



# VIT<sup>®</sup>

## Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

**School of Computer Science and Engineering**

# **ORGAN TRANSPLANT MANAGEMENT SYSTEM**

**CSE2002 – Database Management System - Embedded Project**

**Slot : D1**

**Lab:57,58**

**Rupin Patel : 19BCB0030**

**Monil Modi : 19BCE0463**

**Faculty:- Dr. Pradeep Kumar Roy**

## **Introduction :**

The Organ Donation and Procurement Network Management System is a database management system that uses database technology to construct, maintain and manipulate various kinds of data about a person's donation or procurement of a particular organ.

It is a medical procedure in which an organ is removed from one body and placed in the body of a recipient, to replace a damaged or missing organ. The donor and recipient may be at the same location, or organs may be transported from a donor site to another location.

It maintains a comprehensive medical history and other critical information of every person in the database design.

It maintains a database containing statistical information regarding network of organ donation and procurement of different countries.

Organ Donation and Procurement Organizations play a pivotal role in today's medical institutions. Such organizations are responsible for the evaluation and procurement of organs for organ transplantation. These organizations represent the front-line of organ procurement, having direct contact with the hospital and the family of a recently deceased donor. The work of such organizations includes to identify the best candidates for the available organs and to coordinate with the medical institutions to decide on each organ recipient. They are also responsible for educating the public to increase the awareness of and participation in the organ donation process. Also, it keeps track of all transplantation operations carried till date.

The Organ Donation and Procurement Network Management System is a database management system that uses database technology to construct, maintain and manipulate various kinds of data about a person's donation or procurement of a particular organ.

In short, it maintains a database containing statistical information regarding network of organ donation and procurement of different countries.

## **Problem Statement :**

The main problem in the current scenario is that due to prevailing malpractices, the transplantation and donation of organs are not executed in systematic way.

Organ Wastage is a major issue that can only be solved by having a proper database of all Patient and Donors in a well-formed way, that can be processed easily.

- To deploy a website to keep a track of each organ donated by donor and the organ transplanted into the patient. Thus, creating a transparent and user friendly website for Organ transplantation Management system.
- To reduce the prevailing malpractices.
- To reduce the wastage of organs due to lack of proper database management system.
- To create live statistics of donors (graph between successful and failed transplantation)

```

    erDiagram
        PATIENT ||--o{ DOCTOR : ATTENDED
        DOCTOR ||--o{ ORGANIZATION : WORKS IN
        USER ||--o{ DONOR : DONATES
        DONOR ||--o{ ORGAN : ORGAN DONATED
        ORGANIZATION ||--o{ ORGANIZATION_HEAD : HEADS

        PATIENT {
            string PAT ID PK
            string RELATION
            string STATUS
            string DATE OF TRANSACTION
        }
        DOCTOR {
            string DOCTOR ID PK
            string SPECIALIZATION
            string ORGANIZATION
        }
        USER {
            string USER ID PK
            string DATE OF DONATION
        }
        DONOR {
            string DONOR ID PK
            string ORGAN DONATED
        }
        ORGANIZATION {
            string ORGANIZATION ID PK
            string ORGANIZATION HEAD
        }
        TRANSACTION {
            string TRANSACTION ID PK
            string ORGAN
            string DATE OF TRANSACTION
        }
        ORGANIZATION_HEAD {
            string ORGANIZATION HEAD PK
        }
    
```

## **Entity Sets:**

### **\*1. User**

1. User ID
2. Name
3. Date of birth
4. Phone Number (multi-valued)
5. Medical Insurance
6. Medical History
7. Address

### **\*2. Patient**

1. Patient\_ID
2. Organ Required
3. Reason of procurement
4. User\_ID( foreign key)

### **\*3. Donor**

1. Donor\_ID
2. Organ Donated
3. Reason of donation
4. User\_ID (foreign key)
5. Donor\_name

### **\*4. Organ Available**

1. Organ\_ID
2. Organ Name
3. Donor\_ID (foreign key)

### **\*5. Organization**

1. Organization ID
2. Organization Name
3. Location
4. Government approved organization or not
5. Phone Number (multi-valued)

