AMERICAN INTERNATIONAL UNIVERSITY- BANGLADESH

Project on

"Advance Database Management System "

Prepared For:

Rezwan Ahmed

Faculty, Dept. of Science and information Technology

American International University

Prepared By:

Islam, Md. Raihanul I.D: 17-34988-2(sec:A)	Rahman, Md. Sajjadur I.D: 17-35034-2(sec:B)
Rahil, Md. Abu Hanif I.D: 17-35074-2(sec:B)	Nasif, Nabil Al I.D: 17-35131-2(sec:B)

Date of Submission: 19.08.2020

Table of Contents

System summary:

Blood is universally recognized as the most precious element that sustains life. It saves innumerable lives across the world in a variety of conditions. We have designed a system where patients can easily collect their needed blood. In our system, there are 2 types of users (registered and non-registered members). This system is monitored by admin. Admin can manage the employees of our system. Admin can block any employees as well as a registered member and also see all the transactions in our system. Both registered and non-registered members can order blood. They have to pay a few costs as a service/maintenance cost. There is a benefit for a registered member. They will get a discount when they order blood. A registered member can order and donate blood both. They can update their information and see their own transaction. In the case of order, there is an auto checking option by date which meant a person can not donate blood twice or more with 90 days and can not donate without registration in our system.

There is some limitation in our system. In this system, we didn't test the donor's blood group. Communication between hospital and blood bank is not established and also service the seekers or patients in area wise.

From this system, seekers can get the information the desired blood group from the central inventory easily. Through this system Information of the donors saved permanently and easy to handle patient's records. The main benefit of this project is that it saves time and patients can get blood easily.

ERD diagram:

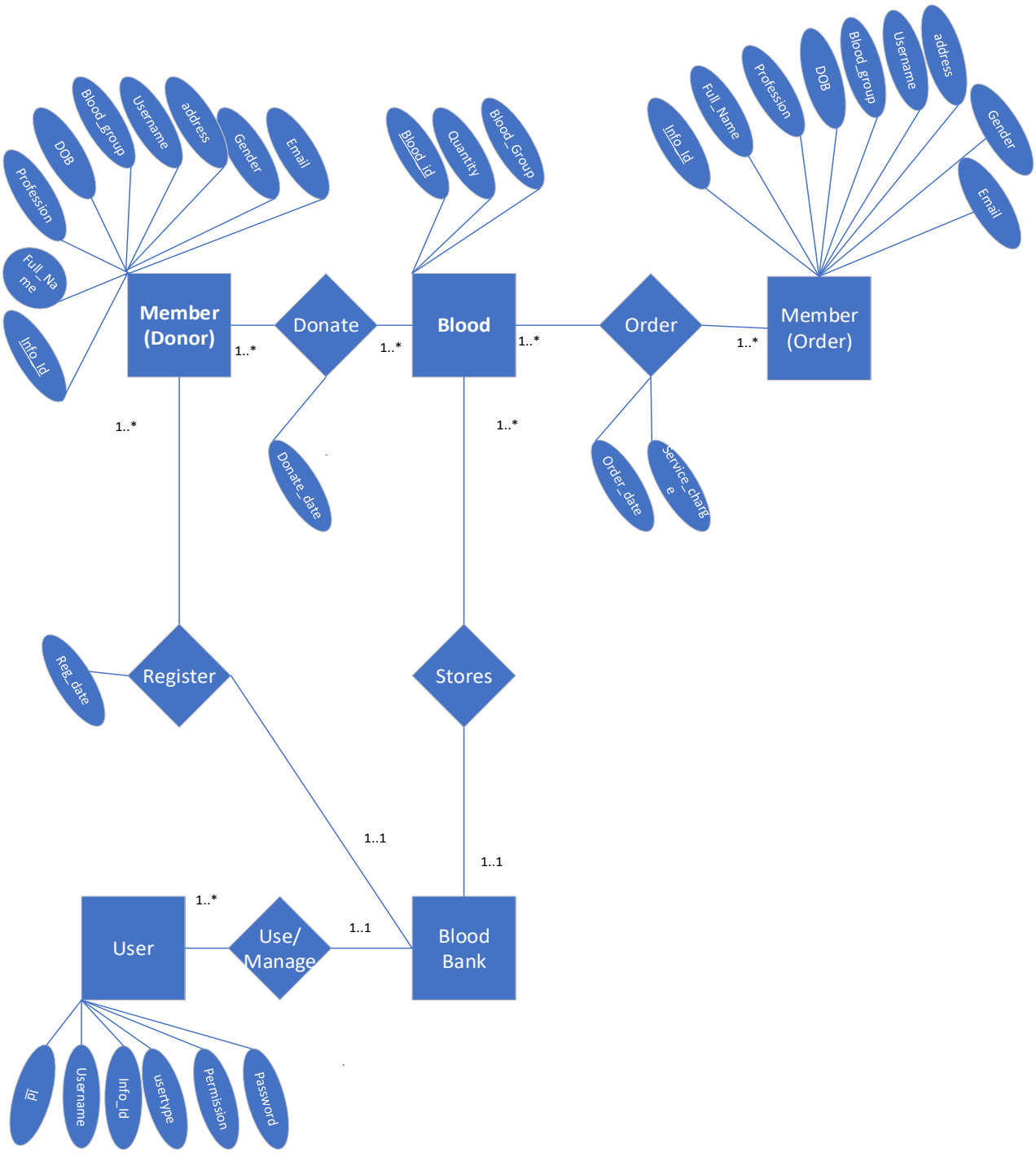


Figure: System ER diagram

Class diagram:

