContext;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
namespace BloodBankManagementWebapi.Controllers
{

```

```

[Route("api/[controller]")]
[ApiController]
public class ViewBloodRequestByBankController : ControllerBase
{
    private readonly AppDbContext _context;
    public ViewBloodRequestByBankController(AppDbContext context)
    {
        _context = context;
    }
    [Route("/Get/{id}")]
    [HttpGet]
    public IActionResult Get(string id) {
        var account= _context.Account.Find(id);
        var bank = _context.AccountUserDetailsAddress.Include(x => x.UserDetails).Include(x => x.Account).Where(x => x.Account == account).FirstOrDefault();
        var bloodrequest = _context.BloodRequest.Where(x => x.Location == bank.UserDetails.Location).Where(x => x.Status == 1).ToList();
        return Ok(bloodrequest);
    }
}

```

ViewBloodRequestController.cs

```

using BloodBankManagementWebapi.DataContext;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
namespace BloodBankManagementWebapi.Controllers

```

```
{  
    [Route("api/[controller]")]  
    [ApiController]  
    public class ViewBloodRequestController : ControllerBase  
    {  
        private readonly AppDbContext _context;  
        public ViewBloodRequestController(AppDbContext context)  
        {  
            _context = context;  
        }  
        [HttpGet]  
        public IActionResult Get()  
        {  
            var BloodRequest = _context.BloodRequest.ToList();  
            return Ok(BloodRequest);  
        }  
    }  
}
```

ViewBloodStockasWholeController.cs

```
using BloodBankManagementWebapi.ApiModel;  
using BloodBankManagementWebapi.DataContext;  
using BloodBankManagementWebapi.Model;  
using Microsoft.AspNetCore.Http;  
using Microsoft.AspNetCore.Mvc;  
using Microsoft.EntityFrameworkCore;  
  
namespace BloodBankManagementWebapi.Controllers  
{
```

```

[Route("api/[controller]")]
[ApiController]
public class ViewBloodStockasWholeController : ControllerBase
{
    private readonly AppDbContext _context;
    public ViewBloodStockasWholeController(AppDbContext context)
    {
        _context = context;
    }

    [HttpGet]
    public IActionResult GetAllStock()
    {
        List<StockAllBank> bloodstock = new List<StockAllBank>();
        var BloodBank = _context.AccountUserDetailsAddress.Include(x => x.UserDetails).Include(a => a.Account).Include(s => s.Address).Where(x => x.UserDetails.rolestatus == 1).Where(x => x.Account.Status != 0).Select(l => new BloodRequestBloodBank
        {
            Id = l.Account.AccountId,
            Name = l.Account.Name,
            Location = l.UserDetails.Location
        }).ToList();
        foreach (var bank in BloodBank)
        {
            var o = _context.BloodStock.Include(c => c.Account).ToList().Select(p => new StockAllBank
            {

```

```

        BloodBankName=bank.Name,
        Location=bank.Location,
        BloodType=p.BloodType,
        Units=p.Units,
        Mobile=p.Account.PhoneNumber

    }).ToList();

    foreach(var i in o)
    {
        bloodstock.Add(i);
    }

};

return Ok(bloodstock);
}

}
}

```

ViewBloodStockController.cs

```

using BloodBankManagementWebapi.DataContext;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;

namespace BloodBankManagementWebapi.Controllers
{
    [Route("api/[controller]/[action]")]
    [ApiController]
    public class ViewBloodStockController : ControllerBase

```

```

{
    private readonly AppDbContext _context;

    public ViewBloodStockController(AppDbContext context)
    {
        _context = context;
    }

    [HttpGet]
    public ActionResult ViewBloodStock()
    {
        var BloodStock = _context.BloodStock.Include(y => y.Account).ToList();
        return Ok(BloodStock);
    }

    [HttpGet]
    public ActionResult ViewBloodStockByIndividual(string id)

    {
        var account = _context.Account.Find(id);

        var BloodStock = _context.BloodStock.Include(y =>
            y.Account).ToList().Where(x => x.Account == account);

        return Ok(BloodStock);
    }
}

```

ViewHospitalController.cs

```

using BloodBankManagementWebapi.DataContext;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;

```

```

namespace BloodBankManagementWebapi.Controllers

{
    [Route("api/[controller]")]
    [ApiController]
    public class ViewHospitalController : ControllerBase
    {
        private readonly AppDbContext _context;
        public ViewHospitalController(AppDbContext context)
        {
            _context = context;
        }

        [HttpGet]
        public IActionResult GetAccountDetails()
        {
            var role = _context.Role.Where(s => s.RoleName == "HOSPITAL").FirstOrDefault();
            var hospitalaccount =
                _context.AccountRole.Include(s=>s.Account).Include(a=>a.Role).Where(e=>e.Role==role).ToList();

            return Ok(hospitalaccount);
        }
    }
}

```

Output

SWAGGER OUTPUT

ADMIN DASHBOARD

The image displays three separate screenshots of the Swagger UI interface, each showing a different endpoint for the Admin Dashboard API. All three screenshots are identical in layout and content, showing the 'Get Basic Details' endpoint.

Endpoint: /api/AdminDashboard/GetBasicDetails

Method: GET

Parameters: No parameters

Responses:

- Call:** curl -X GET 'http://localhost:7089/api/AdminDashboard/GetBasicDetails'
- Request URL:** https://localhost:7089/api/AdminDashboard/GetBasicDetails
- Server response:**

Code: detail

200

Response body:

```
{ "basicDetails": { "id": 1, "name": "Bank A", "address": "123 Main Street", "email": "banka@example.com", "phone": "(555) 123-4567", "status": "Active", "lastUpdated": "2024-03-30T12:00:00Z" } }
```

Response headers:

```
access-control-allow-methods: POST,GET,HEAD,PUT,DELETE,OPTIONS  
access-control-allow-origin: *  
content-type: application/json; charset=UTF-8  
date: Mon, 30 Mar 2024 08:37:00 GMT  
server: Kestrel
```

Responses:

Code: detail

200

Response body:

```
{ "basicDetails": { "id": 1, "name": "Bank A", "address": "123 Main Street", "email": "banka@example.com", "phone": "(555) 123-4567", "status": "Active", "lastUpdated": "2024-03-30T12:00:00Z" } }
```

Response headers:

```
access-control-allow-methods: POST,GET,HEAD,PUT,DELETE,OPTIONS  
access-control-allow-origin: *  
content-type: application/json; charset=UTF-8  
date: Mon, 30 Mar 2024 08:37:00 GMT  
server: Kestrel
```

Responses:

Code: detail

200

Response body:

```
{ "basicDetails": { "id": 1, "name": "Bank A", "address": "123 Main Street", "email": "banka@example.com", "phone": "(555) 123-4567", "status": "Active", "lastUpdated": "2024-03-30T12:00:00Z" } }
```

Response headers:

```
access-control-allow-methods: POST,GET,HEAD,PUT,DELETE,OPTIONS  
access-control-allow-origin: *  
content-type: application/json; charset=UTF-8  
date: Mon, 30 Mar 2024 08:37:00 GMT  
server: Kestrel
```

Accept Requester By Bank

Backend Report

localhost:7089/swagger/index.html

POST /api/AcceptRequesterByBank

Parameters

No parameters

Request body

```
{"requesterId": "123456", "password": "password123$%^#QWEASDFGHJKL"}
```

application/json

Execute Clear

Responses

Code: 200

```
{  "id": 4,  "name": "BloodCamp",  "password": "password123$%^#QWEASDFGHJKL",  "status": "Active"}
```

Request URL: https://localhost:7089/api/AcceptRequesterByBank

Server response

Code: 200

Response headers

```
access-control-allow-methods: POST,GET,PUT,DELETE,OPTIONS
content-type: application/json
date: Mon, 30 Mar 2024 08:39:10 GMT
server: Apache
```

08:39 30-03-2024

Approve Blood Request By Admin

Backend Report

localhost:7089/swagger/index.html

POST /api/approveBloodRequestByAdmin

Parameters

No parameters

Request body

```
{"id": 1, "status": "Approved"}
```

application/json

Responses

Code: 200

```
{  "id": 1,  "name": "BloodCamp",  "password": "password123$%^#QWEASDFGHJKL",  "status": "Approved"}
```

Request URL: https://localhost:7089/api/approveBloodRequestByAdmin

Server response

Code: 200

Response body

```
{  "id": 1,  "name": "BloodCamp",  "password": "password123$%^#QWEASDFGHJKL",  "status": "Approved"}
```

Response headers

```
access-control-allow-methods: POST,GET,PUT,DELETE,OPTIONS
content-type: application/json
date: Mon, 30 Mar 2024 08:41:23 GMT
server: Apache/2.4.41 (Ubuntu)
```

Links

200 Success

ApproveOrRejectAccountByAdmin

POST /api/approveOrRejectAccountByAdmin

BloodCamp

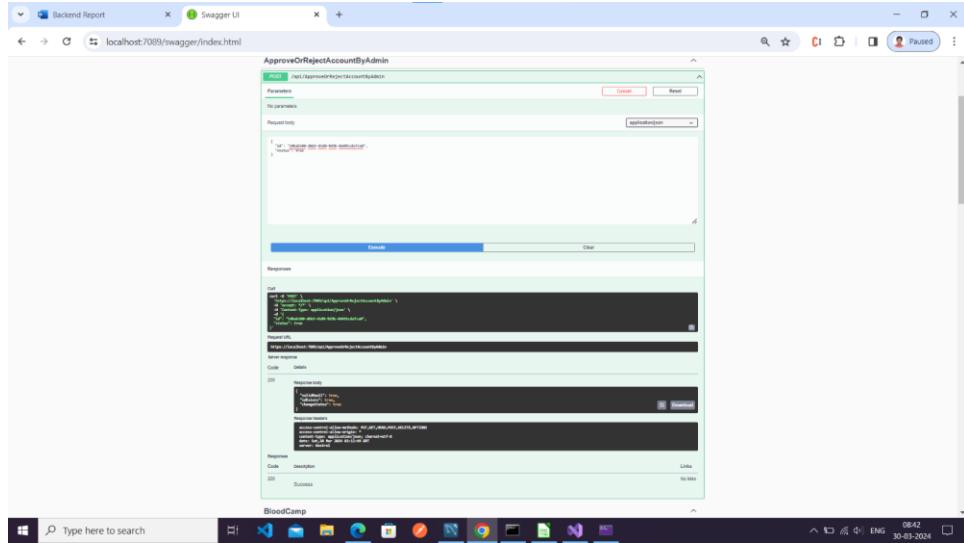
POST /api/BloodCamp

Approve Or

Type here to search

08:41 30-03-2024

Reject Account By Admin



Backend Report > Swagger UI > ApproveOrRejectAccountByAdmin

ApproveOrRejectAccountByAdmin

POST /api/approveorrejectaccount

Parameters:

No parameters

Request body:

```
{ "Action": "reject", "Reason": "User does not have permission" }
```

Responses:

200

```
{ "Status": "Rejected", "Reason": "User does not have permission" }
```

Request URL:

http://localhost:7089/api/approveorrejectaccount

Server response:

Code: 200

Response body:

```
{"Status": "Rejected", "Reason": "User does not have permission"}
```

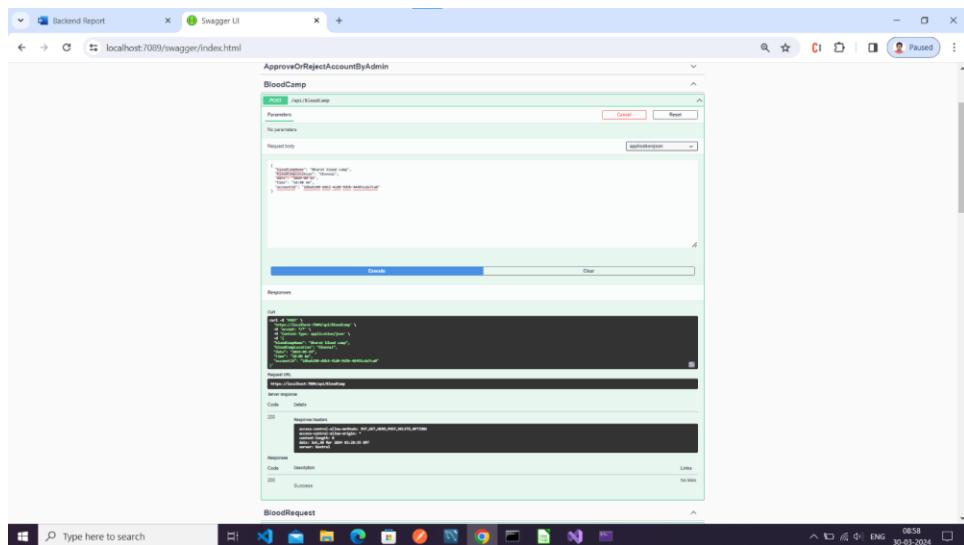
Responses:

Code: 200

Description: Success

BloodCamp

Blood camp



Backend Report > Swagger UI > ApproveOrRejectAccountByAdmin

BloodCamp

POST /api/bloodcamp

Parameters:

No parameters

Request body:

```
{ "Action": "reject", "Reason": "User does not have permission" }
```

Responses:

200

```
{ "Status": "Rejected", "Reason": "User does not have permission" }
```

Request URL:

http://localhost:7089/api/bloodcamp

Server response:

Code: 200

Response body:

```
{"Status": "Rejected", "Reason": "User does not have permission"}
```

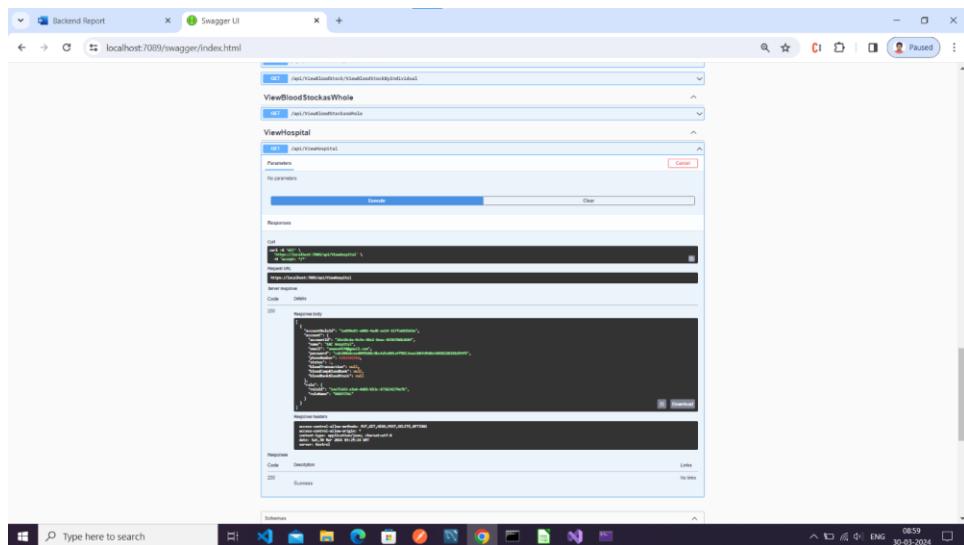
Responses:

Code: 200

Description: Success

BloodRequest

View Hospital



Backend Report > Swagger UI > ViewHospital

ViewHospital

POST /api/viewhospital

Parameters:

No parameters

Request body:

```
{ "Action": "view" }
```

Responses:

200

```
{ "Status": "Viewed", "Reason": "User does not have permission" }
```

Request URL:

http://localhost:7089/api/viewhospital

Server response:

Code: 200

Response body:

