

PROJECT REPORT
ON

iDONATE

Submitted By

MANJUSHA K (CEC19CS031)

RESHMI MOHAN (CEC19CS043)

SREELEKSHMI C J (CEC19CS053)

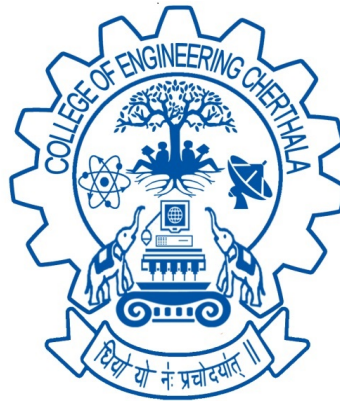
DEVIKA M NATH (CEC17CS063)

under the esteemed guidance of

Mrs. VIMAL VINOD

Assistant Professor

Department Of Computer Engineering



MAY 2023

**DEPARTMENT OF COMPUTER ENGINEERING
COLLEGE OF ENGINEERING, PALLIPPURAM P O, CHERTHALA,
ALAPPUZHA PIN: 688541,
PHONE: 0478 2553416, FAX: 0478 2552714
<http://www.cectl.ac.in>**

A PROJECT REPORT ON

iDONATE

Submitted By

MANJUSHA K (CEC19CS033)

RESHMI MOHAN (CEC19CS043)

SREELEKSHMI C J (CEC19CS053)

DEVIKA M NATH (CEC19CS063)

under the esteemed guidance of

Mrs. VIMAL VINOD

In partial fulfillment of the requirements for the award of the degree

of

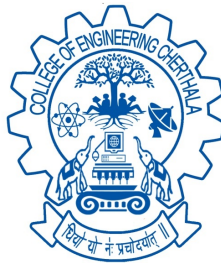
Bachelor of Technology

in

Computer Science and Engineering

of

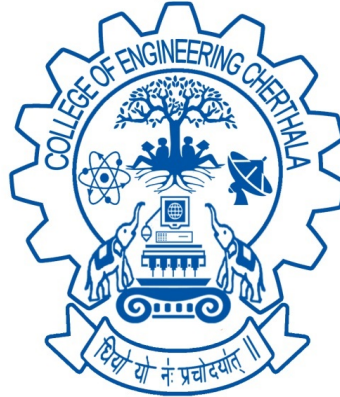
APJ Abdul Kalam Technological University



MAY 2023

**DEPARTMENT OF COMPUTER ENGINEERING
COLLEGE OF ENGINEERING, PALLIPPURAM P O, CHERTHALA,
ALAPPUZHA PIN: 688541,
PHONE: 0478 2553416, FAX: 0478 2552714
<http://www.cectl.ac.in>**

DEPARTMENT OF COMPUTER ENGINEERING
COLLEGE OF ENGINEERING CHERTHALA
ALAPPUZHA-688541



C E R T I F I C A T E

This is to certify that, the project report titled **iDONATE**, is a bonafide record of the **CSD416 Project** presented by **MANJUSHA K (CEC19CS033)**, **RESHMI MOHAN (CEC19CS043)**, **SREELEKSHMI C J (CEC19CS053)**, **DEVIKA M NATH (CEC17CS063)** Eighth Semester B. Tech. Computer Science & Engineering students, under our guidance and supervision, in partial fulfillment of the requirements for the award of the degree, **B. Tech. Computer Science & Engineering** of **APJ Abdul Kalam Technological University**.

Guide

Co-ordinator

HoD

Mrs.Vimal Vinod

Assistant Professor

Dept Of Computer Engg

Mr. Muhammed Ilyas

Assistant Professor

Dept Of Computer Engg

Dr. Priya S

Professor

Dept Of Computer Engg

ACKNOWLEDGEMENT

This work would not have been possible without the support of many people. First and the foremost, we give thanks to Almighty God who gave us the inner strength, resource and ability to complete our project successfully.

We would like to thank **Dr. Jaya V L**, our Principal, who has provided with the best facilities and atmosphere for the project completion and presentation. We would also like to thank our HoD **Dr. Priya S** (Professor, Department Of Computer Engineering), our project coordinator **Mr. Muhammed Ilyas H** (Assistant Professor, Department Of Computer Engineering), and our guide **Mrs. Vimal Vinod** (Assistant Professor, Department Of Computer Engineering) for the help extended and also for the encouragement and support given to us while doing the project.

We would like to thank my dear friends for extending their cooperation and encouragement throughout the project work, without which we would never have completed the project this well. Thank you all for your love and also for being very understanding.

ABSTRACT

Without a supply of blood, health services could not meet their clinical needs. Similarly, organs for transplantation save and transform lives. Donations are acts of generosity that are traditionally seen as altruistic, and accordingly, interventions to recruit and retain blood and organ donors have focused on altruism. There is a disparity exists between the supply and demand of the organs and blood. There doesnt exists a website showing the availability of both blood and organs. The aim of this project is to build a web application which allows people to get a chance to know about the availability of blood and organs in each hospitals and hence can request for the same when need occur.The project will include minimum manual work and maximum optimization, abstraction, and security.

This web application mainly focuses to connect Hospitals, Organizations, Donors and Receivers. This application can be accessed throughout the Hospitals with respective logins provided. This system can be used as an efficient tool for the Hospitals for the blood and organ transplantation and to manage the patients information and also to pass the information regarding the transplantation. After login, the Donors, Receivers, Hospitals and Organizations be able to view the dashboard,from there they can search and update correspondingly. This system could also be used as a central repository that controls/possesses the entire data of patients details in hospitals. This will also helps to fastly crosscheck the availability of blood and organs.

Keywords: Optimization, Abstraction, Security, Transplantation

Contents

1	INTRODUCTION	1
2	PROBLEM STATEMENT	2
2.1	Problem Statement	2
2.2	Objective	2
3	LITERATURE SURVEY	3
3.1	CASE STUDY 1	3
3.2	CASE STUDY 2	4
3.3	CASE STUDY 3	4
3.4	CASE STUDY 4	6
3.5	CASE STUDY 5	6
3.6	Case Study Analysis	6
4	PROPOSED SYSTEM	8
4.1	Solution	8
4.2	Feasibility Study	8
4.2.1	Technical Feasibility	9
4.2.2	Social Feasibility	9
4.2.3	Schedule Feasibility	9
4.2.4	Operational Feasibility	10
4.2.5	Economic Feasibility	10

5	SOFTWARE REQUIREMENT SPECIFICATION	11
5.1	Overall Description	11
5.1.1	Product Perspective	11
5.1.2	Product Functions	11
5.2	User Classes and Characteristics	12
5.2.1	Operating Environment	12
5.2.2	Design and Implementation Constraints	13
5.2.3	Assumptions and Dependencies	13
5.3	External Interface Requirements	13
5.3.1	User Interfaces	13
5.3.2	Hardware Interfaces	14
5.3.3	Software Interfaces	14
5.3.4	Communications Interfaces	14
5.4	System Features	14
5.4.1	Admin User Module	14
5.4.2	Donor User Module	15
5.4.3	Receiver User Module	16
5.4.4	Organization User Module	17
5.5	Other Nonfunctional Requirements	18
5.5.1	Security Requirements	18
5.5.2	Software Quality Attributes	18
6	SYSTEM DESIGNS	19
6.1	Design	19
6.2	MODULES	20
6.2.1	ADMIN	20
6.2.2	DONOR	20
6.2.3	RECEIVER	22
6.2.4	HOSPITAL	23
6.2.5	ORGANIZATION	24

6.3	USE-CASE DIAGRAM	25
6.4	SEQUENCE DIAGRAM	29
6.5	ENTITY RELATION DIAGRAM	34
6.6	Data Flow Diagram	34
6.6.1	Level 0	35
6.6.2	Level 1(Admin)	36
6.6.3	Level 2 (Admin)	37
6.6.4	Level 1(Donor)	38
6.6.5	Level 2(Donor)	39
6.6.6	Level 1(Receiver)	40
6.6.7	Level 2(Receiver)	40
6.6.8	Level 1(Hospital)	41
6.6.9	Level 2(Hospital)	41
6.6.10	Level 1(Organization)	42
6.6.11	Level 2(Organization)	42
7	SOFTWARE AND HARDWARE REQUIREMENT	43
7.1	Software Requirements	43
7.2	Hardware Requirements	43
7.2.1	Computer	43
7.2.2	Smartphone	43
8	IMPLEMENTATION	44
8.1	Coding Environment Used	44
8.2	Admin Module	44
8.3	Donor Module	45
8.4	Receiver Module	45
8.5	Organization Module	45
8.6	Hospital Module	45

9	RESULT & ANALYSIS	49
9.1	Screenshots	49
10	CONCLUSION & FUTURE SCOPE	56
	REFERENCES	57

List of Figures

3.1	Case Study Analysis	7
6.1	Use Case diagram	26
6.2	Admin Sequence diagram	29
6.3	Donor Sequence diagram	30
6.4	Receiver Sequence diagram	31
6.5	Hospital Sequence diagram	32
6.6	Organization Sequence diagram	33
6.7	Entity relation diagram	34
6.8	Level 0 DFD of iDonate	35
6.9	Level 1 DFD of Admin	36
6.10	Level 2 DFD of Admin	37
6.11	Level 1 DFD of Donor	38
6.12	Level 2 DFD of Donor	39
6.13	Level 1 DFD of receiver	40
6.14	Level 2 DFD of Receiver	40
6.15	Level 1 DFD of Hospital	41
6.16	Level 2 DFD of Hospital	41
6.17	Level 1 DFD of Organization	42
6.18	Level 2 DFD of Organization	42
8.1	views.py	46
8.2	views.py	46
8.3	models.py	47

8.4	admin.py	47
8.5	urls.py	48
8.6	Donor User Accepting or Rejecting code in Admin Module	48
9.1	Home page	49
9.2	Home page- about us	50
9.3	Home page - contact details	50
9.4	Sign up page	51
9.5	Login page	51
9.6	ADMIN DASHBOARD	52
9.7	ADMIN PERMISSION	52
9.8	DONOR DASHBOARD	53
9.9	DONOR-SEARCH	53
9.10	DONOR-HOSPITAL DETAILS	54
9.11	QUICK ACCESS	54
9.12	QUICK ACCESS	55

Chapter 1

INTRODUCTION

Many hospitals provide the way to donate organs and blood directly and can supply that to the people who are in need of that. Now-a-days many cases are being reported to death because of the unavailability of blood. Through this we are proposed to have a platform which will give people to get a chance to know about the availability of blood and organs in each hospitals and hence can request for the same when need occur. The need for blood and organ is great as it is life, as there is no replacement for human blood and organ. Every day blood and organ is required in hospitals and emergency treatment facilities for patients with Cancer, Thalassemia and other diseases, for organ transplant recipients, and to help save the lives of accident/trauma victims. With a growing population and advances in medical treatments and procedures requiring blood transfusions, the demand for blood and organ continue to increase. In India many people are losing their lives every day in emergency situations because we are suffering from lack of blood and organ in blood and Organ Banks, and they do not receive the blood and organ timely. There is a disparity exists between the supply and demand of the organs and blood. There doesn't exists a website showing the availability of both blood and organs. The aim of this project is to build a web application which allows people to get a chance to know about the availability of blood and organs in each hospitals and hence can request for the same when need occur.

Chapter 2

PROBLEM STATEMENT

2.1 Problem Statement

Emergency patients, who are in need of blood and organ, usually request through advertising on televisions or social media, with the series of advert placement of donation of blood and organ the patient may still not get the required amount of blood and compactable organs needed at that particular time. There is no platform to co-ordinate these both blood and organ donation activities effectively.

2.2 Objective

The objective is to make a web app to connect Hospitals, Organizations, Donors and Receivers and allows them to know about the availability of blood and organs in each hospitals and hence can request for the same when need occur.

- It eases the complex process which involves finding a donor whose blood group is compatible with the blood group of the patient.
- The Hospital managements or the authorized ones have to update, delete, manage these data.
- The security of user data is also uncertain here.
- As the registration processes are done as a whole, there exists a fear of fake or illegal malpractice.

Chapter 3

LITERATURE SURVEY

3.1 CASE STUDY 1

NHS organ donation and NHS blood and transplants.

These are two websites for organ donation and blood donation respectively under the same organisation. They manage blood and platelet donation, and organ, stem cell and tissue donation and transplantation.

NHS Blood and Transplant (NHSBT) is a joint England and Wales Special Health Authority that provides a blood and transplant service to the National Health Service—supplying blood to hospitals in England, and tissues and solid organs to hospitals across the United Kingdom. Each year, donors give approximately two million donations of blood and 3,500 organs—saving and transforming countless lives. Safeguarding the blood supply and increasing the number of donated organs involves collecting, testing, processing, storing and delivering blood, plasma, and tissue to every NHS Trust in England. NHSBT also matches, allocates, audits, and analyzes organ donations across the whole of the UK.

“NHSBT was embarking on its most complex transformation program ever, initially focusing on the Organ Donation and Transplantation (ODT) area of its business”. “It needed to modernize a significant percentage of its core systems, platforms, and architecture along with re-aligning the infrastructure to more modern cloud-based technologies. The impact on the current business and practices could not be underestimated across the organization—we were anticipating changes in how we work and how the system worked.”

3.2 CASE STUDY 2

HRSA - organdonar.gov

An organ donation management website under Health resource and service administration. Organs, Corneas, Tissues, Hands and Face, Blood Stem Cells, Cord Blood, and Bone Marrow, Blood and transplants are donated using these sites. There also exist android apps like Organ Donation (Life after death), Friends to support, Blood donation etc under certain hospitals. But these apps won't allow the needed ones to register in these apps. Only the donors can register and the hospital authority will provide the organs to the patients consulting there hospitals only.

And also no app provides both blood and organ donation management scheme together. Either blood nor organs are only considered. Organ donar is a website under health resource and service administration but this site also provides only the donar to register.

More than 120,000 men, women, and children across the U.S. are waiting for a life-saving organ transplant, and 22 people die every day because a matching organ can't be found in time. The Division of Transplantation (DoT), HRSA/HHS chose Crosby to help promote registration nationwide for organ, eye, and tissue donation.

Research showed the vast majority of people in the U.S. support donation, but nearly half of eligible adults hadn't signed up. Our job was to reach out to these "passive positives" and spread the message about the importance and ease of registering. Just one donor can save up to 8 lives.

3.3 CASE STUDY 3

e-Rakt Kosh

e-RaktKosh: A Centralized Blood Bank Management System eRaktKosh was Inaugurated on 7th April 2016 by Hon'ble Minister of Health and Family Welfare, Sh. J P Nadda. e-Rakt Kosh enforces Drug Cosmetic Act, National blood policy standards and guidelines ensuring proper collection donation, effective management and monitoring the quality and quantity of the donated blood. Considering the national roll out, e-Rakt Kosh has been developed with modular and scalable approach with configurable rule based architecture allowing customization to easily incorpo-

rate specific requirements from nationwide stakeholders.

e-Rakt Kosh has six major components for management of the blood donation life cycle:

- The bio metric Donor Management System for identifying, tracking and blocking donors based on donor's health, donation history etc.
- It provides features such as blood grouping, TTI screening, antibody screening, component preparation etc. as per the defined processes and rules.
- A centralized Blood Inventory Management System for keeping track of the blood stock across numerous blood banks.
- Bio-Medical Waste Management System for disposal of discarded blood and other waste generated during this process.
- Generation of rare blood group donor registries and the generation of regular repeat donors
- Alert and Notification System

The objectives of e-Rakt Kosh

- Safe and Adequate Blood Supplies
- Reduced Turnaround Time
- Networking of Blood Banks
- Donor Repository
- Salient Features
- Web Based Application
- Aadhar Linkage
- Decision Support
- Enforces Guidelines

- Dashboard
- Statutory Reports

3.4 CASE STUDY 4

www.notto.nic.in

National Organ and Tissue Transplant Organization (NOTTO) is a National level organization set up under Directorate General of Health Services, Ministry of India. A website by the name www.notto.nic.in has been hosted where information with regards to the organ transplantation can be obtained. An online system through website is being developed for establishing network for Removal and Storage of Organs and Tissues from deceased donors and their allocation and distribution in a transparent manner. A computerized system of State/Regional and National Registry of donors and recipients is also going to be put in place.

3.5 CASE STUDY 5

U-Blood

Features of U-Blood:

- Geo-Search With the geo-search feature, finding blood donors has become easier than ever. Enter your location and you will be shown the donors available in the closest proximity.
- Real-time Connect: No delays in receiving blood anymore. Connect with donors and recipients in real-time.
- Notifications: Get updates on blood requests so that you are informed the moment a donor is available or a request is made.