### **\*6. Doctor**

1. Doctor ID
2. Doctor Name
3. Phone Number (multi-valued)

### **\*7. Organization Head**

1. Head Name
2. Date of Joining
3. Term Length

## Software Used :

- HTML
- MYSQL
- CSS
- Python
- Flask

## Functional Dependencies :

### Tables and their Functional Dependencies :-

**1) User**(User\_ID, Name, Date\_of\_birth, Medical\_Insurance, Medical\_History, Street, City, State)

**FD**= {User\_ID → Name, Date\_of\_birth, Medical Insurance, Medical History, Street, City, State}

**2) User\_phone\_no**(User\_ID, phone\_no)

**FD**= {User\_ID → phone\_no}

{User\_ID} is foreign key constraint

**3) Patient**(Patient\_ID, organ\_req, reason\_of\_procurement, Doctor\_ID, User\_ID)

**FD**= {Patient\_ID, organ\_req → reason\_of\_procurement, Doctor\_ID, User\_ID}

{User\_ID, Doctor\_ID} are foreign key constraints

**4) Donor**(Donor\_ID, organ\_donated, reason\_of\_donation, Organization\_ID, User\_ID)

**FD**= {Donor\_ID, organ\_donated → reason\_of\_donation, Organization\_ID, User\_ID}

{User\_ID, Organization\_ID} are foreign key constraints

**5) Organ Available**(Organ\_ID, Organ\_name, Donor\_ID)

**FD**= {Organ\_ID → Organ\_name, Donor\_ID}

{Donor\_ID} is foreign key constraint

**6) Transaction**(Patient\_ID, Organ\_ID, Donor\_ID, Date\_of\_transaction, Status)

**FD**= {Patient\_ID, Organ\_ID → Donor\_ID, Date\_of\_transaction, Status}

{Patient\_ID, Donor\_ID} are foreign key constraints

**7) Organization**(Organization\_ID, Organization\_name, Location, Government\_approved)

**FD**= {Organization\_ID → Organization\_name, Location, Government\_approved}

**8) Organization\_phone\_no**(Organization\_ID, phone\_no)

**FD**={Organization\_ID ->phone\_no}

{Organization\_ID} are foreign key constraints

**9) Doctor**(Doctor\_ID, Doctor\_name, Department\_name, Organization\_id)

**FD**={Doctor\_ID ->Doctor\_name, Organization\_id}

{Organization\_ID} is foreign key constraint

**10) Doctor\_phone\_no**(Doctor\_ID, phone\_no)

**FD**={Doctor\_ID ->phone\_no}

{Doctor\_ID} is foreign key constraint

**11) Organization\_head**(Organization\_ID, Employee\_ID, Name, Date\_of\_joining, Term\_length)

**FD**={Organization\_ID, Employee\_ID -> Name, Date\_of\_joining, Term\_length}

## Triggers :

delimiter //

create trigger ADD\_DONOR\_LOG

after insert

on Donor

for each row

begin

insert into log values

(now(), concat("Inserted new Donor", cast(new.Donor\_Id as char)));

end //

create trigger UPD\_DONOR\_LOG

after update

```
on Donor
for each row
begin
insert into log values
(now(), concat("Updated Donor Details", cast(new.Donor_Id as char)));
end //
```

```
delimiter //
create trigger DEL_DONOR_LOG
after delete
on Donor
for each row
begin
insert into log values
(now(), concat("Deleted Donor ", cast(old.Donor_Id as char)));
end //
```

```
create trigger ADD_PATIENT_LOG
after insert
on Patient
for each row
begin
insert into log values
(now(), concat("Inserted new Patient ", cast(new.Patient_Id as char)));
end //
```

```
create trigger UPD_PATIENT_LOG
```

```
after update
```

```
on Patient
```

```
for each row
```

```
begin
```

```
insert into log values
```

```
(now(), concat("Updated Patient Details ", cast(new.Patient_Id as char)));
```

```
end //
```

```
create trigger DEL_PATIENT_LOG
```

```
after delete
```

```
on Donor
```

```
for each row
```

```
begin
```

```
insert into log values
```

```
(now(), concat("Deleted Patient ", cast(old.Donor_Id as char)));
```

```
end //
```

```
create trigger ADD_TRANSACTION_LOG
```

```
after insert
```

```
on Transaction
```

```
for each row
```

```
begin
```

```
insert into log values
```



```
(now(), concat("Added Transaction :: Patient ID : ", cast(new.Patient_ID as char), "; Donor ID :  
", cast(new.Donor_ID as char)));  
  
end //
```

### **Algorithm and some important queries:**

```
from flask import Flask,render_template,session,request,redirect,url_for,flash  
  
importmysql.connector,hashlib  
  
importmatplotlib.pyplot as plt  
  
importnumpy as np  
  
importpyrebase
```

<simple rule of flask is that one app route is for returning page and other app route is for mechanism and backend work>

->mysql connector syntax

->firebase config

=>("", methods = ['POST', 'GET'])

=>("home".

Identification of session

=>("login", methods = ['GET', 'POST'])

Login for admin

=>("login\_public")

Returning Login for public portal

=>("signup")

Returning Signup page

=>("welcome")

Returning home page for public

=>("result".)

Program for Login mechanism

=>("register".)

Program for Signup mechanism

=>("show\_update\_detail".)

Program for show update detail searching and displaying table

=>("search\_detail".)

Returning searching html pages

#-----Adding Information-----

=>("add\_<id>\_page".)

Returning display page for displaying user ,patient and donors

=>("add\_User"s)

Program for Insertion of user in User table

=>("add\_User\_phone\_no")

Program for Insertion of user phone number in user table

=>("add\_Patient")

Program for Insertion of patient details in patient table

=>("add\_Donor")

Program for Insertion of donor's details in donor table

=>("add\_Doctor")

Program for insertion of Doctor's details in doctor table

=>("add\_Doctor\_phone\_no")

Program for insertion of doctor's phone number in doctor table

=>("add\_Organ\_available")

Program for insertions of available organ

=>("add\_Organization")

Program for insertion of Organizations details in organization table

=>("add\_Organization\_phone\_no")

Program for insertion of organization-phone numbers details

=>("add\_Organization\_head")

Program for insertion of organization-head details

=>("add\_Transaction")

Program for insertion of transaction details of donor and patient

#-----Update details-----  
-----

#-----

=>("update\_user\_page".

Program for returning updated users details page

=>("update\_user\_details")

Program code for updating user details and displaying it with all donors and

Patient guided under that organization user .

=>("update\_patient\_page".

Program code for displaying updated the patient details

=>("update\_patient\_details")

Program code for updating the patient details

=>("update\_donor\_page")

Program code for displaying updated donors details

=>("update\_donor\_details")

Program code for updating donors details

=>("update\_doctor\_page".