```
{"Status": "Viewed", "Reason": "User does not have permission"}
```

Responses:

Code: 200

Description: Success

View Blood Stock As Whole

The screenshot shows the Swagger UI interface for a RESTful API. The main panel displays the 'ViewBloodStocksWhole' endpoint, which is a GET request to '/api/BloodStocks/GetAll'. The 'Parameters' section indicates 'No parameters'. The 'Responses' section shows a successful response (200) with a JSON payload containing multiple blood stock items. Each item includes fields like 'BloodBankID', 'BloodGroup', 'BloodType', 'Count', and 'Status'. Below the main panel, there is another collapsed section labeled 'ViewHospital'.

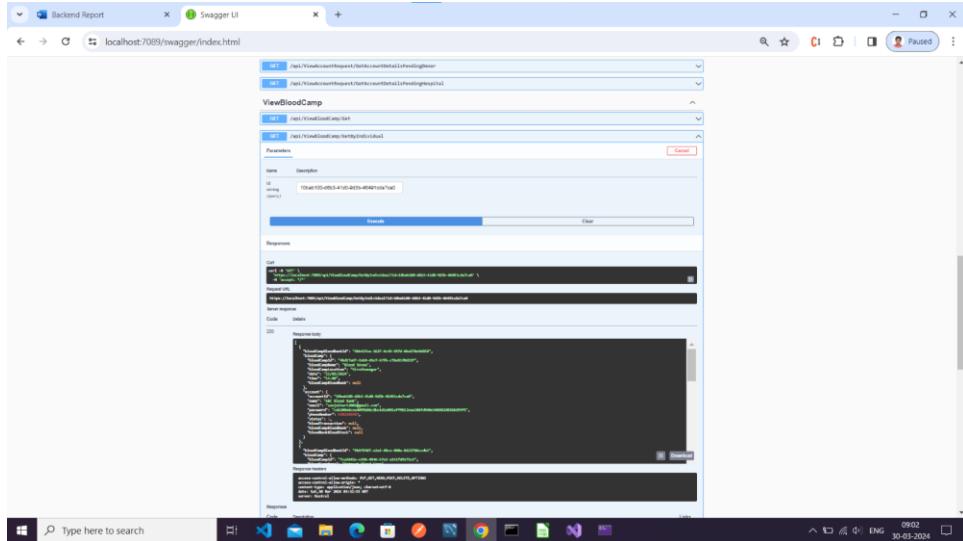
View Blood Request

The screenshot shows the Swagger UI interface for a RESTful API. The main panel displays the 'ViewBloodRequest' endpoint, which is a GET request to '/api/BloodRequest/GetByIndividual'. The 'Parameters' section indicates 'No parameters'. The 'Responses' section shows a successful response (200) with a JSON payload containing a single blood request item. The item includes fields like 'BloodBankID', 'BloodGroup', 'BloodType', 'Count', 'Status', and 'RequesterName'. Below the main panel, there is another collapsed section labeled 'ViewBloodRequestByBank'.

View Blood Camp

The screenshot shows the Swagger UI interface for a RESTful API. The main panel displays the 'ViewBloodCamp' endpoint, which is a GET request to '/api/BloodCamp/GetByIndividual'. The 'Parameters' section indicates 'No parameters'. The 'Responses' section shows a successful response (200) with a JSON payload containing a single blood camp item. The item includes fields like 'BloodBankID', 'BloodGroup', 'BloodType', 'Count', 'Status', and 'RequesterName'. Below the main panel, there is another collapsed section labeled 'ViewBloodCampByBank'.

By individual blood bank



Backend Report Swagger UI

localhost:7089/swagger/index.html

ViewBloodCamp

http://localhost:7089/swagger/index.html#/_/api/v1/bloodcamp/{id}/details

Parameters

Name Description

ID 10aef120-e653-410d-9e13-4f4d110a10a1

Responses

Code 200 Content-Type application/json

Server response

Code 200

Response body

```
{ "id": "10aef120-e653-410d-9e13-4f4d110a10a1", "name": "BloodCamp Name", "description": "BloodCamp Description", "date": "2024-05-30T10:00:00Z", "location": "BloodCamp Location", "status": "Active", "bloodBank": "BloodBank Name", "bloodType": "A+", "quantity": 100, "isAccepted": false } 
```

Responses

Code 200 Content-Type application/json

Server response

Code 200

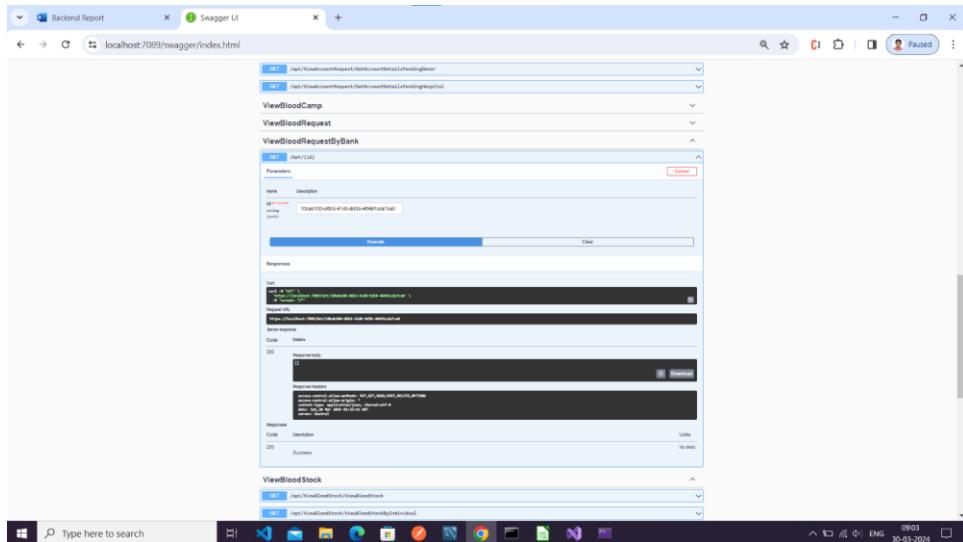
Response body

```
{ "id": "10aef120-e653-410d-9e13-4f4d110a10a1", "name": "BloodCamp Name", "description": "BloodCamp Description", "date": "2024-05-30T10:00:00Z", "location": "BloodCamp Location", "status": "Active", "bloodBank": "BloodBank Name", "bloodType": "A+", "quantity": 100, "isAccepted": false } 
```

Type here to search

0902 30-03-2024

View Blood Request by Bank



Backend Report Swagger UI

localhost:7089/swagger/index.html

ViewBloodCamp

ViewBloodRequest

ViewBloodRequestByBank

http://localhost:7089/swagger/index.html#/_/api/v1/bloodrequest/bybank/{id}

Parameters

Name Description

ID 10aef120-e653-410d-9e13-4f4d110a10a1

Responses

Code 200 Content-Type application/json

Server response

Code 200

Response body

```
{ "id": "10aef120-e653-410d-9e13-4f4d110a10a1", "name": "BloodRequest Name", "description": "BloodRequest Description", "date": "2024-05-30T10:00:00Z", "location": "BloodRequest Location", "status": "Active", "bloodBank": "BloodBank Name", "bloodType": "A+", "quantity": 100, "isAccepted": false } 
```

Responses

Code 200 Content-Type application/json

Server response

Code 200

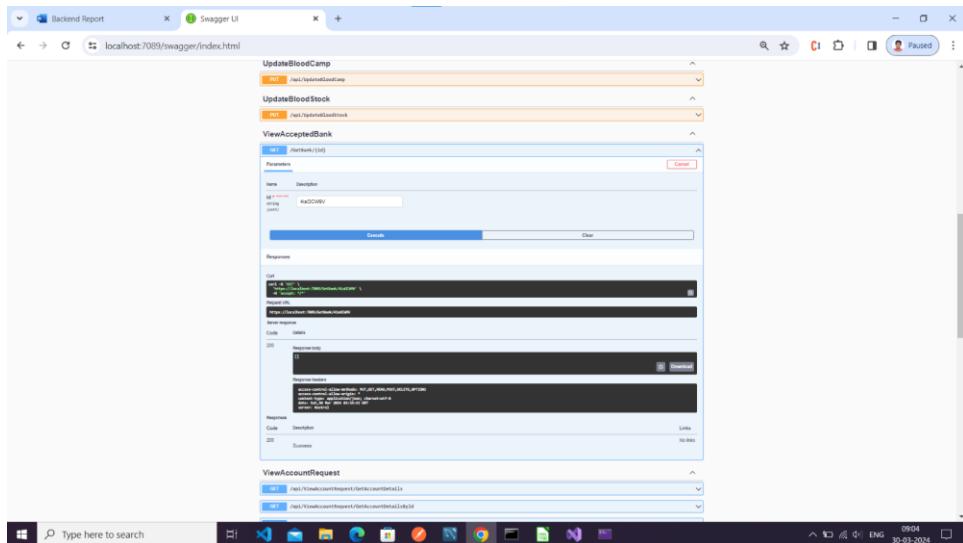
Response body

```
{ "id": "10aef120-e653-410d-9e13-4f4d110a10a1", "name": "BloodRequest Name", "description": "BloodRequest Description", "date": "2024-05-30T10:00:00Z", "location": "BloodRequest Location", "status": "Active", "bloodBank": "BloodBank Name", "bloodType": "A+", "quantity": 100, "isAccepted": false } 
```

Type here to search

0903 30-03-2024

View Accepted bank



Backend Report Swagger UI

localhost:7089/swagger/index.html

UpdateBloodCamp

UpdateBloodStock

ViewAcceptedBank

http://localhost:7089/swagger/index.html#/_/api/v1/bloodcamp/{id}/details

Parameters

Name Description

ID AcDQUBV

Responses

Code 200 Content-Type application/json

Server response

Code 200

Response body

```
{ "id": "AcDQUBV", "name": "Accepted Bank Name", "description": "Accepted Bank Description", "date": "2024-05-30T10:00:00Z", "location": "Accepted Bank Location", "status": "Active", "bloodBank": "Accepted Bank Name", "bloodType": "A+", "quantity": 100, "isAccepted": true } 
```

Responses

Code 200 Content-Type application/json

Server response

Code 200

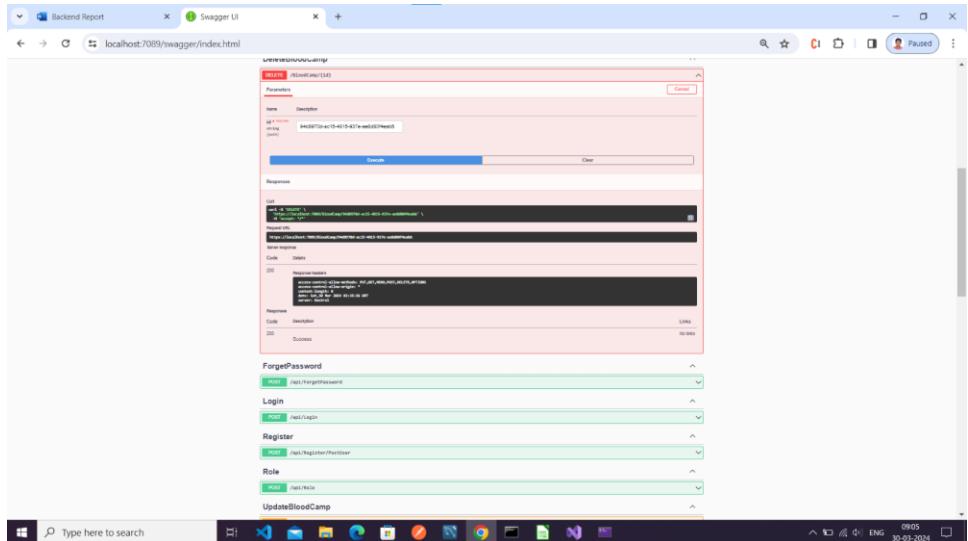
Response body