3.6 Case Study Analysis

EXISTING SITES AND APPS	ADVANTAGES	DISADVANTAGES
NHS organ donation and NHS blood and transplants	They manage blood and platelet donation, and organ, stem cell and tissue donation and transplantation.	Different websites for organ and blood donation.
HRSA - organdonar.gov	Organs, Corneas, Tissues, Hands and Face, Blood Stem Cells, Cord Blood, and Bone Marrow, Blood and transplants are donated using these sites.	Only the donors can register and the hospital authority will provide the organs to the patients consulting there hospitals only.
e-Rakt Kosh	Ensures the proper collection donation, effective management and monitoring the quality and quantity of the donated blood.	High maintenance cost and not user-friendly.
www.notto.nic.in	Reducing turnaround time, networking of blood banks and donor repository.	Backlog of entries- Inadequate staffing results in one person only entering donor features while the entry regarding issue of the blood is not done for weeks. This also results in mismatch between online data and in-house availability of blood products as they do not match in real time.
U-Blood	No delays in receiving blood since the donors available in the closest proximity finds using location access.	Security issues and not efficient

Fig. 3.1: Case Study Analysis

Chapter 4

PROPOSED SYSTEM

4.1 Solution

The aim is to develop a web application with advanced features. The main purpose is to connect hospitals, organizations, donars and recievers and allows them to get a chance to know about the availability of blood and organs in each hospitals and hence can request for the same when need occur. The main functions are as follows:

- Patients's information is maintained in the database so that the data will be organized and, can be accessed by hospital management.
- Gives more security to data, ensures data accuracy.
- Reduces paper work and save a lot of time.
- Since the availability of blood and organs are shown in a single website, it will be easy to connect the needed ones and helps to save their lives.
- Since the data is stored in a database, the chance of data duplication is probably less.
- The system will be cost effective.

4.2 Feasibility Study

The main objective of this study is to determine whether the proposed system is feasible or not. Mainly there are three types of feasibility study to which the proposed system is subjected as

described below:

Five key considerations are involved in this feasibility

- Technical Feasibility
- Social Feasibility
- Schedule Feasibility
- Operational Feasibility
- Economic Feasibility

The proposed system must be evaluated from a technical viewpoint first, and if technically feasible, their impact on the organization must be assessed. If compatible, the operational system can be devised. Then those must be tested for economic feasibility.

4.2.1 Technical Feasibility

The technologies required for the development is identified. Since, both the hardware and software requirements are satisfied, it is technically feasible.

4.2.2 Social Feasibility

The proposed project will be socially feasible. The social feasibility determines whether the project would be accepted by the people. This assumption would in general examine the probability that the project would have to be accepted by the group of people that are directly affected by the proposed system

4.2.3 Schedule Feasibility

The primary analysis depicts that the project can be completed by the schedule. Thus the project is feasible.

4.2.4 Operational Feasibility

The proposed project is beneficial because it helps in making faster and reliable decisions. So, users will be encouraged to use it, and it is expected to serve the user's need, which means helping the user in making decisions.

4.2.5 Economic Feasibility

The system will be developed at reasonable cost with the available hardware, software and manpower. So, its benefits outweigh the cost. So, it is economically feasible.

Chapter 5

SOFTWARE REQUIREMENT SPECIFICATION

5.1 Overall Description

The purpose of this document is to present a detailed description of iDonate a platform which connects hospitals, organizations, donors and receivers. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. It aims at digitizing the activities of blood and organ donation. An efficient system for the hospitals for patients data management. Admins can manage the users database.

5.1.1 Product Perspective

To provide the user friendly system to all the people and institutions who are in need of medical help (blood and organs). It is our one step towards helping those people to face the challenging world of internet today. To provide them the facility of technology through this they have the chance to overcome this disability.

5.1.2 Product Functions

This project has many functions. They are listed below:

- This system will provide 5 modules: Admin, Donor, Receiver, Organization, Hospital.
- Organizations may be any institutions whom act as a middle factor between the organ/blood seeker/donor.

- Hospital can have the access on the donated or received blood/organ.
- Donor can donate blood/organs in the hospitals by applying preferred date in the hospitals through this app.
- Receiver can order/apply for the blood/organ through this app.
- The admin will check the validity and after done with the matching, the admin can notify the Receiver about the matching and the hospital details.
- This will help the organ/blood bank system more efficient and beneficial.

5.2 User Classes and Characteristics

- 2.3.1. Login: This is the very first page and will ask user to enter login credentials. After receiving user name it will prompt again for password. After receiving all of the details from organization, it will encrypt and check the validity of the details entered by organization. If valid, then the particular organization will be redirected to dashboard else will be sent back to login page.
- 2.3.2. Dashboard: After successful login, that particular organization will be redirected to this page and this is the main page from where organization/donor/receiver/hospital can perform all the activities like Add details, search, order blood/organ (if receiver or the hospital managements were the users), donate blood/organs (if institutions or the donor were the users) etc.
- 2.3.3. Search: This module is used to search other accounts. 2.3.4. Notification: This page will store all of the notifications regarding the results of availability of blood/organs.

5.2.1 Operating Environment

The application will be operating on an environment with a processor above intel core i3 and a minimum of 2GB RAM memory. For the most part, you'll get faster CPU performance from the Core i5 parts over the Core i3. Some Core i5 processors are dual-core and some are quad-core. Most of the time, a true quad-core CPU will perform better than a dual-core processor.

- Computer with any web browser.
- Smartphone with any Web browser.

5.2.2 Design and Implementation Constraints

This application is constrained by the data provided by the website and the verification at the block level. If the data provided by the site is not accurate and the block officials don't pay attention or do fraud then the system will have problem showing the stats. The internet connection is also a constraint for the application. Since the application fetches data from the database over the Internet. It is crucial that there is an Internet connection for the application to function. The application will be constrained by the capacity of the database. The database may be forced to queue incoming requests and therefore increase the time it takes to fetch data.

5.2.3 Assumptions and Dependencies

One assumption about the product is that the data on the database is accurate and the chances of fraud being done at block level are minimal. The user will have enough space in his/her phone to store the data fetched from the database as the application will store the data in memory so that in case of zero updation or slow network it can show the last successfully retrieved the data.

- Computer or Smartphone with any browser.
- A good internet connection.

5.3 External Interface Requirements

5.3.1 User Interfaces

There will be a user interface for the help of the user. This interface will be responsible for the communication. It is made possible by implementing the modular concept in which each module deal with its own functionalities.

UI will be mainly styled using css. Opening of the application takes the user to landing page. From there the user can login as an administrator or as a donor or organization or receiver or

hospital management and they will be logged into to the respective dashboard.

5.3.2 Hardware Interfaces

Basic Requirement: The primary requirement is a smartphone or a computer. The smart phone or the computer must have a stable internet connection to enable the smooth working of the application.

- Processor: Above intel core i3
- Memory: 2GB RAM(minimum)

5.3.3 Software Interfaces

The web app opened in a browser is connected to MongoDB database. The data to be stored includes information about students and alumni.

- Operating System: Windows 7/Windows 8/Windows 10
- Frontend: HTML, CSS, JavaScript
- Backend: Python-Django

5.3.4 Communications Interfaces

Sqlite3 server is used for storing the data.

5.4 System Features

5.4.1 Admin User Module

Description and Priority

Admin can login to admin dashboard.

Stimulus/Response Sequences

Admin can login into their profile with respective email and password. The Admin ID and password will be validated. After successful validation, the admin gains access to his/her profile. After login admin can update his/her profile details. As a response admin will get a message saying his/her profile is updated. Admin can add Donation or Received details and also send Mail to people into the registered email address. Admin can deactivate any user's profile whenever any disagreements happens. Admin can view all the updations happened in every profile except encrypted one.

Functional Requirements

- Admin login - The user is able to log into the web application.
- Profile Updation - The user is able to update his/her profile.
- Profile encryption - Admin can deactivate any of the user's profile whenever any disagreements happens.
- Database Management - The Admin is able to manage the database.

5.4.2 Donor User Module**Description and Priority**

Donor can login to Donor's profile and can view the dashboard.

Stimulus/Response Sequences

Donor Can view the status of hospital's donating system. He/She can Contact with the Hospitals to donate an organ, Whenever any request on donating that particular organ/blood came she/he can do it from the hospital. He/She can delete or update their own profiles. Donor can search for other users. They can donate blood/organ accordingly and can notify in the application.

Functional Requirements

- Donor login - The Donor is able to log into the web application.
- Can view the status of hospital's donating system
- He/She can Contact with the Hospitals to donate an organ.
- Whenever any request on donating that particular organ/blood came she/he can do it from the hospital.
- He/She can delete or update their own profiles.
- Donor can search for other users.
- Can donate blood/organ accordingly and can notify in the application

5.4.3 Receiver User Module**Description and Priority**

Receiver can login to receiver's profile and can view the dashboard.

Stimulus/Response Sequences

Receiver Can view the status of hospital's donating system. He/She can Contact with the Hospitals to donate an organ, Whenever any request on donating that particular organ/blood came she/he can do it from the hospital. He/She can delete or update their own profiles. Receiver can search for other users. They can donate blood/organ accordingly and can notify in the application.

Functional Requirements

- Can view the status of Hospital's system.
- He/She can Contact with the Hospitals in need of an organ/blood.
- Whenever he/she needs to get transplanted, He/she can check the hospital's status on availability of that organ and the further procedures will be done by the hospitals.

- He/She can delete or update their own profiles.
- Receiver can search for other users.
- Receiver can purchase blood/organs

5.4.4 Organization User Module

Description and Priority

An organization can login to the respective profile.

Stimulus/Response Sequences

Organizations Can view the status of hospital's system. He/She can Contact with the Hospitals to donate an organ, Whenever any request on donating that particular organ/blood came she/he can do it from the hospital. He/She can delete or update their own profiles. Organizations can search for other users. They can donate blood/organ accordingly and can notify in the application.

Functional Requirements

- Can view the status and matching.
- Organization can Contact with the Hospitals to donate an organ.
- Whenever any request on donating that particular organ/blood came Organization can do it from the hospital.
- Organization can delete or update their own profiles. Donor can search for other users.
- Can donate blood/organ accordingly and can notify in the application

5.5 Other Nonfunctional Requirements

Performance Requirements

This web application requires a decent web browser preferably Google Chrome or Mozilla Firefox.

5.5.1 Security Requirements

Since these apps store the information in the remote server, data won't be lost if the client crashes. Also the app will be regularly updated for fixing bugs and adding new features. In case of problems the user can contact the app support team for any help.

5.5.2 Software Quality Attributes

iDonate System provides a simple and easy to use UI. Users can access the web application from anywhere and anytime using a web browser.

Chapter 6

SYSTEM DESIGNS

6.1 Design

Design of the system includes mainly two steps:

- System Design
- Detailed Design

In System design a structural framework for the entire system is created. It is done in such a way that related part come under particular groups. Thus after the system design, a network of different groups is obtained. It is the high-level strategy for solving the problem and building a solution. It includes the decision about the organization of the system into subsystems, the allocation of subsystems to hardware and software components, and major conceptual and policy decisions that form the framework for the detailed design.

In detailed design, each group is studied in detail and the internal operations are decided. Based on this, the data structures and the programming language to be used are decided. Apart from detailed design, the system design can be grouped into physical design and structural design. The physical design maps out the details of the physical system and plans the system implementation and specifies the hardware and software requirements.

Structured design is an attempt to minimize the complexity and make a problem manageable by subdividing into smaller segments, which is called modularization or decomposition. In this way structuring minimizes intuitive reasoning and promotes maintainable provable of systems. The structured design partitions a program into small, independent modules. They are arranged in a

hierarchy that approximates a model of the business are and is organized in a top-down manner. Logical design proceeds in a top-down manner. General features, such as reports and inputs are identified first. Then each is studied individually and in more detail. Hence the structured design is an attempt to minimize the complexity and make a problem.

6.2 MODULES

6.2.1 ADMIN

Input : Username & Password

Output : Logged in to the Admin dashboard.

1. start
2. Logged in using username & password.
3. Send this data to the server.
4. If an error occurred, returns back to the homepage.
5. If success, logged in to the admin dashboard.
6. Can view all profile & can accept or reject the request.
7. Stop

6.2.2 DONOR

LOGIN

Input : Username & Password

Output : Logged in to the donor dashboard.

1. Logged in using username & password.
2. Send this data to the server.

3. If an error occurred, returns back to the homepage.
4. If success, logged in to the donor dashboard.
5. Can view their profile, identity and contact details.
6. Donor dashboard shows the Notifications and sent requests.
7. Donor can send request to any of the receiver, hospitals and organizations.
8. Stop

SIGN UP

Input : Username, Mail id & Password

Output : Signin to the Donor registration.

1. Donor can sign Up using username, Mail id & password.
2. Send this data to the server.
3. Sign In to the donor registration.
4. Enter valid donor details.
5. If admin reject the request, returns back to the homepage.
6. If admin accepts the request, can login to the donor dashboard.
7. Can view their profile, identity and contact details.
8. Donor dashboard shows the Notifications and sent requests.
9. Donor can send and accept request to any of the receiver, hospitals and organizations.
10. Stop

6.2.3 RECEIVER

LOGIN

Input : Username & Password

Output : Logged in to the Receiver dashboard.

1. Logged in using username & password.
2. Send this data to the server.
3. If an error occurred, returns back to the homepage.
4. If success, logged in to the receiver dashboard.
5. Can view their profile, identity and contact details.
6. Receiver dashboard shows the Notifications and sent requests.
7. Receiver can send and accept request to any of the donor, hospitals and organizations.
8. Stop

SIGN UP

Input : Username, Mail id & Password

Output : Sign In to the receiver registration.

1. Receiver can sign Up using username, Mail id& password.
2. Send this data to the server.
3. Sign In to the receiver registration.
4. Enter valid receiver details.
5. If admin reject the request, returns back to the homepage.

6. If admin accepts the request, can login to the receiver dashboard.
7. Can view their profile, identity and contact details.
8. Receiver dashboard shows the Notifications and sent requests.
9. Receiver can send and accept request to any of the donor, hospitals and organizations.
10. Stop

6.2.4 HOSPITAL

LOGIN

Input : Username & Password

Output : Logged in to the Hospital dashboard.

1. Logged in using username, Mail id & password.
2. Send this data to the server.
3. If an error occurred, returns back to the homepage.
4. If success, logged in to the hospital dashboard.
5. Can view their profile, identity and contact details.
6. Hospital dashboard shows the Notifications and sent requests.
7. Hospital can send and accept request to any of the receivers, donors and organizations.
8. Stop

SIGN UP

Input : Username, Mail id & Password

Output : Sign In to the Hospital registration.

1. Hospital can sign Up using username, Mail id & password.
2. Send this data to the server.
3. Sign In to the hospital registration.
4. Enter valid hospital details.
5. If admin reject the request, returns back to the homepage.
6. If admin accepts the request, can login to the hospital dashboard.
7. Can view their profile, identity and contact details.
8. Hospital dashboard shows the Notifications and sent requests.
9. Hospital can send and accept request to any of the receiver, donors and organizations.
10. Stop

6.2.5 ORGANIZATION

LOGIN

Input : Username & Password

Output : Logged in to the Organization dashboard.

1. Logged in using username & password.
2. Send this data to the server.
3. If an error occurred, returns back to the homepage.
4. If success, logged in to the organization dashboard.
5. Can view their profile, identity and contact details.
6. Organization dashboard shows the Notifications and sent requests.
7. Organization can send request to any of the receiver, donor and hospitals.

8. Stop

SIGN UP

Input : Username, Mail id & Password

Output : Sign In to the Organization registration.

1. Organization can sign Up using username, Mail id & password.
2. Send this data to the server.
3. Sign In to the Organization registration.
4. Enter valid organization details.
5. If admin reject the request, returns back to the homepage.
6. If admin accepts the request, can login to the organization dashboard.
7. Can view their profile, identity and contact details.
8. Donor dashboard shows the Notifications and sent requests.
9. Donor can send request to any of the receiver, donors and hospitals.
10. Stop

6.3 USE-CASE DIAGRAM

Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform.