Program code for displaying updated doctor details

=>("update\_doctor\_details")

Program code for updating doctors details

=>("update\_organization\_page")

Program code for displaying updated details of organizations

=>("update\_organization\_details")

Program code for updating organization details

#-----Searching Information-----

=>("search\_User\_details")

Program code for searching and displaying user details

=>("search\_Patient\_details")

Program code for searching and displaying patient details

=>("search\_Donor\_details")

Program code for searching and displaying donor details

=>("search\_Organ\_details")

Program code for searching and displaying organ details of failed  
transaction details

=>("search\_Organization\_details")

Program code for searching and displaying organization details

=>("search\_Organization\_head\_details")

Program code for searching and displaying organization head details

=>("search\_Doctor\_details")

Program code for searching and displaying patient details

=>("search\_Transaction")

Program code for searching and displaying transaction details of patient and donor.

```
=>("search_log")
```

Program code for searching and displaying recent modification in tables

```
#-----Remove Pages-----
```

```
=>('remove_user')
```

```
=>('remove_patient')
```

```
=>('remove_donor')
```

```
=>('remove_doctor')
```

```
=>('remove_organization')
```

```
=>('remove_organization_head')
```

```
#-----Actual Deletion from database-----
```

```
=>('del_user')
```

```
qry = "delete from User where User_ID="+str(request.form['User_ID'])
```

```
=>('del_patient')
```

```
qry = "delete from Patient where Patient_ID="+str(request.form['Patient_ID'])+" and  
organ_req=\'%s\'"%(request.form['organ_req'])
```

```
=>('del_donor')
```

```
qry = "delete from Donor where Donor_ID="+str(request.form['Donor_ID'])+" and  
organ_donated=\'%s\'" %request.form['organ_donated']
```

```
=>('del_doctor')
```

```
qry = "delete from Doctor where Doctor_ID="+str(request.form['Doctor_ID'])
```

```
=>('del_organization')
```

```
qry = "delete from Organization where Organization_ID="+str(request.form['Organization_ID'])
```

```
=>('del_organization_head')
```

```
qry = "delete from Organization_head where  
Organization_ID="+str(request.form['Organization_ID'])+" and  
Employee_ID="+str(request.form['Employee_ID'])
```

```
=>('see_messages',methods=['GET','POST'])
```

```
qry = "Select * from Messages"
```

```
=>('seen_message')
```

```
qry = "delete from Messages where message_id=\'"+msg_id+\'\""
```

```
=>('statistics')
```

```
qry = "select organ_donated, count(Donor_ID) from Donor group by organ_donated"
```

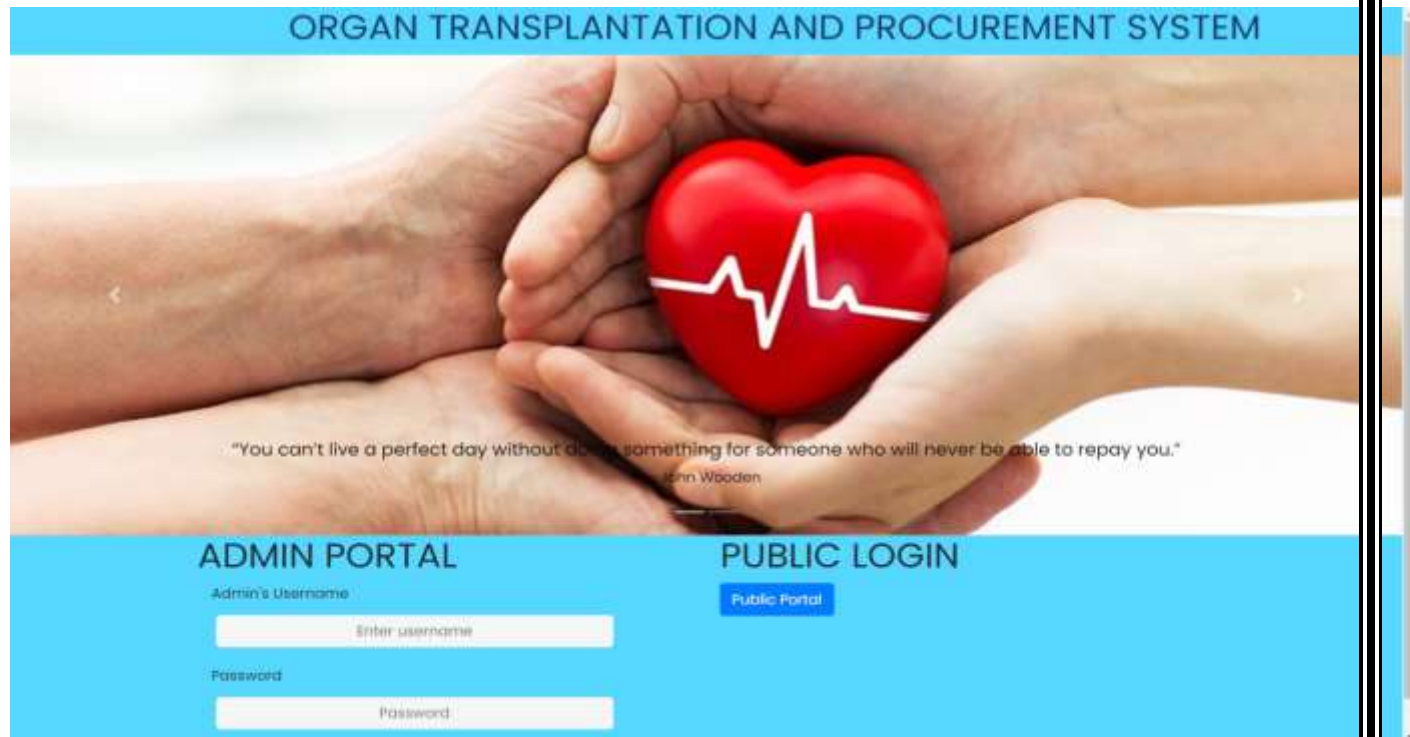
```
qry = "select organ_req, count(Patient_Id) from Patient group by organ_req"
```

```
qry = "select distinct Organ_donated from Transaction inner join Donor on  
Transaction.Donor_ID = Donor.Donor_ID"
```

```
qry = "select count(*) from Transaction inner join Donor on Donor.Donor_ID =  
Transaction.Donor_ID where Organ_donated = '%s' and Status = 1" %organ
```