```
{ "id": "AcDQUBV", "name": "Accepted Bank Name", "description": "Accepted Bank Description", "date": "2024-05-30T10:00:00Z", "location": "Accepted Bank Location", "status": "Active", "bloodBank": "Accepted Bank Name", "bloodType": "A+", "quantity": 100, "isAccepted": true } 
```

Type here to search

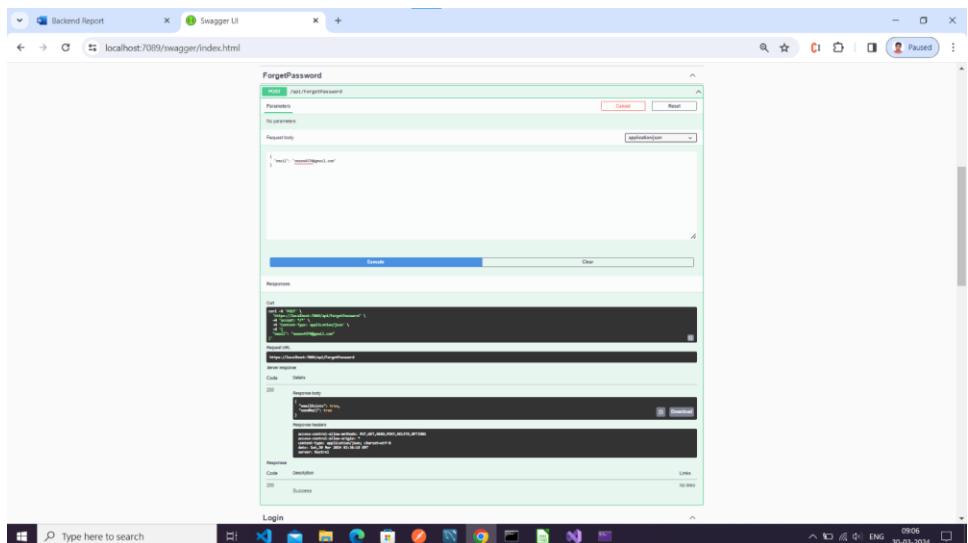
0904 30-03-2024

Delete blood camp



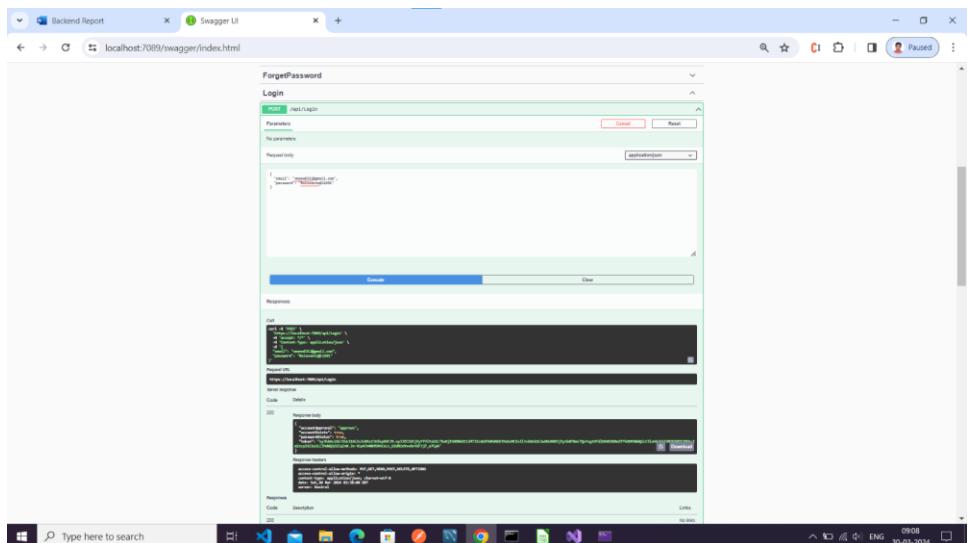
The screenshot shows the Swagger UI interface for a REST API. The main panel displays the `DELETE /bloodCamp/{id}` endpoint. The request body is set to `{id: '6408f09a15401a87eadd7eac'}`. The response section shows a successful `200 OK` status with the message "Success". Below this, a detailed view of the response body is shown, containing JSON data related to a blood camp deletion.

Forgot Password



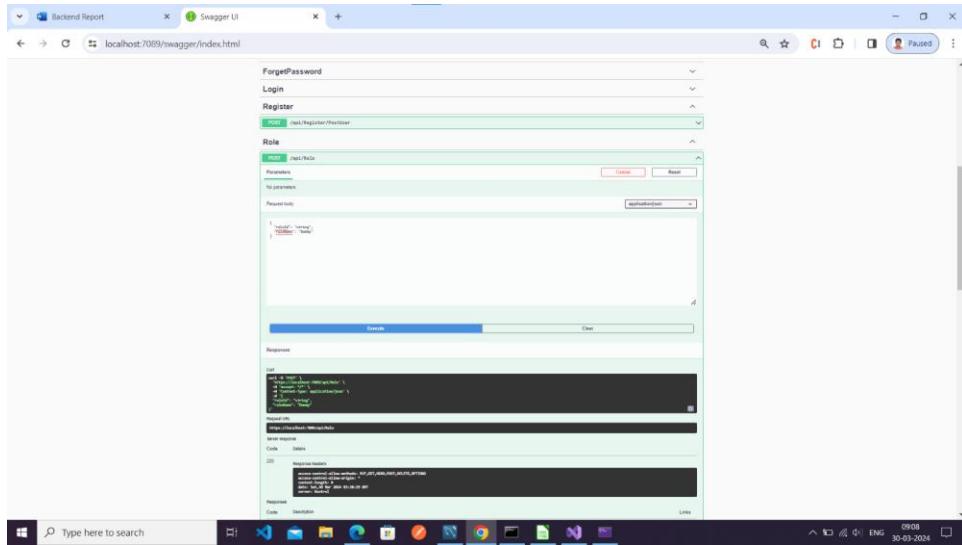
The screenshot shows the Swagger UI interface for a REST API. The main panel displays the `POST /api/forgotPassword` endpoint. The request body is set to `{email: 'test@gmail.com'}`. The response section shows a successful `200 OK` status with the message "Success". Below this, a detailed view of the response body is shown, containing JSON data related to a password reset link generation.

Login



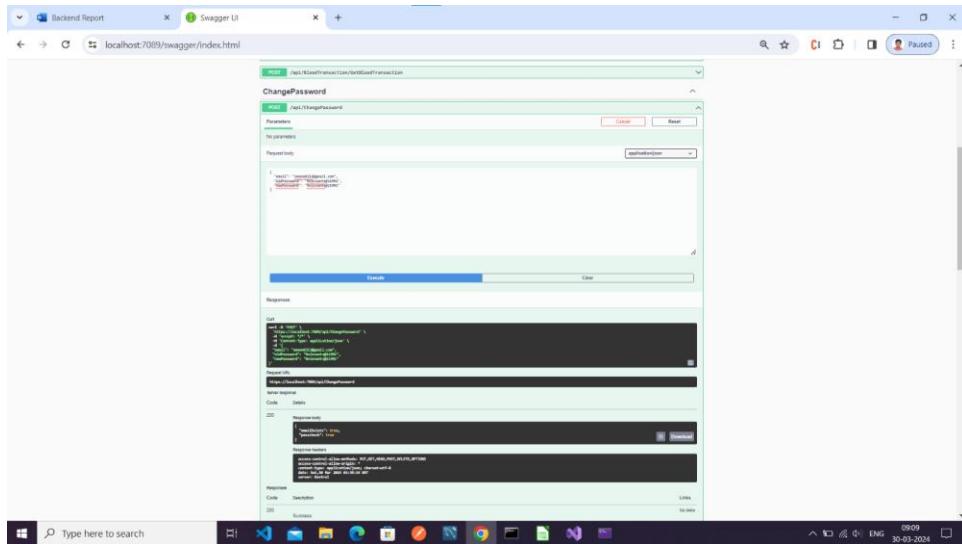
The screenshot shows the Swagger UI interface for a REST API. The main panel displays the `POST /api/login` endpoint. The request body is set to `{email: 'test@gmail.com', password: '1234567890'}`. The response section shows a successful `200 OK` status with the message "Success". Below this, a detailed view of the response body is shown, containing JSON data related to a user login.

Role



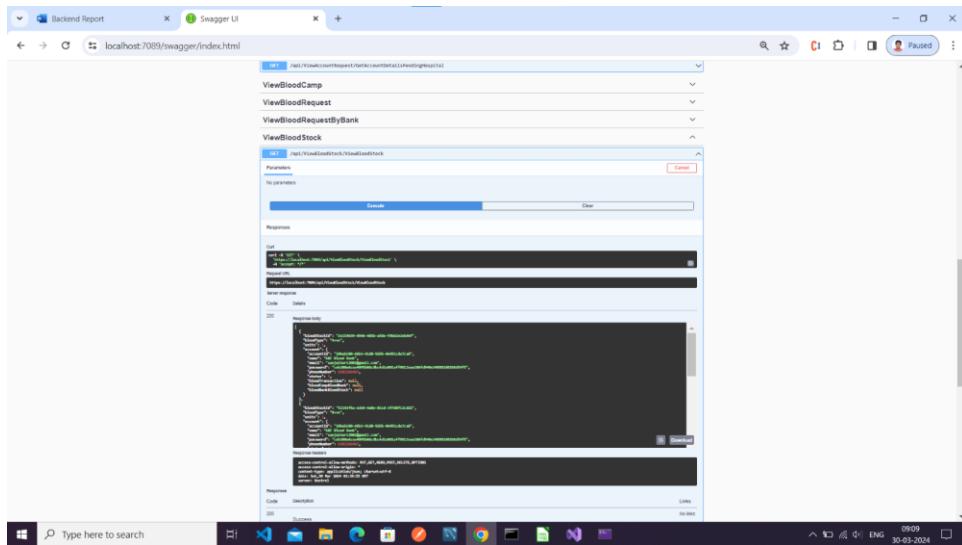
The screenshot shows the Swagger UI interface for a RESTful API. The URL is `localhost:7089/swagger/index.html`. The left sidebar lists several endpoints under the `Role` category, with `POST /api/Role` being the active one. The request body is set to `{"Role": "Admin", "Status": "Active"}`. The response section shows a JSON object with fields like `Role`, `Status`, and `CreatedDate`.

Change Password



The screenshot shows the Swagger UI interface for a RESTful API. The URL is `localhost:7089/swagger/index.html`. The left sidebar lists several endpoints under the `ChangePassword` category, with `POST /api/ChangePassword` being the active one. The request body is set to `{"Oldpassword": "*****", "Newpassword": "*****", "ConfirmNewpassword": "*****"}`. The response section shows a JSON object with fields like `Message`, `Code`, and `Description`.

View Blood Stock



The screenshot shows the Swagger UI interface for a RESTful API. The URL is `localhost:7089/swagger/index.html`. The left sidebar lists several endpoints under the `ViewBloodStock` category, with `POST /api/ViewBloodStock` being the active one. The request body is set to `{"BloodGroup": "AB+", "BloodBank": "ABC Bank", "BloodCamp": "Camp ABC", "BloodRequest": "Request ABC", "BloodStock": "Stock ABC"}`. The response section shows a JSON object with fields like `BloodGroup`, `BloodBank`, `BloodCamp`, `BloodRequest`, and `BloodStock`.

View Blood Stock By Bank

The screenshot shows the Swagger UI interface for a RESTful API. The URL is `localhost:7089/swagger/index.html`. The left sidebar lists several endpoints under the `ViewBloodStock` category. One endpoint, `http://localhost:7089/api/viewbloodstock/bloodstockbyid/{id}`, is selected. A modal dialog is open for this endpoint, showing a parameter named `id` with a value of `10bank123-abcd-1234-5678-901234567890`. Below the modal, the response section shows a JSON object with various fields like `id`, `name`, `type`, `quantity`, and `status`. The status code is 200, and the response body contains a detailed JSON structure of the blood stock record.

View Account Details

The screenshot shows the Swagger UI interface for a RESTful API. The URL is `localhost:7089/swagger/index.html`. The left sidebar lists several endpoints under the `ViewAcceptedBank` category. One endpoint, `http://localhost:7089/api/viewaccountrequest/bankaccountdetails`, is selected. A modal dialog is open for this endpoint, showing no parameters. Below the modal, the response section shows a JSON object with fields like `id`, `name`, `type`, `quantity`, and `status`. The status code is 200, and the response body contains a detailed JSON structure of the account details.

View Donor By Id

The screenshot shows the Swagger UI interface for a RESTful API. The URL is `localhost:7089/swagger/index.html`. The left sidebar lists several endpoints under the `ViewDonorById` category. One endpoint, `http://localhost:7089/api/viewaccountrequest/bankaccountdetails/{id}`, is selected. A modal dialog is open for this endpoint, showing a parameter named `id` with a value of `47761772-2800-4423-a150-7d5b4f110508`. Below the modal, the response section shows a JSON object with fields like `id`, `name`, `type`, `quantity`, and `status`. The status code is 200, and the response body contains a detailed JSON structure of the donor record.

Get Donor Details

The screenshot shows the Swagger UI interface for a REST API. The left sidebar lists several endpoints under the 'ViewAcceptedBank' category. The 'ViewAcceptedBank' endpoint is selected, showing its details. The 'Responses' section displays a successful response (HTTP 200) with a JSON schema. The JSON schema defines a 'Donor' object with properties: 'id', 'name', 'age', 'gender', 'blood_group', 'city', 'state', 'pin_code', 'phone_number', 'email', 'status', and 'last_donation_date'. Below the schema is a large preview window showing a sample JSON response. The response body contains a single 'Donor' object with values like 'id': 1, 'name': 'John Doe', etc.

View Hospital Details

This screenshot is identical to the previous one, showing the 'ViewAcceptedBank' endpoint in the Swagger UI. The JSON schema for the 'Donor' object remains the same, and the sample response body also shows a single 'Donor' object with the same fields and values as before.

View Blood Bank Details

This screenshot is identical to the previous ones, showing the 'ViewAcceptedBank' endpoint in the Swagger UI. The JSON schema for the 'Donor' object and the sample response body remain consistent with the previous screenshots.

View Pending Bank Request

The screenshot shows the Swagger UI interface for a REST API. The left sidebar lists several endpoints under the 'ViewAcceptedBank' category. The 'ViewAcceptedBank' endpoint is expanded, showing its detailed configuration. The 'Responses' section for this endpoint displays a successful response (HTTP 200) with a JSON payload. The JSON response includes fields such as 'id', 'name', 'status', 'requester', 'date', and 'details'. Below the responses, there is a 'Responses' section for the 'ViewPendingBank' endpoint, which also shows a successful response (HTTP 200) with a similar JSON structure.

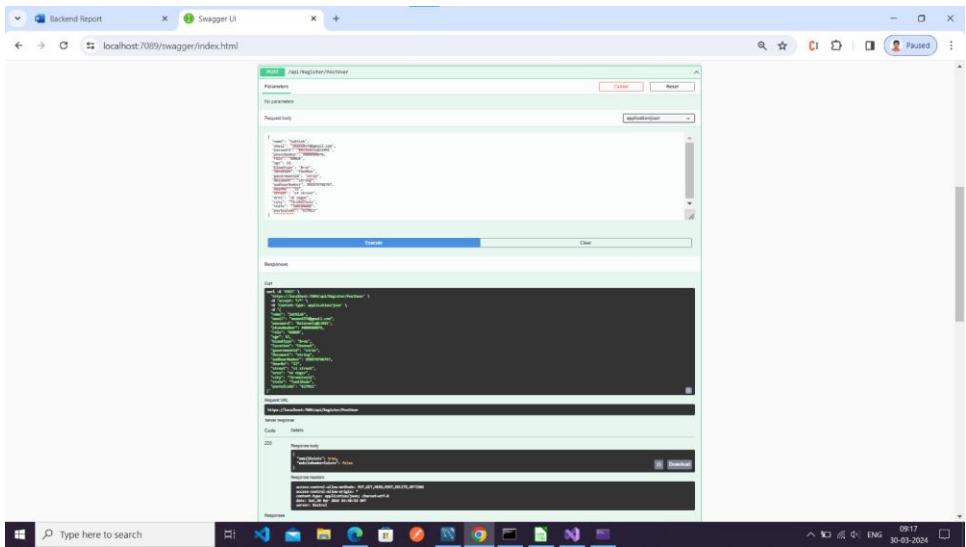
View Pending Donor Request

This screenshot is identical to the previous one, showing the Swagger UI interface for the 'ViewAcceptedBank' endpoint. It highlights the 'ViewAcceptedBank' endpoint and its detailed configuration, including the 'Responses' section for both 'ViewAcceptedBank' and 'ViewPendingBank'.

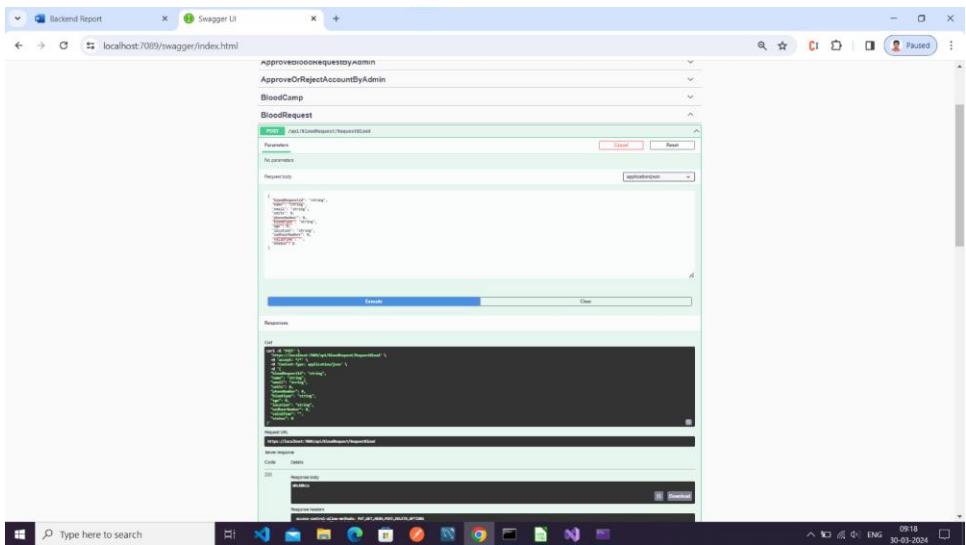
View Pending Hospital Request

This screenshot is identical to the previous ones, showing the Swagger UI interface for the 'ViewAcceptedBank' endpoint. It highlights the 'ViewAcceptedBank' endpoint and its detailed configuration, including the 'Responses' section for both 'ViewAcceptedBank' and 'ViewPendingBank'.

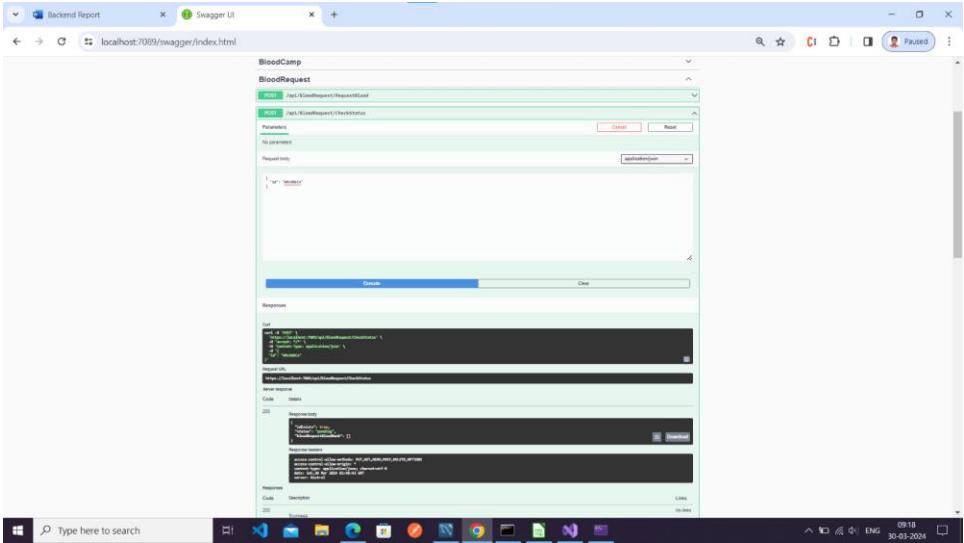
Register



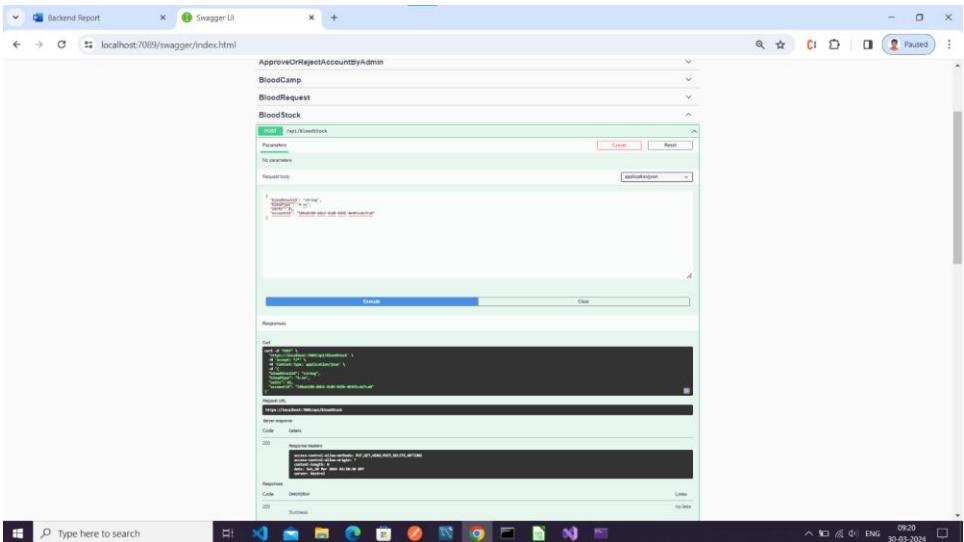
Add Blood Request



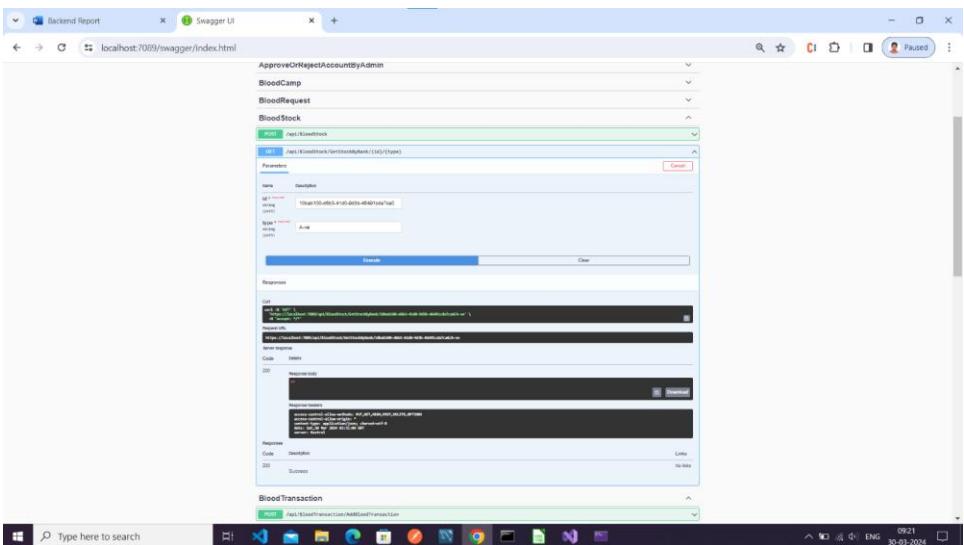
Check Request Status



Add Blood Stock



View Stock by bank and type



Add blood transaction

The screenshot shows a browser window with the title "Backend Report" and the address bar "localhost:7089/swagger/index.html". The main content is a Swagger UI interface for a REST API. On the left, there's a tree view with nodes: BloodRequest, BloodStock, and BloodTransaction. Under BloodTransaction, there's a "POST /api/Bloodtransaction/AddBloodTransaction" endpoint. The "Parameters" section is empty. The "Request body" section contains a JSON object:

```
{ "BloodRequest": "101", "BloodStock": "101", "BloodType": "A+", "Quantity": "100", "Status": "In Stock" }
```

The "Responses" section shows a successful response (200 OK) with the following JSON content:

```
{ "Message": "Blood Transaction Added successfully", "Status": "Success", "StatusText": "Success", "StatusColor": "#008000" }
```

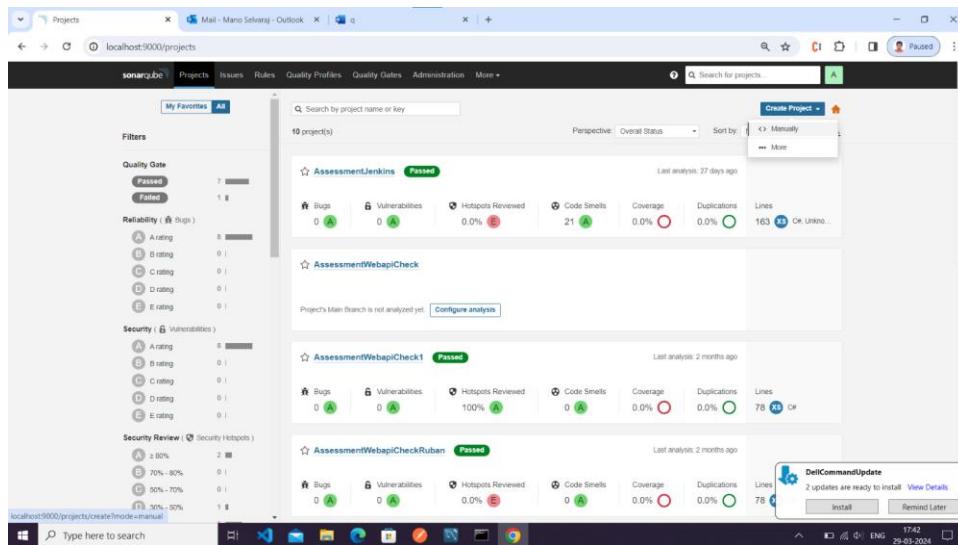
At the bottom of the screen, the taskbar shows various icons for system tools like Task Manager, File Explorer, and a terminal window.

SONARQUBE REPORT

SonarQube – Checking quality of code

Step –1 :Start the sonarqube. Open the sonarqube.

Step – 2 :Create the project manually

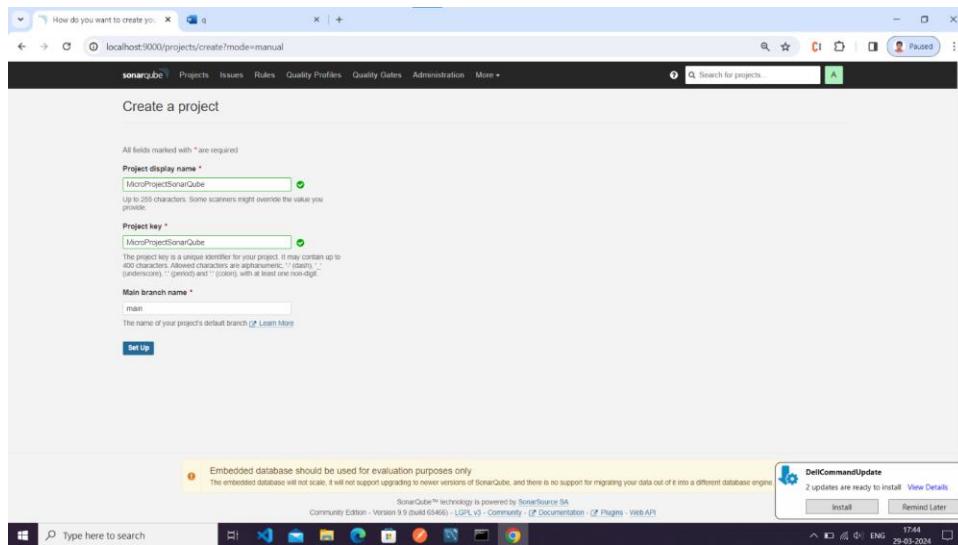


The screenshot shows the SonarQube web interface with the following details:

- Projects:** 10 project(s) listed.
- AssessmentJenkins:** Passed, Last analysis: 27 days ago. Metrics: Bugs 0, Vulnerabilities 0, Hotspots Reviewed 0.0%, Code Smells 21, Coverage 0.0%, Duplications 0.0%, Lines 163.
- AssessmentWebapiCheck:** Project's Main Branch is not analyzed yet. Configure analysis.
- AssessmentWebapiCheck1:** Pending, Last analysis: 2 months ago. Metrics: Bugs 0, Vulnerabilities 0, Hotspots Reviewed 100%, Code Smells 0, Coverage 0.0%, Duplications 0.0%, Lines 78.
- AssessmentWebapiCheckRuban:** Passed, Last analysis: 2 months ago. Metrics: Bugs 0, Vulnerabilities 0, Hotspots Reviewed 0.0%, Code Smells 0, Coverage 0.0%, Duplications 0.0%, Lines 78.

A sidebar on the left shows filters for Quality Gate (Passed, Failed), Reliability (A rating, B rating, C rating, D rating, E rating), and Security (Vulnerabilities). A message at the bottom right says "DellCommandUpdate 2 updates ready to install View Details".

Step – 3 :Name the project and key and click the setup button.

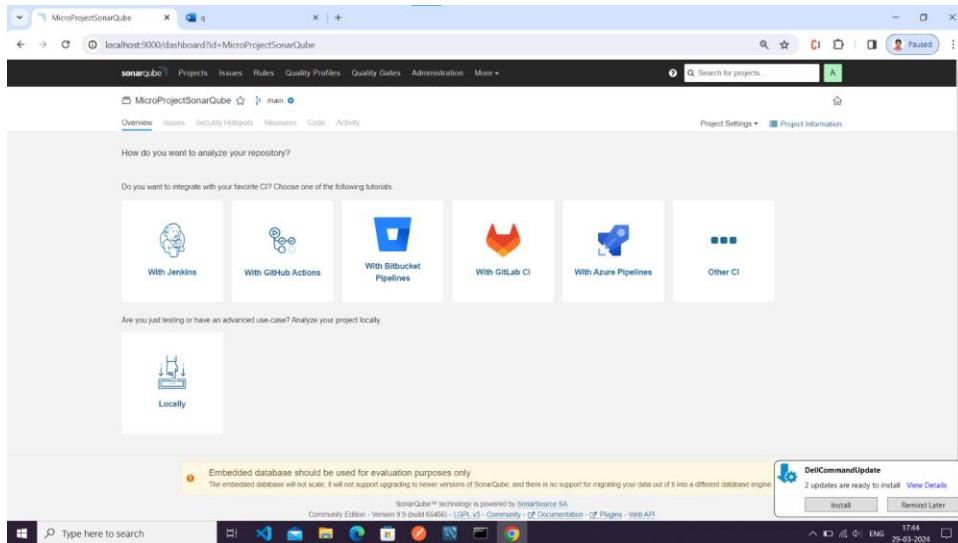


The screenshot shows the "Create a project" form with the following fields filled:

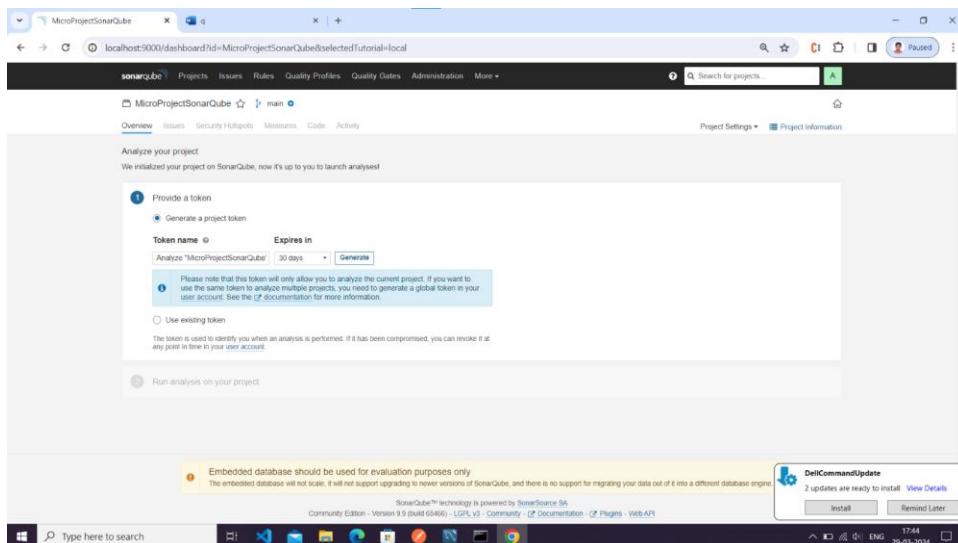
- Project display name ***: MicroProjectSonarQube
- Project key ***: MicroProjectSonarQube
- Main branch name ***: main

Below the form, a note states: "Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine." A "Setup" button is visible at the bottom left. A message at the bottom right says "DellCommandUpdate 2 updates ready to install View Details".

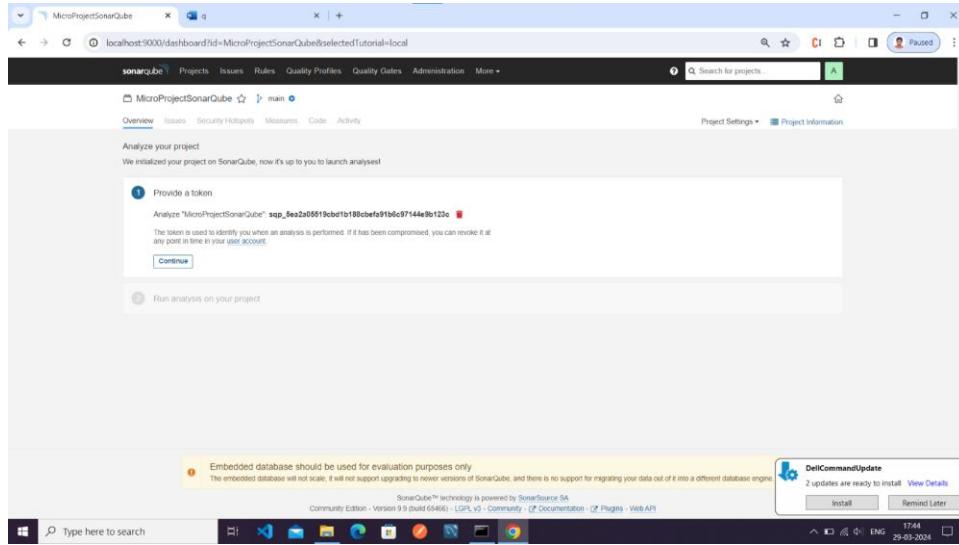
Step – 4: Now go with locally to analysis the project locally



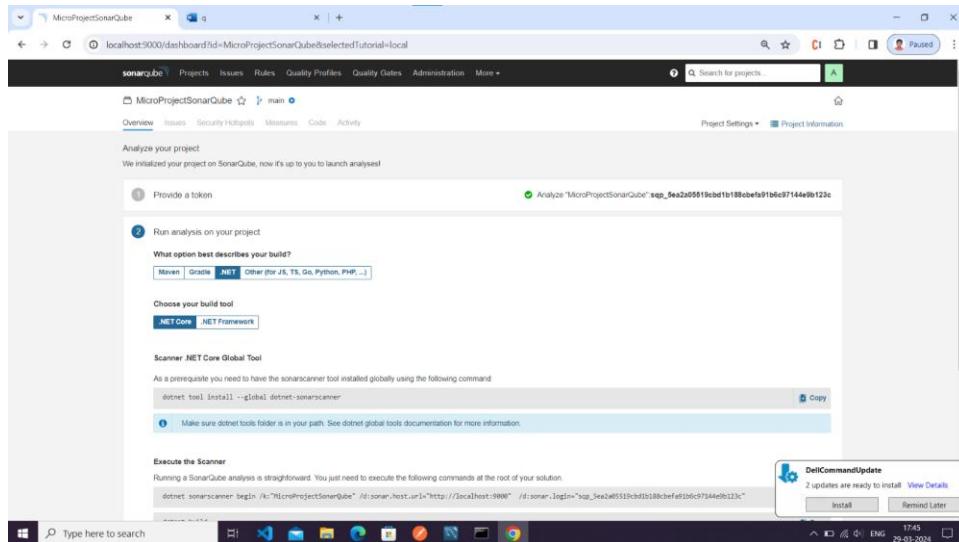
Step – 5 : Generate the token for 30 days.



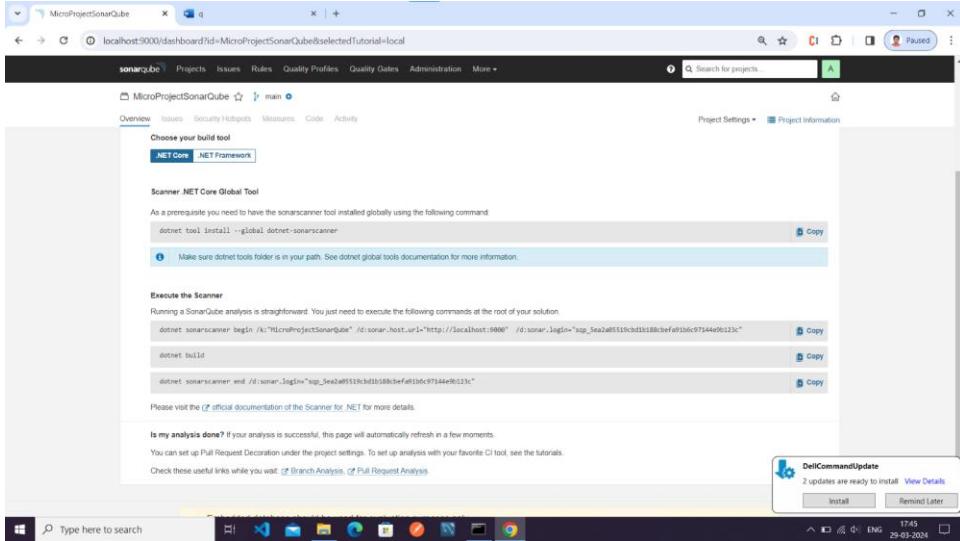
Step – 6 : Token is provided , then click on continue to continue the project.



Step – 7 : Then Choose the project .net => ,net core to analysis the dotnet webapi project.



Step – 8 : Copy the command .



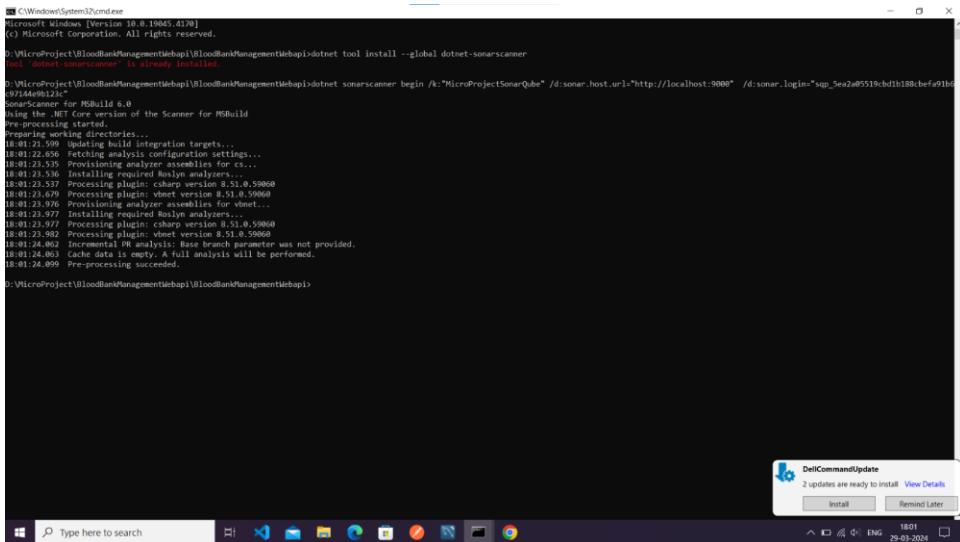
Step – 9 : Copy the command:

Dotnet tool install –global dotnet-sonarscanner.

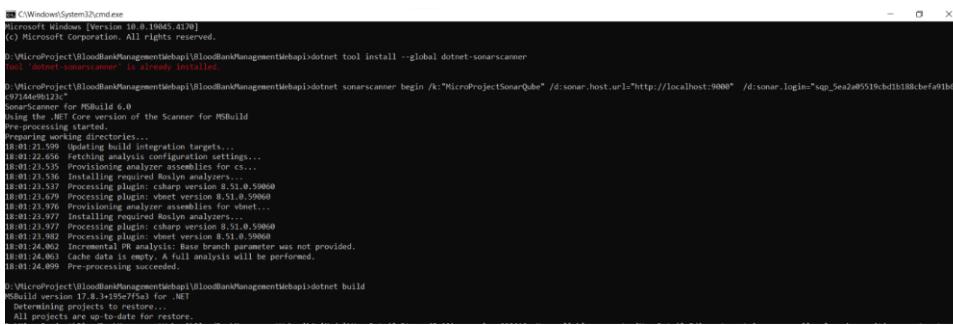
Ignore this step , if it is already installed.

Step – 10 : dotnet sonarscanner begin /k:"MicroProjectSonarQube"
/d:sonar.host.url="http://localhost:9000"
/d:sonar.login="sqp_5ea205519cbd1b188cbefa91b6c97144e9b123c"

This begin the scanner in your project by creating the link between scanner and project.



Step – 11 : Then run dotnet build command .



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.4170]
(c) Microsoft Corporation. All rights reserved.

D:\MicroProject\BloodBankManagementWebapi\BloodBankManagementWebapi>dotnet tool install --global dotnet-sanscanner
Tool 'dotnet-sanscanner' is already installed.

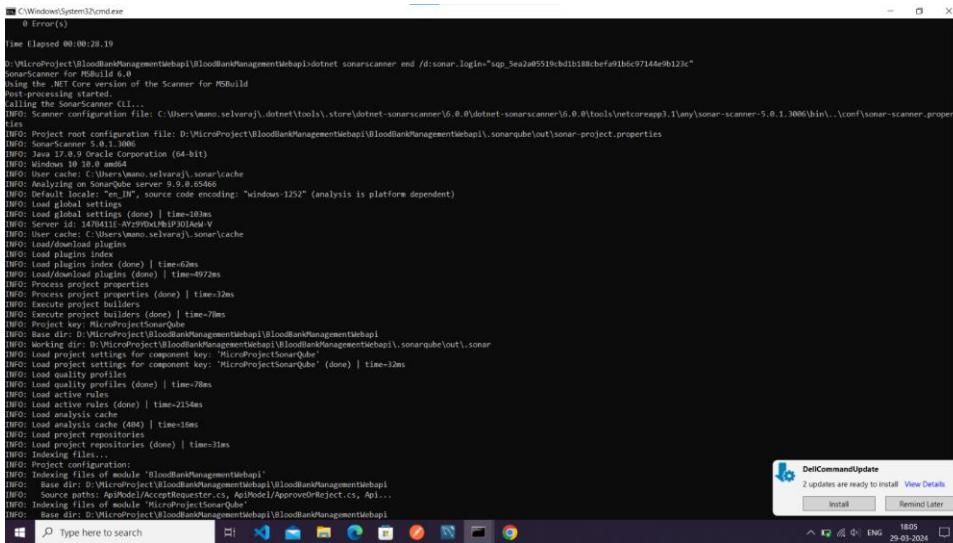
D:\MicroProject\BloodBankManagementWebapi\BloodBankManagementWebapi>dotnet sonarscanner begin /k:"MicroProjectSonarQube" /d:sonar.host.url="http://localhost:9000" /d:sonar.login="sqp_5ea2a05519cbd1b188cbefa91b6c97144e9b123c"
SonarScanner for MSBuild 4.0
using the .NET Core version of the Scanner for MSBuild
Pre-processing started.
Preparing working directories...
18:01:21.999 Updating build integration targets...
18:01:21.999 Processing analyzers assemblies for cs...
18:01:23.535 Provisioning analyzer assemblies for cs...
18:01:23.536 Installing required Roslyn analyzers...
18:01:23.537 Processing plugin: charr version 8.51.0.59860
18:01:23.539 Processing plugin: charr version 8.51.0.59860
18:01:23.976 Provisioning analyzer assemblies for vmt...
18:01:23.977 Installing required Roslyn analyzers...
18:01:23.978 Processing plugin: charr version 8.51.0.59860
18:01:23.982 Processing plugin: vmt version 8.51.0.59860
18:01:24.062 Incremental PR analysis: Basic branch parameter was not provided.
18:01:24.063 Cache data is empty. A full analysis will be performed.
18:01:24.069 Pre-processing succeeded.

D:\MicroProject\BloodBankManagementWebapi\BloodBankManagementWebapi>dotnet build
Restoring packages for D:\MicroProject\BloodBankManagementWebapi\BloodBankManagementWebapi\BloodBankManagementWebapi.csproj...
Determining projects to restore...
All projects are up-to-date for restore.
```

