

**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**  
JNANASANGAMA, BELAGAVI - 590018



**Mini Project Report**  
**on**

**ORGAN DONATION AND PROCUREMENT MANAGEMENT  
SYSTEM**

*Submitted in partial fulfillment for the award of degree of*

**Bachelor of Engineering**  
**in**  
**COMPUTER SCIENCE AND ENGINEERING**

Submitted by  
**DHAVIN U**  
(1BG18CS032)



*Vidyaya Amrutham Ashnutha*

*B.N.M. Institute of Technology*

Approved by AICTE, Affiliated to VTU, Accredited as grade A Institution by NAAC.

All UG branches – CSE, ECE, EEE, ISE & Mech.E accredited by NBA for academic years 2018-19 to 2020-21 & valid upto 30.06.2021

Post box no. 7087, 27<sup>th</sup> cross, 12<sup>th</sup> Main, Banashankari 2<sup>nd</sup> Stage, Bengaluru- 560070, INDIA

Ph: 91-80- 26711780/81/82 Email: principal@bnmit.in, www. bnmit.org

**Department of Computer Science and Engineering**  
**2020-21**

# *B.N.M. Institute of Technology*

Approved by AICTE, Affiliated to VTU, Accredited as grade A Institution by NAAC.

All UG branches – CSE, ECE, EEE, ISE & Mech.E accredited by NBA for academic years 2018-19 to 2020-21 & valid upto 30.06.2021

Post box no. 7087, 27<sup>th</sup> cross, 12<sup>th</sup> Main, Banashankari 2<sup>nd</sup> Stage, Bengaluru- 560070, INDIA

Ph: 91-80- 26711780/81/82 Email: principal@bnmit.in, www. bnmit.org

## Department of Computer Science and Engineering



*Vidyaya Amrutham Ashnutha*

## **CERTIFICATE**

Certified that the Mini Project entitled **ORGAN DONATION AND PROCUREMENT MANAGEMENT SYSTEM** carried out by **Mr. DHAVIN U USN 1BG18CS032** a bonafide student of V Semester B.E., **B.N.M Institute of Technology** in partial fulfillment for the Bachelor of Engineering in COMPUTER SCIENCE AND ENGINEERING of the **Visvesvaraya Technological University**, Belagavi during the year 2020-21. It is certified that all corrections/ suggestions indicated for internal Assessment have been incorporated in the report. The project report has been approved as it satisfies the academic requirements in respect of Database Management Systems Laboratory with Mini Project as prescribed for the said degree.

**Prof. Asha K**  
**Assistant Professor & Lab-Incharge**  
**Department of CSE**  
**BNMIT, Bengaluru**

**Dr. Sahana D. Gowda**  
**Professor & HOD**  
**Department of CSE**  
**BNMIT, Bengaluru**

**Name & Signature**

**Examiner1:**

**Examiner2:**

# **ABSTRACT**

Organ transplantation 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. Organ Donation and Procurement Organizations play an important role in today's medical institutions. Such organizations are responsible for the evaluation and procurement of organs for organ transplantation. These organizations have direct contact with the hospital and the family of a recently deceased donor. The work of such organizations includes identifying the best candidates for the available organs and to coordinate with the medical institutions to decide on each organ recipient.

The ORGAN DONATION AND PROCUREMENT MANAGEMENT SYSTEM is developed mainly for general hospitals, clinics and other health centers to manage the donor and patient registration and user maintenance. The public can retrieve information about organ donation in this web site. People who interested can register themselves through this system. The application will be processed by the administrator and each donor will receive feedback about their application status. Furthermore, the authorized user's account will be maintained by the administrator. The donor record will be managed by four main users such as administrator, doctor, medical assistant and management staff. Only administrator has the authority and privileges to print organ list report and total donation report according to district from this system. The methodology of this system is Structured System Analysis and Design .An analysis study has been done based on the current manual system and all the problems statements and requirements have been identified. Moreover, ORGAN DONATION AND PROCUREMENT MANAGEMENT SYSTEM is three tier architecture system which involves client tier, business tier and database management tier. The interfaces for ORGAN DONATION AND PROCUREMENT MANAGEMENT SYSTEM have been designed according to the requirement and needs of the current market Rather than that, this system also has been tested and evaluated in real life. This ORGAN DONATION AND PROCUREMENT MANAGEMENT SYSTEM will help to improve the performance of current situation and overcome the problems that arise nowadays.

# ACKNOWLEDGEMENT

The success and final outcome of this project required a lot of guidance and assistance from many people and I am extremely privileged to have got this all along the completion of my project.

I would like to thank **Shri. Narayan Rao R Maanay**, Secretary, BNMIT, Bengaluru for providing excellent academic environment in the college.

I would like to sincerely thank **Prof. T J Rama Murthy**, Director, BNMIT, Bengaluru for having extended his support and encouragement during the course of the work.

I would like to express my gratitude to **Prof. Eishwar N Maanay**, Dean, BNMIT, Bengaluru for his relentless support, guidance and assistance.

I would like to thank **Dr. Krishnamurthy G N**, Principal, BNMIT, Bengaluru for his constant encouragement.

I would like to thank, **Dr. Sahana D Gowda**, Professor and Head of the Department of Computer Science and Engineering who has shared her opinions and thoughts which helped me in giving my presentation successfully.

I would like to thank. **Prof Asha K**, Assistant Professor in the Department of Computer Science and Engineering for her assistance and guidance in doing this project.

Finally, I take this opportunity to extend my earnest gratitude and respect to my parents, teaching & non-teaching staffs of the department and all my friends, for giving me valuable advice and support at all times in all possible ways.

**Dhavin U**  
**1BG18CS032**

# Table of Contents

<b>CONTENTS</b>	<b>Page No.</b>
<b>ABSTRACT</b>	<b>I</b>
<b>ACKNOWLEDGEMENT</b>	<b>II</b>
<b>1. INTRODUCTION</b>	<b>1</b>
1.1 Overview of Database Management Systems	1
1.2 Problem statement	3
1.3 Objectives	4
1.4 Dataset description	4
<b>2. SYSTEM REQUIREMENTS</b>	<b>6</b>
2.1 Software requirements	6
2.2 Hardware requirements	6
<b>3. SYSTEM DESIGN</b>	<b>7</b>
3.1 E R Diagram	7
3.2 Schema Diagram	8
3.3 Overview of GUI	9
3.4 Normalization	10
<b>4. IMPLEMENTATION</b>	<b>12</b>
4.1 Table Creation	12
4.2 Description of Tables	16
4.3 Populated Tables	18
4.4 SQL Triggers and Stored Procedure	19
4.5 Database Connectivity	20
<b>5. RESULTS</b>	<b>22</b>
<b>CONCLUSION</b>	<b>32</b>
<b>FUTURE ENHANCEMENTS</b>	<b>33</b>

## List of Figures

<b>Figure No.</b>	<b>Figure Name</b>	<b>Page No.</b>
3.1	Entity Relationship Diagram	7
3.2	Schema Diagram	8
4.1	User Description	16
4.2	Organization Description	16
4.3	Doctor Description	16
4.4	Donor Description	17
4.5	Patient Description	17
4.6	Organ_availability Description	17
4.7	User Table	18
4.8	Donor Table	18
4.9	Patient Table	18
4.10	Doctor Table	18
4.11	Organization Table	19
4.12	Trigger to add donor information	19
4.13	Trigger to update donor information	19
4.14	Trigger to delete donor information	20
4.15	Stored Procedure to add transaction	20
5.1	Main Page	22
5.2	Admin Login	22
5.3	Admin home	23
5.4	Admin User Menu	23
5.5	Admin Search Menu	23
5.6	Admin Add Menu	24
5.7	Admin Update Menu	24
5.8	Admin Remove Menu	25
5.9	User Details (a)	25
5.10	User Details (b)	26
5.11	Search User	26
5.12	Search Transaction	27

5.13	Add User	27
5.14	Add User_phone	28
5.15	Add Donor Page	28
5.16	Log Details Page	29
5.17	User Login Page	29
5.18	User Home Page	30
5.19	Donate Page	30
5.20	Procure Page	31

# List of Tables

Table No.	Table Name	Page No.
3.1	User table in 1NF	10
3.2	User table in 2NF	11
3.3	User_Phoneno table in 2NF	11





# Chapter 1

## INTRODUCTION

### 1.1 Overview of Database Management System

A Database is a collection of related data organized in a way that data can be easily accessed, managed and updated and various operations can be performed on it. By Data, we mean known facts that can be recorded and that have implicit meaning. A DBMS is a collection of programs that enables users to create and maintain a database. The DBMS is a general-purpose software system that facilitates the processes of defining, constructing, manipulating, and sharing databases among various users and applications. DBMS also provides protection and security to database. It maintains data consistency in case of multiple users. Here are some examples of popular DBMS, MySQL, Oracle, Sybase, Microsoft Access and IBM DB2 etc.