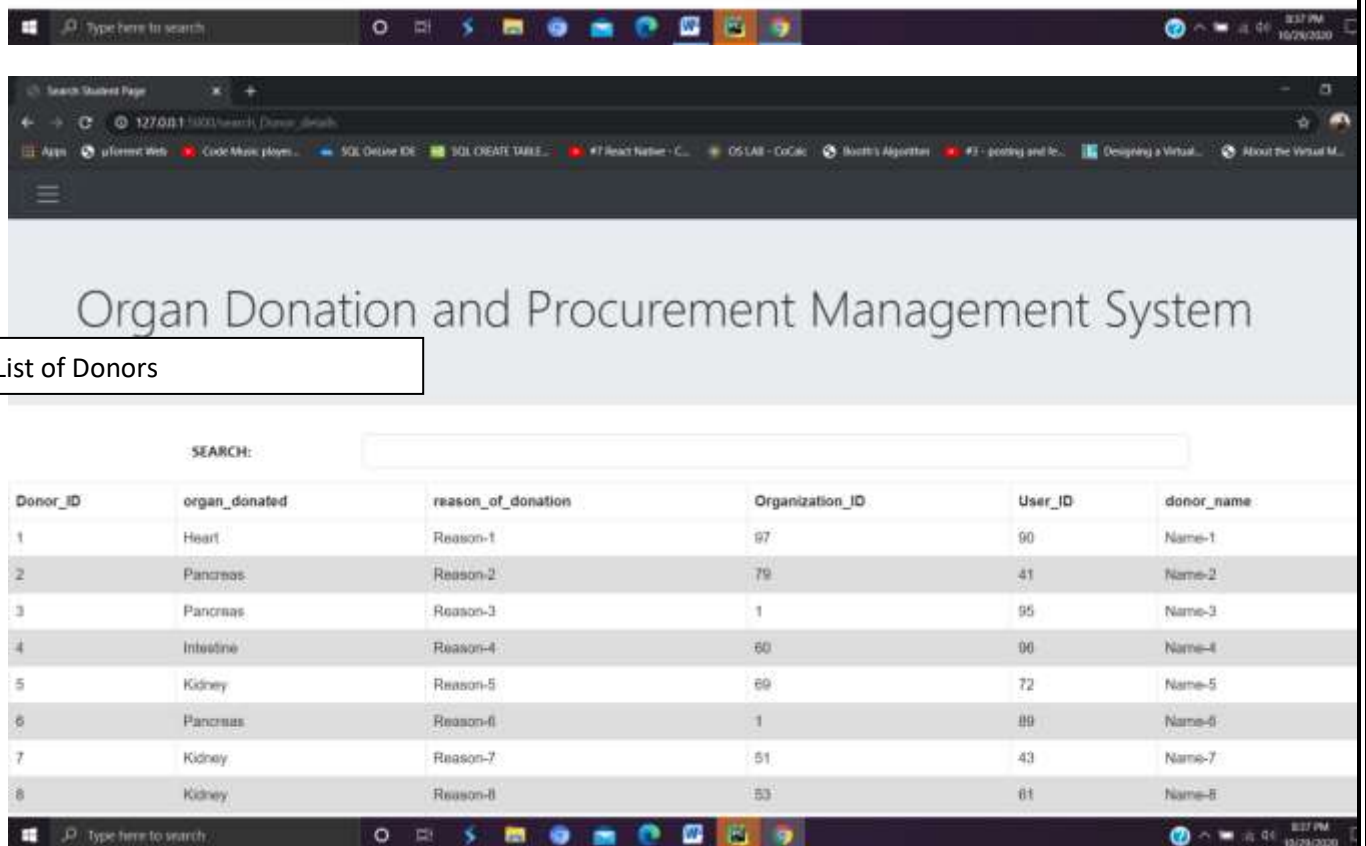
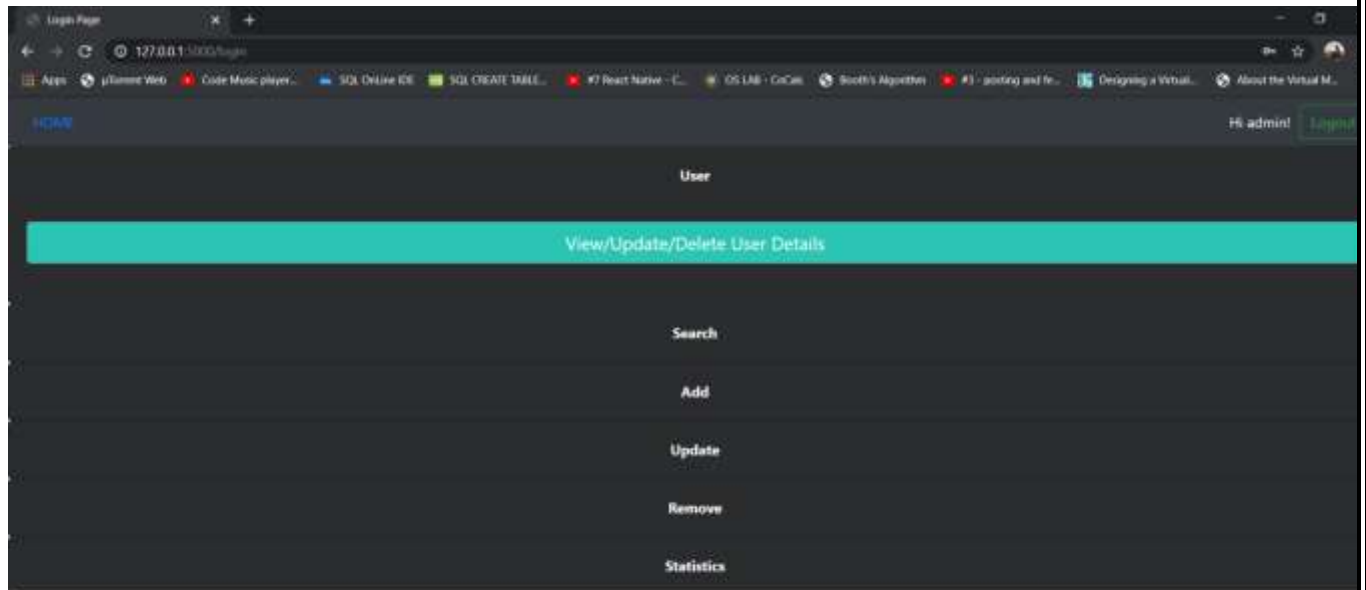
```
qry = "select count(*) from Transaction inner join Donor on Donor.Donor_ID =  
Transaction.Donor_ID where Organ_donated = '%s' and Status = 0" %organ
```