The below mentioned use case diagram has five actors :

- Admin
- Organization
- Donor

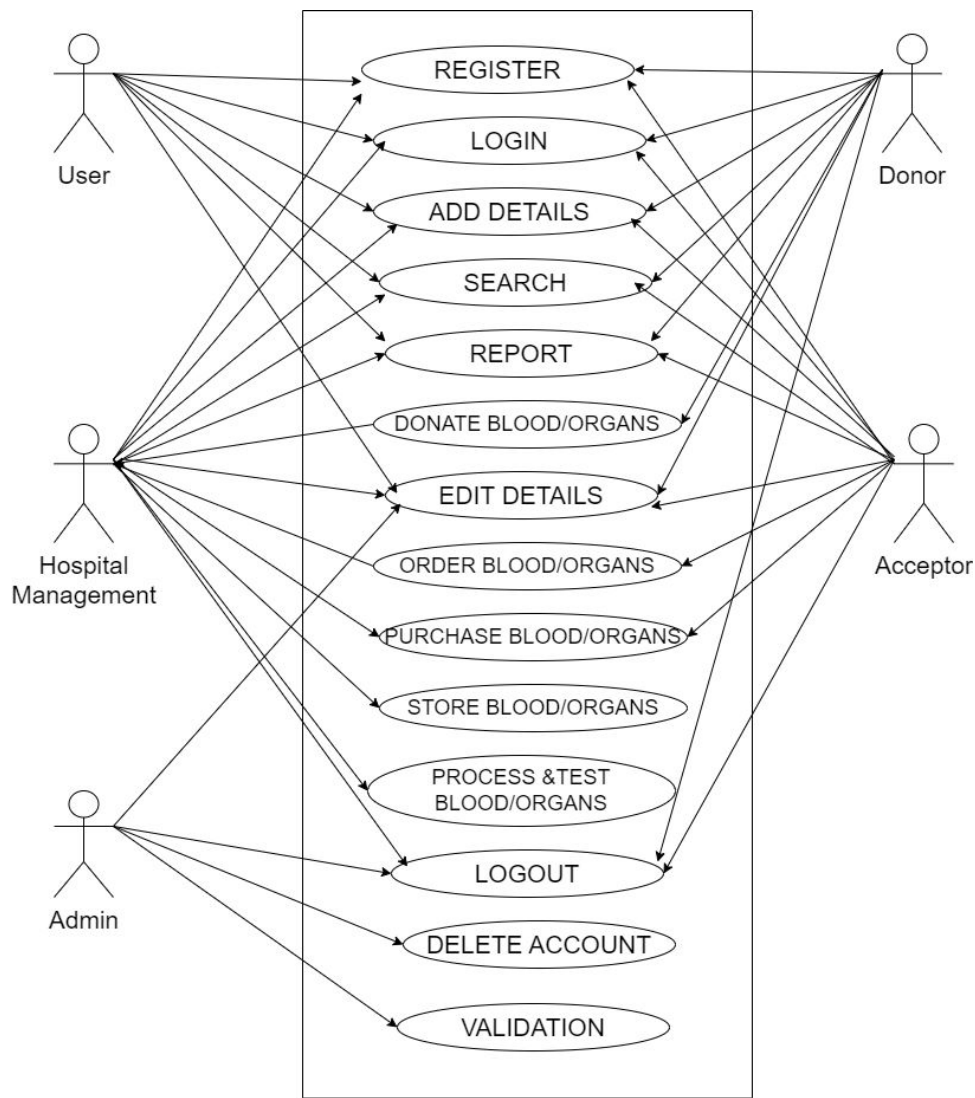


Fig. 6.1: Use Case diagram

- Receiver
- Hospital

Here the Admin can:

- Add Donation/ Received details.
- Send Mail to people into the registered email address.
- Admin can deactivate any user's profile whenever any disagreements happens.

- Admin can change any user password.
- Admin can create any new user.
- Admin can update all user's profile.
- Stores datas in the database.
- Admin can update his/her password.
- Admin can view all the updations happened in every profile except encrypted one.

Here the Organizations can:

- View blood/organ details in the registered hospitals.
- Organizations can request and can book for blood/organ from suggested hospitals and can contact the hospitals for blood/organ.
- Organizations can search for any hospitals or other users.

Here the Donor can:

- View the status of hospital's donating system
- He/She can Contact with the Hospitals to donate an organ.
- Whenever any request on donating that particular organ/blood came she/he can do it from the hospital.
- He/She can delete or update their own profiles.
- Donor can search for other users.
- Can donate blood/organ accordingly and can notify in the application

Here the Receiver can:

- View the status of hospital's donating system.
- He/She can Contact with the Hospitals in need of an organ/blood.

- Whenever he/she needs to get transplanted, He/she can check the hospital's status on availability of that organ and the further procedures will be done by the hospitals.
- He/She can delete or update their own profiles.
- Receiver can search for other users.
- Receiver can purchase blood/organs.

6.4 SEQUENCE DIAGRAM

A Sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these operations take place.