Step –12 : Finally run the command

Dotnet sonarscanner end /d:sonar.login="sqp_5ea2a05519cbd1b188cbefa91b6c97144e9b123c"

This help to end the analysis and send report to sonarqube



```
C:\Windows\System32\cmd.exe
0 Error(s)
Time Elapsed: 00:00:38.19

D:\MicroProject\BloodBankManagementWebapi\BloodBankManagementWebapi>dotnet sonarscanner end /d:sonar.login="sqp_5ea2a05519cbd1b188cbefa91b6c97144e9b123c"
SonarScanner for MSBuild 4.0
using the .NET Core version of the Scanner for MSBuild
Post-process started.
Starting the SonarScanner CLI...
INFO: Scanner configuration file: C:\Users\mamo.selvaraj\dotnet\store\dotnet-sonarscanner\6.0.0\tools\netcoreapp3.1\any\sonar-scanner-5.0.1.3006\bin..\conf\sonar-scanner.properties
INFO: Project root configuration file: D:\MicroProject\BloodBankManagementWebapi\BloodBankManagementWebapi.sonarqube\out\sonar-project.properties
INFO: Java 17.0.9 Oracle Corporation (64-bit)
INFO: Windows 10.0.0 amd64
INFO: User cache: C:\Users\mamo.selvaraj\.sonar\cache
INFO: SonarQube server: http://sonarqube:9000
INFO: Default locale: "en_IN", server code encoding: "windows-1252" (analysis is platform dependent)
INFO: Load global settings
INFO: Load global properties (done) | time=10ms
INFO: Server id: 1470411:AV9D0XHbIP30IAwV
INFO: User cache: C:\Users\mamo.selvaraj\.sonar\cache
INFO: Load/download plugins
INFO: Load active rules
INFO: Load plugin index (done) | time=6ms
INFO: Load/download plugins (done) | time=497ms
INFO: Load global properties (done) | time=2ms
INFO: Process project properties (done) | time=32ms
INFO: Execute project builders
INFO: Execute project builders (done) | time=70ms
INFO: Load quality profile
INFO: Load quality profile (done) | time=7ms
INFO: Load quality profiles (done) | time=78ms
INFO: Load active rules
INFO: Load active rules (done) | time=215ms
INFO: Load analysis cache
INFO: Load analysis cache (done) | time=10ms
INFO: Load project repositories
INFO: Load project repositories (done) | time=31ms
INFO: Load project filters
INFO: Load project filters (done) | time=1ms
INFO: Project key generation:
INFO: Indexing files of module 'BloodBankManagementWebapi'
INFO: Base dir: D:\MicroProject\BloodBankManagementWebapi\BloodBankManagementWebapi\BloodBankManagementWebapi
INFO: File 'D:\MicroProject\BloodBankManagementWebapi\BloodBankManagementWebapi\BloodBankManagementWebapi\ApproveOrReject.cs' skipped: file is newer than its parent.
INFO: Indexing files of module 'MicroProjectSonarQube'
INFO: Base dir: D:\MicroProject\BloodBankManagementWebapi\BloodBankManagementWebapi
```



```

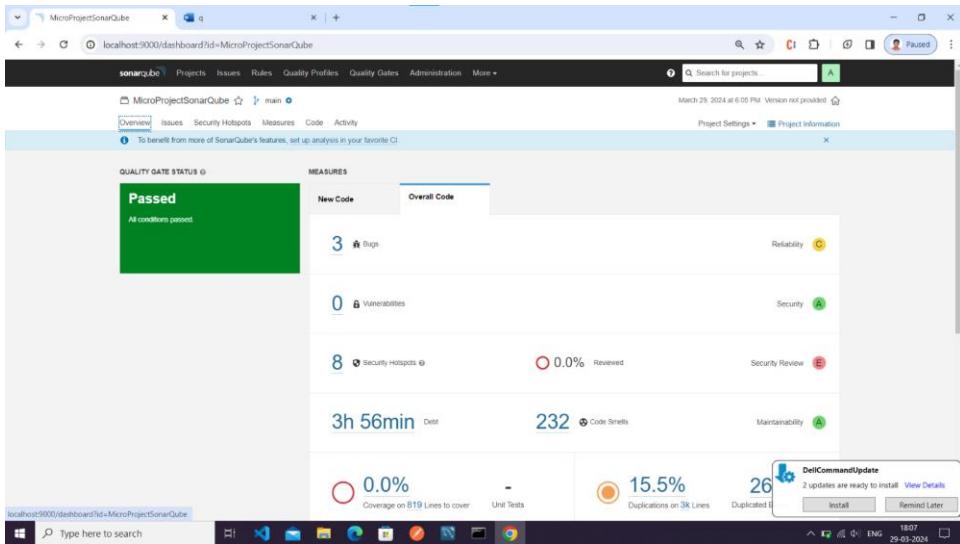
C:\Windows\System32\cmd.exe
INFO: Sensor SCM [csharp] (done) | time=81ms
INFO: Sensor SCM [csharp] (done) | time=1ms
INFO: Sensors Analysis Warnings Import [csharp] (done) | time=3ms
INFO: Sensor SCM File Caching Sensor [csharp] (done) | time=107ms
INFO: Sensor SCM File Caching Sensor [csharp] (done) | time=107ms
INFO: Sensor Zero Coverage Sensor
INFO: Sensor Zero Coverage Sensor [done] | time=142ms
INFO: SCM Publisher SCM provider for this project is: git
INFO: SCM Publisher 77/77 source files have been analyzed (done) | time=751ms
INFO: CPD Executor 18 files had no CPD blocks
INFO: CPD Executor 18 files had no CPD blocks
INFO: CPD Executor CPD calculation finished (done) | time=219ms
INFO: Analysis report generated in 97ms, dir size=411.5 kB
INFO: Analysis report compressed in 659ms, zip size=204.0 kB
INFO: ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=MicroProjectSonarQube
INFO: Note that you will not be able to access the updated dashboard once the server has processed the submitted analysis report
INFO: Post-processing completed at http://localhost:9000/api/ci/taskId=Aj0KXThnx4tXVF60IEY
INFO: Analysis total time: 26.365 s
INFO: EXECUTION SUCCESS
INFO: Total time: 33.880s
INFO: Final Memory: 16M/68M
INFO: The SonarScanner CLI has finished
18:05:27,949 Post-processing succeeded.

D:\MicroProject\BloodBankManagementWebapi\BloodBankManagementWebapi>

```

Step – 13:Check the report in sonarqube

Initially it contain bug, vulnerability and code smell.



Step – 14: After clearing all issues, rerun the command to send the report to sonarqube

Step –15 : Check the sonarqube final report with 0 bug,0 vulnerability and 0 codesmells.

The screenshot shows the SonarQube dashboard for the project 'MicroProjectSonarQube'. The top navigation bar includes links for sonarqube, Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, and More. A search bar is at the top right. The main content area displays the 'MEASURES' section under 'Overall Code'. It shows the following data:

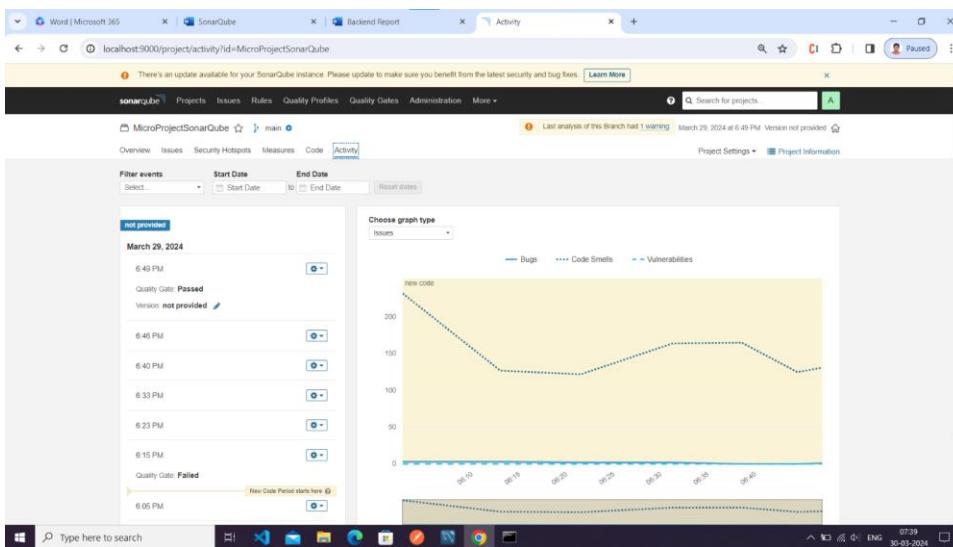
Measure	Value	Status
New Code	Since March 29, 2024 Started 49 minutes ago	
New Bugs	0	Reliability (A)
New Vulnerabilities	0	Security (A)
New Security Hotspots	0	Security Review (A)
Added Debt	0	Maintainability (A)
New Code Smells	0	

Coverage information: Coverage on 0 New Lines to cover, 0.0% completion.

A notification at the bottom right says 'DellCommandUpdate' with '2 updates are ready to install' and buttons for 'Install' and 'Remind Later'.

Step – 16: Finally end the analysis.

Graph:



Security hotspot

The screenshot shows the SonarQube interface for the project "MicroProjectSonarQube". The "Security Hotspots" tab is selected. A prominent message at the top states: "There are no Security Hotspots to review." Below this, a note says: "Next time you analyze a piece of code that contains a potential security risk, it will show up here." A link "Learn more about Security Hotspots" is provided. At the bottom of the page, a yellow box contains a warning: "Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine." The SonarQube footer includes links for "Community Edition", "Version 9.9 (build 6546)", "LGPL v3", "Community", "Documentation", "Plugins", and "Web API".

Measure

The screenshot shows the SonarQube "Project Overview" page for the project "MicroProjectSonarQube". The "Measures" tab is selected. On the left, a sidebar lists metrics: Reliability, Security, Security Review, Maintainability, Coverage, Duplications, Size, Complexity, and Issues. The main area displays a chart titled "Color: Worse of Reliability Rating and Security Rating; Size: Lines of Code". The chart has "Risk" on the y-axis (0.0% to 100%) and "Coverage" on the x-axis (0.0% to 100%). A large green circle is positioned in the bottom-left quadrant, indicating low risk and high coverage. A note above the chart says: "New Code: Since March 29, 2024". The SonarQube footer includes links for "Community Edition", "Version 9.9 (build 6546)", "LGPL v3", "Community", "Documentation", "Plugins", and "Web API".