## Screenshots :





## Admin Portal:-



Search Student Page

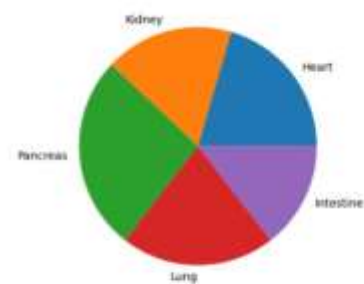
127.0.0.1:5000/search/Organ\_details

## Organ Donation and Procurement Management System

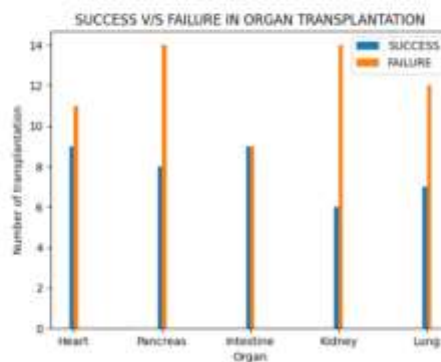
Details of donors and Patient where transplantation get failed

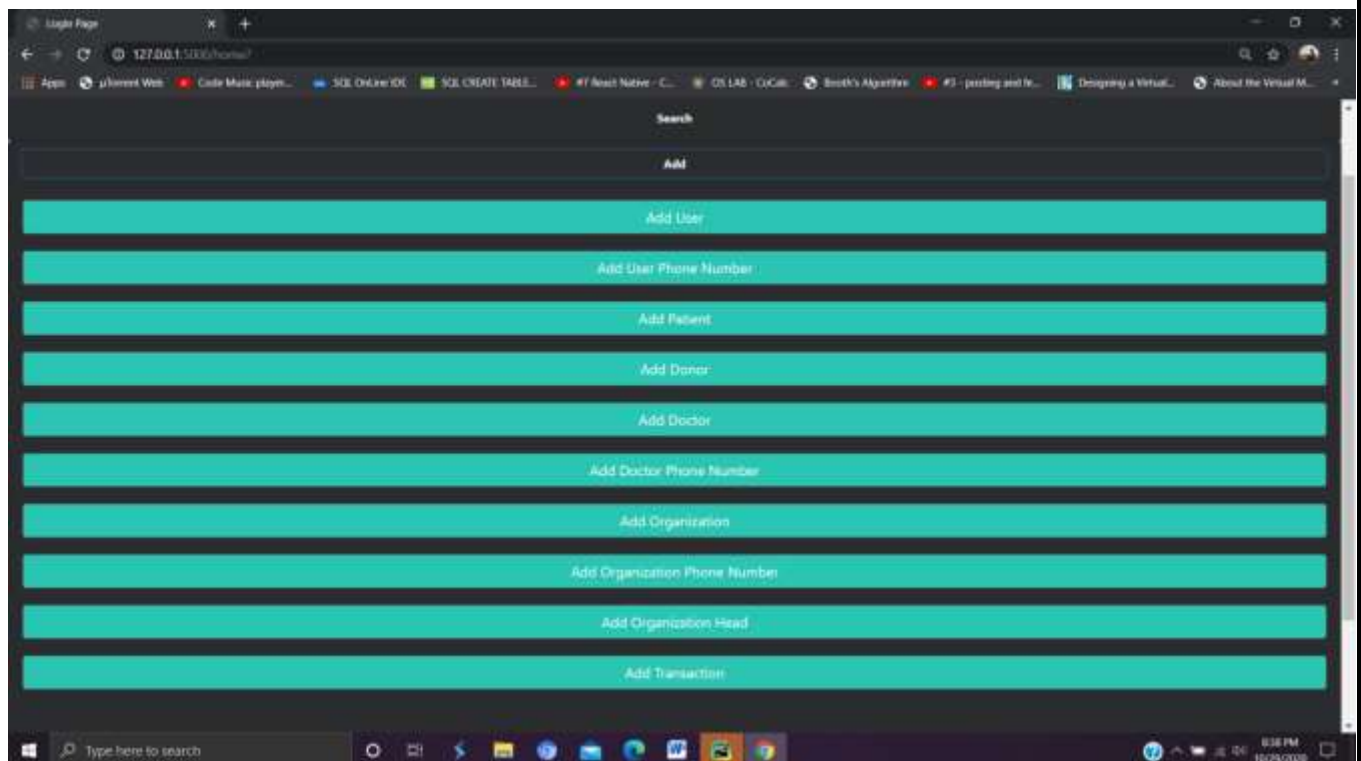