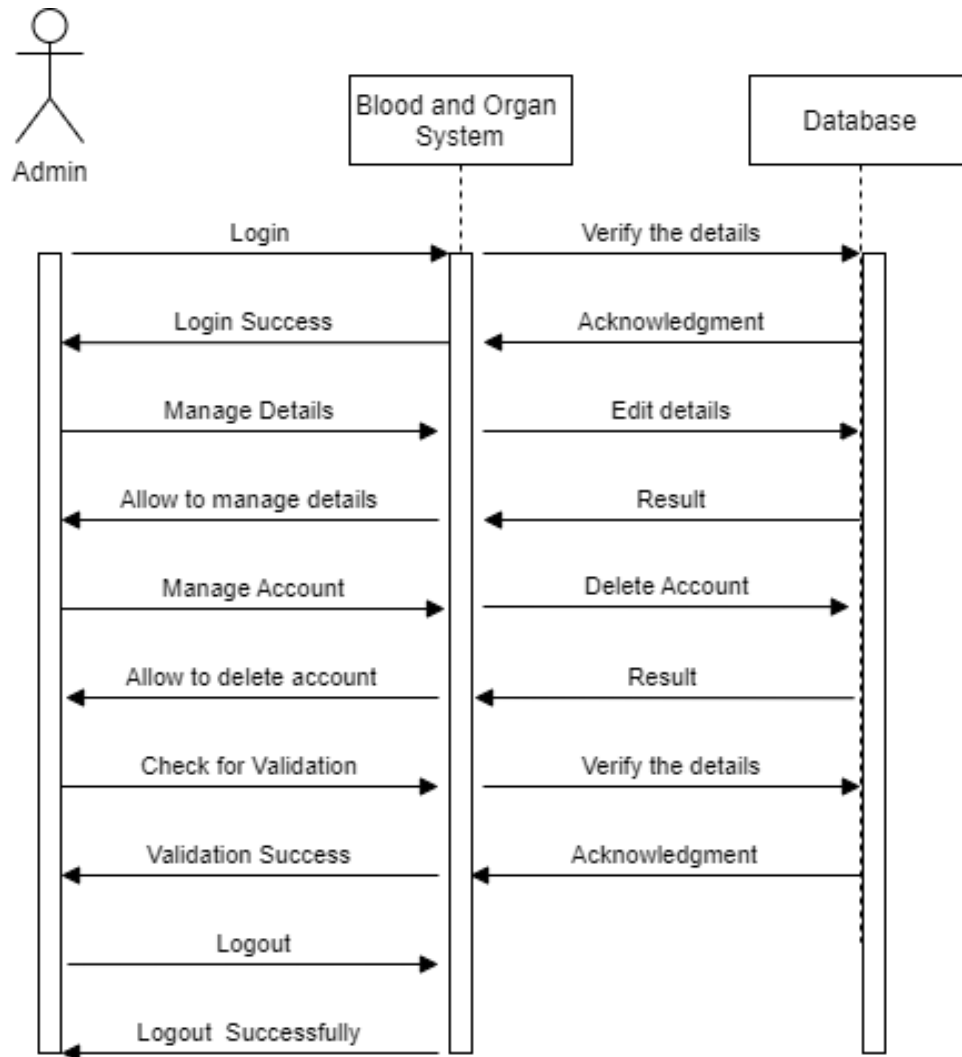


Fig. 6.2: Admin Sequence diagram

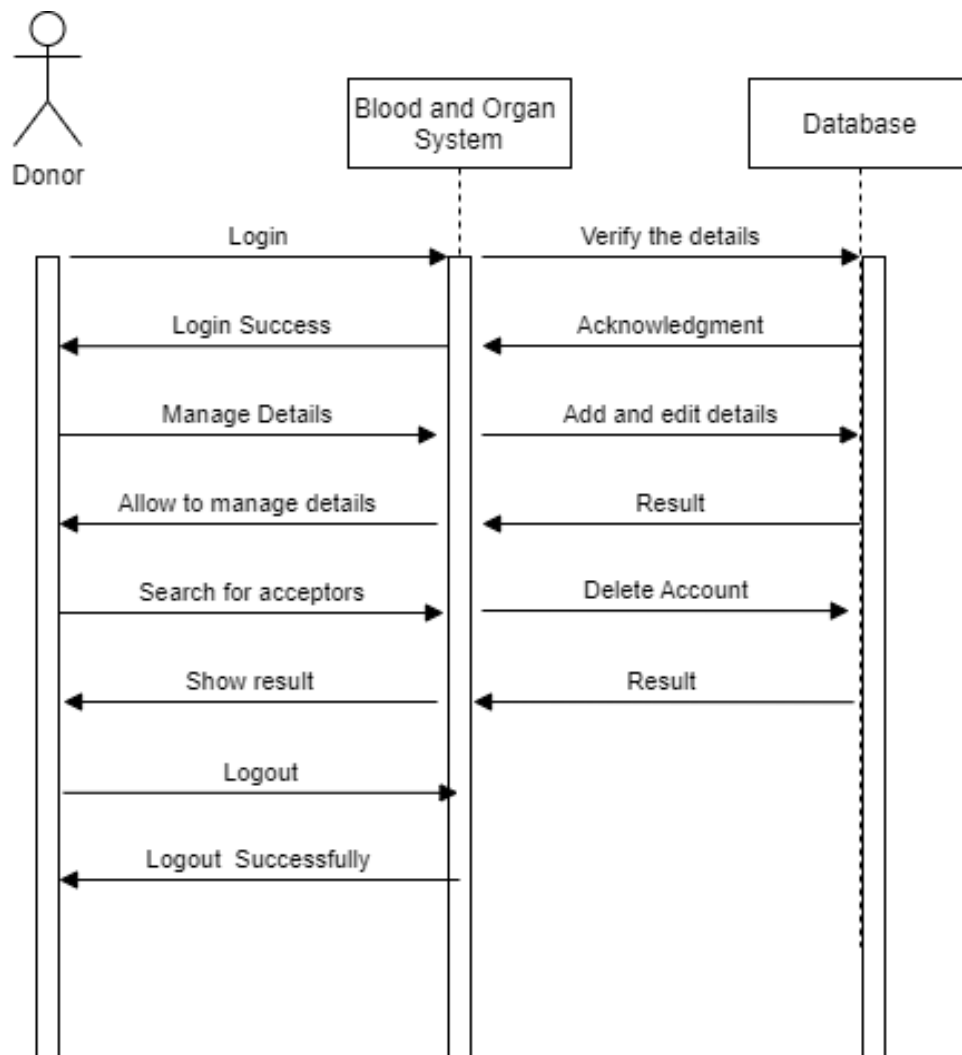


Fig. 6.3: Donor Sequence diagram

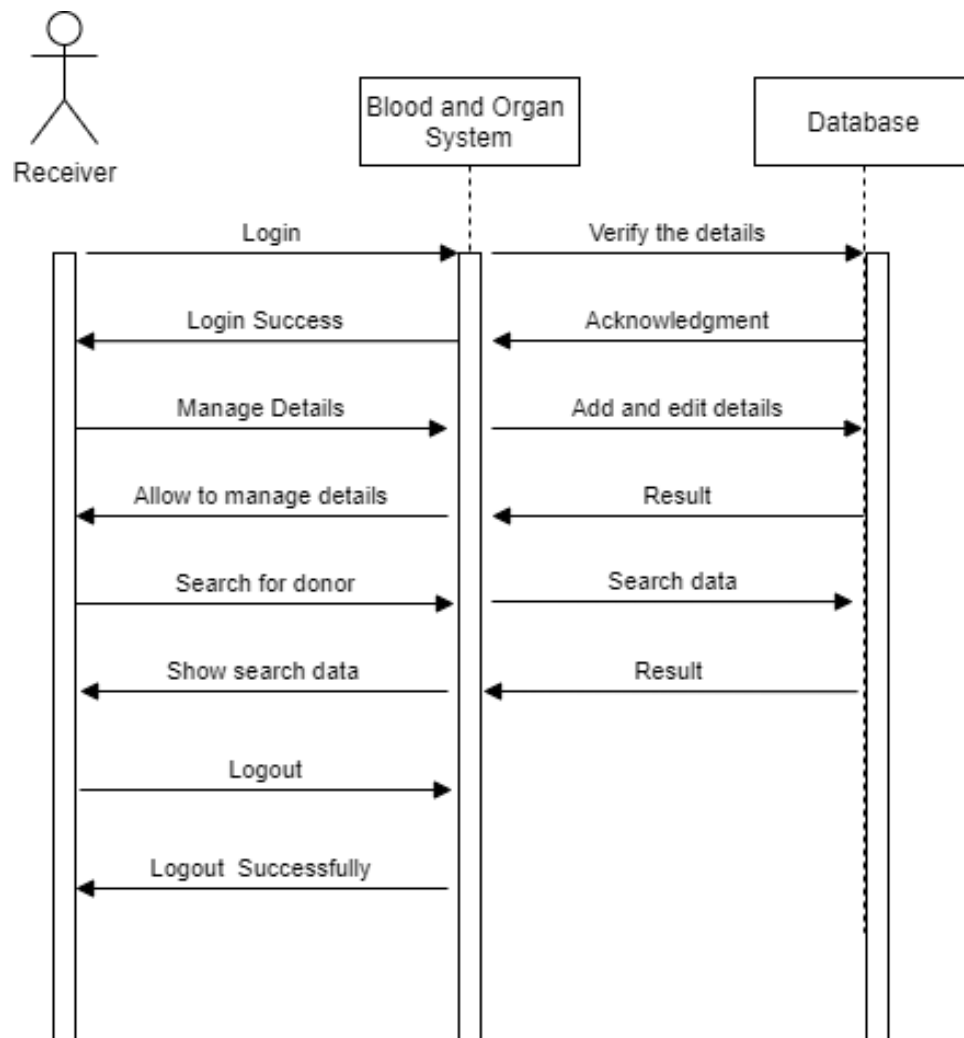


Fig. 6.4: Receiver Sequence diagram

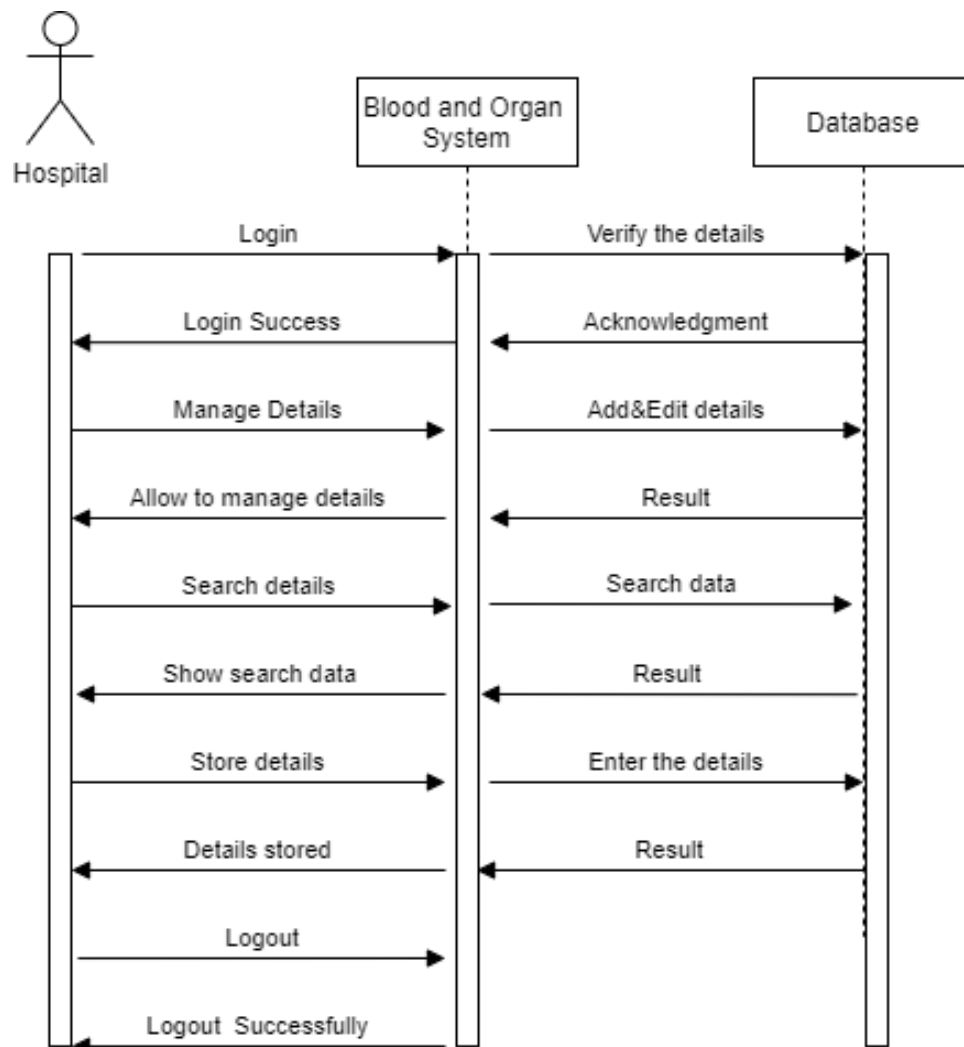


Fig. 6.5: Hospital Sequence diagram

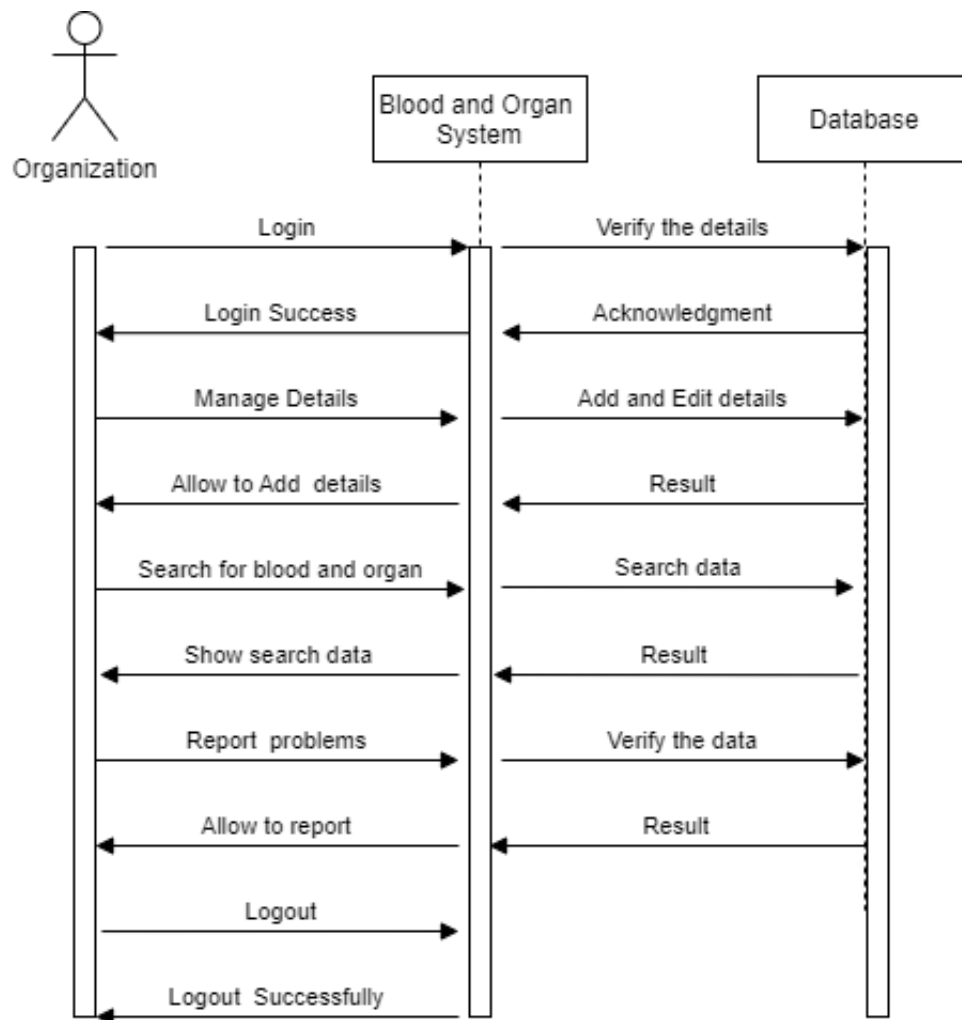


Fig. 6.6: Organization Sequence diagram

6.5 ENTITY RELATION DIAGRAM

The ER or (Entity Relational Model) is a high-level conceptual data model diagram. Entity-Relation model is based on the notion of real-world entities and the relationship between them.

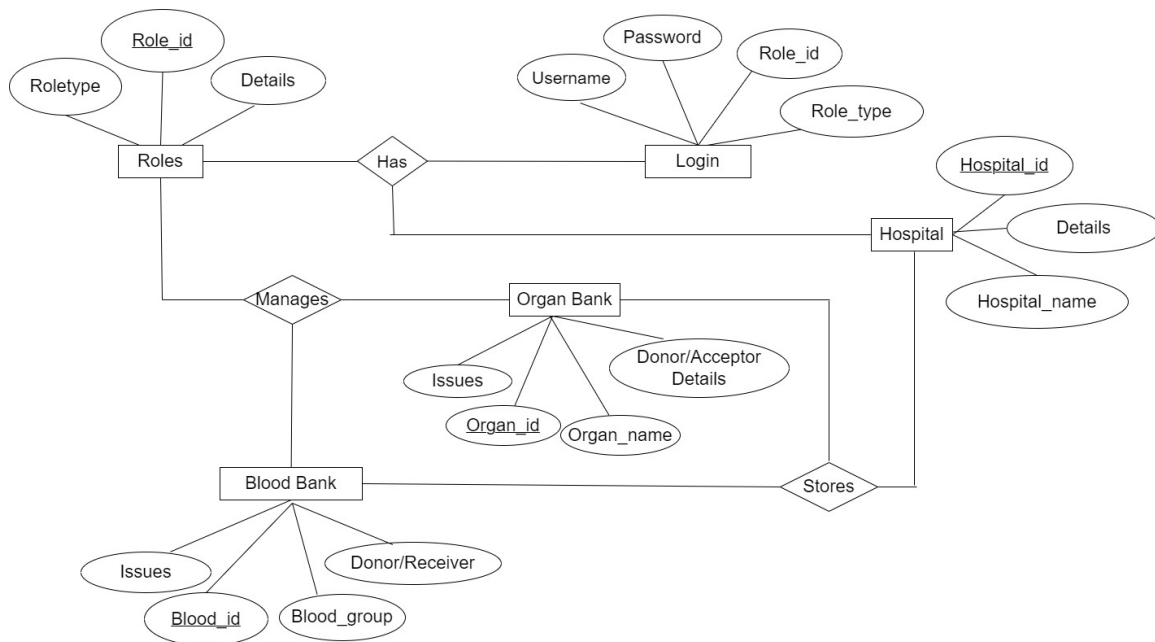


Fig. 6.7: Entity relation diagram

6.6 Data Flow Diagram

The data flow diagram (DFD) is used for classifying system requirements to major transformation that will become programs in system design. This is starting point of the design phase that functionally decomposes the required specifications down to the lower level of details

Bubbles: Represent the data transformations.

Lines: Represent the logic flow of data.

Data can trigger events and can be processed to useful information. Systems analysis recognizes the central goal of data in organizations.

Description

- Process: Describes how each input data is converted to output data
- Data Store: Describes the repositories of data in a system.
- Data Flow: Describes the data flowing between process, Data stores and entities.
- Source: An external entity causing the origin of data
- Sink: An external entity, which consumes the data