TESTING REPORT

Manual Testing

Project Name :	Blood Bank Management System
Module Name :	Admin Module, Donor Module, Blood Bank Module, Hospital Module, Blood Request Module, Login and register page, welcome page
Created By :	Mano Selvaraj - 11991

Test Case	Test Scenario	Test Case	Pre-Condition	Test Step	Test Data	Expected Result	Post Condition	Actual Result	Status (Pass/Fail)
-----------	---------------	-----------	---------------	-----------	-----------	-----------------	----------------	---------------	--------------------

se ID									
BB M S_ 00 01	Verify Login Page	Check login with non exists email	Non exists email and correct password	Enter the non exists email and correct password	Non exists email	Error message shown like "Email is not exists"	Login page with error	Error message shown like "Email is not exists"	Pass
BB M S_ 00 02	Verify Login Page	Check login with non invalid password	exists email and incorrect password	Enter the exists email and incorrect password	Incorr ect passw ord	Error message shown like "Invalid Password"	Login page with error	Error message shown like "Invalid Password"	Pass
BB M S_ 00 03	Verify Login Page	Check login wth empty data	-	Click Login button with no data entry	-	Error message shown like "*Required! enter a email" and "*Required!enter a password"	Login page with error	Error message shown like "*Required!enter a email" and "*Required!enter a password"	Pass
BB M S_ 00 04	Verify Login Page	Check login with correct data , redirect to home page	Correct email and password	Enter the exists email and correct password	Corre ct email and passw ord	Redirect to home page	Home page	Redirect to home page	Pass
BB M S_ 00 05	Verify Forget Password Page	Check forget password with non existe email	Non exists email	Enter non exist email	Non exists email	Error message shown like "Email is not exists"	Error messa ge is displa yed	Error message shown like "Email is not exists"	Pass
BB M S_ 00 06	Verify Forget Password Page	Check with correct email	exists email	Enter the exists email	Exists email	Sent Mail with random password and redirects to change password page	Chang e Passw ord Page	Sent Mail with random password and redirects to change password page	Pass
BB M S_ 00 07	Verify Change Password Page	Check with incorrect password sent in email	Incorrect Password	Enter incorrect password	Incorr ect passw ord	Error message shown like "Old Password is wrong"	Error messa ge is displa yed	Error message shown like "Old Password is wrong"	Pass

BB M S_ 00 08	Verify Change Passwor d Page	Check with correct old password	correct old password	Enter the correct old password	Corre ct passw ord	Redirect to login page	Login page	Redirect to login page	Pass
BB M S_ 00 09	Login with admin creden tial goes to admin dashbo ard	Check with admin email and password	admin email and password	Enter the admin email and password	admin crede ntial	Redirect to admin dashboard	Admin Dashb oard	Redirect to admin dashboard	Pass
BB M S_ 00 10	Login with blood bank creden tial goes to blood bank dashbo ard	Check with blood bank email and password	blood bank email and password	Enter the blood bank email and password	blood bank crede ntial	Redirect to blood bank dashboard	Blood bank Dashb oard	Redirect to blood bank dashboard	Pass
BB M S_ 00 11	Login with hospital creden tial goes to hospital dashbo ard	Check with hospital email and password	hospital email and password	Enter the hospital email and password	hospi tal crede ntial	Redirect to hospital dashboard	Hospit al Dashb oard	Redirect to hospital dashboard	Pass
BB M S_ 00 12	Login with donor creden tial goes to donor home	Check with donor email and password	donor email and password	Enter the donor email and password	donor crede ntial	Redirect to donor home page	Donor home page	Redirect to donor home page	Pass
BB M S_ 00 13	Verify Blood Request Page	Check with entering all data	correct data	Enter the correct data and click the submit button	Corre ct data	Redirect to pending page with request id	Pendin g page	Redirect to pending page with request id	Pass
BB M S_	Verify Blood	Check with	no data	Click button without	-	Error message shown here like	Error mess age is	Error message shown here like	Pass

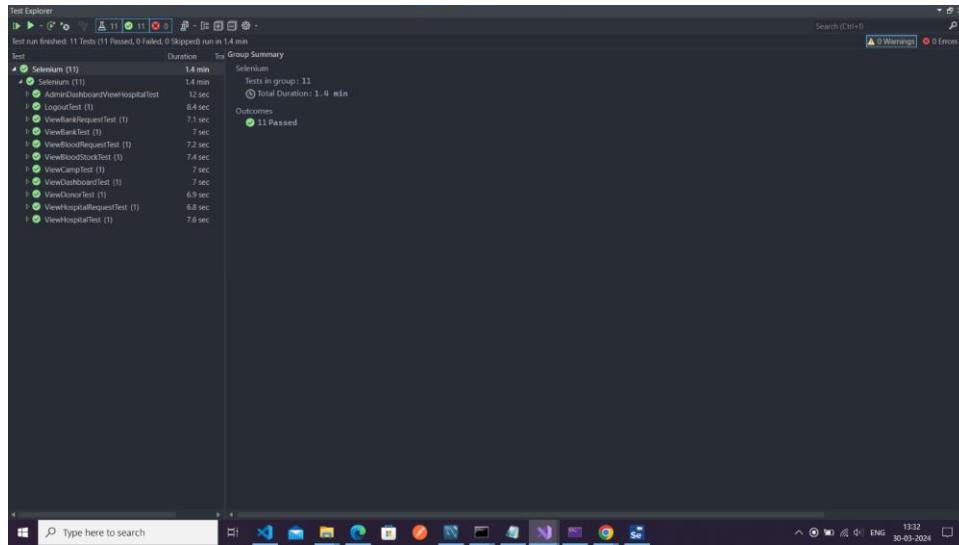
00 14	Request Page	entering no data		entering data		"**Required!" in all field	displayed	"**Required!" in all field	
BB M S_ 00 15	Verify the check blood request page	Check with incorrect id	incorrect id	Enter the incorrect id	incorrect id	Error message shown here like "Invalid Id"	Error message is displayed	Error message shown here like "Invalid Id"	Pass
BB M S_ 00 16	Verify the check blood request page	Check with correct pending request id	correct pending request id	Enter the correct pending request id	correct pending request id	Error message shown here like "Pending"	Error message is displayed	Error message shown here like "Pending"	Pass
BB M S_ 00 17	Verify the check blood request page	Check with correct rejected request id	correct rejected request id	Enter the correct rejected request id	correct rejected request id	Error message shown here like "Rejected"	Error message is displayed	Error message shown here like "Rejected"	Pass
BB M S_ 00 18	Verify the check blood request page	Check with correct approved request id	correct approved request id	Enter the correct approved request id	correct approved request id	List of available blood bank is shown with accepted bank is also shown	Bank details is displayed	List of available blood bank is shown with accepted bank is also shown	Pass
BB M S_ 00 19	Verify the donor registration page	Check with entering all data	correct data	Enter the correct data and click the submit button	Correct data	Redirect to pending page	Pending page	Redirect to pending page	Pass
BB M S_ 00 20	Verify the donor registration page	Check with entering no data	no data	Click button without entering data	—	Error message shown here like "**Required!" in all field	Error message is displayed	Error message shown here like "**Required!" in all field	Pass
BB M S_ 00 21	Verify the blood bank registration page	Check with entering all data	correct data	Enter the correct data and click the submit button	Correct data	Redirect to pending page	Pending page	Redirect to pending page	Pass
BB M	Verify the	Check with	no data	Click button	—	Error message shown here like	Error messa	Error message shown here	Pass

S_0022	blood bank registration page	entering no data		without entering data		"**Required!" in all field	ge is displayed	like "***Required!" in all field	
BB M S_0023	Verify the hospital registration page	Check with entering all data	correct data	Enter the correct data and click the submit button	Correct data	Redirect to pending page	Pending page	Redirect to pending page	Pass
BB M S_0024	Verify the hospital registration page	Check with entering no data	no data	Click button without entering data	—	Error message shown here like "***Required!" in all field	Error message is displayed	Error message shown here like "***Required!" in all field	Pass
BB M S_0025	Verify the Login Page	Check the login with rejected or pending account details	correct email and password where account status is pending / rejected	Enter the correct email and password where account status is pending / rejected and click the login button	correct email and password where account status is pending / rejected	Redirect to pending or rejected page	Pending/Rejected Page	Redirect to pending or rejected page	Pass
BB M S_0026	Verify the add blood stock by blood bank user	Check the blood stock after adding the blood stock	blood type and units	Enter blood type and units	blood type and units	Blood Stock is increased	Check the stock in view page	Blood Stock is increased	Pass
BB M S_0027	Verify the add blood stock by blood bank user	Check the blood stock with no data	—	No data entered and click the add button	—	Error message shown here like "***Required!" in all field	Error message is displayed	Error message shown here like "***Required!" in all field	Pass

BB M S_ 00 28	Verify the add blood camp by blood bank user	Check the blood camp after adding the camp	Blood Camp name,location,date and time	Enter the Blood Camp name,location,date and time data and click add blood camp button	Blood Camp name,location,date and time	Blood camp is added	Check the camp data in view page	Blood camp is added	Pass
BB M S_ 00 29	Verify the add blood camp by blood bank user	Check with no data	-	No data entered and click the add button	-	Error message shown here like "***Required!" in all field	Error message is displayed	Error message shown here like "***Required!" in all field	Pass
BB M S_ 00 30	Verify the add blood transaction by blood bank user	Check the blood transaction after adding the blood transaction details	request id, blood type and units	Enter the request id, blood type and units	request id, blood type and units	Blood Stock is decreased and data is added in view page	Check data in view page	Blood Stock is decreased and data is added in view page	Pass
BB M S_ 00 31	Verify the add blood transaction by blood bank user	Check with no data	-	No data entered and click the add button	-	Error message shown here like "***Required!" in all field	Error message is displayed	Error message shown here like "***Required!" in all field	Pass

Selenium

Output



Check Blood Bank Page Visibility

```
using System;  
  
using System.Collections;  
  
using System.Collections.Generic;  
  
using System.Linq;  
  
using System.Threading;  
  
using OpenQA.Selenium;  
  
using OpenQA.Selenium.Chrome;  
  
using OpenQA.Selenium.Firefox;  
  
using OpenQA.Selenium.Remote;  
  
using OpenQA.Selenium.Support.UI;  
  
using OpenQA.Selenium.Interactions;
```

```
using NUnit.Framework;
namespace Selenium
{
    [TestFixture]
    public class ViewBankTest
    {
        private IWebDriver driver;
        public IDictionary<string, object> vars { get; private set; }
        private IJavaScriptExecutor js;
        [SetUp]
        public void SetUp()
        {
            driver = new ChromeDriver();
            js = (IJavaScriptExecutor)driver;
            vars = new Dictionary<string, object>();
        }
        [TearDown]
        protected void TearDown()
        {
            driver.Quit();
        }
        [Test]
        public void viewBank()
        {
            driver.Navigate().GoToUrl("http://localhost:3000/admindashboard");
            driver.Manage().Window.Size = new System.Drawing.Size(1552, 840);
            driver.FindElement(By.CssSelector(".nav-item:nth-child(4) .d-none")).Click();
            driver.Close();
        }
    }
}
```

```
}
```

```
}
```

Check Dashboard Page Visibility

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Threading;
using OpenQA.Selenium;
using OpenQA.Selenium.Chrome;
using OpenQA.Selenium.Firefox;
using OpenQA.Selenium.Remote;
using OpenQA.Selenium.Support.UI;
using OpenQA.Selenium.Interactions;
using NUnit.Framework;
namespace Selenium
{
    [TestFixture]
    public class ViewDashboardTest
    {
        private IWebDriver driver;
        public IDictionary<string, object> vars { get; private set; }
        private IJavaScriptExecutor js;
        [SetUp]
        public void SetUp()
        {
            driver = new ChromeDriver();
            js = (IJavaScriptExecutor)driver;
```

```
vars = new Dictionary<string, object>();

}

[TearDown]

protected void TearDown()

{

    driver.Quit();

}

[Test]

public void viewDashboard()

{

    driver.Navigate().GoToUrl("http://localhost:3000/admindashboard");

    driver.Manage().Window.Size = new System.Drawing.Size(1552, 840);

    driver.FindElement(By.CssSelector(".nav-item:nth-child(1) .d-none")).Click();

    driver.FindElement(By.CssSelector(".nav-item:nth-child(1) .d-none")).Click();

    driver.Close();

}

}

}


```

Check Hospital Page Visibility

```
using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Threading;

using OpenQA.Selenium;

using OpenQA.Selenium.Chrome;

using OpenQA.Selenium.Firefox;
```

```
using OpenQA.Selenium.Remote;
using OpenQA.Selenium.Support.UI;
using OpenQA.Selenium.Interactions;
using NUnit.Framework;
namespace Selenium
{
    [TestFixture]
    public class AdminDashboardViewHospitalTest
    {
        private IWebDriver driver;
        public IDictionary<string, object> vars { get; private set; }
        private IJavaScriptExecutor js;
        [SetUp]
        public void SetUp()
        {
            driver = new ChromeDriver();
            js = (IJavaScriptExecutor)driver;
            vars = new Dictionary<string, object>();
        }
        [TearDown]
        protected void TearDown()
        {
            driver.Quit();
        }
        [Test]
        public void adminDashboardViewHospital()
        {
            driver.Navigate().GoToUrl("http://localhost:3000/viewhospital");
            driver.Manage().Window.Size = new System.Drawing.Size(1552, 840);
```

```
    driver.FindElement(By.CssSelector(".nav-item:nth-child(2) .d-none")).Click();  
}  
}  
  
}
```

Check Logout Button

```
using System;  
using System.Collections;  
using System.Collections.Generic;  
using System.Linq;  
using System.Threading;  
using OpenQA.Selenium;  
using OpenQA.Selenium.Chrome;  
using OpenQA.Selenium.Firefox;  
using OpenQA.Selenium.Remote;  
using OpenQA.Selenium.Support.UI;  
using OpenQA.Selenium.Interactions;  
using NUnit.Framework;  
namespace Selenium  
{  
    [TestFixture]  
    public class LogoutTest  
    {  
        private IWebDriver driver;  
        public IDictionary<string, object> vars { get; private set; }  
        private IJavaScriptExecutor js;  
        [SetUp]  
        public void SetUp()  
    }
```

```

{
    driver = new ChromeDriver();
    js = (IJavaScriptExecutor)driver;
    vars = new Dictionary<string, object>();
}

[TearDown]

protected void TearDown()
{
    driver.Quit();
}

[Test]

public void logout()
{
    driver.Navigate().GoToUrl("http://localhost:3000/admindashboard");
    driver.Manage().Window.Size = new System.Drawing.Size(1552, 840);
    driver.FindElement(By.CssSelector(".navbg > .d-none")).Click();
    driver.Close();
}

}
}

}

```

Check Bank Request Page Visibility

```

using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Threading;
using OpenQA.Selenium;

```

```
using OpenQA.Selenium.Chrome;
using OpenQA.Selenium.Firefox;
using OpenQA.Selenium.Remote;
using OpenQA.Selenium.Support.UI;
using OpenQA.Selenium.Interactions;
using NUnit.Framework;
namespace Selenium
{
    [TestFixture]
    public class ViewBankRequestTest
    {
        private IWebDriver driver;
        public IDictionary<string, object> vars { get; private set; }
        private IJavaScriptExecutor js;
        [SetUp]
        public void SetUp()
        {
            driver = new ChromeDriver();
            js = (IJavaScriptExecutor)driver;
            vars = new Dictionary<string, object>();
        }
        [TearDown]
        protected void TearDown()
        {
            driver.Quit();
        }
        [Test]
        public void viewBankRequest()
        {

```

```
        driver.Navigate().GoToUrl("http://localhost:3000/admindashboard");

        driver.Manage().Window.Size = new System.Drawing.Size(1552, 840);

        driver.FindElement(By.CssSelector(".nav-item:nth-child(8) .d-none")).Click();

        driver.Close();

    }

}

}

}
```

Check Blood Request Page Visibility

```
using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Threading;

using OpenQA.Selenium;

using OpenQA.Selenium.Chrome;

using OpenQA.Selenium.Firefox;

using OpenQA.Selenium.Remote;

using OpenQA.Selenium.Support.UI;

using OpenQA.Selenium.Interactions;

using NUnit.Framework;

namespace Selenium

{

    [TestFixture]

    public class ViewBloodRequestTest

    {

        private IWebDriver driver;

        public IDictionary<string, object> vars { get; private set; }

        private IJavaScriptExecutor js;
```

```

[SetUp]
public void SetUp()
{
    driver = new ChromeDriver();
    js = (IJavaScriptExecutor)driver;
    vars = new Dictionary<string, object>();
}

[TearDown]
protected void TearDown()
{
    driver.Quit();
}

[Test]
public void viewBloodRequest()
{
    driver.Navigate().GoToUrl("http://localhost:3000/admindashboard");
    driver.Manage().Window.Size = new System.Drawing.Size(1552, 840);
    driver.FindElement(By.CssSelector(".nav-item:nth-child(8) .d-none")).Click();
    driver.FindElement(By.CssSelector(".nav-item:nth-child(9) .d-none")).Click();
    driver.Close();
}

}

```

}

Check Blood Stock Page Visibility

```

using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;

```

```
using System.Threading;
using OpenQA.Selenium;
using OpenQA.Selenium.Chrome;
using OpenQA.Selenium.Firefox;
using OpenQA.Selenium.Remote;
using OpenQA.Selenium.Support.UI;
using OpenQA.Selenium.Interactions;
using NUnit.Framework;
namespace Selenium
{
    [TestFixture]
    public class ViewBloodStockTest
    {
        private IWebDriver driver;
        public IDictionary<string, object> vars { get; private set; }
        private IJavaScriptExecutor js;
        [SetUp]
        public void SetUp()
        {
            driver = new ChromeDriver();
            js = (IJavaScriptExecutor)driver;
            vars = new Dictionary<string, object>();
        }
        [TearDown]
        protected void TearDown()
        {
            driver.Quit();
        }
        [Test]
```