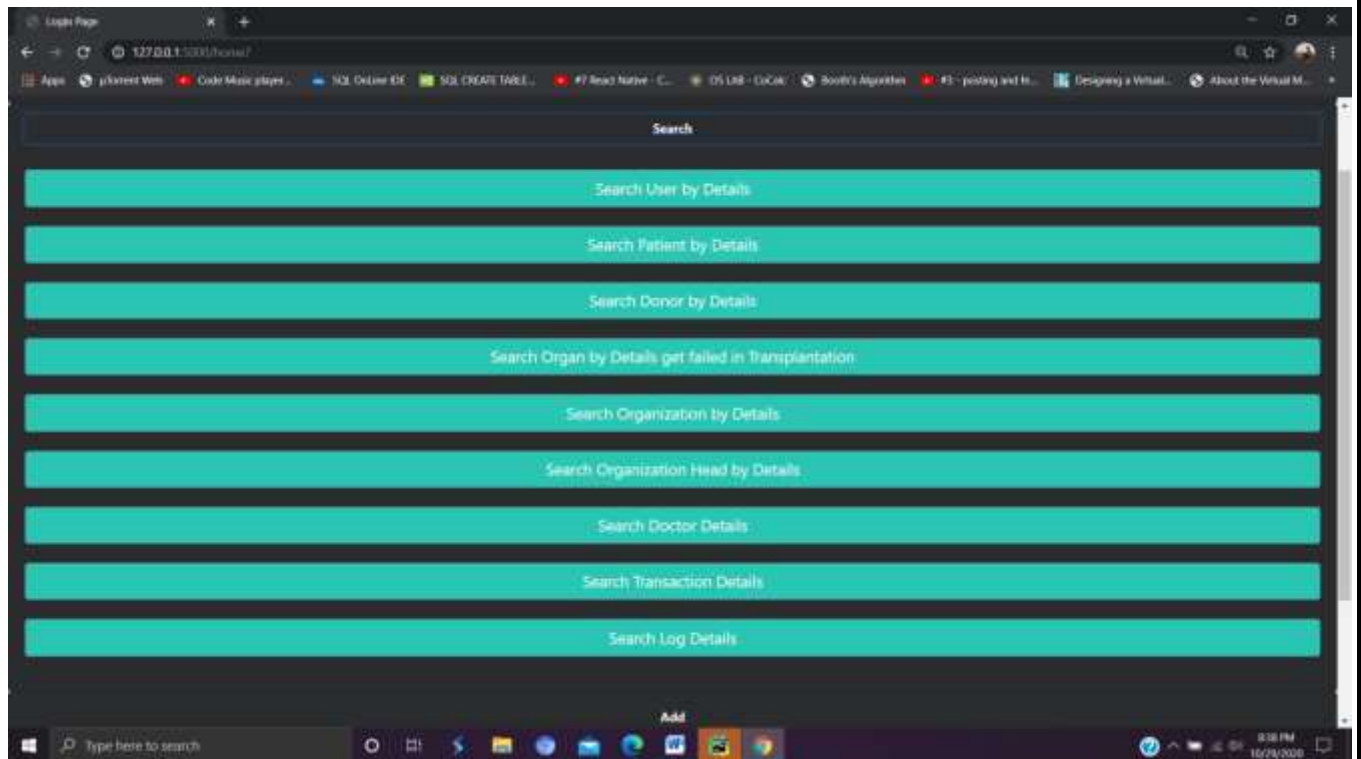
SEARCH:

Donor_ID	organ_donated	donor_name	Patient_ID
93	Kidney	Name-93	1
155	Kidney	Name-155	4
58	Heart	Name-58	6
172	Kidney	Name-172	7
71	Pancreas	Name-71	8
147	Intestine	Name-147	10
145	Heart	Name-145	12
4	Intestine	Name-4	18
129	Intestine	Name-129	20
144	Heart	Name-144	21