The database system can be divided into four components:

- The database system can be divided into System developer and End users.
- Database application: Database application may be Personal, Departmental, Enterprise and Internal
- DBMS: Software that allow users to define, create and manages database access, Ex: MySQL, Oracle etc.
- Database: Collection of logical data.

Functions of database management system:

- Provides Recovery services
- Provides utility
- Provides data Independence
- Provides a clear and logical view of the process that manipulates data

Advantages of DBMS:

- Segregation of application program
- Minimal data duplicity and Easy retrieval of data

### **1.1.2 Advantages of a Database Management System**

#### **Controlling data redundancy:**

In non-database systems each application program has its own private files. In this case, the duplicated copies of the same data are created in many places. In DBMS, all data of an organization is integrated into a single database file. The data is recorded in only one place in the database and it is not duplicated.

#### **Sharing of data:**

In DBMS, data can be shared by authorized users of the organization. The Database administrator manages the data and gives rights to users to access the data. Many users can be authorized to access the same piece of information simultaneously. The remote users can also share same data. Similarly, the data of same database can be shared between different application programs.

#### **Data consistency:**

By controlling the data redundancy, the data consistency is obtained. If a data item appears only once, any update to its value has to be performed only once and the updated value is immediately available to all users.

#### **Data security:**

Form is very important object of DBMS. It can be created very easily and quickly in DBMS. Once a form is created, it can be used many times and it can be modified very easily. The created forms are also saved along with database and behave like a software component. A form provides very easy way (user-friendly) to enter data into database, edit data and display data from database. The non-technical users can also perform various operations on database through forms without going into technical details of a database.

#### **Integration of data:**

In Database management system, data in database is stored in tables. A single database contains multiple tables and relationships can be created between tables (or associated data entities). This makes easy to retrieve and update data.

**Integration constraints:**

Integrity constraints or consistency rules can be applied to database so that the correct data can be entered into database. The constraints may be applied to data item within a single record or may be applied to relationships between records.

### **1.1.3 Advantages of Database Management System**

Provides data abstraction and segregation of application program from the data.

Reduced redundancy of data ensures maximum cost efficiency for the storage of data

Reduced development time while building applications that use database

## **1.2 Problem Statement**

Organ transplantation 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. Organ Donation and Procurement Organizations play an important role in today's medical institutions. Such organizations are responsible for the evaluation and procurement of organs for organ transplantation. These organizations have direct contact with the hospital and the family of a recently deceased donor. The work of such organizations includes identifying the best candidates for the available organs and to coordinate with the medical institutions to decide on each organ recipient.

The **Organ Donation and Procurement 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 maintains a comprehensive medical history and other critical information like blood group, age, etc. of every person in the database design. In short, it maintains a database containing statistical information regarding the network of organ donation and procurement of different countries.

## 1.3 Objectives

- Organ Donation and Procurement Management System is a platform where in a user can log in and either become a donor or a procurer to donate or procure a specific organ.
- Organ Donation and Procurement Management System contains a database of users who are spread worldwide as a result it can be beneficial for anyone in need for organ which can save a person's life.
- A patient can come and look for which organization is offering a particular organ so that the patient can contact the donor to make a transaction.
- A donor can be called by a doctor through an organization if any patient is in need of any organ if they are agreeing to give.
- The Donor and Patient transaction can be competed in the same platform which is implemented using stored procedures.

## 1.4 Dataset Description

Given below are the entities along with its attributes and relations present in the database of this application that are used to retrieve information from the database as per requirement of user.

- An entity type '**USER**' with attribute user\_id, name, dob, medical\_insurance, medical\_history, street, city, state with primary key as user\_id.
- An entity type '**ORGANIZATION**' with attributes organization\_id, organization\_name, location, government\_approved with primary key as organization\_id,
- An entity type '**DOCTOR**' with attributes doctor\_id, doctor\_name, department\_name, organization\_id with foreign key as organization\_id and primary key as doctor\_id.
- An entity type '**PATIENT**' with patient\_id, organ\_required, reason\_of\_procurement, doctor\_id, user\_id with foreign key as user\_id from

USER table and doctor\_id from DOCTOR table and patient\_id and organ\_required as primary key.

- An entity type '**DONOR**' with attribute donor\_id, organ\_donated, reason\_of\_donation, organization\_id and user\_id with foreign key as user\_id and organization\_id and primary key as donor\_id and organ\_donated.
- An entity type '**ORGAN\_AVAILABILITY**' with attribute patient\_id, organ\_id, donor\_id, date\_of\_transaction and status having foreign key as patient\_id and donor\_id and primary key as patient\_id and organ\_id.

## **Chapter 2**

# **SYSTEM REQUIREMENTS**

## **2.1 Software Requirements**

Software requirements deal with defining software resource requirements and prerequisites that need to be installed on a computer to provide optimal functioning of an application.

### **2.1.1 Front End**

HTML5/CSS/JavaScript

Google Chrome (Web Browser)

### **2.1.2 Back End**

Live Server VSCode Extension

MySQL (v8.0.12) for Database Management System

Python (Flask Framework)

Visual Studio Code (Source Code Editor)

Windows 10

## **2.2 Hardware Requirements**

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware.

Cores: Single-Core (Dual-Core is recommended)

RAM: minimum 4GB (6GB recommended)

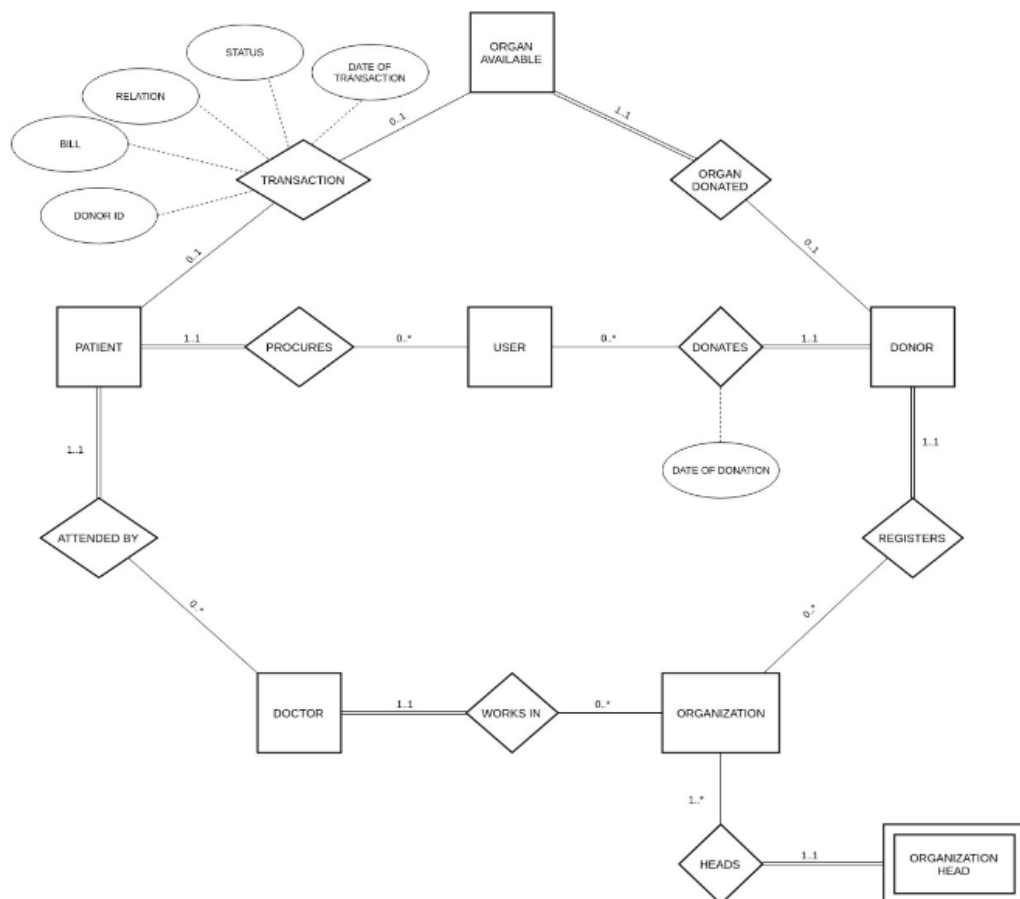
Hard disk: 40GB hard disk

## Chapter 3

# SYSTEM DESIGN

### 3.1 Entity Relationship Diagram

An Entity Relationship Diagram describes interrelated things of interest in a specific domain of knowledge. A basic ER model is composed of entity types and specifies relationships that can exist between instances of those entity types. In software engineering, an ER model is commonly formed to represent things that a business needs to remember in order to perform business processes. Consequently, the ER model becomes an abstract data model that defines a data or information structure which can be implemented in a database, typically a relational database.