```
public void viewBloodStock()
{
    driver.Navigate().GoToUrl("http://localhost:3000/admindashboard");
    driver.Manage().Window.Size = new System.Drawing.Size(1552, 840);
    driver.FindElement(By.CssSelector(".nav-item:nth-child(10) .d-none")).Click();
    driver.Close();
}

}
```



```
}
```

Coding:

View Hospital Component Test

```
import React from 'react';
import { render, screen, waitFor } from '@testing-library/react';
import axios from 'axios';
import ViewHospitalComponent from '../Component/ViewHospitalComponent';

jest.mock('axios');

describe('ViewHospitalComponent', () => {
  it('renders hospital data fetched from API', async () => {
    const mockData = {
      data: [
        {
          "id": "26e36c6a-0c9a-48e2-8eac-0296706b2604",
          "name": "ABC Hospital",
          "email": "smano4570@gmail.com",
          "phoneNumber": 6382202586,
          "location": "Chennai",
          "governmentId": "1245433",
          "document": "document",
          "doorNo": "21/2",
          "street": "Salai Street,",
          "area": "Ktc Nagar",
          "city": "Tirunelveli",
          "state": "Tamil Nadu",
          "postalCode": "627011",
          "status": 1
        }
      ]
    };

    await render();
    expect(screen.getByText("ABC Hospital")).toBeInTheDocument();
  });
});
```

```

        }
    ]
}

axios.get.mockResolvedValue(mockData);

const { getByTestId, getAllByTestId } = render(<ViewHospitalComponent />);

await waitFor(() => {
    expect(getByTestId('row-26e36c6a-0c9a-48e2-8eac-0296706b2604')).toBeInTheDocument();
});

});
});
});

```

View Donor Component Test

```

import React from 'react';
import { render, waitFor } from '@testing-library/react';
import axios from 'axios';
import ViewDonorComponent from '../Component/ViewDonorComponent';

```

```

jest.mock('axios');

```

```

describe('ViewDonorComponent', () => {
    it('renders donor information', async () => {
        const data = {

```

```
data: [
  {
    "accountUserDetailsAddressId": "50fa9fae-bfb2-4e5c-a219-2ebdc1fb4f47",
    "account": {
      "accountId": "477b7d70-299c-4428-a130-7b5bf5176586",
      "name": "Keerthi Vasan",
      "email": "keerthivasan280901@gmail.com",
      "password": "ceb200a6cea4095b86c8bc4d1e801aff9812eaa106fd940e348882d0268d59f5",
      "phoneNumber": 9089678776,
      "status": 1,
      "bloodTransaction": null,
      "bloodCampBloodBank": null,
      "bloodBankBloodStock": null
    },
    "userDetails": {
      "userDetailsId": "1b60eba4-66f9-496f-a764-7afb10d7ee87",
      "age": 21,
      "bloodType": "B+ve",
      "location": "Chennai",
      "governmentId": "",
      "document": "",
      "aadhaarNumber": 345623458900,
      "rolestatus": 3
    },
    "address": {
      "addressId": "b6872d86-1eb1-4fca-b2ad-3f2c798440b7",
      "doorNo": "21",
      "street": "Vk Street",
      "area": "Narayana nagar",
    }
  }
]
```

```

        "city": "Perambalur",
        "state": "Tamil Nadu",
        "postalCode": "627001"

    }
}

];

};

axios.get.mockResolvedValue(data);

const { getByTestId } = render(<ViewDonorComponent />);

await waitFor(() => {

expect(getByTestId('row-477b7d70-299c-4428-a130-7b5bf5176586')).toBeInTheDocument();

});

});

});

};


```

View Blood Transaction Component Test

```

import React from 'react';

import { render, waitFor } from '@testing-library/react';

import axios from 'axios';

import ViewBloodTransaction from '../Component/ViewBloodTransaction';

import Cookies from 'js-cookie';

jest.mock('axios');

jest.mock('js-cookie');

describe('ViewBloodTransaction', () => {

```

```
it('renders blood transaction information', async () => {
  const data = {
    data: [
      {
        "bloodTransactionId": "cbb7f7ec-2ebf-45b3-af88-12729cb4fa41",
        "accountId": "10bab100-d6b3-41d0-9d3b-46491cda7ca0",
        "account": {
          "accountId": "10bab100-d6b3-41d0-9d3b-46491cda7ca0",
          "name": "ABC Blood Bank",
          "email": "sanjaihari2002@gmail.com",
          "password": "ceb200a6cea4095b86c8bc4d1e801aff9812eaa106fd940e348882d0268d59f5",
          "phoneNumber": 6382202487,
          "status": 1,
          "bloodTransaction": null,
          "bloodCampBloodBank": null,
          "bloodBankBloodStock": null
        },
        "bloodRequestId": "T2937kCU",
        "bloodRequest": {
          "bloodRequestId": "T2937kCU",
          "name": "Sanjai",
          "email": "sanjairock85@gmail.com",
          "units": 2,
          "phoneNumber": 9867541243,
          "bloodType": "B+ve",
          "age": 20,
          "location": "Chennai",
          "aadhaarNumber": 235489098762,
          "validTime": "03/29/2024 17:11:20",
        }
      }
    ]
  }
})
```

```

    "status": 4,
    "acceptStatus": 1,
    "bloodTransaction": null
  },
  "bloodType": "B+ve",
  "units": 0,
  "date": "string",
  "time": "string"
}

]
};

axios.post.mockResolvedValue(data);
Cookies.get.mockReturnValue('sample-id');

const { getByTestId } = render(<ViewBloodTransaction />);

await waitFor(() => {
  expect(getByTestId('row-cbb7f7ec-2ebf-45b3-af88-12729cb4fa41')).toBeInTheDocument();
});

});
});

});
});

```

Unauthorized Component Test

```

import React from 'react';
import { render } from '@testing-library/react';
import UnAuthorizedComponent from '../Component/UnAuthorizedComponent';

```

```

describe('UnAuthorizedComponent', () => {
  it('renders without crashing', () => {
    const { getByTestId } = render(<UnAuthorizedComponent />);
    expect(getByTestId('unauthorized-section')).toBeInTheDocument();
  });

  it('displays the unauthorized image', () => {
    const { getByTestId } = render(<UnAuthorizedComponent />);
    const img = getByTestId('unauthorized-image');
    expect(img).toBeInTheDocument();
    expect(img).toHaveAttribute('src', expect.stringContaining('accessdenied.png'));
  });
});

```

Reject Component Test

```

import React from 'react';
import { render } from '@testing-library/react';
import RejectComponent from '../Component/RejectComponent';

describe('RejectComponent', () => {
  it('renders the reject image', () => {
    const { getByTestId } = render(<RejectComponent />);
    const imgElement = getByTestId('reject-image');
    expect(imgElement).toBeInTheDocument();
    expect(imgElement).toHaveAttribute('src', expect.stringContaining('reject.png'));
  });
});

```

Pending Component Test

```

import React from 'react';

```

```
import { render, screen } from '@testing-library/react';
import { BrowserRouter } from 'react-router-dom';
import PendingComponent from '../Component/PendingComponent';

describe('PendingComponent', () => {
  it('renders the pending message', () => {
    render(
      <BrowserRouter>
        <PendingComponent />
      </BrowserRouter>
    );
  });

  const pendingMessage = screen.getByText(/Your Request has been waiting for management approval/i);
  expect(pendingMessage).toBeInTheDocument();
});

});
```

Not Found Component Test

```
import React from 'react';
import { render } from '@testing-library/react';
import NotFoundComponent from '../Component/NotFoundComponent';

describe('NotFoundComponent', () => {
  it('renders without crashing', () => {
    const { getByTestId } = render(<NotFoundComponent />);
    expect(getByTestId('notfound-section')).toBeInTheDocument();
  });
});
```

```
});

it('displays the not found image with correct dimensions', () => {
  const { getByTestId } = render(<NotFoundComponent />);
  const img = getByTestId('notfound-image');
  expect(img).toBeInTheDocument();
  expect(img).toHaveAttribute('src', expect.stringContaining('notfound.png'));
  expect(img).toHaveAttribute('width', '350px');
  expect(img).toHaveAttribute('height', '150px');
});
});
```

Login Component Test

```
import React from 'react';
import { render, fireEvent, waitFor } from '@testing-library/react';
import LoginComponent from '../Component/LoginComponent';
import axios from 'axios'
import { BrowserRouter } from 'react-router-dom';

const mockedNavigate = jest.fn();
jest.mock('react-router-dom', () => ({
  ...jest.requireActual('react-router-dom'),
  useNavigate: () => mockedNavigate,
}));

describe('LoginComponent', () => {
  it('renders email and password inputs', () => {
    const { getByTestId } = render(
      <BrowserRouter><LoginComponent /></BrowserRouter>);
  });
});
```

```
const emailInput = getByTestId('email-input');

const passwordInput = getByTestId('password-input');

expect(emailInput.toBeInTheDocument());
expect(passwordInput.toBeInTheDocument());

});

it('submits form with valid email and password', async () => {
  const mockAxios = jest.spyOn(axios, 'post').mockResolvedValueOnce({
    data: {
      accountExists: true,
      passwordStatus: true,
      accountApproval: 'approve',
      token: 'mockToken',
    },
  });
});

const { getByTestId } = render(<BrowserRouter><LoginComponent /></BrowserRouter>);

const emailInput = getByTestId('email-input');

const passwordInput = getByTestId('password-input');

const loginButton = getByTestId('login-button');

fireEvent.change(emailInput, { target: { value: 'validemail@example.com' } });
fireEvent.change(passwordInput, { target: { value: 'validpassword' } });
fireEvent.click(loginButton);

await waitFor(() => {
```

Forget Password Test

```
import React from 'react';
import { render, fireEvent, waitFor } from '@testing-library/react';
import ForgetPasswordComponent from '../Component/ForgetPasswordComponent';
import axios from 'axios';
import { BrowserRouter } from 'react-router-dom';

jest.mock('axios');

describe('ForgetPasswordComponent', () => {
  it('renders correctly', () => {
    const { getByTestId } = render(
      <BrowserRouter>
        <ForgetPasswordComponent />
      </BrowserRouter>
    );
    expect(getByTestId('email-input')).toBeInTheDocument();
    expect(getByTestId('submit-button')).toBeInTheDocument();
  });
})
```

```
});

it('submits the form with valid email', async () => {
  axios.post.mockResolvedValue({ data: { emailExists: true, sendMail: true } });
  const { getByTestId } = render(
    <BrowserRouter>
      <ForgetPasswordComponent />
    </BrowserRouter>
  );
  fireEvent.change(getByTestId('email-input'), { target: { value: 'test@example.com' } });
  fireEvent.click(getByTestId('submit-button'));

  await waitFor(() => {
    expect(axios.post).toHaveBeenCalledWith('https://localhost:7089/api/ForgetPassword', {
      email: 'test@example.com',
    });
  });
  });

});
```

Change Password Test

```
import React from 'react';
import { render, fireEvent, waitFor } from '@testing-library/react';
import ChangePasswordComponent from '../Component/ChangePasswordComponent';
import axios from 'axios';
import { BrowserRouter as Router } from 'react-router-dom';
jest.mock('axios');
const mockedNavigate = jest.fn();
jest.mock('react-router-dom', () => ({
...jest.requireActual('react-router-dom'),
useNavigate: () => mockedNavigate,
}));
```

```
describe('ChangePasswordComponent', () => {
it('renders correctly', () => {
const { getByTestId } = render(<Router>
<ChangePasswordComponent />
</Router>);
expect(getByTestId('email-input')).toBeInTheDocument();
expect(getByTestId('old-password-input')).toBeInTheDocument();
expect(getByTestId('new-password-input')).toBeInTheDocument();
expect(getByTestId('submit-button')).toBeInTheDocument();
});
```

```
it('submits the form with valid data', async () => {
axios.post.mockResolvedValue({ data: { emailExists: true, passcheck: true } });
});
```

```
const { getByTestId } = render(<ChangePasswordComponent />);

fireEvent.change(getByTestId('email-input'), { target: { value: 'test@example.com' } });

fireEvent.change(getByTestId('old-password-input'), { target: { value: 'oldPassword123' } });

fireEvent.change(getByTestId('new-password-input'), { target: { value: 'newPassword123' } });

fireEvent.click(getByTestId('submit-button'));

await waitFor(() => {

  expect(axios.post).toHaveBeenCalledWith('https://localhost:7089/api/ChangePassword', {

    email: 'test@example.com',

    oldPassword: 'oldPassword123',

    newPassword: 'newPassword123',

  });

});

});});
```

Blood Bank Request Component Test

```
import React from 'react';

import { render, fireEvent, waitFor } from '@testing-library/react';

import axios from 'axios';

import BloodBankRequestComponent from '../Component/BloodBankRequestComponent'; // Adjust the import path as necessary

jest.mock('axios');

describe('BloodBankRequestComponent', () => {

  it('renders hospital data and can approve a request', async () => {
```

```
const hospitalData = [  
  {  
    "id": "10bab100-d6b3-41d0-9d3b-46491cda7ca0",  
    "name": "ABC Blood Bank",  
    "email": "sanjaihari2002@gmail.com",  
    "phoneNumber": 6382202487,  
    "location": "Chennai",  
    "governmentId": "1245433",  
    "document": "document",  
    "doorNo": "24",  
    "street": "VK Street,",  
    "area": "Ktc Nagar",  
    "city": "Viruthunagar",  
    "state": "Tamil Nadu",  
    "postalCode": "627011",  
    "status": 0  
  },  
];  
  
axios.get.mockResolvedValue({ data: hospitalData });  
  
const { getByTestId } = render(<BloodBankRequestComponent />);  
  
await waitFor(() => expect(getByTestId('row-10bab100-d6b3-41d0-9d3b-46491cda7ca0')).toBeInTheDocument());
```

```
});
```

```
});
```

Add Blood Stock Component Test

```
import React from 'react';

import { render, fireEvent, waitFor } from '@testing-library/react';

import AddBloodStockComponent from '../Component/AddBloodStockComponent';

import axios from 'axios';

import Cookies from 'js-cookie';

jest.mock('axios');

jest.mock('js-cookie', () => ({
  get: jest.fn(),
}));
```



```
describe('AddBloodStockComponent', () => {
  beforeEach(() => {
    jest.clearAllMocks();
  });

  it('renders correctly', () => {
    const { getByTestId } = render(<AddBloodStockComponent />);

    expect(getByTestId('headname')).toHaveTextContent('Add Blood Stock');
  });
});
```

```
it('submits the form with correct blood type and units', async () => {
  Cookies.get.mockReturnValue('scbc238wej873cewd72');

  const { getByTestId } = render(<AddBloodStockComponent />);

  const bloodTypeSelect = getByTestId('BloodType');

  const unitInput = getByTestId('Unit');

  const addButton = getByTestId('addbutton');

  fireEvent.change(bloodTypeSelect, { target: { value: 'A+ve' } });

  fireEvent.change(unitInput, { target: { value: '5' } });

  axios.post.mockResolvedValue({ status: 200 });

  fireEvent.click(addButton);

  await waitFor(() => {
    expect(axios.post).toHaveBeenCalledWith('https://localhost:7089/api/BloodStock', {
      bloodStockId: '',
      bloodType: 'A+ve',
      units: 5,
      accountId: 'scbc238wej873cewd72',
    });
  });
});});
```

Add Blood Camp Test

```
import React from 'react';
```

```
import { render, fireEvent, waitFor } from '@testing-library/react';
import '@testing-library/jest-dom';
import AddBloodCampComponent from '../Component/AddBloodCampComponent';
import axios from 'axios';
import Cookies from 'js-cookie';

jest.mock('axios');
jest.mock('js-cookie');

describe('AddBloodCampComponent', () => {
  beforeEach(() => {
    Cookies.get.mockReturnValue('test-id');
  });

  it('renders without crashing', () => {
    const { getByTestId } = render(<AddBloodCampComponent />);
    expect(getByTestId('headname')).toHaveTextContent('Blood Camp');
  });

  it('submits the form with correct data', async () => {
    axios.post.mockResolvedValue({ status: 200 });
    const { getByTestId } = render(<AddBloodCampComponent />);

    fireEvent.change(getByTestId('bloodCampName'), { target: { value: 'Test Camp' } });
    fireEvent.change(getByTestId('bloodCampLocation'), { target: { value: 'Test Location' } });
    fireEvent.change(getByTestId('date'), { target: { value: '2024-01-30' } });

    await waitFor(() => {
      expect(axios.post).toHaveBeenCalledWith('http://localhost:3001/blood-camps', {
        bloodCampName: 'Test Camp',
        bloodCampLocation: 'Test Location',
        date: '2024-01-30'
      });
    });
  });
});
```

```

fireEvent.change(getByTestId('time'), { target: { value: '10:00' } });

fireEvent.click(getByTestId('bloodcampbtn'));


await waitFor(() => {

  expect(axios.post).toHaveBeenCalledWith('https://localhost:7089/api/BloodCamp', {

    bloodCampName: 'Test Camp',

    bloodCampLocation: 'Test Location',

    date: '2024-01-30',

    time: '10:00',

    accountId: 'test-id'