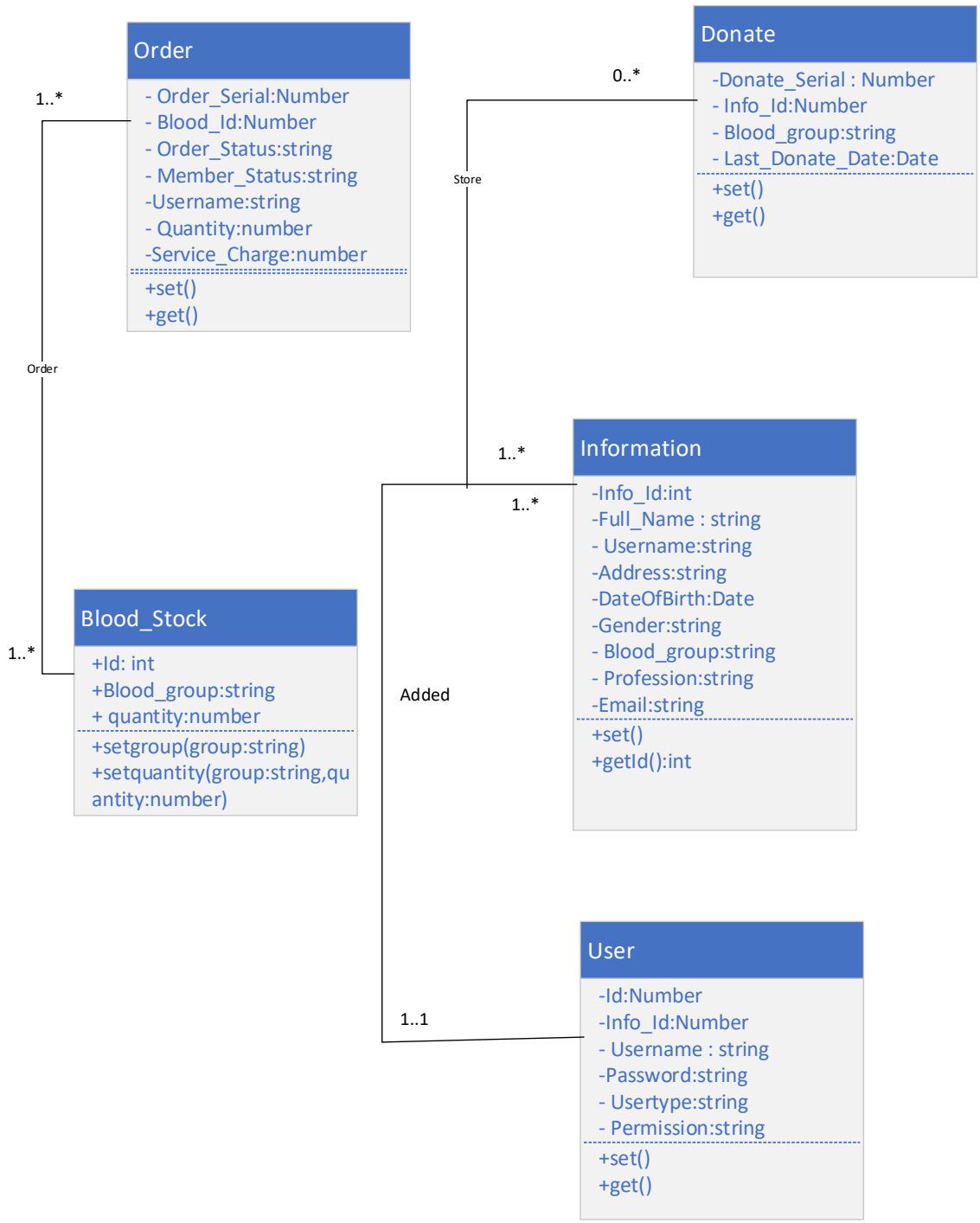


Figure :System class diagram

Use case diagram:

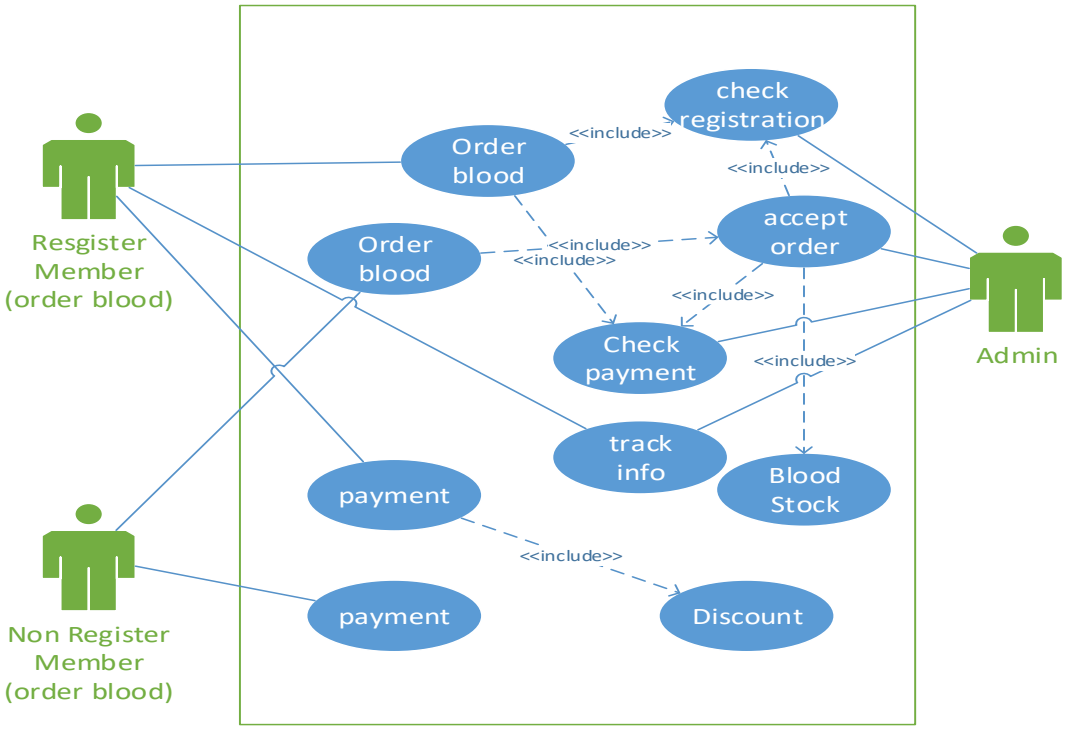


Figure :Blood order use case diagram

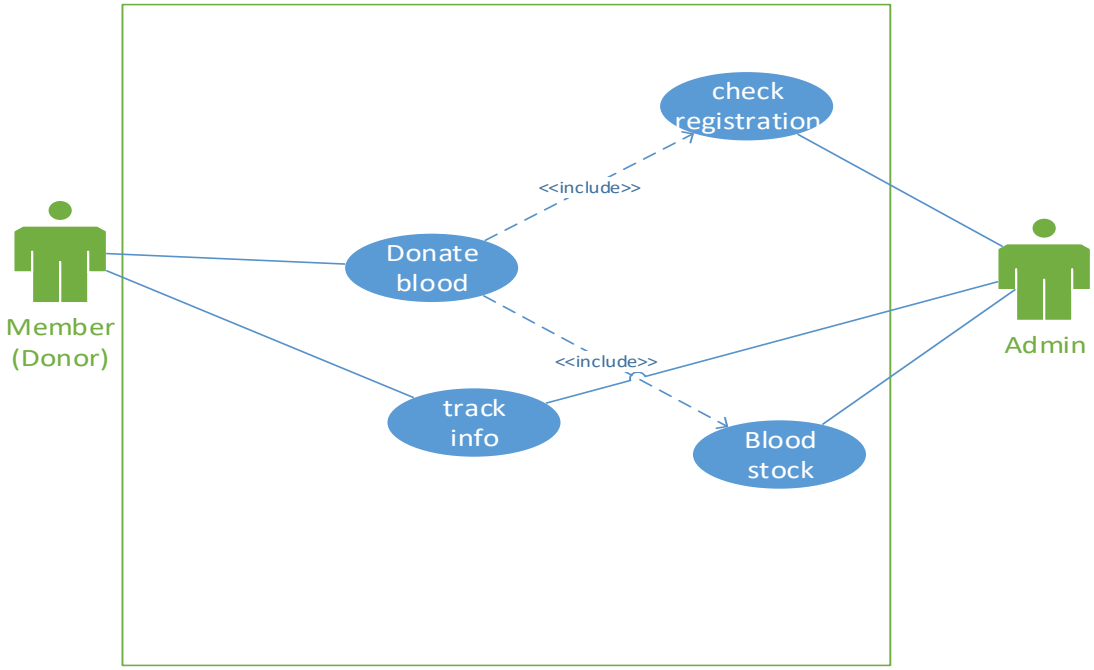


Figure :Blood donate use case diagram

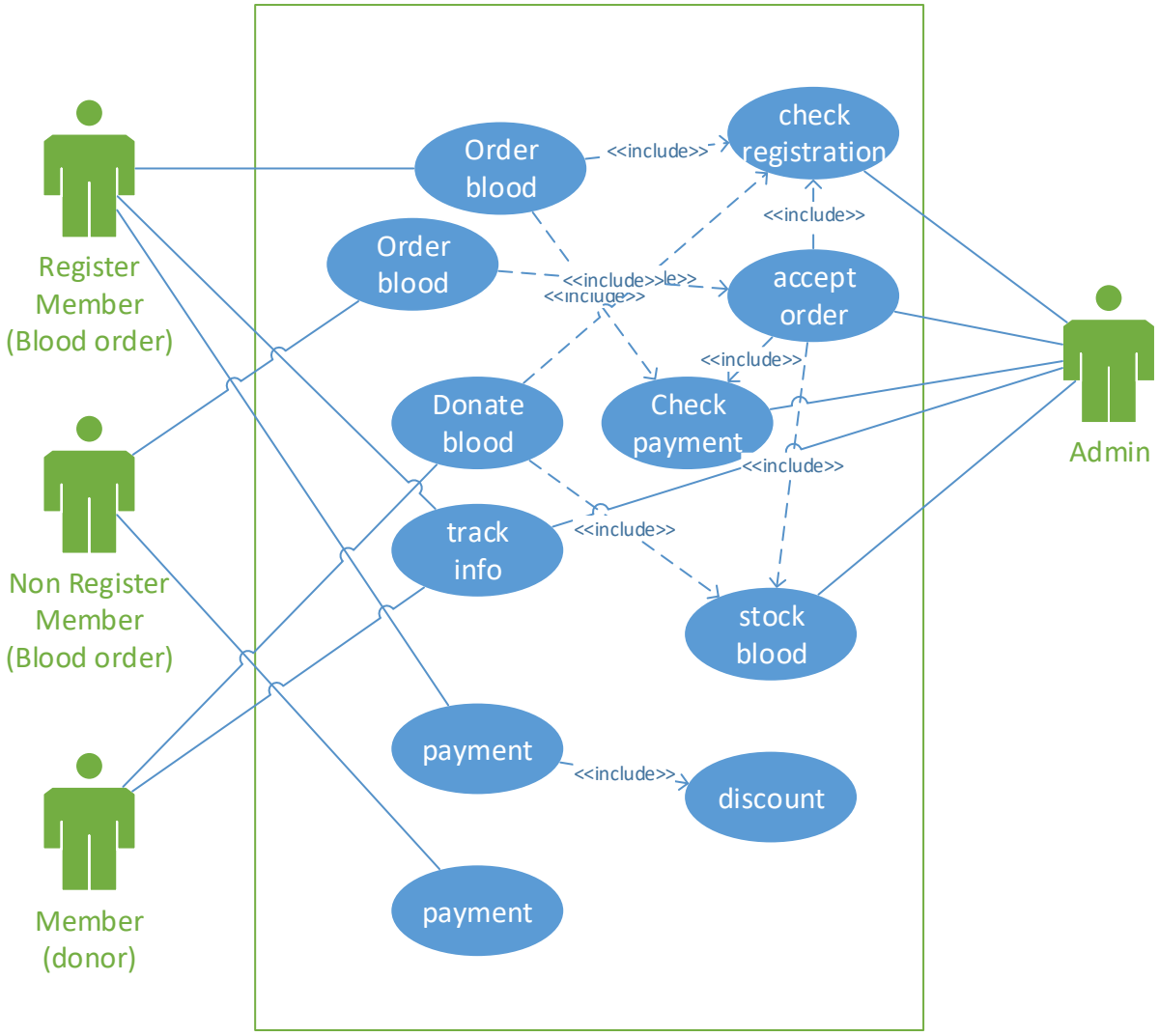


Figure :Combined system use case diagram

Activity diagram:

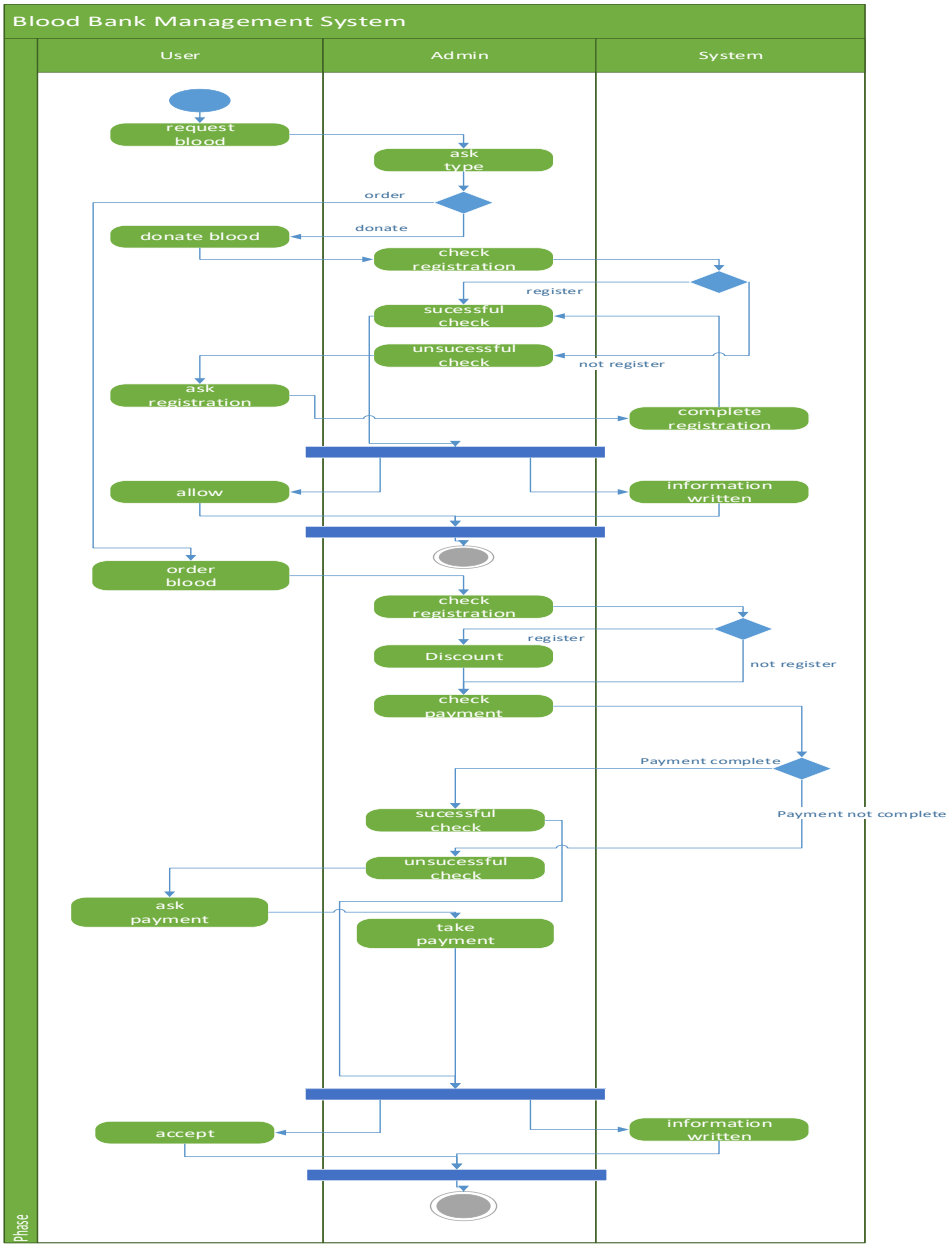


Figure :System activity diagram

Schema diagram:

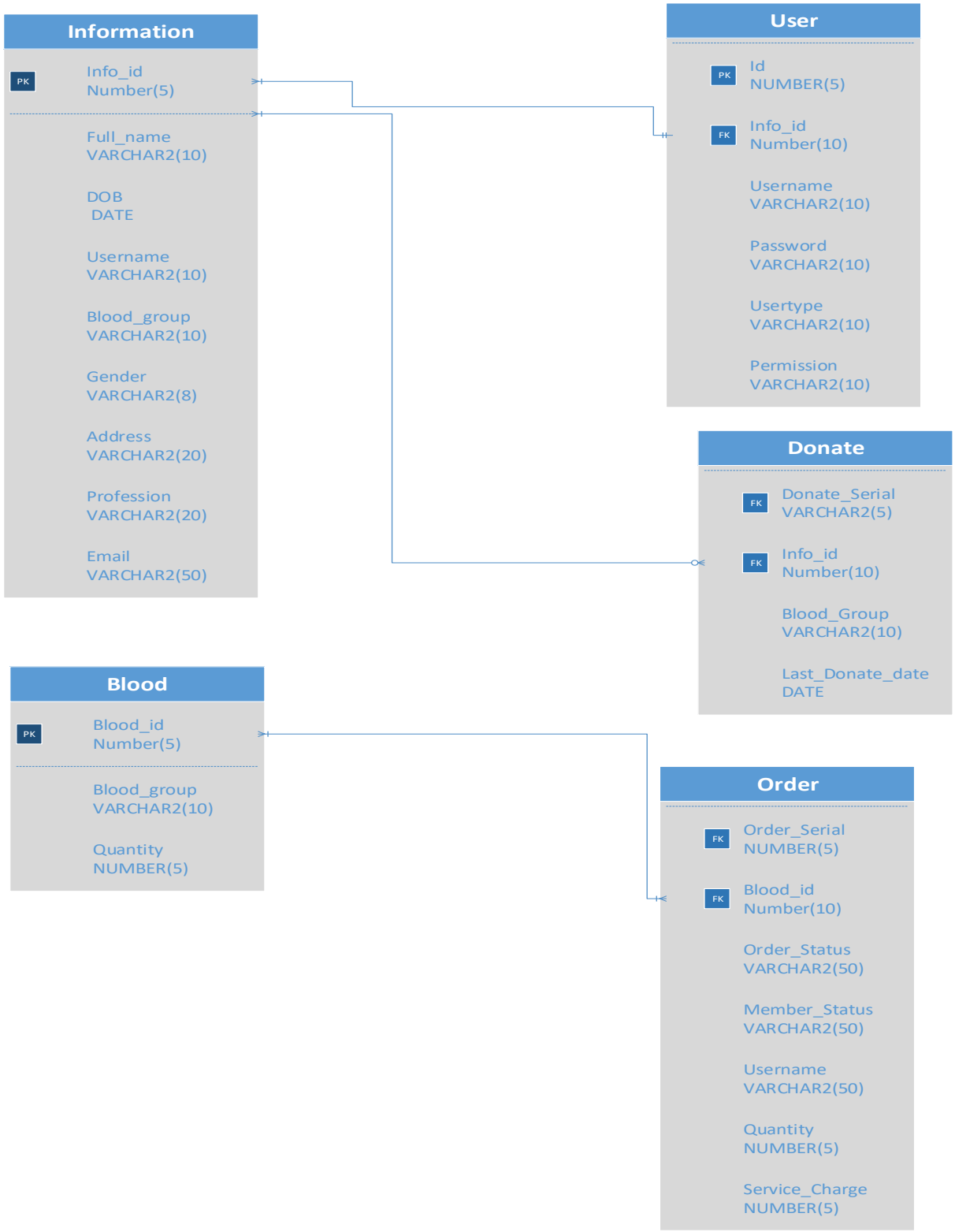


Figure :System schema diagram

Table with description:

Autocommit Rows 10 Save Run

```
desc informations
```

Results Explain **Describe** Saved SQL History

Object Type **TABLE** Object **INFORMATIONS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
INFORMATIONS	INFO_ID	NUMBER	-	5	0	1	-	-	-
	FULL_NAME	VARCHAR2	255	-	-	-	-	-	-
	USERNAME	VARCHAR2	255	-	-	-	-	-	-
	ADDRESS	VARCHAR2	255	-	-	-	-	-	-
	BLOOD_GROUP	VARCHAR2	255	-	-	-	-	-	-
	GENDER	VARCHAR2	255	-	-	-	-	-	-
	DOB	DATE	7	-	-	-	-	-	-
	PROFESSION	VARCHAR2	255	-	-	-	-	-	-
	GMAIL	VARCHAR2	255	-	-	-	-	-	-

1 - 9

Autocommit Rows 10 Save Run

```
desc Users
```

Results Explain **Describe** Saved SQL History

Object Type **TABLE** Object **USERS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
USERS	ID	NUMBER	-	5	0	1	-	-	-
	USERNAME	VARCHAR2	255	-	-	-	-	-	-
	PASSWORD	VARCHAR2	255	-	-	-	-	-	-
	TYPE	VARCHAR2	255	-	-	-	-	-	-
	PERMISSION	VARCHAR2	255	-	-	-	-	-	-
	INFO_ID	NUMBER	-	5	0	-	✓	-	-

1 - 6

Autocommit Rows 10 Save Run

```
desc blood_stocks
```

Results Explain **Describe** Saved SQL History

Object Type **TABLE** Object **BLOOD_STOCKS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
BLOOD_STOCKS	BLOOD_ID	NUMBER	-	5	0	1	-	-	-
	BLOOD_GROUP	VARCHAR2	255	-	-	-	-	-	-
	QUANTITY	NUMBER	-	5	0	-	-	-	-

1 - 3

Autocommit Rows 10 Save Run

```
desc donates
```

Results Explain **Describe** Saved SQL History

Object Type **TABLE** Object **DONATES**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DONATES	SERIAL	NUMBER	-	5	0	1	-	-	-
	INFO_ID	NUMBER	-	5	0	-	✓	-	-
	BLOOD_GROUP	VARCHAR2	255	-	-	-	-	-	-
	LAST_TIME_DONATED	DATE	7	-	-	-	-	-	-

1 - 4

Autocommit Rows 10 Save Run

```
desc orders
```

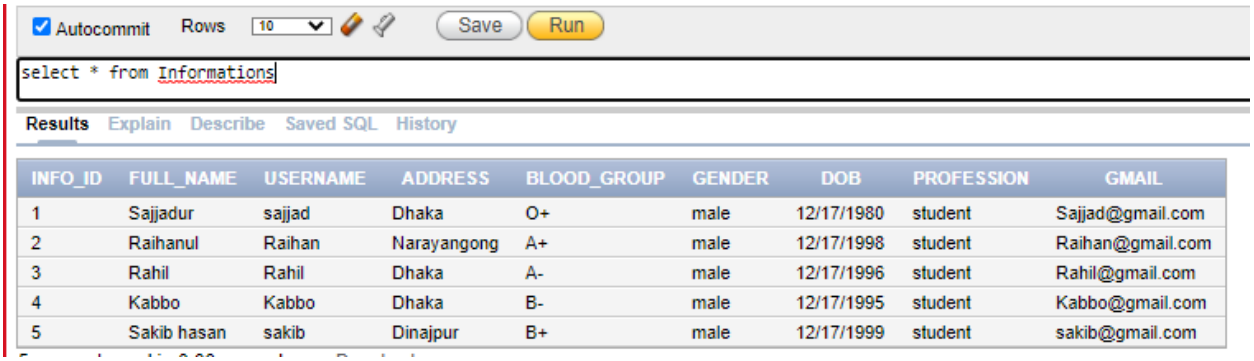
Results Explain **Describe** Saved SQL History