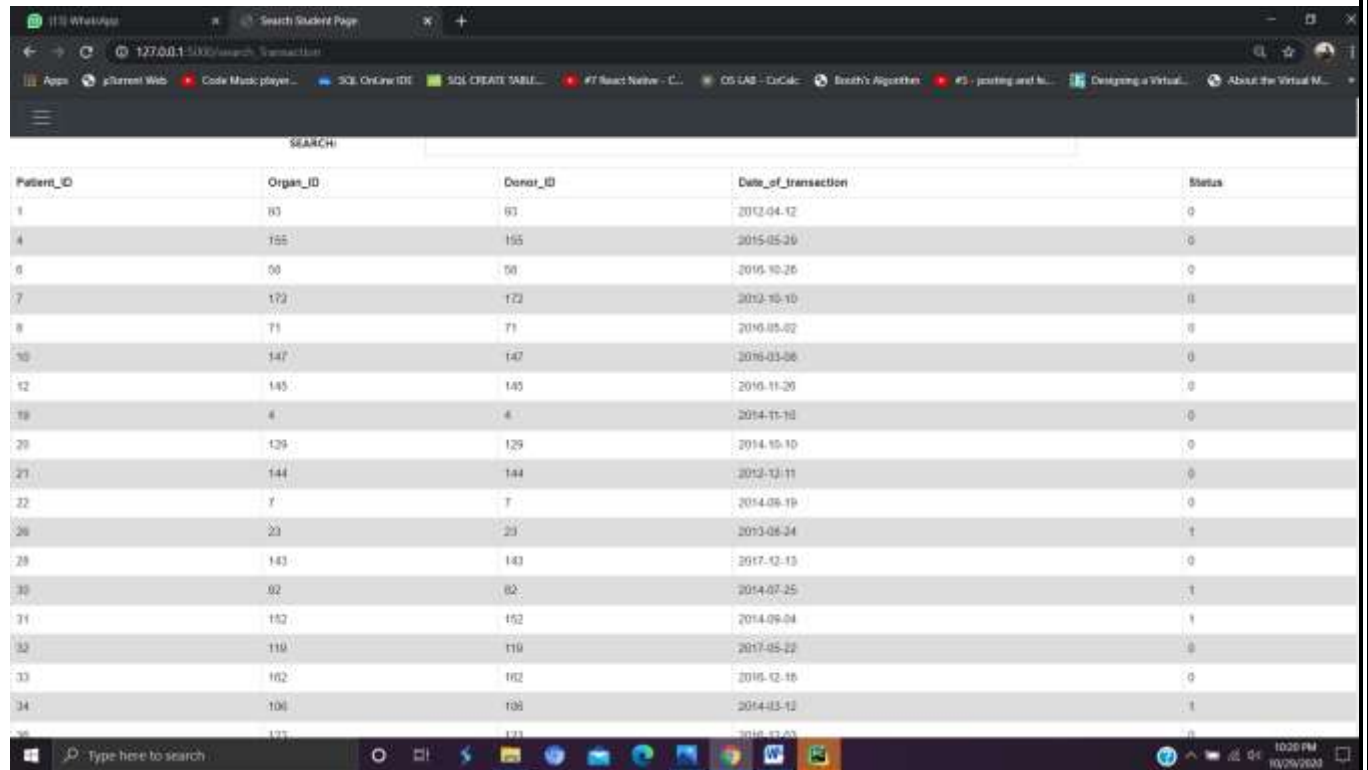


Live Statistics of data between success and failure transplantation.





## Transactions of Donor and Patient:-



Patient_ID	Organ_ID	Donor_ID	Date_of_transaction	Status
1	83	83	2012-04-12	0
4	155	155	2015-05-29	0
6	50	50	2016-10-26	0
7	172	172	2012-10-10	0
8	71	71	2016-05-02	0
10	147	147	2016-03-06	0
12	145	145	2016-11-20	0
19	4	4	2014-11-18	0
20	129	129	2014-10-10	0
21	144	144	2012-12-11	0
22	7	7	2014-06-19	0
26	23	23	2013-06-24	1
28	143	143	2017-12-13	0
30	82	82	2014-07-25	1
31	152	152	2014-09-04	1
32	119	119	2017-05-22	0
33	162	162	2016-12-16	0
34	106	106	2014-03-12	1
35	173	173	2016-11-03	0

127.0.0.1:5000/add\_donor\_page

## Organ Procurement and Donation Management System

Donor_ID	<input type="text"/>
organ_donated	<input type="text"/>
reason_of_donation	<input type="text"/>
Organization_ID	<input type="text"/>
User_ID	<input type="text"/>
donor_name	<input type="text"/>

Add Donor

127.0.0.1:5000/add\_transaction\_page

## Organ Procurement and Donation Management System

Patient_ID	<input type="text"/>
Organ_ID	<input type="text"/>
Donor_ID	<input type="text"/>
Date_of_transaction	<input type="text"/>
Status	<input type="text"/>

Add Transaction

# Organ Donation and Procurement Management System

## USER DETAILS

1	User_ID	1
2	Name	Name-1
3	Date_of_Birth	1978-08-21
4	Medical_Insurance	1
5	Medical_History	Nil
6	Street	Street-1
7	City	New Delhi
8	State	Delhi
12	Phone Numbers	Y009430002 - 9308904019 ,

User details with all medical history and all other details.