**Figure 3.1: Entity Relationship Diagram.**



### 3.2 Schema Diagram

A database schema is the skeleton structure that represents the logical view of the entire database. It formulates all the constraints that are to be applied on the data. A database schema defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagrams.

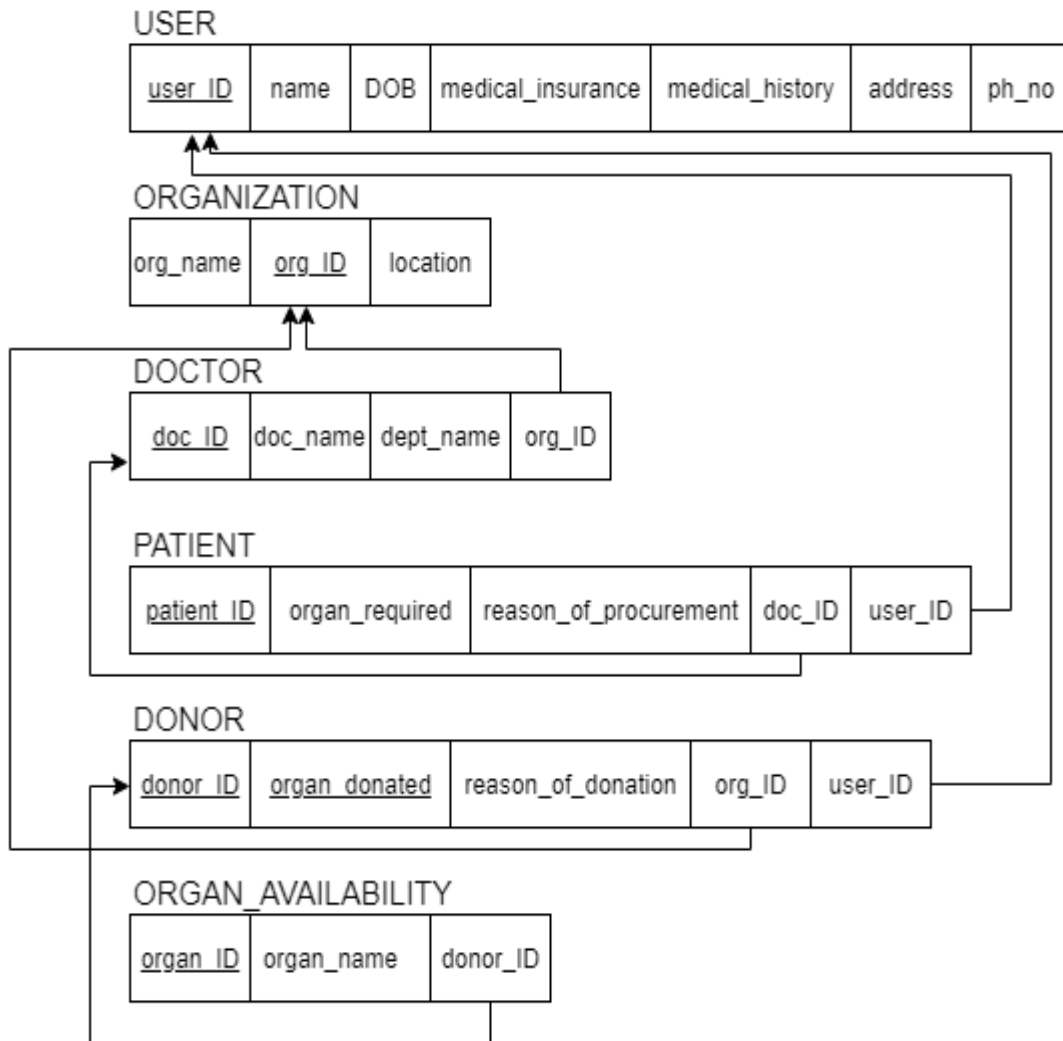


Figure 3.2: Schema diagram of Organ Donation and Procurement Management System.

### 3.3 Overview of Graphical User Interface

GUI is program interface that takes advantages of the computer's graphics capabilities to make the program easier to user. Well-designed graphical user interfaces can free the user from learning complex command language. On the other hand, many users worked more efficiently with a command-driven interface, especially if they already know the command language.

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a web server or from local storage and render them into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document and is applicable to rendering in speech, or on other media. CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

**Flask** is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, and upload handling, various open authentication technologies and several common framework related tools.

### 3.4 Normalization

Normalization is a process of analyzing the given relation schema based on their functional dependencies and primary key to achieve desirable properties of minimizing redundancy and minimizing insert, delete, update anomaly. The normalization process takes a relation schema through a series of tests to certify whether it satisfies a certain normal form. The normal form of a relation refers to the highest normal form condition that it meets, and hence the degree to which it has been normalized. There are two goals of the normalization process: eliminating redundant data (for example, storing the same data in more than one table) and ensuring data dependencies make sense (only storing related data in a table). Both of these are worthy goals as it reduces the amount of space a database consumes and ensure that data is logically stored.

#### 3.4.1 1NF (First Normal Form):

1NF Rules:

-Each table cell should contain a single value.

-Each record needs to be unique the table below is in 1NF:

User_ID	Name	DOB	Med_ins	Med_his	Street	City	State
1	Dhavin	1987-08-21	1	NIL	Street-1	Bangalore	K'taka
2	Virat	1978-09-23	0	NIL	Street-2	Chennai	TN

**Table 3.1 User table in 1NF**

It is in 1NF as it does not have any composite or multivalued attributes.

#### 3.4.2 2NF (Second Normal Form):

2NF Rules:

-Should be in 1NF

-Single column primary key the table below is in 2NF:

It is clear that to move forward to make our simple database in 2<sup>nd</sup>

Normalization form the table above needs to be partitioned:

<u>User_ID</u>	Name	DOB	Med_ins	Med_his	Street	City	State	Phno
----------------	------	-----	---------	---------	--------	------	-------	------

User_ID	<u>Phno</u>
---------	-------------

**Table 3.2 User table in 2NF**

<u>User_ID</u>	Name	DOB	Med_ins	Med_his	Street	City	State
1	Dhavin	1987-08-21	1	NIL	Street-1	Bangalore	K'taka
2	Virat	1978-09-23	0	NIL	Street-2	Chennai	TN

**Table 3.3 User\_Phoneno table in 2NF**

User_ID	<u>Phno</u>
1	8971727374
1	8971727373
2	8971727376
2	8971727378

A User can have one or more phone numbers, to accommodate the phone numbers in a single table will cause us deletion and updation anomalies. Hence, we separate the Phno attribute and make Phno as primary key in a separate table

It is in 2NF as it is fully functionally dependent.

As the relation does not contain a non-key attribute functionally determining other non-key attributes, it is in 3NF (there is no transitivity in attributes).