6.6.1 Level 0

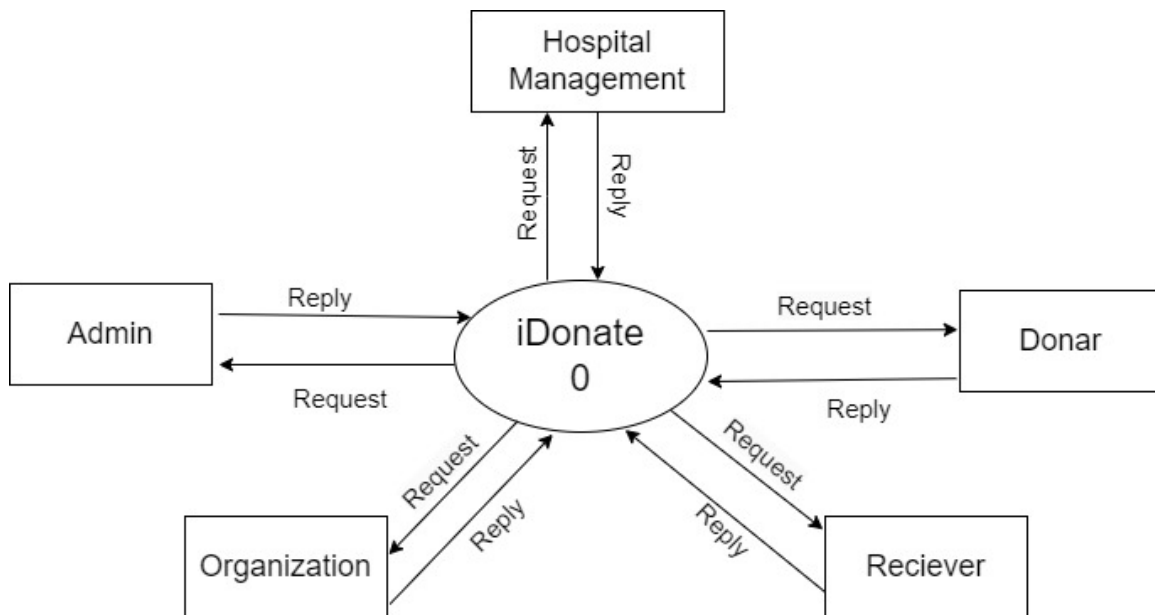


Fig. 6.8: Level 0 DFD of iDonate

6.6.2 Level 1(Admin)

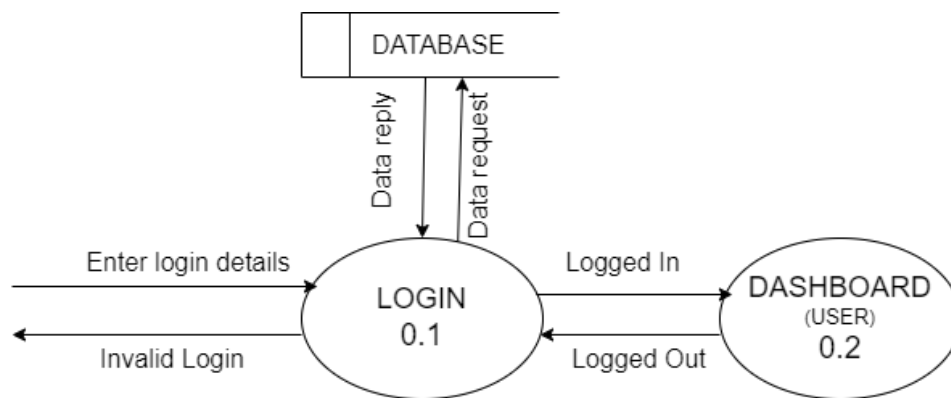


Fig. 6.9: Level 1 DFD of Admin

6.6.3 Level 2 (Admin)

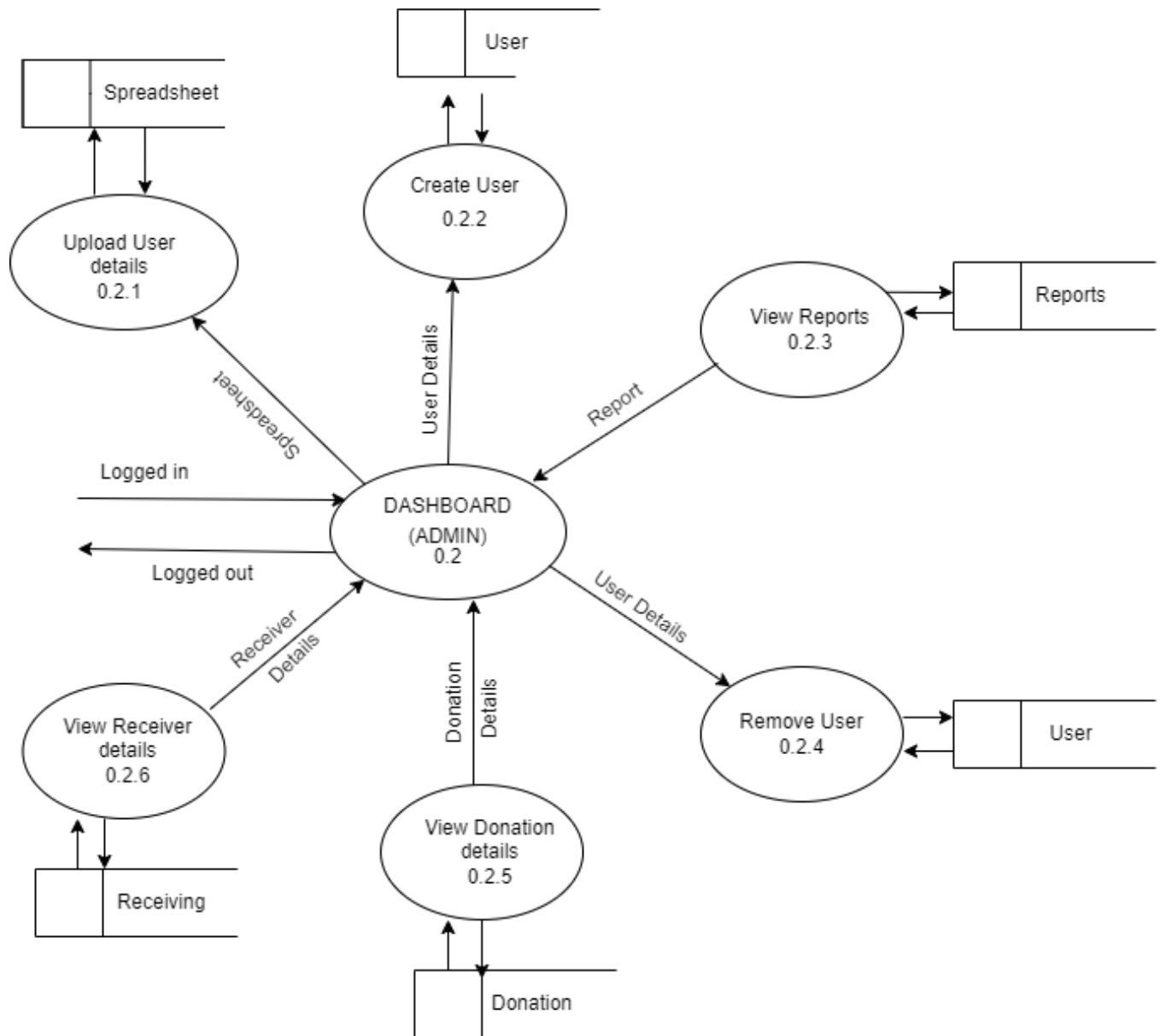


Fig. 6.10: Level 2 DFD of Admin

6.6.4 Level 1(Donor)

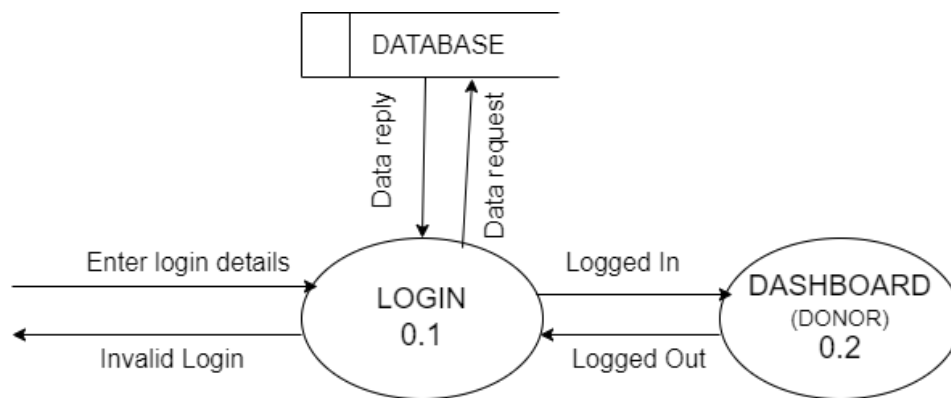


Fig. 6.11: Level 1 DFD of Donor

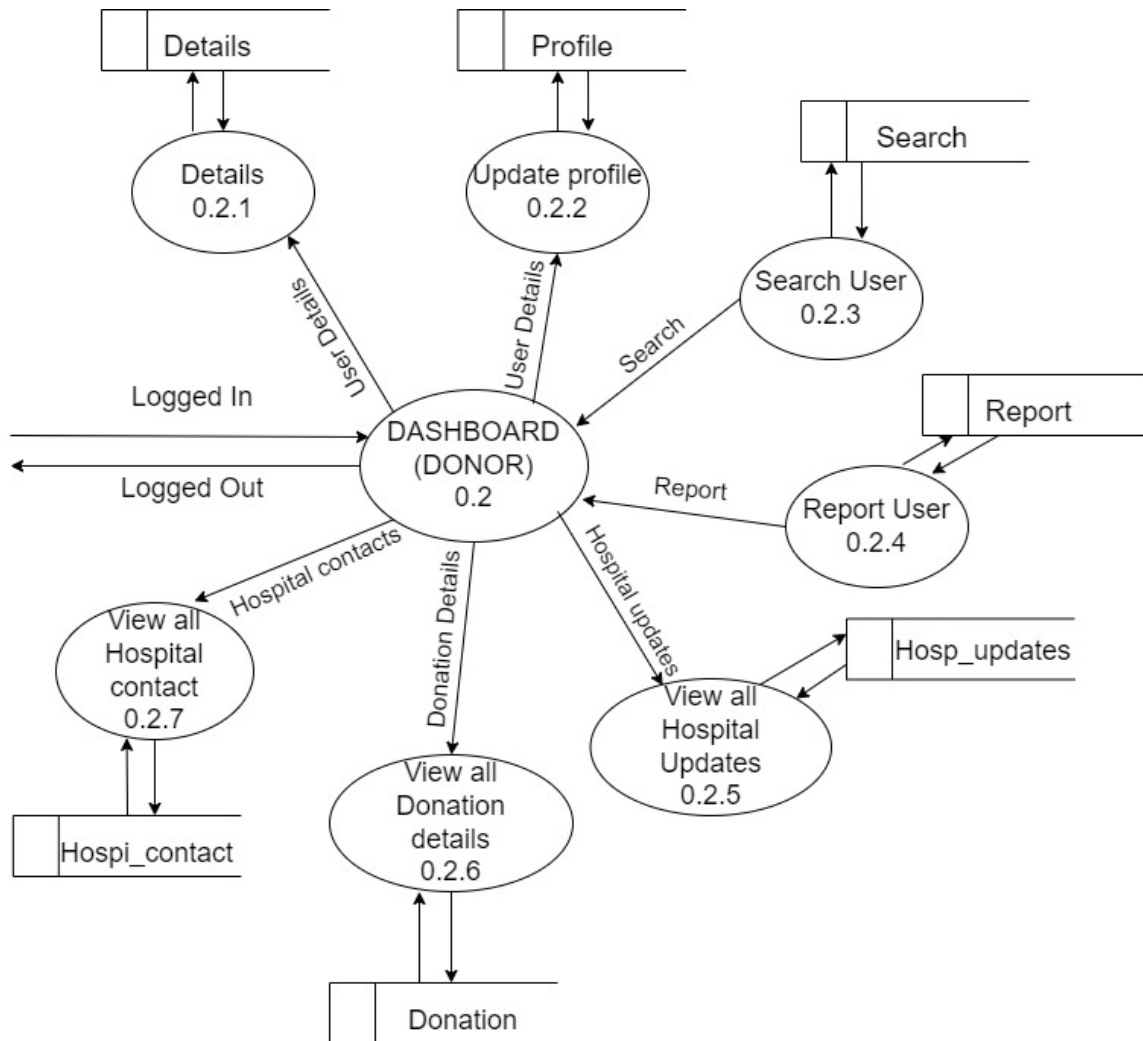
6.6.5 Level 2(Donor)

Fig. 6.12: Level 2 DFD of Donor

6.6.6 Level 1(Receiver)

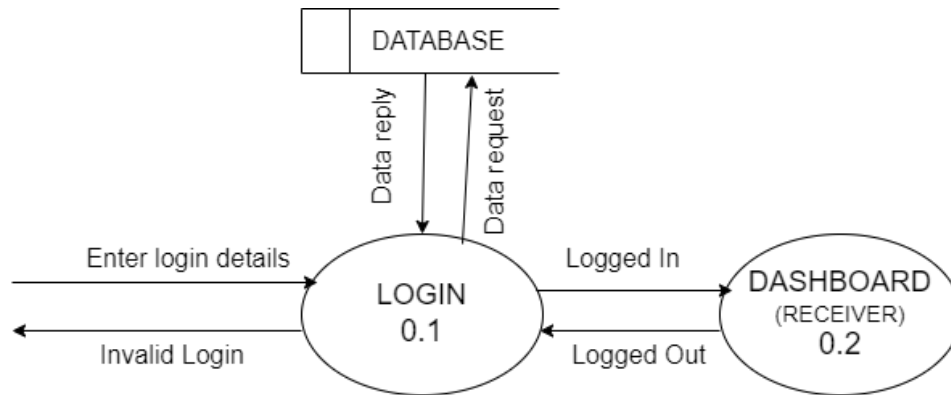


Fig. 6.13: Level 1 DFD of receiver

6.6.7 Level 2(Receiver)

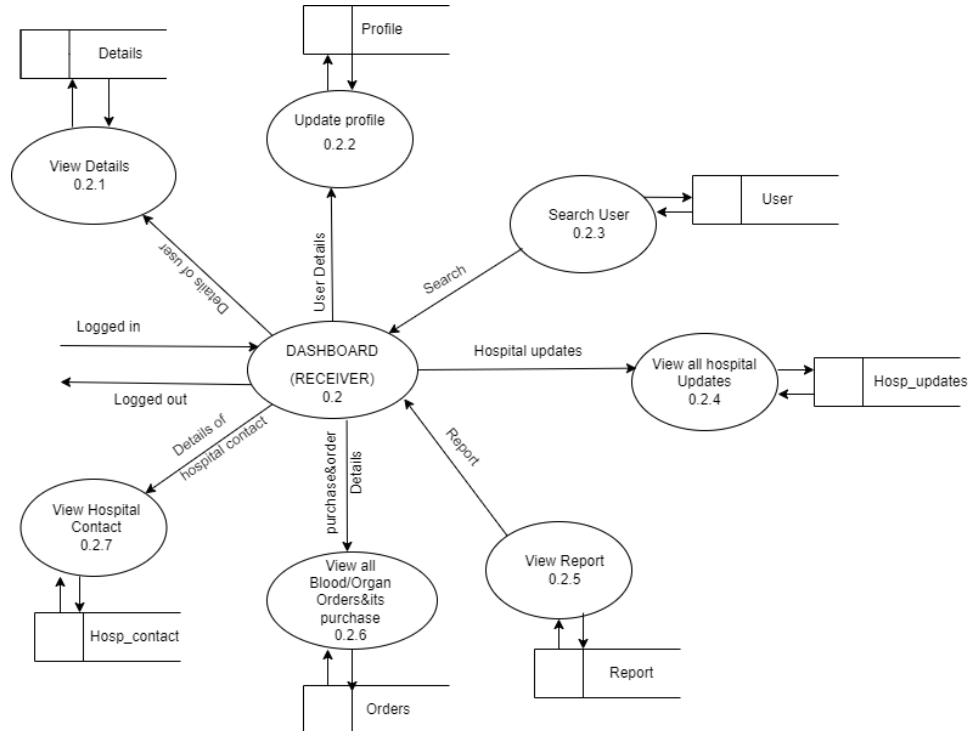


Fig. 6.14: Level 2 DFD of Receiver

6.6.8 Level 1(Hospital)

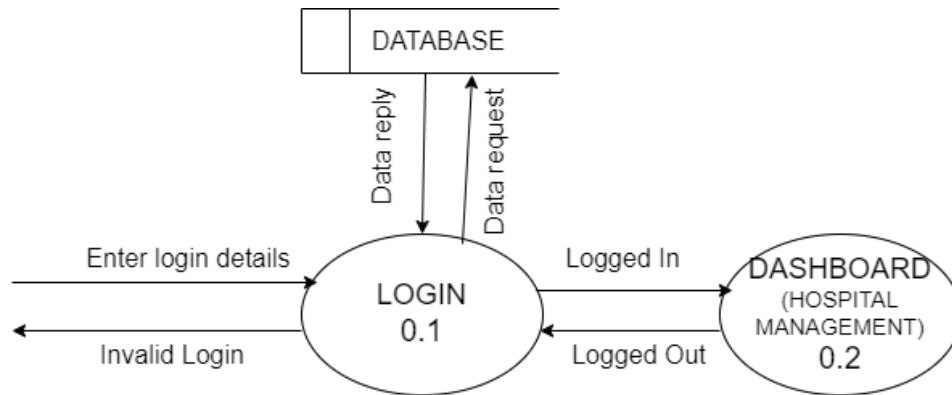


Fig. 6.15: Level 1 DFD of Hospital

6.6.9 Level 2(Hospital)

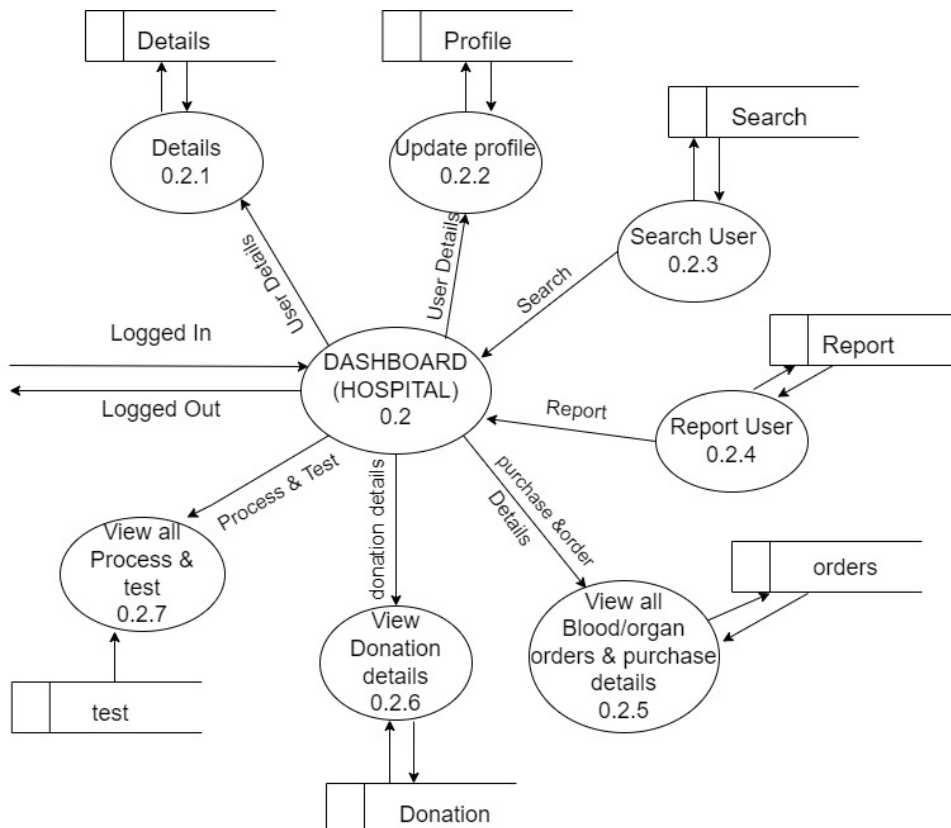


Fig. 6.16: Level 2 DFD of Hospital

6.6.10 Level 1(Organization)

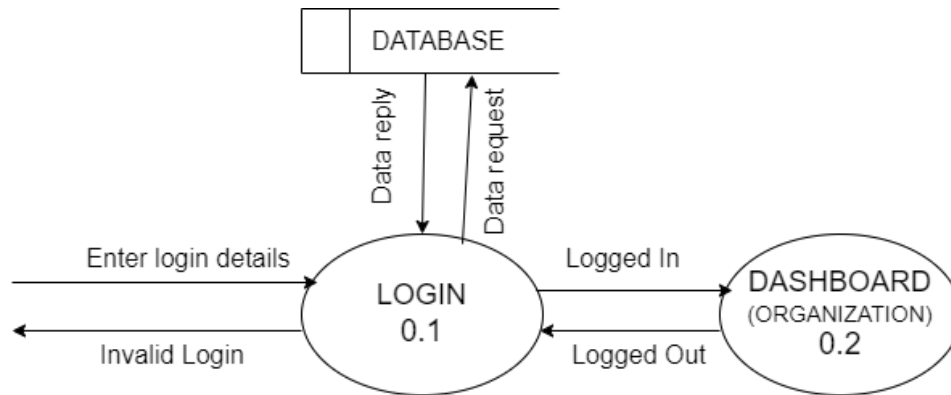


Fig. 6.17: Level 1 DFD of Organization

6.6.11 Level 2(Organization)

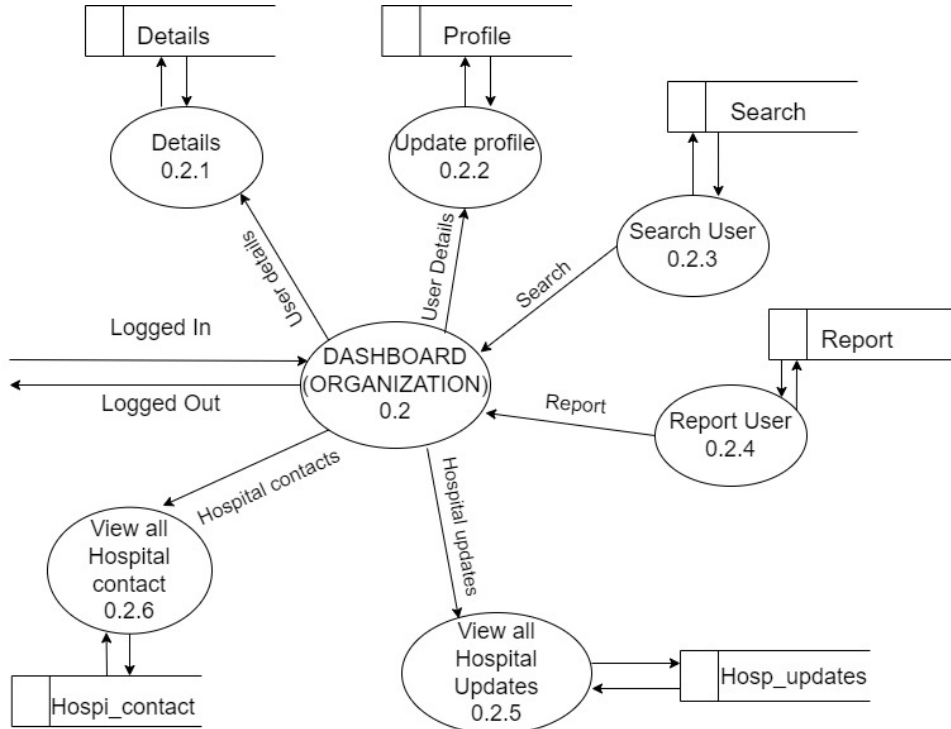


Fig. 6.18: Level 2 DFD of Organization

Chapter 7

SOFTWARE AND HARDWARE REQUIREMENT

7.1 Software Requirements

- Frontend : html,css,js
- Server : python,Django
- Database : Sqlite3
- Web Browser : Chrome or Firefox preferred.

7.2 Hardware Requirements

7.2.1 Computer

- Processor : Pentium 4
- Main memory : Minimum : 2GB

7.2.2 Smartphone

- RAM : 2GB

Chapter 8

IMPLEMENTATION

An important aspect of system analyst's job is to make sure that the new design implemented to establish standards. Implementation involves all these activities that take place to convert from the old system to new. A proper implementation is essential to provide reliable system to meet the requirements of a new computerized system will improve the efficiency of the entire system and reduce the labours involved.

8.1 Coding Environment Used

Visual Studio Code is a source-code editor developed by Microsoft for Windows, Linux and macOS. It includes support for debugging, embedded Git control, syntax highlighting, intelligent code completion, snippets, and code refactoring. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages (such as C++, C, Java, Python, PHP, Go) and runtimes (such as .NET and Unity). It is also customizable, so users can change the editor's theme, keyboard shortcuts, and preferences.