Object Type **TABLE** Object **ORDERS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
ORDERS	ORDER_SERIAL	NUMBER	-	5	0	1	-	-	-
	USERNAME	VARCHAR2	255	-	-	-	-	-	-
	ORDER_STATUS	VARCHAR2	255	-	-	-	-	-	-
	BLOOD_ID	NUMBER	-	5	0	-	✓	-	-
	QUANTITY	NUMBER	-	5	0	-	-	-	-
	MEMBER_STATUS	VARCHAR2	255	-	-	-	-	-	-
	SERVICE_CHARGE	VARCHAR2	255	-	-	-	-	-	-

1 - 7

Table with data:

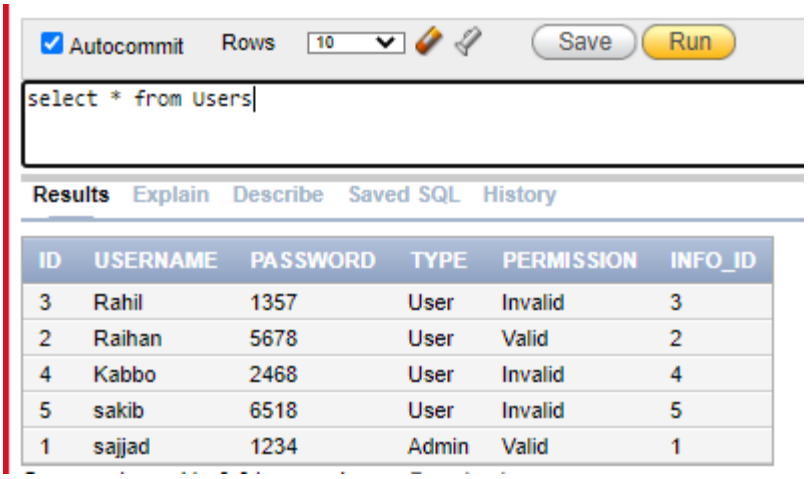


Autocommit Rows 10 Save Run

```
select * from Informations
```

Results Explain Describe Saved SQL History

INFO_ID	FULL_NAME	USERNAME	ADDRESS	BLOOD_GROUP	GENDER	DOB	PROFESSION	GMAIL
1	Sajjadur	sajjad	Dhaka	O+	male	12/17/1980	student	Sajjad@gmail.com
2	Raihanul	Raihan	Narayangong	A+	male	12/17/1998	student	Raihan@gmail.com
3	Rahil	Rahil	Dhaka	A-	male	12/17/1996	student	Rahil@gmail.com
4	Kabbo	Kabbo	Dhaka	B-	male	12/17/1995	student	Kabbo@gmail.com
5	Sakib hasan	sakib	Dinajpur	B+	male	12/17/1999	student	sakib@gmail.com



Autocommit Rows 10 Save Run

```
select * from Users
```

Results Explain Describe Saved SQL History

ID	USERNAME	PASSWORD	TYPE	PERMISSION	INFO_ID
3	Rahil	1357	User	Invalid	3
2	Raihan	5678	User	Valid	2
4	Kabbo	2468	User	Invalid	4
5	sakib	6518	User	Invalid	5
1	sajjad	1234	Admin	Valid	1

Autocommit Rows 10 Save Run

```
select * from blood_stocks
```

Results Explain Describe Saved SQL History

BLOOD_ID	BLOOD_GROUP	QUANTITY
1	O+	50
2	O-	45
3	A+	40
4	A-	30
5	B-	10

5 rows returned in 0.05 seconds Download

Autocommit Rows 10 Save Run

```
select * from donates
```

Results Explain Describe Saved SQL History

SERIAL	INFO_ID	BLOOD_GROUP	LAST_TIME_DONATED
6	1	O+	12/17/2019
7	2	O-	12/17/2019
8	3	A+	12/17/2019
9	4	A-	12/17/2019
10	5	B-	12/17/2019

5 rows returned in 0.05 seconds Download

Autocommit Rows 10 Save Run

```
select * from orders
```

Results Explain Describe Saved SQL History

ORDER_SERIAL	USERNAME	ORDER_STATUS	BLOOD_ID	QUANTITY	MEMBER_STATUS	SERVICE_CHARGE
1	sajjad	Accepted	1	2	Registered	30Tk
2	Raihan	Accepted	2	3	Registered	45Tk
3	Unknown	Rejected	3	2	Not Registered	30Tk
4	Unknown	Rejected	3	5	Not Registered	30Tk
5	Unknown	Accepted	1	2	Not Registered	50Tk

5 rows returned in 0.05 seconds Download

Test query:

1.Question: Find out the users who have the minimum quantity blood group in our stock.

=> Select * from Informations where Blood_Group in(select Blood_Group from Blood_Stocks where Quantity=(select min(quantity) from Blood_Stocks))

INFO_ID	FULL_NAME	USERNAME	ADDRESS	BLOOD_GROUP	GENDER	DOB	PROFESSION	GMAIL
4	Kabbo	Kabbo	Dhaka	B-	male	12/17/1995	student	Kabbo@gmail.com

1 rows returned in 0.23 seconds Download

2.Question: Find out the profession having a minimum of 2 users in our system.

=> select profession,count(profession) as "Total Users" from Informations group by profession having count(profession)>2

PROFESSION	Total Users
student	5

1 rows returned in 0.08 seconds

3.Question: find the name & location of most senior donor.

=> select Full_name,address from informations,donates where informations.info_id=donates.info_id and last_time_donated=(select min(last_time_donated) from informations,donates where informations.info_id=donates.info_id)

FULL_NAME	ADDRESS
Raihanul	Narayangong
Sajjadur	Fulbari
Rahil	Dhaka
Kabbo	Dhaka
Sakib hasan	Dinajpur

5 rows returned in 0.19 seconds

4.Question: Find out the blood group with quantity of 3rd highest quantity.