## Chapter 4

# IMPLEMENTATION

### 4.1 Table Creation

```
CREATE DATABASE DBMS_PROJECT;  
USE DBMS_PROJECT;
```

```
CREATE TABLE login(  
    username VARCHAR(20) NOT NULL,  
    password VARCHAR(20) NOT NULL  
);
```

```
INSERT INTO login VALUES ('admin','admin');
```

#table 1

```
CREATE TABLE User(  
    User_ID int NOT NULL,  
    Name varchar(20) NOT NULL,  
    Date_of_Birth date NOT NULL,  
    Medical_insurance int,  
    Medical_history varchar(20),  
    Street varchar(20),  
    City varchar(20),  
    State varchar(20),  
    PRIMARY KEY(User_ID)  
);
```

#table 2

```
CREATE TABLE User_phone_no(  
    User_ID int NOT NULL,  
    phone_no varchar(15),
```

```
FOREIGN KEY(User_ID) REFERENCES User(User_ID) ON DELETE  
CASCADE  
);
```

#table 3

```
CREATE TABLE Organization(  
    Organization_ID int NOT NULL,  
    Organization_name varchar(20) NOT NULL,  
    Location varchar(20),  
    Government_approved int, # 0 or 1  
    PRIMARY KEY(Organization_ID)  
);
```

#table 4

```
CREATE TABLE Doctor(  
    Doctor_ID int NOT NULL,  
    Doctor_Name varchar(20) NOT NULL,  
    Department_Name varchar(20) NOT NULL,  
    organization_ID int NOT NULL,  
    FOREIGN KEY(organization_ID) REFERENCES Organization(organization_ID)  
ON DELETE CASCADE,  
    PRIMARY KEY(Doctor_ID)  
);
```

#table 5

```
CREATE TABLE Patient(  
    Patient_ID int NOT NULL,  
    organ_req varchar(20) NOT NULL,  
    reason_of_procurement varchar(20),  
    Doctor_ID int NOT NULL,  
    User_ID int NOT NULL,  
    FOREIGN KEY(User_ID) REFERENCES User(User_ID) ON DELETE  
CASCADE,
```

```
FOREIGN KEY(Doctor_ID) REFERENCES Doctor(Doctor_ID) ON DELETE  
CASCADE,  
PRIMARY KEY(Patient_Id, organ_req)  
);
```

#table 6

```
CREATE TABLE Donor(  
Donor_ID int NOT NULL,  
organ_donated varchar(20) NOT NULL,  
reason_of_donation varchar(20),  
Organization_ID int NOT NULL,  
User_ID int NOT NULL,  
FOREIGN KEY(User_ID) REFERENCES User(User_ID) ON DELETE  
CASCADE,  
FOREIGN KEY(Organization_ID) REFERENCES Organization(Organization_ID)  
ON DELETE CASCADE,  
PRIMARY KEY(Donor_ID, organ_donated)  
);
```

#table 7

```
CREATE TABLE Organ_available(  
Organ_ID int NOT NULL AUTO_INCREMENT,  
Organ_name varchar(20) NOT NULL,  
Donor_ID int NOT NULL,  
FOREIGN KEY(Donor_ID) REFERENCES Donor(Donor_ID) ON DELETE  
CASCADE,  
PRIMARY KEY(Organ_ID)  
);
```

#table 8

```
CREATE TABLE Transaction(  
Patient_ID int NOT NULL,  
Organ_ID int NOT NULL,  
Donor_ID int NOT NULL,
```

```
Date_of_transaction date NOT NULL,  
Status int NOT NULL, #0 or 1  
FOREIGN KEY(Patient_ID) REFERENCES Patient(Patient_ID) ON DELETE  
CASCADE,  
FOREIGN KEY(Donor_ID) REFERENCES Donor(Donor_ID) ON DELETE  
CASCADE,  
PRIMARY KEY(Patient_ID,Organ_ID)  
);
```

#table 9

```
CREATE TABLE Organization_phone_no(  
Organization_ID int NOT NULL,  
Phone_no varchar(15),  
FOREIGN KEY(Organization_ID) REFERENCES Organization(Organization_ID)  
ON DELETE CASCADE  
);
```

#table 10

```
CREATE TABLE Doctor_phone_no(  
Doctor_ID int NOT NULL,  
Phone_no varchar(15),  
FOREIGN KEY(Doctor_ID) REFERENCES Doctor(Doctor_ID) ON DELETE  
CASCADE  
);
```

#table 11

```
CREATE TABLE Organization_head(  
Organization_ID int NOT NULL,  
Employee_ID int NOT NULL,  
Name varchar(20) NOT NULL,  
Date_of_joining date NOT NULL,  
Term_length int NOT NULL,  
FOREIGN KEY(Organization_ID) REFERENCES Organization(Organization_ID)  
ON DELETE CASCADE,
```



PRIMARY KEY(Organization\_ID,Employee\_ID)  
);

## 4.2 Description of Tables

```
mysql> desc user;
```

Field	Type	Null	Key	Default	Extra
User_ID	int	NO	PRI	NULL	
Name	varchar(20)	NO		NULL	
Date_of_Birth	date	NO		NULL	
Medical_insurance	int	YES		NULL	
Medical_history	varchar(20)	YES		NULL	
Street	varchar(20)	YES		NULL	
City	varchar(20)	YES		NULL	
State	varchar(20)	YES		NULL	

8 rows in set (0.18 sec)

Fig 4.1 – User Description

```
mysql> desc organization;
```

Field	Type	Null	Key	Default	Extra
Organization_ID	int	NO	PRI	NULL	
Organization_name	varchar(20)	NO		NULL	
Location	varchar(20)	YES		NULL	
Government_approved	int	YES		NULL	

4 rows in set (0.29 sec)

Fig 4.2 – Organization Description

```
mysql> desc doctor;
```

Field	Type	Null	Key	Default	Extra
Doctor_ID	int	NO	PRI	NULL	
Doctor_Name	varchar(20)	NO		NULL	
Department_Name	varchar(20)	NO		NULL	
organization_ID	int	NO	MUL	NULL	

4 rows in set (0.00 sec)

Fig 4.3 – Doctor Description

```
mysql> desc donor;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Donor_ID       | int           | NO   | PRI | NULL    |       |
| organ_donated   | varchar(20)   | NO   | PRI | NULL    |       |
| reason_of_donation | varchar(20)   | YES  |     | NULL    |       |
| Organization_ID | int           | NO   | MUL | NULL    |       |
| User_ID        | int           | NO   | MUL | NULL    |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.03 sec)
```

Fig 4.4 – Donor Description

```
mysql> desc patient;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Patient_ID     | int           | NO   | PRI | NULL    |       |
| organ_req       | varchar(20)   | NO   | PRI | NULL    |       |
| reason_of_procurement | varchar(20)   | YES  |     | NULL    |       |
| Doctor_ID      | int           | NO   | MUL | NULL    |       |
| User_ID        | int           | NO   | MUL | NULL    |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.02 sec)
```

Fig 4.5 – Patient Description

```
mysql> desc organ_available;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Organ_ID       | int           | NO   | PRI | NULL    | auto_increment |
| Organ_name     | varchar(20)   | NO   |     | NULL    |               |
| Donor_ID       | int           | NO   | MUL | NULL    |               |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

Fig 4.6 – Organ\_availability Description

### 4.3 Populated Tables

```
mysql> select * from user;
```

User_ID	Name	Date_of_Birth	Medical_insurance	Medical_history	Street	City	State
1	Dhavin	1978-08-21	1	NIL	Street-1	Bangalore	Karnataka
2	Virat	1978-08-25	0	NIL	Street-2	Bangalore	Karnataka

2 rows in set (0.07 sec)

Fig 4.7 – User Table

```
mysql> select * from donor;
```

Donor_ID	organ_donated	reason_of_donation	Organization_ID	User_ID
1	Heart	Reason-1	1	1
2	Kidney	Reason-2	2	2

2 rows in set (0.03 sec)

Fig 4.8 – Donor Table

```
mysql> select * from patient;
```

Patient_ID	organ_req	reason_of_procurement	Doctor_ID	User_ID
1	Heart	Reason-1	1	1
2	Kidney	Reason-2	2	2

2 rows in set (0.04 sec)

Fig 4.9 – Patient Table

```
mysql> select * from doctor;
```

Doctor_ID	Doctor_Name	Department_Name	organization_ID
1	Dr.Elon Musk	Cardiology	1
2	Dr. Mukesh Ambani	Neurology	2

2 rows in set (0.03 sec)

Fig 4.10 – Doctor Table

```
mysql> select * from organization;
+-----+-----+-----+-----+
| Organization_ID | Organization_name | Location | Government_approved |
+-----+-----+-----+-----+
| 1 | BNM Hospital | Banashankari | 1 |
| 2 | Apollo Hospital | JP Nagar | 0 |
+-----+-----+-----+-----+
2 rows in set (0.07 sec)
```

**Fig 4.11 – Organization Table**

## 4.4 SQL Triggers & Stored Procedures

### Triggers

A database trigger is procedural code that is automatically executed in response to certain events on a table or view in a database. The trigger is mostly used for maintaining the integrity of the information on the database.