8.2 Admin Module

This allows admin to login into the application using username and password. It will redirect to the admin dashboard. Admin can view the user profile and verify the details then assign the status by accept/reject.

8.3 Donor Module

Donor can sign up using username, email-id and password and select the role. Then by using username and password donor can login to the dashboard. Donor dashboard shows the notifications, sent requests and Can view the profile, identity and contact details. Donor can send request to any of the receiver, hospitals and organizations.

8.4 Receiver Module

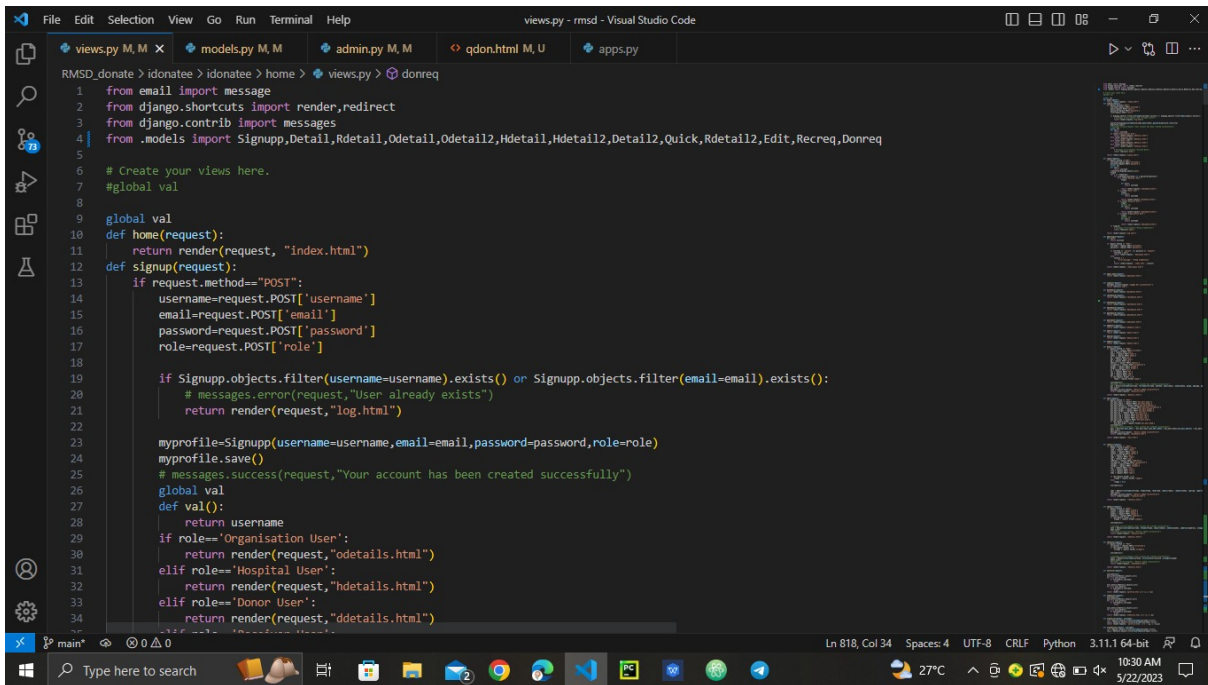
Receiver can sign up using username, email-id and password and select the role. Then by using username and password receiver can login to the dashboard. Receiver dashboard shows the notifications, sent requests and can view the profile, identity and contact details. Receiver can send and accept request to any of the donor, hospitals and organizations.

8.5 Organization Module

Organization can sign up using username, email-id and password and select the role. Organization will give the licence id and proof. Then by using username and password organization can login to the dashboard. Organization can send and accept request to any of the donor, hospitals and receivers.

8.6 Hospital Module

Hospital can sign up using username, email-id and password and select the role. Then by using username and password hospital can login to the dashboard. Hospitals can send and accept request to any of the donor, receiver and organizations.

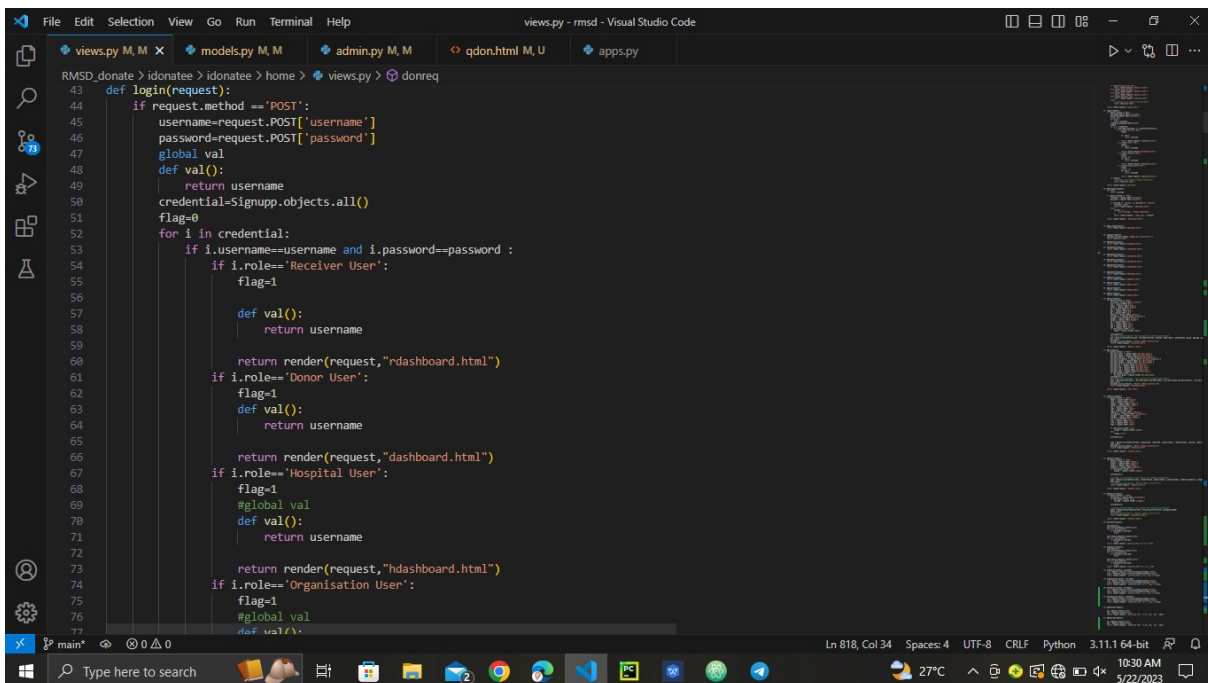


```

1 from email import message
2 from django.shortcuts import render, redirect
3 from django.contrib import messages
4 from .models import Signupp, Detail, Odetail, Odetail2, Hdetail, Hdetail2, Detail2, Quick, Rdetail2, Edit, Recreq, Donreq
5
6 # Create your views here.
7 global val
8
9 global val
10 def home(request):
11     return render(request, "index.html")
12 def signup(request):
13     if request.method=="POST":
14         username=request.POST['username']
15         email=request.POST['email']
16         password=request.POST['password']
17         role=request.POST['role']
18
19         if Signupp.objects.filter(username=username).exists() or Signupp.objects.filter(email=email).exists():
20             # messages.error(request,"User already exists")
21             return render(request,"log.html")
22
23         myprofile=Signupp(username=username,email=email,password=password,role=role)
24         myprofile.save()
25         # messages.success(request,"Your account has been created successfully")
26         global val
27         def val():
28             return username
29         if role=="Organisation User":
30             return render(request,"odetails.html")
31         elif role=="Hospital User":
32             return render(request,"hdetails.html")
33         elif role=="Donor User":
34             return render(request,"ddetails.html")

```

Fig. 8.1: views.py

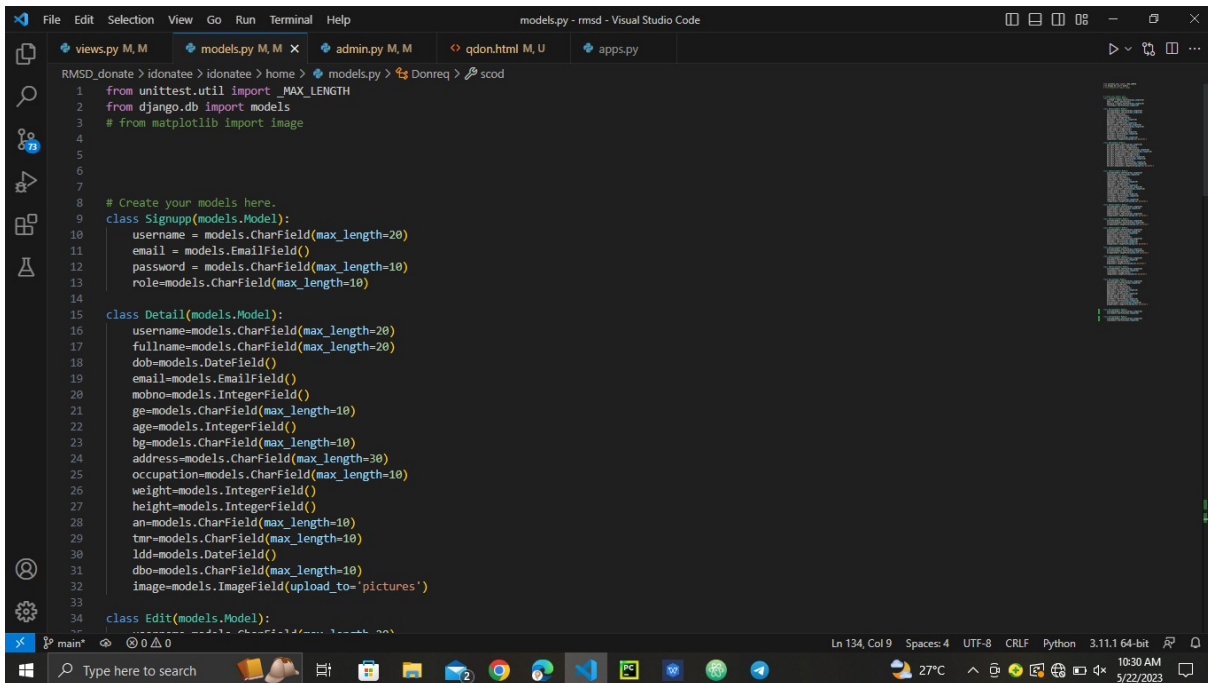


```

43 def login(request):
44     if request.method=="POST":
45         username=request.POST['username']
46         password=request.POST['password']
47         global val
48         def val():
49             return username
50         credential=Signupp.objects.all()
51         flag=0
52         for i in credential:
53             if i.username==username and i.password==password :
54                 if i.role=="Receiver User":
55                     flag=1
56
57                 def val():
58                     return username
59                 return render(request,"dashboard.html")
60             if i.role=="Donor User":
61                 flag=1
62                 def val():
63                     return username
64                 return render(request,"dashboard.html")
65             if i.role=="Hospital User":
66                 flag=1
67                 #global val
68                 def val():
69                     return username
70                 return render(request,"hdashboard.html")
71             if i.role=="Organisation User":
72                 flag=1
73                 #global val
74                 def val():

```

Fig. 8.2: views.py

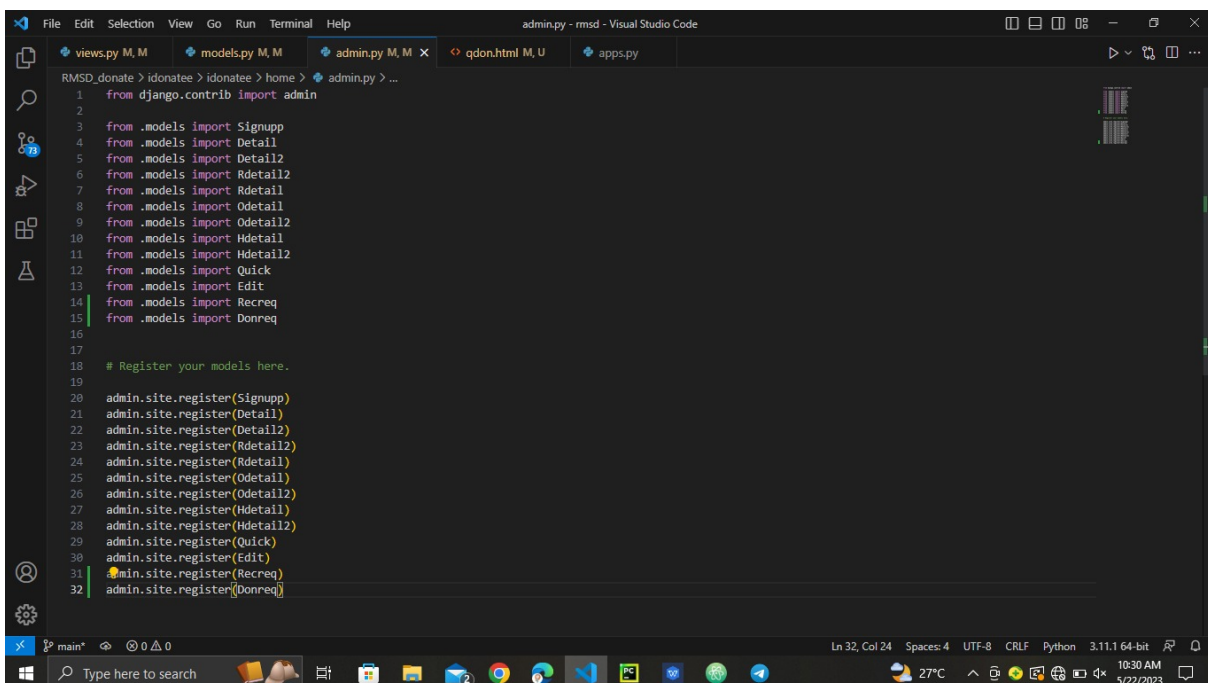


```

1  from unittest.util import MAX_LENGTH
2  from django.db import models
3  # from matplotlib import image
4
5
6
7
8  # Create your models here.
9  class Signupp(models.Model):
10     username = models.CharField(max_length=20)
11     email = models.EmailField()
12     password = models.CharField(max_length=10)
13     role=models.CharField(max_length=10)
14
15
16     class Detail(models.Model):
17         username=models.CharField(max_length=20)
18         fullname=models.CharField(max_length=20)
19         dob=models.DateField()
20         email=models.EmailField()
21         mobno=models.IntegerField()
22         ge=models.CharField(max_length=10)
23         age=models.IntegerField()
24         hg=models.CharField(max_length=10)
25         address=models.CharField(max_length=30)
26         occupation=models.CharField(max_length=10)
27         weight=models.IntegerField()
28         height=models.IntegerField()
29         an=models.CharField(max_length=10)
30         tmr=models.CharField(max_length=10)
31         ldd=models.DateField()
32         dbo=models.CharField(max_length=10)
33         image=models.ImageField(upload_to='pictures')
34
35     class Edit(models.Model):

```

Fig. 8.3: models.py



```

1  from django.contrib import admin
2
3  from .models import Signupp
4  from .models import Detail
5  from .models import Detail2
6  from .models import Rdetail2
7  from .models import Rdetail
8  from .models import Odetail
9  from .models import Odetail2
10 from .models import Hdetail
11 from .models import Hdetail2
12 from .models import Quick
13 from .models import Edit
14 from .models import Recreq
15 from .models import Donreq
16
17
18 # Register your models here.
19
20 admin.site.register(Signupp)
21 admin.site.register(Detail)
22 admin.site.register(Detail2)
23 admin.site.register(Rdetail2)
24 admin.site.register(Rdetail)
25 admin.site.register(Odetail)
26 admin.site.register(Odetail2)
27 admin.site.register(Hdetail)
28 admin.site.register(Hdetail2)
29 admin.site.register(Quick)
30 admin.site.register(Edit)
31 admin.site.register(Recreq)
32 admin.site.register(Donreq)