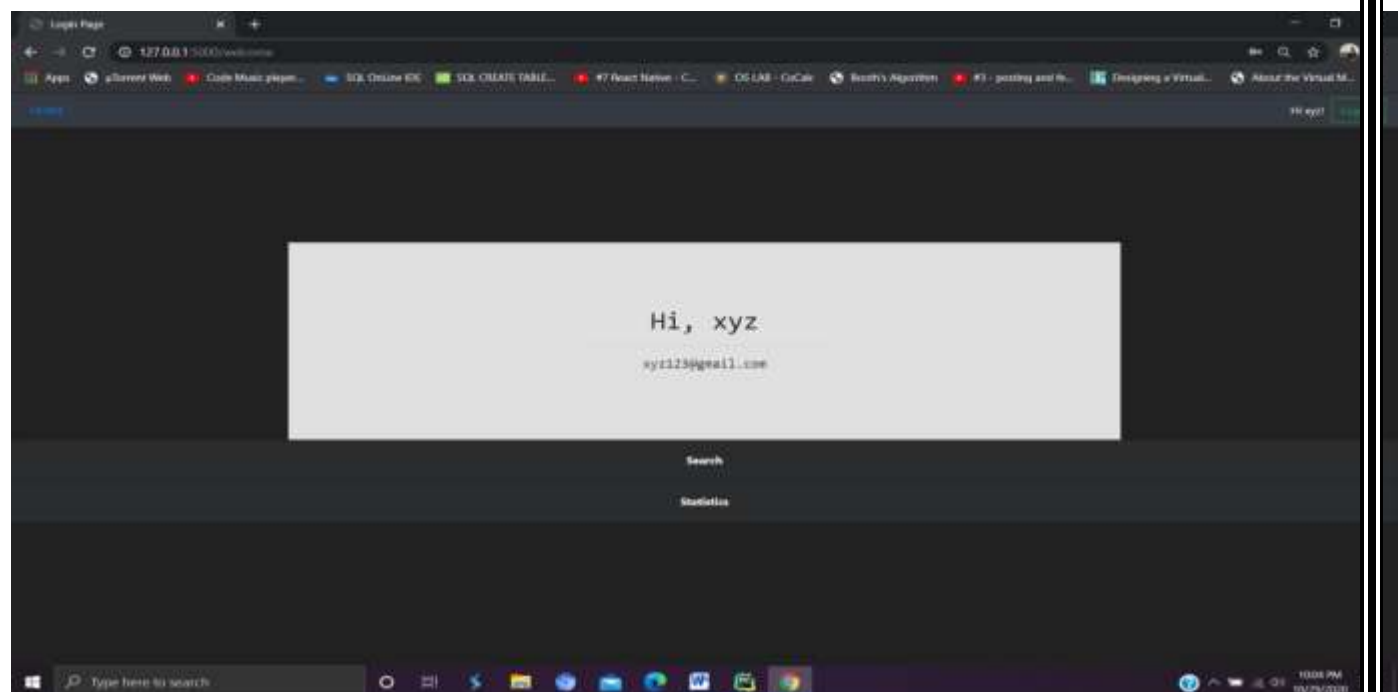
## PATIENT DETAILS

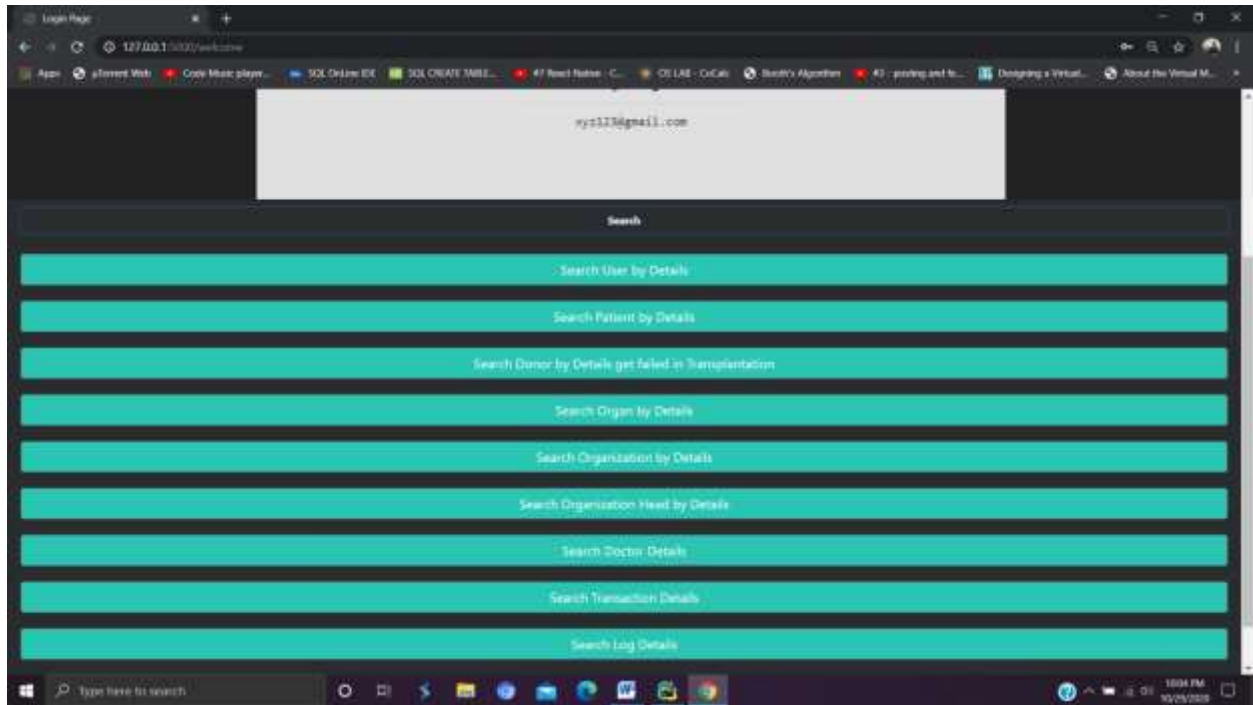
Sorry Not Applicable

## DONOR DETAILS

1	Donor_ID	90
2	organ_donated	Intestine
3	reason_of_donation	Reason-00
4	Organization_name	Organization-43
1	Donor_ID	167
2	organ_donated	Pancreas
3	reason_of_donation	Reason-167
4	Organization_name	Organization-58
1	Donor_ID	180
2	organ_donated	Heart
3	reason_of_donation	Reason-180
4	Organization_name	Organization-88

## Public portal:-





## Conclusion :

Every organ in the human body has significance and performs a particular task required for the survival of the human being. Thus, organ donation plays a very vital role and through this we can save many lives. Thus, with this project we tried to point out the loopholes in the current system and tried to fill them with an effective solution.

By implementing this idea we tried to reduce prevailing malpractices and wastage of organs due to lack of communication platform between donors and patients. It reduces the workload of admin as they can retrieve data easily. We also tried to spread awareness about organ donation. Through this database we can retrieve the data of past patients and donors. We can also predict the availability of live donors in future by using statistics and since statistics can also be seen by public, so it's a proof that people must donate organ when needed as they can see number of failed and passed transaction. Thus, with this project we tried to provide complete transparency between the donors and patients. So, as anyone can view the database, different malpractices can be completely avoided.

### **Future aspects:-**

- 1) Need to be more secure system for public usage.
- 2) Live tracking system
- 3) Improve GUI
- 4) Using data scored in our database, we can suggest suitable donor and patient pair using various biological and geographical factors.

**To run this project follow the steps given in procedure.txt file**