  });

});

});

});

});

```

Output

```

✓ renders correctly (229 ms)
✓ submits the form with valid data (45 ms)

Test Suites: 13 passed, 13 total
Tests:       20 passed, 20 total
Snapshots:   0 total
Time:        25.688 s
Ran all test suites related to changed files.

```

```

(c) Microsoft Corporation. All rights reserved.
D:\MicroProject\BloodBankManagementApp>npm test
> bloodbankmanagementapp@0.1.0 test
> react-scripts test --verbose

```

```
C:\Windows\system32\cmd.exe
at validateInputTypeInDevelopment (node_modules/react-dom/cjs/react-dom.development.js:9541:5)
at getInitialValueForFormInputs (node_modules/react-dom/cjs/react-dom.development.js:10958:17)
at finalizeInitialChild (node_modules/react-dom/cjs/react-dom.development.js:10958:3)
at completeWork (node_modules/react-dom/cjs/react-dom.development.js:22393:17)
at performUnitOfWork (node_modules/react-dom/cjs/react-dom.development.js:26508:16)
at workLoopSync (node_modules/react-dom/cjs/react-dom.development.js:26506:5)
at renderRootSync (node_modules/react-dom/cjs/react-dom.development.js:26444:7)
at renderRoot (node_modules/react-dom/cjs/react-dom.development.js:25738:74)
at flushBatchQueue (node_modules/react-dom/cjs/react-dom.development.js:2667:24)
at act (node_modules/react/cjs/react.development.js:2582:11)
at renderRoot (node_modules/react-dom/cjs/react-dom.development.js:25738:29)
at render (node_modules/@testing-library/react/dist/pure.js:159:26)
at render (node_modules/@testing-library/react/dist/pure.js:246:10)
at Object. (src/test/ChangePasswordTest.test.js:15:39)

console.log
{ data: { emailExists: true, sendMail: true } }

at src/Component/ForgetPasswordComponent.js:17:17
src/test/ForgetPasswordTest.test.js (16.095 ms)
ForgetPasswordComponent
✓ renders correctly (223 ms)
✓ submits the form with valid email (41 ms)

console.log
{ data: { emailExists: true, passcheck: true } }

at src/Component/ChangePasswordComponent.js:54:17
src/test/ChangePasswordTest.test.js (16.191 ms)
ChangePasswordComponent
✓ renders correctly (229 ms)
✓ submits the form with valid data (45 ms)

Test Suites: 13 passed, 13 total
Tests:    20 passed, 20 total
Snapshots: 0 total
Time:    0:00:08.688 s
Ran all test suites related to changed files.

watch Usage
› Press a to run all tests.
› Press f to run only failed tests.
› Press p to filter by a path.
› Press g to filter by a filename regex pattern.
› Press t to filter by a test name regex pattern.
› Press Enter to trigger a test run.
```

8:42 30-03-2024

CI / CD

- **Docker**
- **Jenkins**

Docker

React

```
FROM node:latest
```

```
WORKDIR /app/reactapp
```

```
COPY package*.json /app/reactapp
```

```
RUN npm install
```

```
COPY . /app/reactapp
```

```
CMD [ "npm" , "start" ]
```

Dotnet Webapi

```
FROM --platform=$BUILDPLATFORM mcr.microsoft.com/dotnet/sdk:8.0-alpine AS build
```

```
COPY . /source
```

```
WORKDIR /source
```

```
ARG TARGETARCH
```

```
RUN --mount=type=cache,id=nuget,target=/root/.nuget/packages \
```

```
dotnet publish -a ${TARGETARCH/amd64/x64} --use-current-runtime --self-contained false -o /app
```

```
FROM mcr.microsoft.com/dotnet/aspnet:8.0-alpine AS final
```

```
WORKDIR /app
```

```
COPY --from=build /app .
```

```
USER $APP_UID
```

```
ENTRYPOINT ["dotnet", "BloodBankManagementWebapi.dll"]
```

Compose File

```
version: '3.4'

networks:

dev:
  driver: bridge

services:
  myapp:
    depends_on:
      - "simple"
    build:
      context: ./bloodbankmanagementapp
      dockerfile: Dockerfile
    ports:
      - "8090:3000"
    networks:
      - dev
  simple:
    depends_on:
      - "app_db"
    ports:
      - "8081:80"
    build:
      context: ./BloodBankManagementWebapi/BloodBankManagementWebapi
      dockerfile: Dockerfile
```

environment:

- ConnectionStrings__DefaultConnectionString=Server=app_db;Port=3306;Database=sampledbs;Uid=root;Pwd=root

- ASPNETCORE_URLS=http://+:80

networks:

- dev

app_db:

image: mysql:latest

container_name: app_db

environment:

- MYSQL_ROOT_PASSWORD=root

- MYSQL_DATABASE=sampledbs

- MYSQL_USER=mano

- MYSQL_PASSWORD=admin123

ports:

- "6703:3306"

restart: always

volumes:

- app_data:/var/lib/mysql

networks:

- dev

volumes:

app_data:

Output

In command line

The screenshot shows the Visual Studio Code interface with the Docker extension. The Explorer sidebar on the left lists files and folders related to the project, including `compose.yml`, `Dockerfile`, and `Program.cs`. The main editor area displays the `compose.yml` file:

```
version: '3.8'
services:
  app_db:
    image: mysql:5.7
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: bloodbankmanagement
      MYSQL_USER: bloodbank
      MYSQL_PASSWORD: bloodbank
    ports:
      - "8090:3000"
  simple-1:
    image: mcr.microsoft.com/dotnet/core/sdk:3.1
    ports:
      - "8090:3000"
  myapp-1:
    image: mcr.microsoft.com/dotnet/core/aspnet:3.1
    depends_on:
      - app_db
    ports:
      - "8090:3000"
    networks:
      - dev
```

The terminal tab at the bottom shows the command `docker compose up` being run, and the output indicates that three containers are running: `app_db`, `simple-1`, and `myapp-1`.

This screenshot is identical to the one above, showing the Visual Studio Code interface with the Docker extension. The Explorer sidebar, main editor area with `compose.yml`, and terminal output all remain the same.

In Docker Desktop

Images

The screenshot shows the Docker Desktop application window. The left sidebar includes options for `Containers`, `Images` (which is selected), `Volumes`, `Builds` (marked as `NEW`), `Dev Environments` (marked as `BETA`), and `Docker Scout`. The main pane is titled `Images` and shows a table of local images:

Name	Tag	Status	Created	Size	Actions
mysql	latest	In use	3 months ago	618.82 MB	[...]
microproject-simple	latest	In use	4 hours ago	176.79 MB	[...]
microproject-myapp	latest	In use	29 seconds ago	1.46 GB	[...]

At the bottom, there are sections for `Walkthroughs` (with a link to "How do I run a container?") and `Run Docker Hub images`. The status bar at the bottom shows "Engine running" and "RAM 3.76 GB CPU 40.86% Signed in".

Container

Docker Desktop interface showing the 'Containers' tab. The sidebar includes 'Containers', 'Images', 'Volumes', 'Builds', 'Dev Environments (BETA)', and 'Docker Scout'. Under 'Extensions', there's an option to 'Add Extensions'. The main area shows a table of running containers:

	Name	Image	Status	CPU (%)	Port(s)	Last started	Actions
<input type="checkbox"/>	microproject	Running (3/3)	0%	23 seconds ago	.../microproject	.../microproject	.../microproject
<input type="checkbox"/>	myapp-1	microproject-myapp: Running	0%	23 seconds ago	.../myapp-1	.../myapp-1	.../myapp-1
<input type="checkbox"/>	simple-1	microproject-simple: Running	0%	25 seconds ago	.../simple-1	.../simple-1	.../simple-1
<input type="checkbox"/>	app_db	mysql:latest	Running	0%	.../app_db	.../app_db	.../app_db

Showing 4 items

System status at the bottom: Engine running, RAM 3.74 GB, CPU 65.65%, Signed in.

Docker Desktop interface showing the 'Containers' tab for the 'microproject' service. The sidebar includes 'Containers', 'Images', 'Volumes', 'Builds', 'Dev Environments (BETA)', and 'Docker Scout'. Under 'Extensions', there's an option to 'Add Extensions'. The main area shows a table of running containers and a terminal window displaying logs for the 'app_db' container:

```

2024-03-30 22:01:21 app_db | 2024-03-30T16:31:20.992231Z 0 [System] [MY-013602]
[Server] Channel mysql_main configured to support TLS. Encrypted connections are no
w supported for this channel.
2024-03-30 22:01:21 app_db | 2024-03-30T16:31:21.005154Z 0 [Warning] [MY-011810]
[Server] Insecure configuration for --pid-file: location '/var/run/mysqld' in the
path is accessible to all OS users. Consider choosing a different directory.
2024-03-30 22:01:21 app_db | 2024-03-30T16:31:21.121563Z 0 [System] [MY-011323]
[Server] X Plugin ready for connections. Bind-address: '::' port: 3306, socket: /v
ar/run/mysqld/mysqld.sock
2024-03-30 22:01:21 app_db | 2024-03-30T16:31:21.122039Z 0 [System] [MY-010931]
[Server] /usr/sbin/mysqld: ready for connections. Version: '8.2.0' socket: '/var/r
un/mysqld/mysqld.sock' port: 3306 MySQL Community Server - GPL.
2024-03-30 22:00:36 myapp-1 |
2024-03-30 22:00:38 myapp-1 | > bloodbankmanagementapp@0.1.0 start
2024-03-30 22:00:38 myapp-1 | > react-scripts start
2024-03-30 22:00:38 myapp-1 |
2024-03-30 22:00:43 myapp-1 | (node:26) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_M
IDDLEWARE] DeprecationWarning: 'onAfterSetupMiddleware' option is deprecated. Please
use the 'setupMiddlewares' option.
2024-03-30 22:00:43 myapp-1 | (Use 'node --trace-deprecation ...' to show where t
he warning was created)
2024-03-30 22:00:43 myapp-1 | (node:26) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_M
IDDLEWARE] DeprecationWarning: 'onBeforeSetupMiddleware' option is deprecated. Plea
se use the 'setupMiddlewares' option.
2024-03-30 22:00:43 myapp-1 | Starting the development server...
2024-03-30 22:00:43 myapp-1 |

```

System status at the bottom: Engine running, RAM 3.75 GB, CPU 75.96%, Signed in.

In Browser

React

Browser screenshot showing a React application for blood donation. The URL is 'localhost:8090'. The page has a header with 'Blood Request Status | Login' and a main section titled 'Donate Blood Save Life' featuring an illustration of people donating blood. Below the illustration are four buttons: 'Blood Request', 'Blood Bank', 'Donor', and 'Hospital'. A 'Start' button is located at the bottom left.

Dotnet webapi

The screenshot shows the Swagger UI interface for a BloodBankManagementWebapi v1. The top navigation bar includes tabs for Chat, Mail, and React App. The main content area is titled "BloodBankManagementWebapi" and "1.0 (OAS2)". A search bar at the top right says "Select a definition" and "BloodBankManagementWebapi v1". Below the title, there are sections for "AcceptRequesterByBank", "AdminDashboard", "ApproveBloodRequestByAdmin", and "ApproveOrRejectAccountByAdmin", each listing specific API endpoints with their methods (e.g., POST, GET) and URLs.

Jenkins

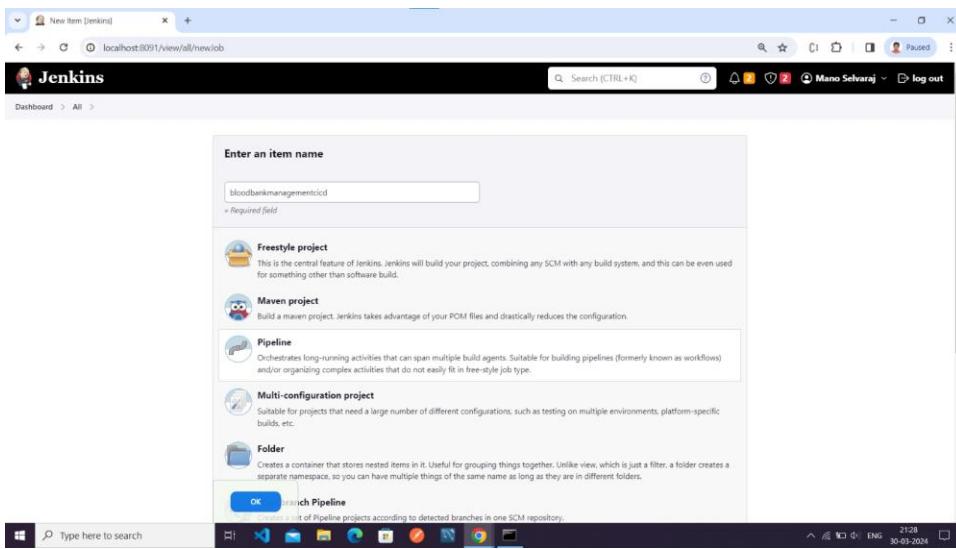
Step – 1 : Open jenkins

The screenshot shows the Jenkins dashboard. On the left, there are links for "New Item", "People", "Build History", "Manage Jenkins", and "My Views". The main area displays a table of build jobs with columns for "S" (Status), "W" (Last build), "Name", "Last Success", "Last Failure", and "Last Duration". The table entries are:

S	W	Name	Last Success	Last Failure	Last Duration
○	✖	AssessmentJenkins	28 days #4	N/A	38 sec
○	✖	AssessmentDemo	28 days #10	28 days #9	31 min
○	✖	FirstProject	1 mo 3 days #15	1 mo 3 days #13	14 sec
✖	✖	MyCI/CDPipeline	1 mo 2 days #5	1 mo 2 days #11	3 min 21 sec
○	✖	ReactSample	N/A	N/A	N/A
○	✖	ReactSample1	16 days #8	16 days #7	6 min 15 sec

At the bottom, there are links for "Icon: S M L", "Icon legend", "Atom feed for all", "Atom feed for failures", and "Atom feed for just latest builds". The status bar at the bottom right shows "REST API Jenkins 2.426.3", "21:28", "ENR", and the date "30-03-2024".

Step – 2 : Create a pipeline project



Step – 3 : Do script for automation

Script ?

```

1 node
2 {
3   stage('1. GIT Checkout') {
4     git 'https://github.com/smano-20052002/MicroProjectFinal.git'
5   }
6   dir('BloodBankManagementWebapi'){
7     stage('2. Project Build') {
8       bat 'dotnet build'
9     }
10    stage('2. Project Test') {
11      bat 'dotnet test'
12    }
13    stage('3. CodeQuality Test') {
14      bat 'dotnet sonarscanner begin /k:"MicroProjectCICD" /d:sonar.host.url="http://localhost:9000" /d:sonar.login="sqp_669dbc79281094d7a0a490d73b9'
15    }
16    stage('4. CodeQuality Build'){
17      bat 'dotnet build'
18    }
19    stage('5. CodeQuality Report'){
20      bat 'dotnet sonarscanner end /d:sonar.login="sqp_669dbc79281094d7a0a490d73b94c5889d47d6a2"'
21  }
22 }
23 stage('6. Dockerization'){
24   bat 'docker compose up --build'
25 }
26
27

```

Step – 4: Run the automation

In Docker

Docker Desktop interface showing the Images section. The sidebar includes options for Containers, Images (selected), Volumes, Builds (NEW), Dev Environments (BETA), Docker Scout, Extensions, and Add Extensions. The main area displays 3 images with the following details:

Name	Tag	Status	Created	Size	Actions
mysql	latest	In use	3 months ago	618.82 MB	[More]
microproject-simple	latest	In use	4 hours ago	176.79 MB	[More]
microproject-myapp	latest	In use	29 seconds ago	1.46 GB	[More]

At the bottom, a "Walkthroughs" section shows "How do I run a container?" and "Run Docker Hub images". The status bar at the bottom indicates "Engine running" with RAM usage of 3.76 GB and CPU usage of 40.86%.

In SonarQube

SonarQube project dashboard for MicroProjectSonarQube. The top navigation bar includes Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, and More. The dashboard shows the following information:

- QUALITY GATE STATUS:** Passed (All conditions passed)
- LAST ANALYSIS:** March 29, 2024 at 6:46 PM. Version not provided.
- OVERVIEW:** Issues, Security Hotspots, Measures, Code, Activity.
- MEASURES:** Overall Code (New Code: Since March 29, 2024, started 49 minutes ago). Metrics include:
 - New Bugs: 0 (Reliability: A)
 - New Vulnerabilities: 0 (Security: A)
 - New Security Hotspots: 0 (Security Review: A)
 - Added Debt: 0 (Maintainability: A)
 - New Code Smells: 0
- Coverage:** Coverage on 0 New Lines to cover (0.0%)
- NOTIFICATIONS:** DellCommandUpdate (2 Updates are ready to install) with Install and Remind Later buttons.