=> select blood_group,quantity from blood_stocks where quantity=(SELECT MAX(quantity) FROM blood_stocks WHERE quantity < (SELECT MAX(quantity) FROM blood_stocks where quantity<(SELECT MAX(quantity) FROM blood_stocks)));

BLOOD_GROUP	QUANTITY
A+	40

1 rows returned in 0.02 seconds

5.Question: Find out the Location where stay at least 2 users.

=> select address, count(*) from informations,users where informations.info_id=users.info_id and type='User' group by address having count(*)>=2

ADDRESS	COUNT(*)
Dhaka	2

1 rows returned in 0.16 seconds

6. Question: Taking user id from the user and display type of that user using the procedure and handle any exception.

create procedure Id_Tpe(uid in Users.Info_Id%type,tp out Users.Type%type)

is

begin

select type into tp from Users where Info_Id=uid;

end;

```
declare
uid Users.Info_Id%type:=:Enter_your_id;
tpe Users.Type%type;

invalid_id exception;
begin



if uid<0 then

raise invalid_id;

else
Id_Tpe(uid,tpe);
dbms_output.put_line(tpe);

end if;

exception

when invalid_id then
dbms_output.put_line('Id can not be negetive');
when no_data_found then
dbms_output.put_line('No user exist!');
when others then
```

```
dbms_output.put_line('Error occured!');



end;
```


```
Admin

Statement processed.

0.51 seconds
```

7.Question: Show name, Blood group, mail of an invalid user Using plsql.

```
declare



p Users.Permission%type;


i number(4);
Cursor c
Is
select * from Informations;


begin
for i in c loop
select Permission into p from Users where Info_id=i.Info_id;
```

```
if p='Invalid' then

dbms_output.put_line('Full Name: '||i.full_name||' Blood group: '||i.Blood_Group||' Gmail: '||i.Gmail);


end if;


end loop;


end;
```

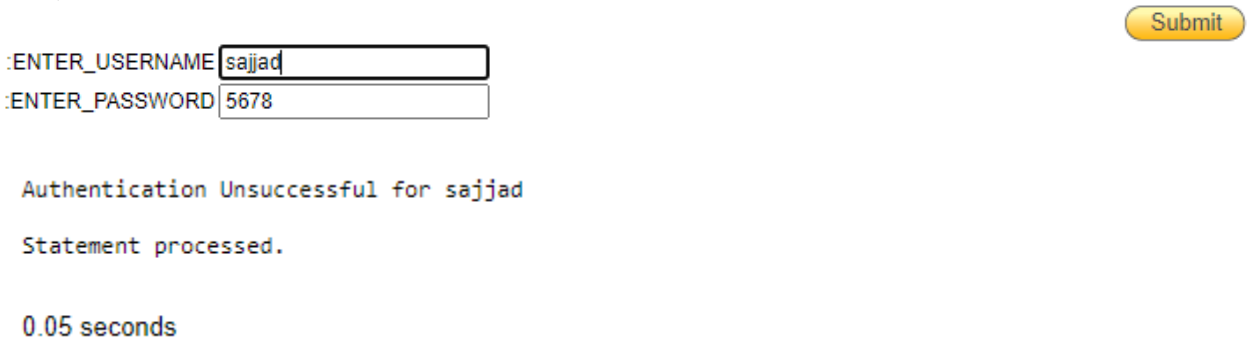
```
Full Name: Rahil Blood group: A- Gmail: Rahil@gmail.com
Full Name: Kabbo Blood group: B- Gmail: Kabbo@gmail.com
Full Name: Sakib hasan Blood group: B+ Gmail: sakib@gmail.com

Statement processed.


0.43 seconds
```

8.Question: Check User Athentication for any user by taking Username and password from the user.

```
declare


uname Users.Username%type:=:Enter_Username;

upass Users.Password%type:=:Enter_Password;


i number(4);


flg number(4);
```

```
Cursor c Is select USERNAME,PASSWORD from USERS where USERNAME=uname;




begin


i:=0;
flg:=0;



for i in c loop
flg:=1;

if i.PASSWORD=upass then
dbms_output.put_line('Authentication Successful for '||i.USERNAME);
else
dbms_output.put_line('Authentication Unsuccessful for '||i.USERNAME);
end if;

end loop;
if flg=0 then
dbms_output.put_line('User '||uname||' does not exist');
end if;
```

end;



9.Question: Display the rejected order of all blood groups with the quantity.

declare

bldgrp Blood_Stocks.Blood_Group%type;

i number(4);

Cursor c

Is

select * from Orders;

begin

for i in c loop

select Blood_Group into bldgrp from Blood_Stocks where Blood_Id=i.Blood_Id;

if i.Order_status='Rejected' then

dbms_output.put_line('Blood Group :'||bldgrp||' Quantity :'||i.Quantity);

end if;

```
end loop;


end;
```

```
Blood Group :A+ Quantity :2
Blood Group :A+ Quantity :5

Statement processed.

0.36 seconds
```

10.Question: Show all the orders of non registered member.

```
declare


bldgrp Blood_Stocks.Blood_Group%type;

i number(4);

Cursor c

Is

select * from Orders;




begin


for i in c loop

select Blood_Group into bldgrp from Blood_Stocks where Blood_Id=i.Blood_Id;

if i.Member_status='Not Registered' then

dbms_output.put_line('Blood Group :'||bldgrp||' Quantity :'||i.Quantity||' Order Status :'||i.Order_Status);

end if;
```