```

Fig. 8.4: admin.py

```

1 from django.urls import path
2 from . import views
3
4
5 urlpatterns = [
6     path('', views.home, name='home'),
7     path('login', views.login, name='login'),
8     path('adminlogin', views.adminlogin, name='adminlogin'),
9     path('signup', views.signup, name='signup'),
10    path('signout', views.signout, name='signout'),
11
12    path('detail', views.detail, name='detail'),
13    path('detail2', views.detail2, name='detail2'),
14    path('dashboard', views.dashboard, name='dashboard'),
15    path('profile', views.profile, name='profile'),
16    path('dsearch', views.dsearch, name='dsearch'),
17    path('edit', views.edit, name='edit'),
18    path('didentity', views.didentity, name='didentity'),
19    path('donsearch/<str:username>', views.donsearch, name='donsearch'),
20    path('donidentity/<str:username>', views.donidentity, name='donidentity'),
21    path('donacceptreject/<str:username>', views.donacceptreject, name='donacceptreject'),
22
23    path('rdetail', views.rdetail, name='rdetail'),
24    path('rdetail2', views.rdetail2, name='rdetail2'),
25    path('rdashboard', views.rdashboard, name='rdashboard'),
26    path('ridentity', views.ridentity, name='ridentity'),
27    path('rprofile', views.rprofile, name='rprofile'),
28    path('rsearch', views.rsearch, name='rsearch'),
29    path('recidentity/<str:username>', views.recidentity, name='recidentity'),
30    path('recsearch/<str:username>', views.recsearch, name='recsearch'),
31    path('recacceptreject/<str:username>', views.recacceptreject, name='recacceptreject'),
32    path('rnotification', views.rnotification, name='rnotification'),
33
34

```

Fig. 8.5: urls.py

```

551 def donacceptreject(request, username):
552     def don_delete_user_data(username):
553         try:
554             user_detail = Detail.objects.filter(username=username).first()
555             if user_detail:
556                 user_detail.delete()
557             user_signup = Signup.objects.filter(username=username).first()
558             if user_signup:
559                 user_signup.delete()
560             user_detail2 = Detail2.objects.filter(username=username).first()
561             if user_detail2:
562                 user_detail2.delete()
563             return True
564         except Exception as e:
565             print(e)
566             return False
567     d_p = Detail.objects.filter(username=username).first()
568     if request.method == "POST":
569         status = request.POST.get("password")
570         if status == "reject":
571             if don_delete_user_data(username):
572                 messages.success(request, "User data deleted successfully.")
573             else:
574                 messages.error(request, "Error deleting user data.")
575             signup = Signup.objects.filter(username=username).first()
576             if signup:
577                 signup.delete()
578             else:
579                 messages.error(request, "Signup not found.")
580             return redirect("/admdonar")
581         elif status == "accept":
582             # Handle the accept case here
583             pass # replace this with your code
584     return render(request, 'admDacceptreject.html', {'i': d_p})