```
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 //
```

**Fig 4.12- Trigger to add donor information to Log Table**

```
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 //
```

**Fig 4.13- Trigger to update donor information to Log Table**

```
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 //
```

**Fig 4.14- Trigger to delete donor information to Log Table**

### **Stored Procedure**

A stored procedure is a set of Structured Query Language (SQL) statements with an assigned name, which are stored in a relational database management system as a group, so it can be reused and shared by multiple programs.

Code for Stored Procedure is as follows:

```
CREATE PROCEDURE ADD_TRANSACTION_LOG
BEGIN
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 //
```

**Fig 4.15- Stored Procedure to add transaction to Log table**

## **4.5 Database Connectivity**

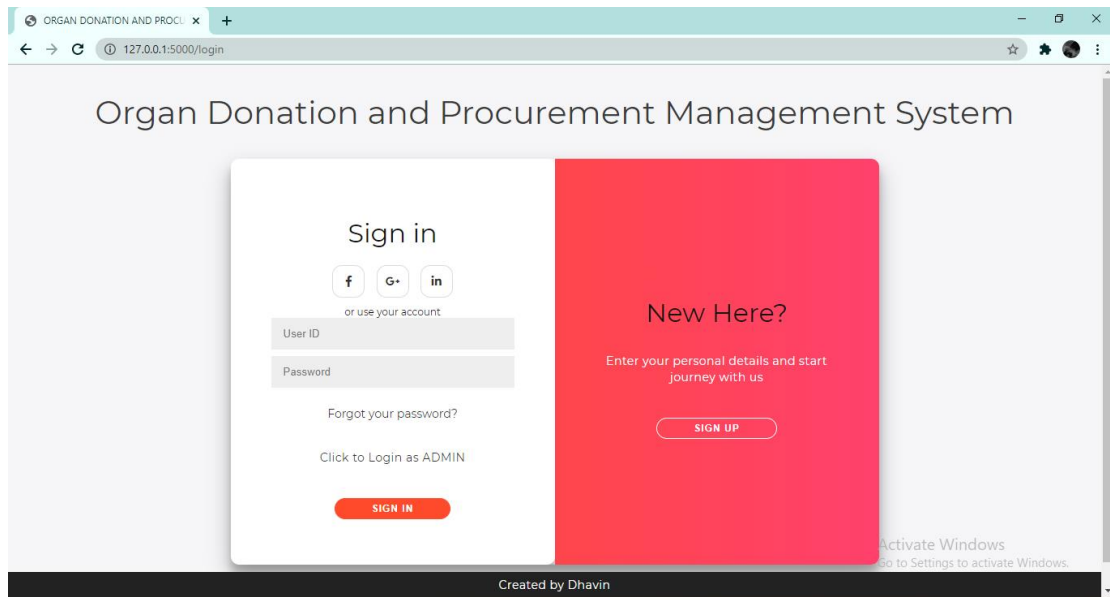
In computer science, a database connection is the means by which a database server and its client software communicate with each other. The term is used whether the client and the server are on different machines. The client uses a database connection to send commands to and receive replies from the server. A database is stored as a file or a set of files on magnetic disk or tape, optical disk, or some other secondary storage device. The information in these files may be broken down into records, each of which consists of one or more fields. The connection to MySQL database is obtained by the following code snippet:

```
mydb = mysql.connector.connect(  
    host='localhost',  
    user='root',  
    password='**MY_PASSWORD**',  
    database = 'DBMS_PROJECT'  
)
```

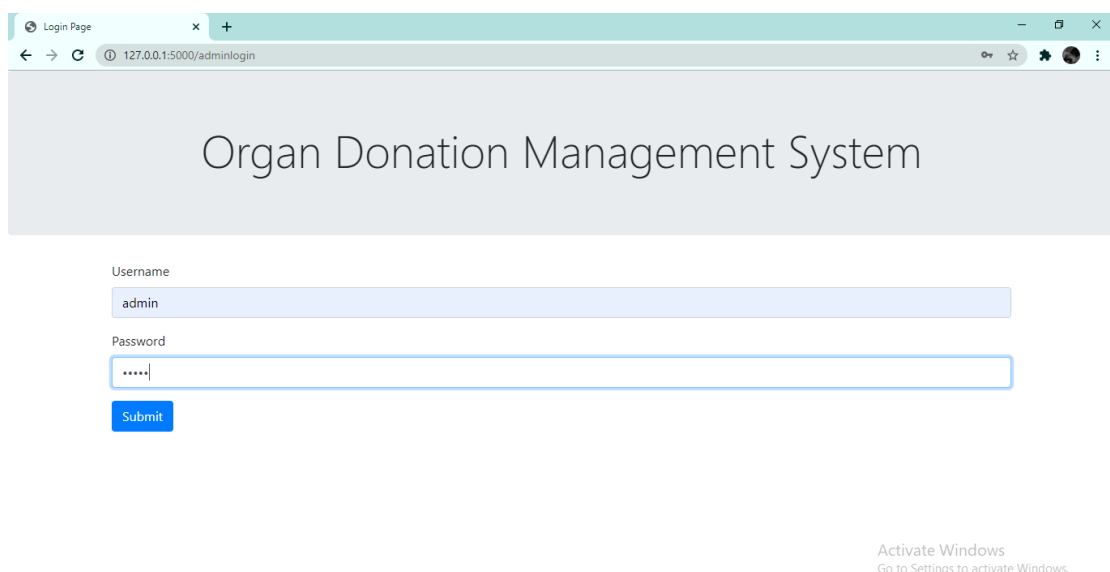
## Chapter 5

# RESULTS

### Main Page of ORGAN DONATION AND PROCUREMENT MANAGEMENT SYSTEM



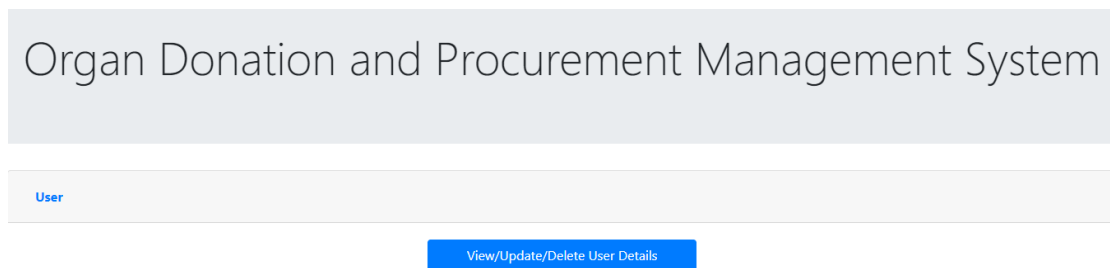
**Figure 5.1: Main page.**



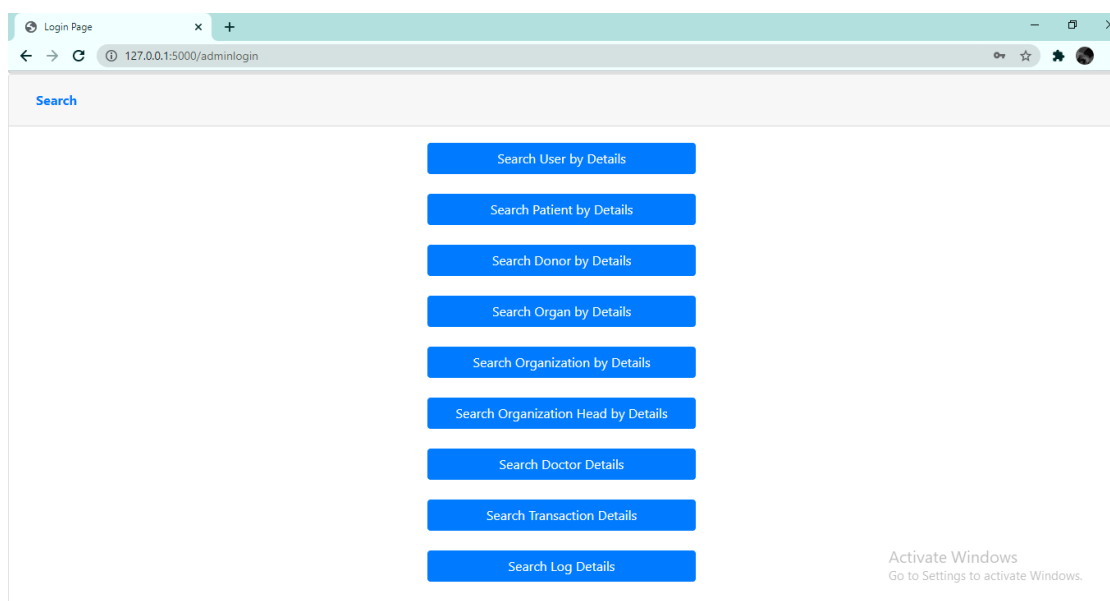
**Figure 5.2: Admin Login**



**Figure 5.3: Admin home**

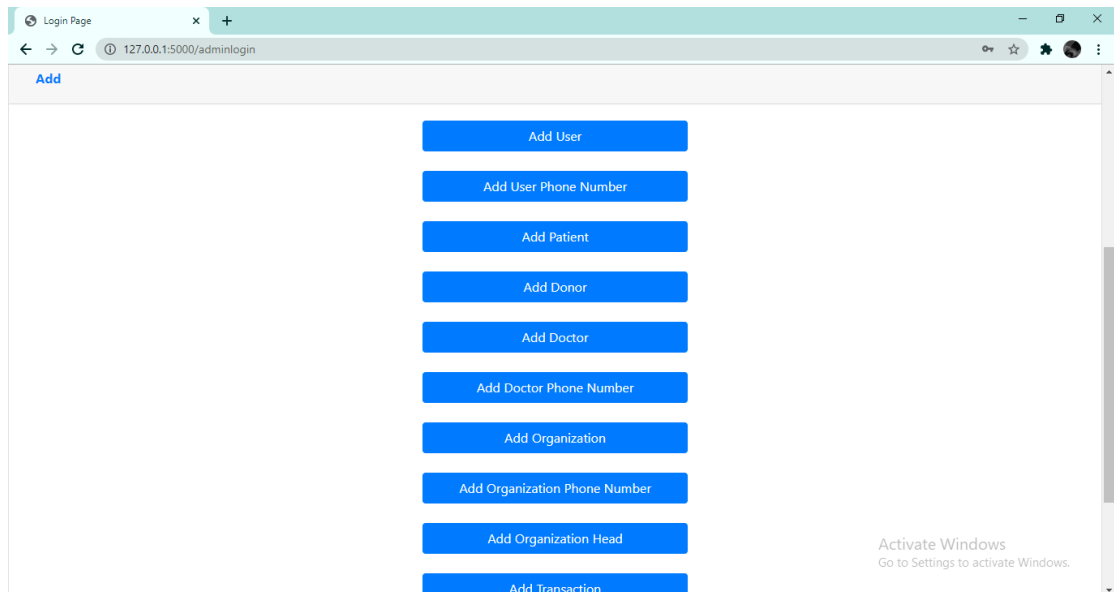


**Figure 5.4: Admin User Menu**

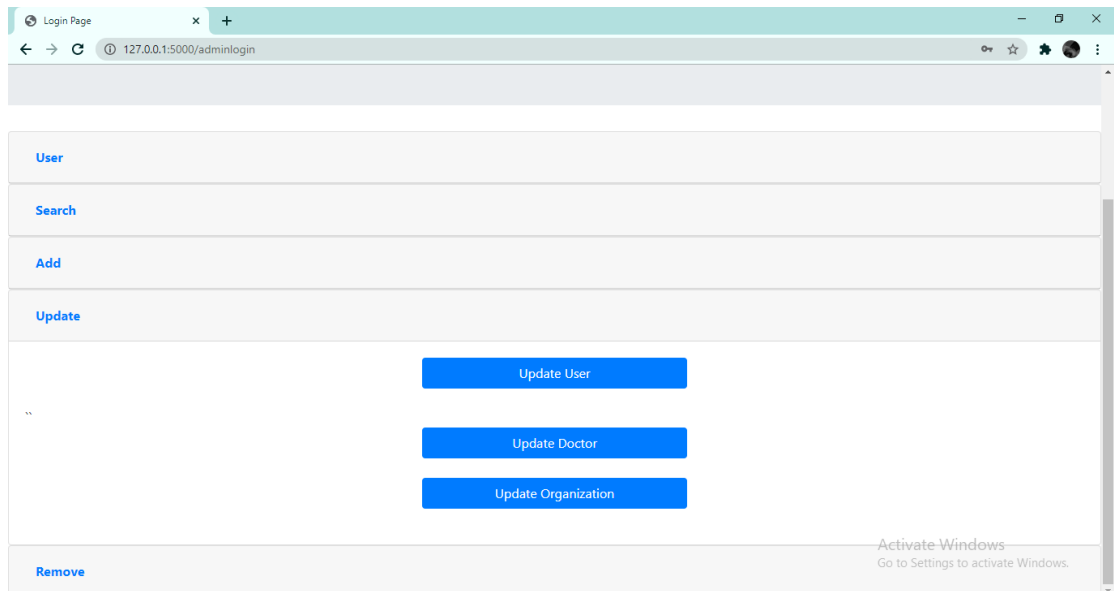


**Figure 5.5: Admin Search Menu**

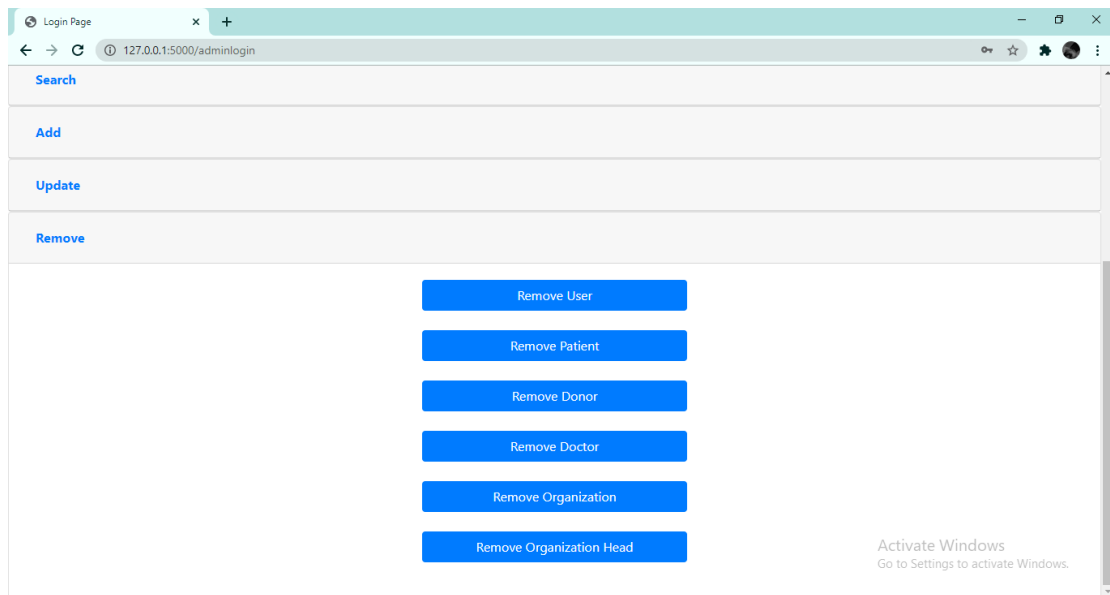




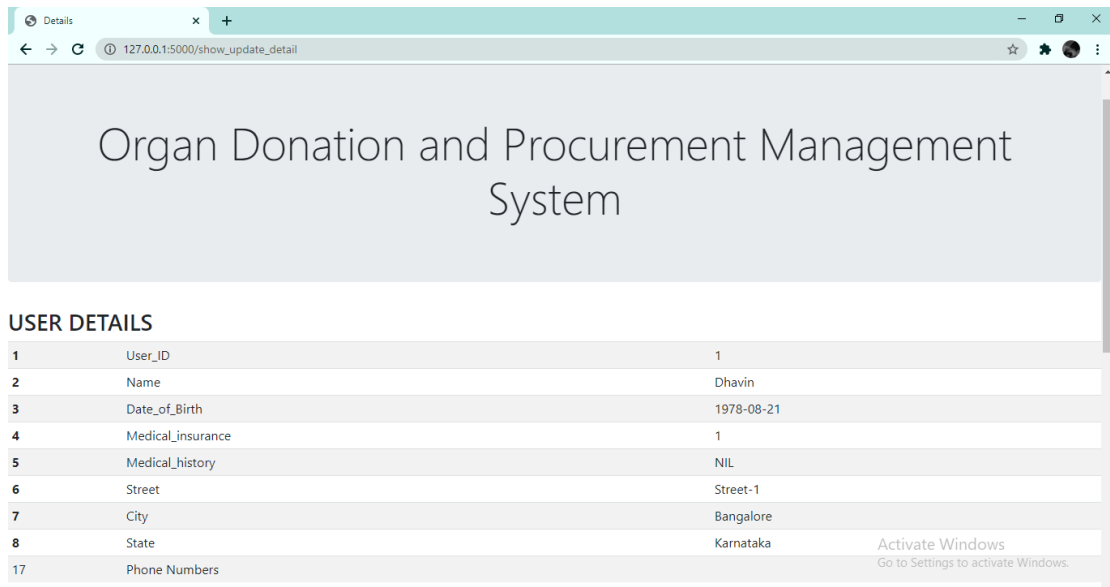
**Figure 5.6: Admin Add Menu**



**Figure 5.7: Admin Update Menu**



**Figure 5.8: Admin Remove Menu**



**Figure 5.9: User Details (a).**

Details x +

127.0.0.1:5000/show\_update\_detail

17 Phone Numbers

### PATIENT DETAILS

1	Patient_ID	1
2	organ_req	Heart
3	reason_of_procurement	Reason-1
4	Doctor_name	Dr.Elon Musk

### DONOR DETAILS

1	Donor_ID	1
2	organ_donated	Heart
3	reason_of_donation	Reason-1
4	Organization_name	BNM Hospital

### TRANSACTION DETAILS

1	Patient_ID	1
2	Donor_ID	1
3	Organ_ID	1
4	Date_of_transaction	1978-08-21
5	Status	1

Figure 5.10: User Details (b).

Search Student Page x +

127.0.0.1:5000/search\_User\_details

HOME Hi admin! Logout

## Organ Donation and Procurement Management System

SEARCH:

User_ID	Name	Date_of_Birth	Medical_insurance	Medical_history	Street	City	State
1	Dhavin	1978-08-21	1	NIL	Street-1	Bangalore	Karnataka
2	Virat	1978-08-25	0	NIL	Street-2	Bangalore	Karnataka
3	Rahul	1934-08-21	1	NIL	Street-1	Bangalore	Karnataka
4	Rohit	1978-08-25	1	NIL	Street-3	Mumbai	Maharashtra
5	Gill	1978-08-26	0	NIL	Street-4	Chennai	Tamil Nadu

Figure 5.11: Search User

Search Student Page x +

127.0.0.1:5000/search\_Transaction

HOME Hi admin! Logout

### Organ Donation and Procurement Management System

SEARCH:

Patient_ID	Organ_ID	Donor_ID	Date_of_transaction	Status
1	1	1	1978-08-21	1
2	2	2	1978-08-21	1

Activate Windows  
Go to Settings to activate Windows.

**Figure 5.12: Search Transaction**

Transaction x +

127.0.0.1:5000/add\_User\_page

### Organ Procurement and Donation Management System

User\_ID

Name

Date\_of\_Birth

Medical\_insurance

Medical\_history

Street

City

State

Add User

Activate Windows  
Go to Settings to activate Windows.

**Figure 5.13: Add User**

Transaction x +

127.0.0.1:5000/add\_User\_phone\_no\_page

HOME Hi admin! Logout

# Organ Procurement and Donation Management System

User\_ID

phone\_no

Add User\_phone\_no

Activate Windows  
Go to Settings to activate Windows.

**Figure 5.14: Add User\_phone**

Transaction x +

127.0.0.1:5000/add\_Donor\_page

HOME Hi admin! Logout

# Organ Procurement and Donation Management System

Donor\_ID

organ\_donated

reason\_of\_donation

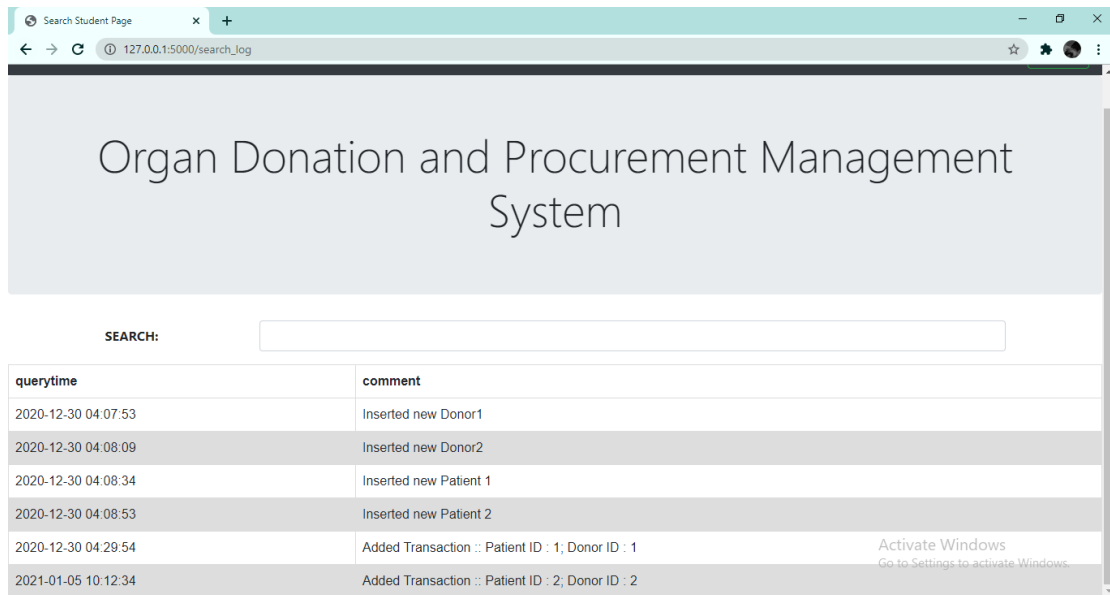
Organization\_ID

User\_ID

Add Donor

Activate Windows  
Go to Settings to activate Windows.

**Figure 5.15: Add Donor**



Search Student Page

127.0.0.1:5000/search\_log

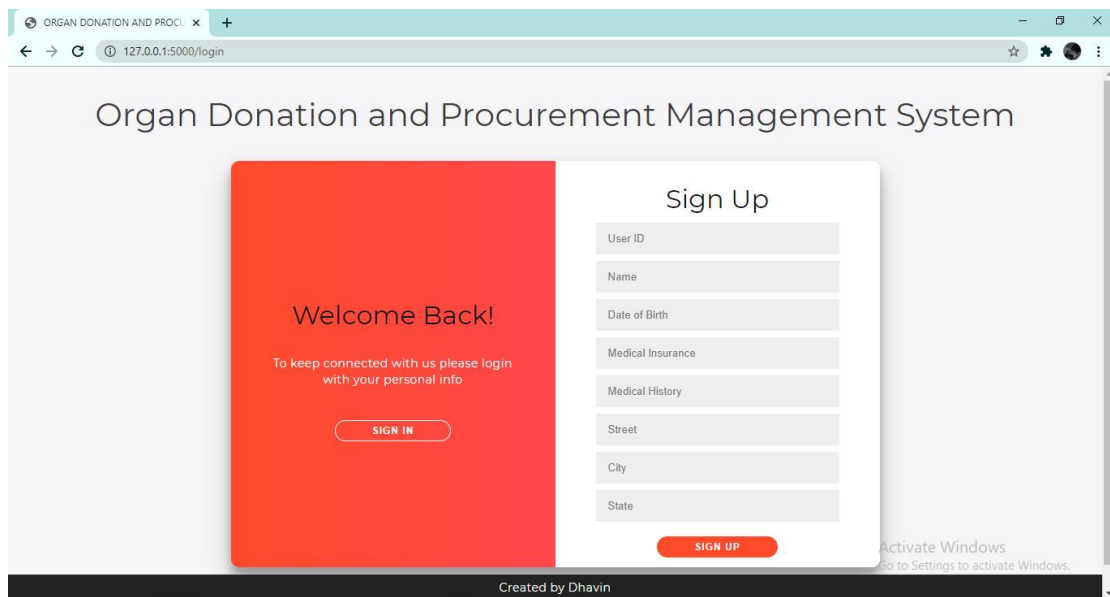
### Organ Donation and Procurement Management System

SEARCH:

querytime	comment
2020-12-30 04:07:53	Inserted new Donor1
2020-12-30 04:08:09	Inserted new Donor2
2020-12-30 04:08:34	Inserted new Patient 1
2020-12-30 04:08:53	Inserted new Patient 2
2020-12-30 04:29:54	Added Transaction :: Patient ID : 1; Donor ID : 1
2021-01-05 10:12:34	Added Transaction :: Patient ID : 2; Donor ID : 2

Activate Windows  
Go to Settings to activate Windows.

**Figure 5.16: Log Details**



ORGAN DONATION AND PROCI

127.0.0.1:5000/login

### Organ Donation and Procurement Management System

#### Welcome Back!

To keep connected with us please login with your personal info

[SIGN IN](#)

#### Sign Up

User ID

Name

Date of Birth