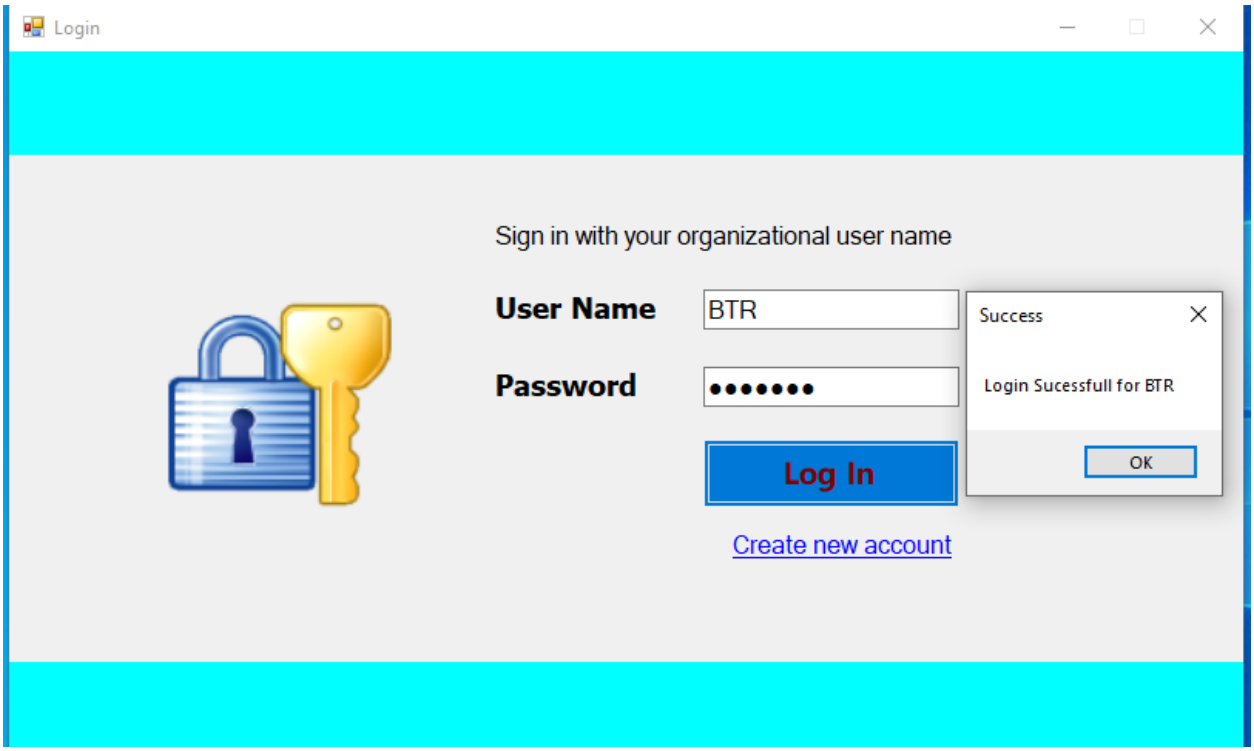
end loop;

end;

```
Blood Group :A+ Quantity :2 Order Status :Rejected
Blood Group :A+ Quantity :5 Order Status :Rejected
Blood Group :O+ Quantity :2 Order Status :Accepted

Statement processed.

0.03 seconds
```

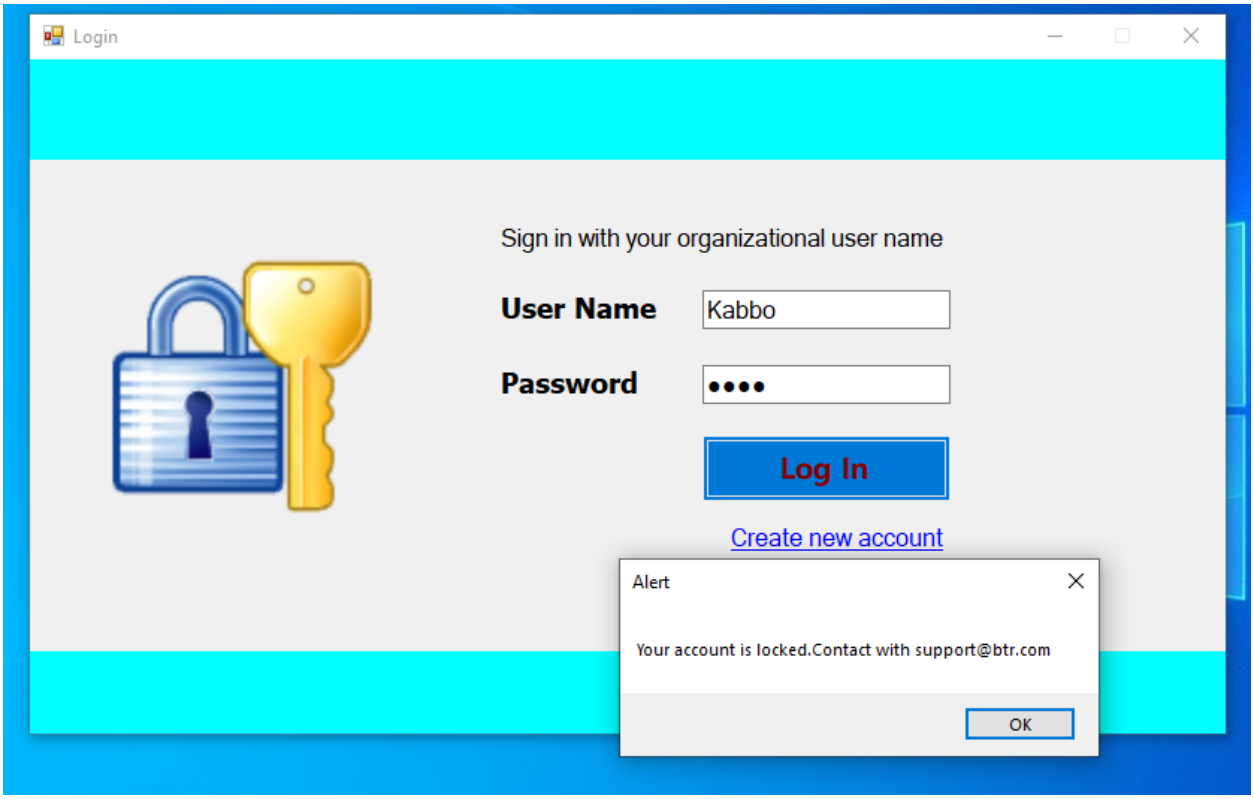
Login UI (successful login with database):

Successful login-



Unsuccessful login-

Login for invalid permission:



-End-