```

Fig. 8.6: Donor User Accepting or Rejecting code in Admin Module

Chapter 9

RESULT & ANALYSIS

9.1 Screenshots

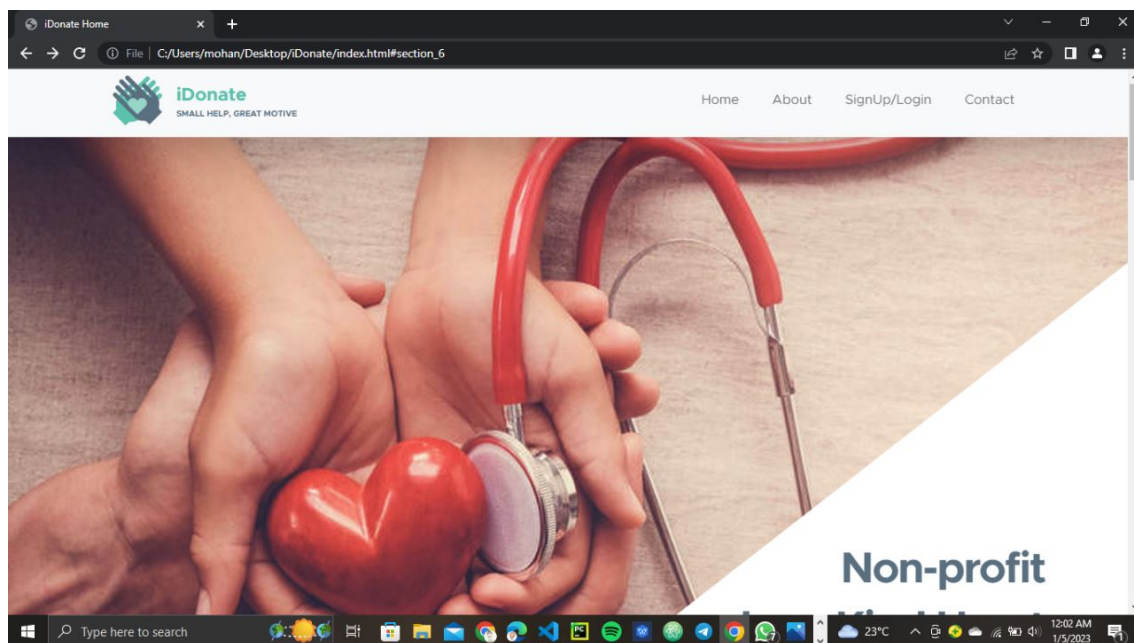


Fig. 9.1: Home page

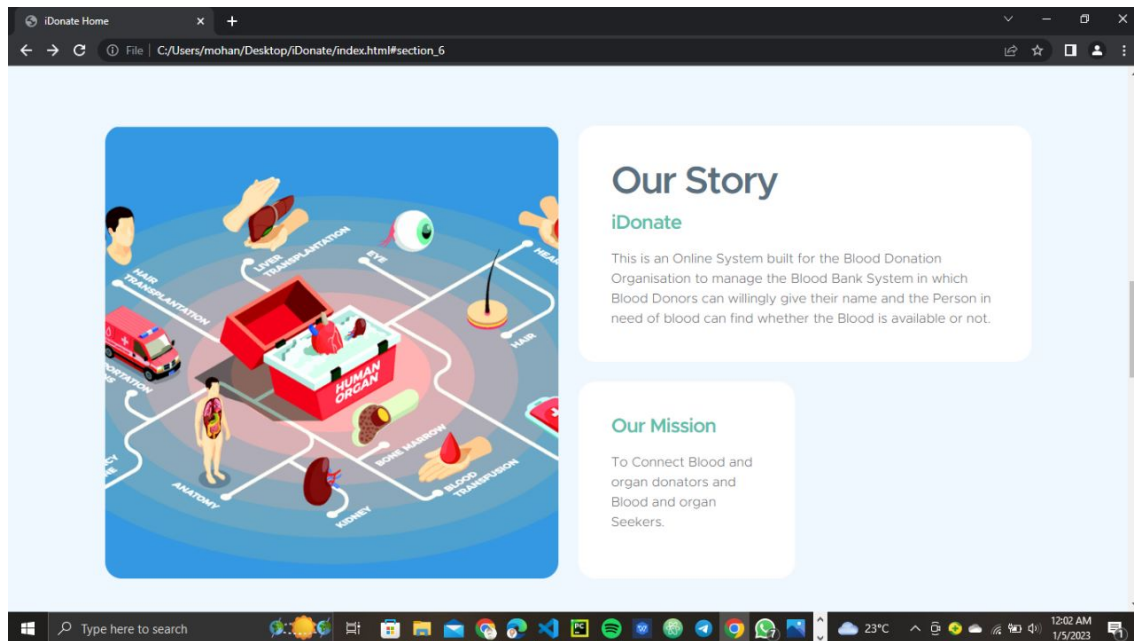


Fig. 9.2: Home page- about us

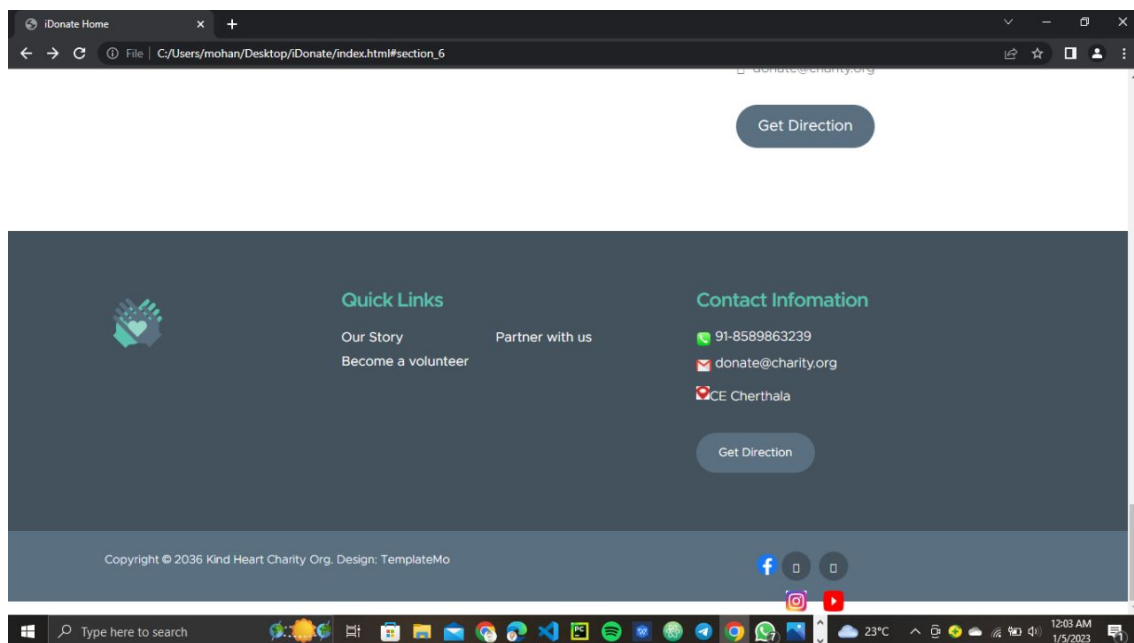


Fig. 9.3: Home page - contact details

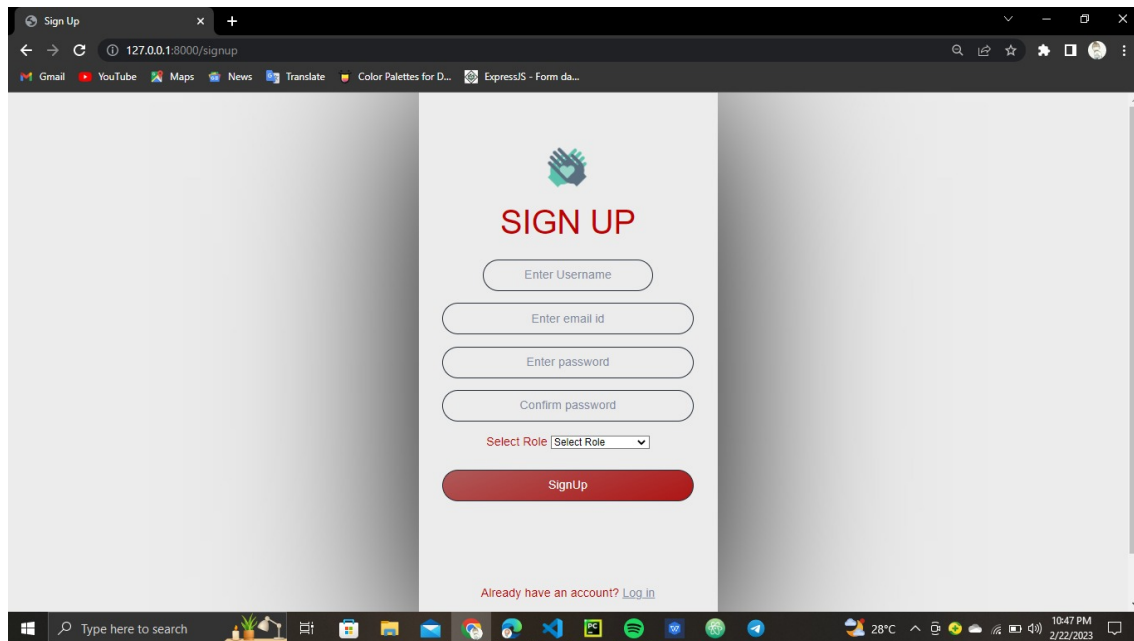


Fig. 9.4: Sign up page

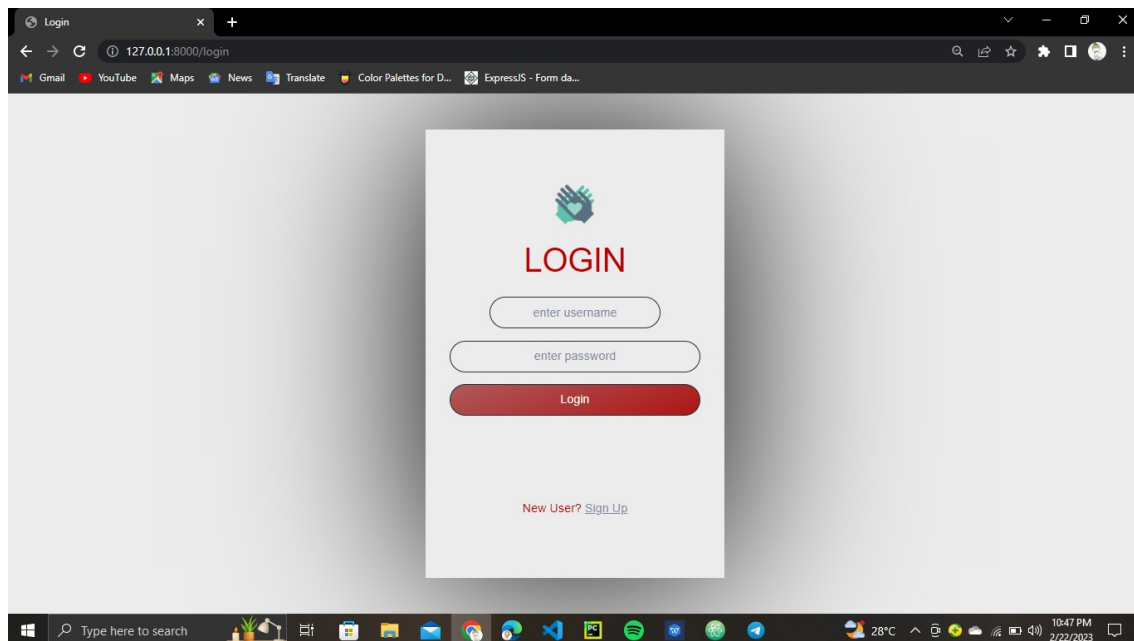


Fig. 9.5: Login page

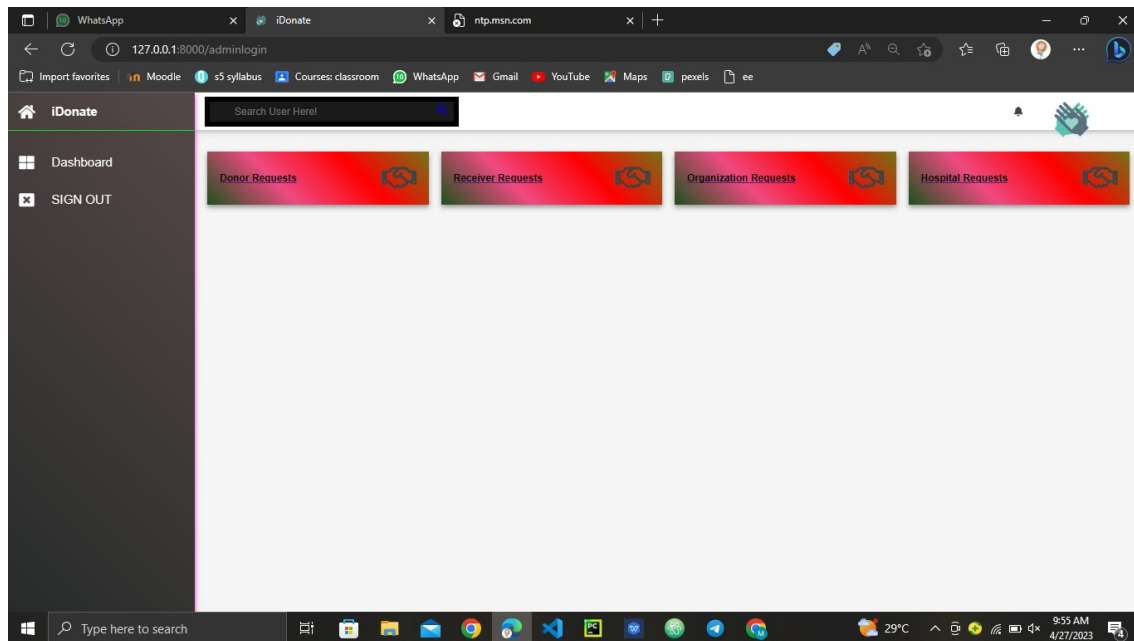


Fig. 9.6: ADMIN DASHBOARD

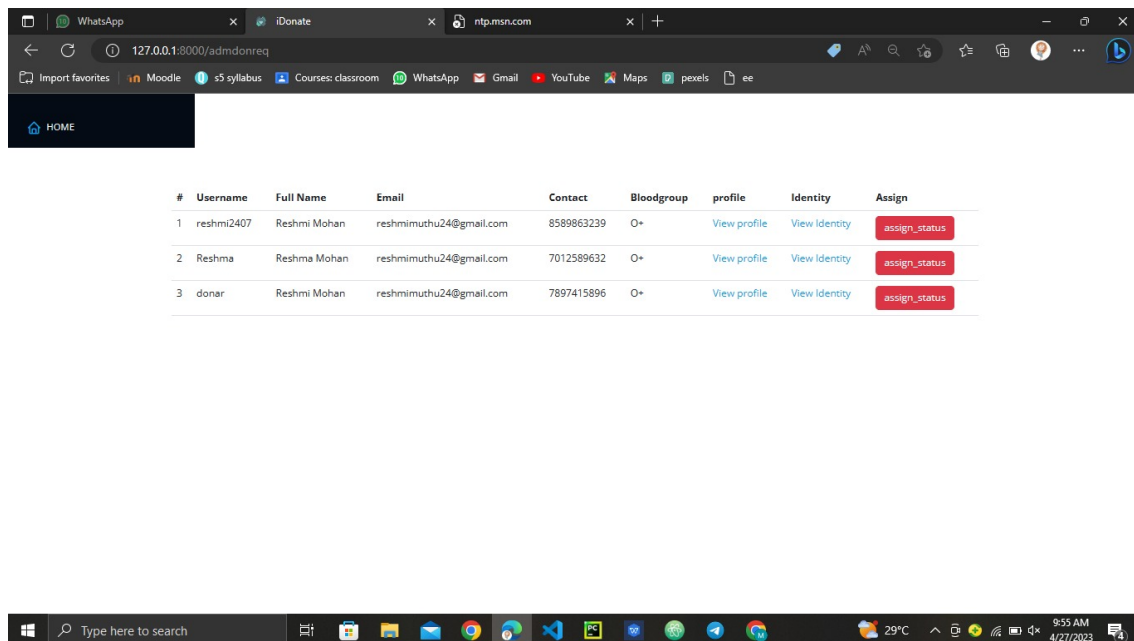


Fig. 9.7: ADMIN PERMISSION

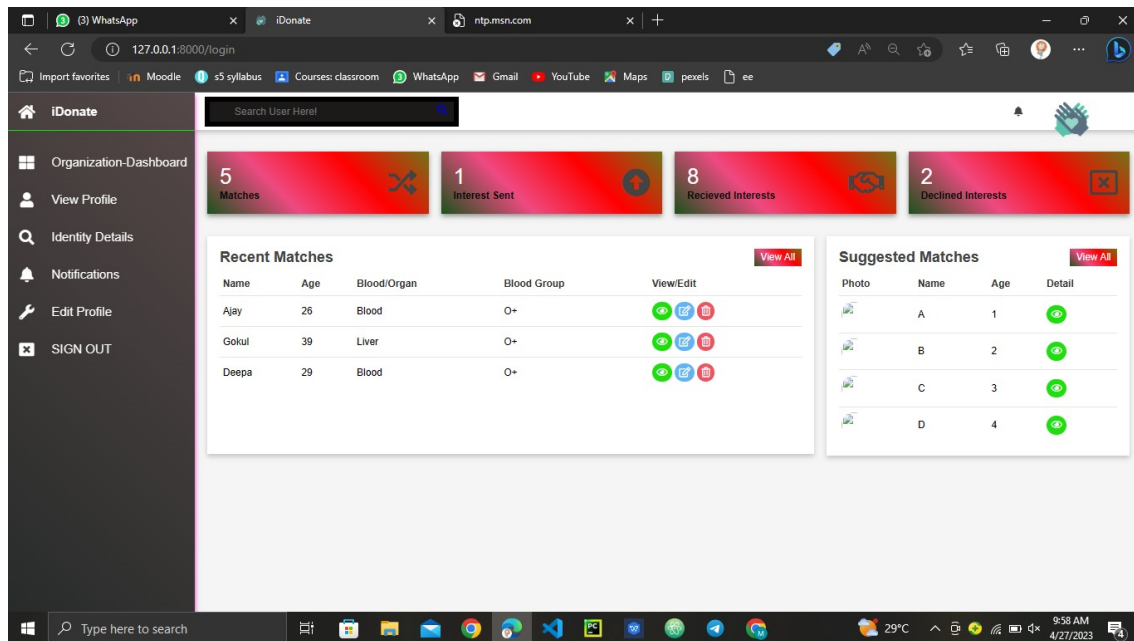


Fig. 9.8: DONOR DASHBOARD

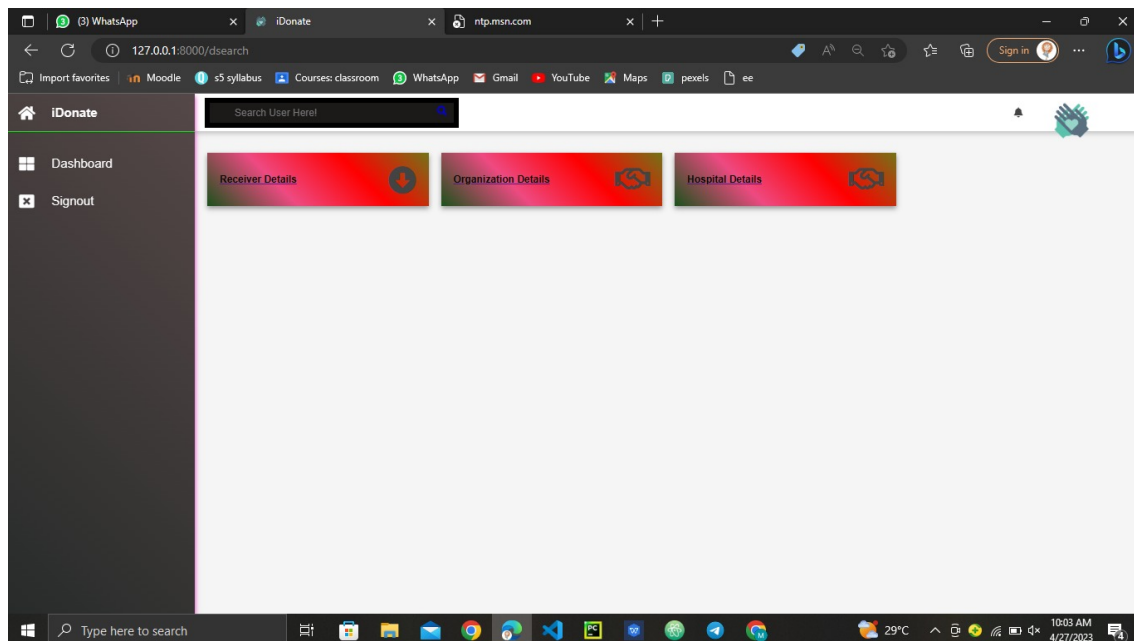
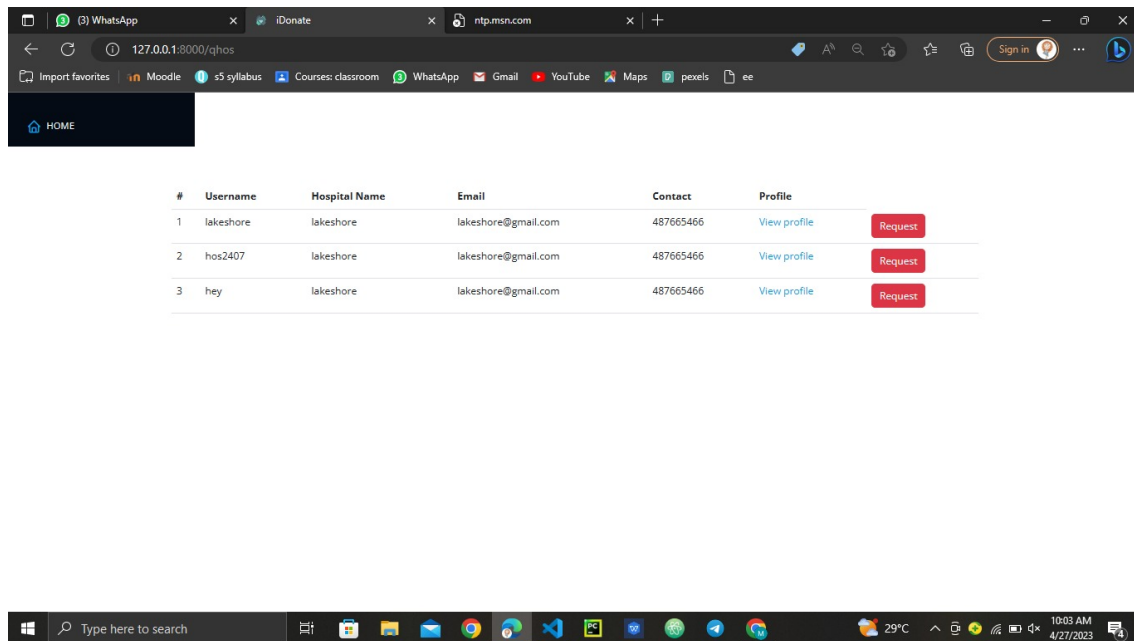


Fig. 9.9: DONOR-SEARCH



#	Username	Hospital Name	Email	Contact	Profile	
1	lakeshore	lakeshore	lakeshore@gmail.com	487665466	View profile	Request
2	hos2407	lakeshore	lakeshore@gmail.com	487665466	View profile	Request
3	hey	lakeshore	lakeshore@gmail.com	487665466	View profile	Request

Fig. 9.10: DONOR-HOSPITAL DETAILS

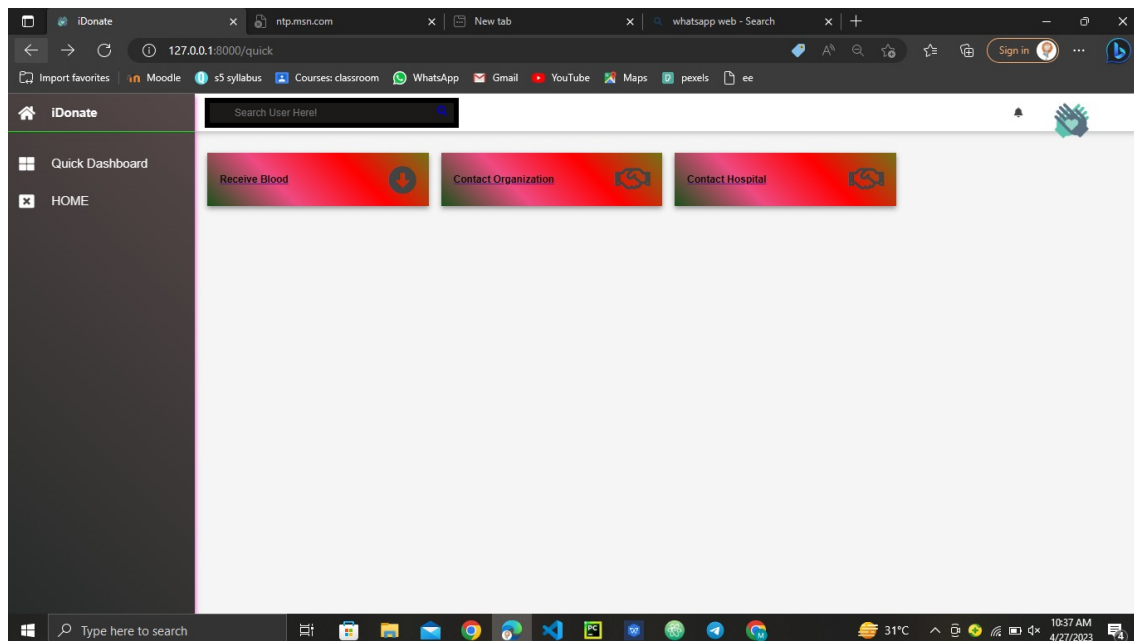


Fig. 9.11: QUICK ACCESS

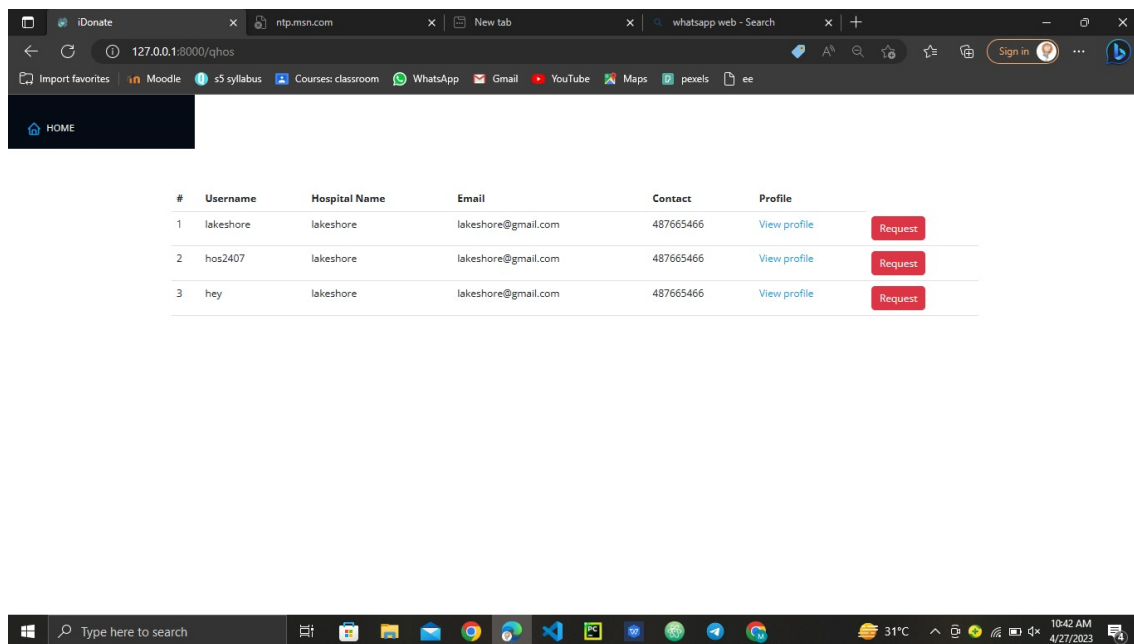


Fig. 9.12: QUICK ACCESS

Chapter 10

CONCLUSION & FUTURE SCOPE

Future Scope of this project are as follows:

- In future we can use the technique of machine learning and artificial intelligence for reading the medical prescription and reports and carry out the major matching between donor and recipient through the application itself.
- It can be the most self evolved application which can read the medical report and can find matches by specially taking time of requirement as a major consideration.

This application will be the most user-friendly platform to enroll people in such good activity and will save many lives of the people who are dying due to lack of organ donation. It will create a long-term lifesaving role for every citizen in our country. Our prime target is to provide organs to the seeker when they are in need and make it a life saver platform for those who are dying due to the lack of organs and also targets the human race and spreads the concept of saving life after one's death .

REFERENCES

- [1] P. Priya, V. Saranya, S. Shabana, Kavitha Subramani : "The Optimization of Blood Donor Information and Management System by Technopedia" [IJIRSET] - Feb 2014.
- [2] Tushar Pandit, Satish Niloor, A.S. Shinde : "A Survey Paper on E-Blood Bank and an Idea to use on Smartphone" [IJCA] - March 2015.
- [3] Prof. Snigdha, Varsha Anabhavane, Pratiksha lokhande, Siddhi Kasar, Pranita More : "Android Blood Bank" [IJARCCE] – Nov 2015.
- [4] R.Vanitha, M.E, P.Divyarani, BCloud App: "Blood Donor Application for Android Mobile", International Journal of Innovations in Engineering and Technology (IJIET), Vol. 2 Issue 1 February 2013.
- [5] Iraky Khalifa, Hala Abd Al-Glil, Mohamed M. Abbassy : "Mobile Hospitalization for Kidney Transplantation" [IJCA] – Apr 2014.
- [6] Nikita M. Lunawat, Chetan D. Kshirsagar, Ashish A. Gawhande, Rohini M. Rathod, Apurva D. Thool, Shrika C. Chuwble "Blood And Organ For Patient Using Android Application" [IJRET] May 6.