Medical Insurance

Medical History

Street

City

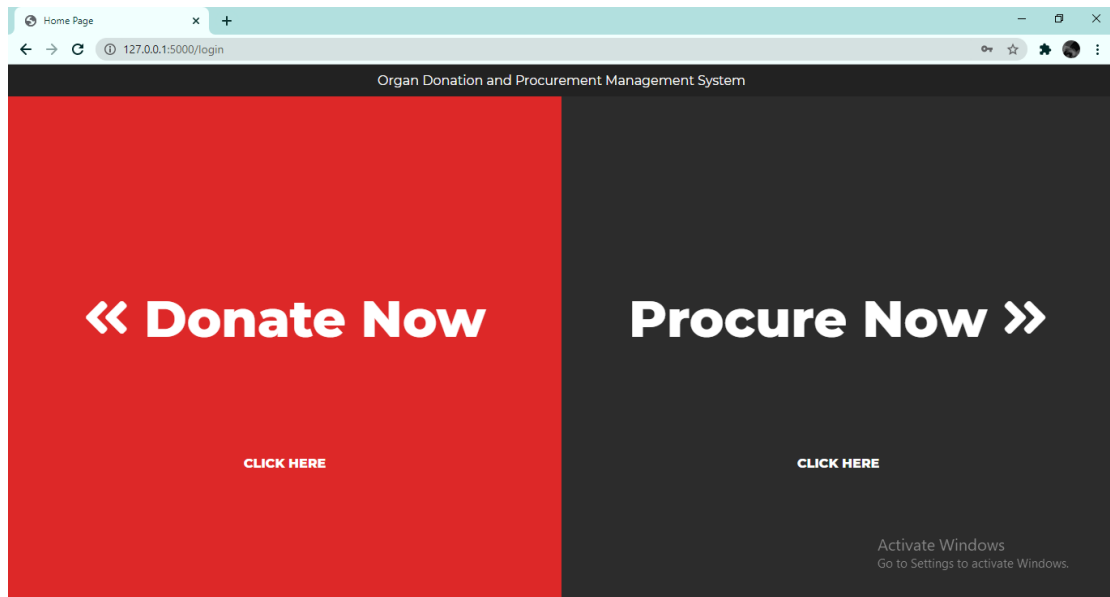
State

[SIGN UP](#)

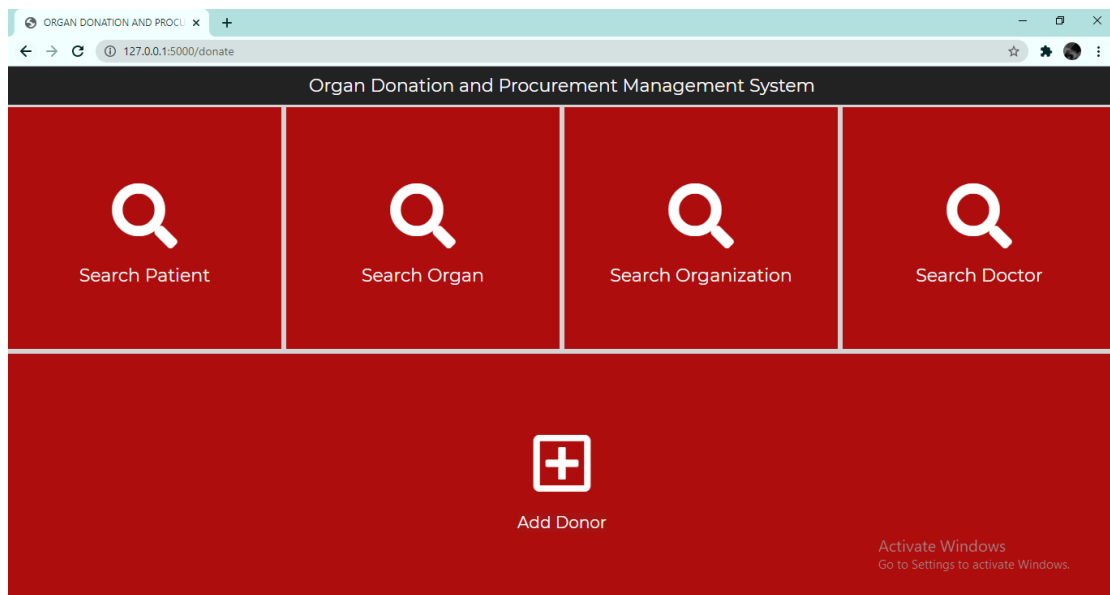
Activate Windows  
Go to Settings to activate Windows.

Created by Dhavin

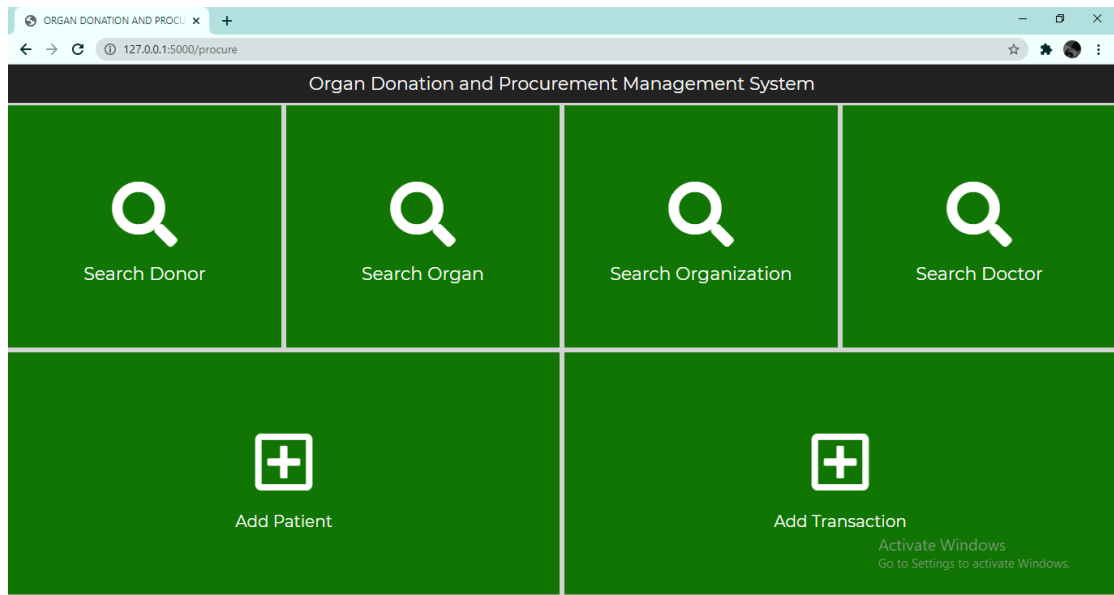
**Figure 5.17: User Login**



**Figure 5.18: User Home**



**Figure 5.19: Donate Page.**



**Figure 5.20: Procure Page.**



## CONCLUSION

Thus we have successfully implemented organ donation and procurement database management which helps us in centralizing the data used for managing the tasks performed in an organ donation we have successfully implemented various functionalities of MySQL and created the fully functional database management system for organ donation and procurement.

Organ Donation and Procurement Management System project is designed to meet the requirements of Donor and Patients. It has been developed in Python, HTML, and CSS keeping in mind the specification of the system. For designing the system, HTML, CSS, JavaScript is being used as the front end.

Environment. Overall the project teaches us the essential skills like:

- Understanding the database handling and query processing
- Implement, analyze and evaluate the project developed for an application
- Demonstrate the working of different concepts of DBMS

## **FUTURE ENHANCEMENT**

- It is not possible to develop a application that makes all the requirements of the user. User requirements keep on changing.so, Some of the future enhancements that can be done to this system are:
- As the technology emerges, it is possible to upgrade the application and can be adaptable to desired environment.
- We can also applicable this to Oracle and MySQL instead of SQL Server.
- Based on the future security issues, security can be improved using encryption and decryption techniques.
- We can also provide administrative tools like Backup, Replication and Linked Server.