PROJECT REPORT

PLASMA DONOR APPLICATION

submitted by

PNT2022TMID00767

Vishnudhar G R	–	211419104309
Tharunkumar C	–	211419104293
Sathish kumar G S	–	211419104240
Surya H	–	211419104278

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

Recently concern grows about the plasma donation for COVID-19 during the pandemic situation. This convalescent plasma was used to recover patients who are critically ill as it helps to grow antibodies on their body. Recent researches show that many people are willing to help someone in need through money, blood and plasma donation etc. but they find it difficult to identify and approach the needy people who arenot aware of technological innovations, including the use of social media. Plasma is used to various infectious diseases and it is one of the oldest methods known as plasma therapy. Plasma therapy is a Process where blood is donated by recovered patients in order to establish anti bodies that fights the infection. This system comprises of Admin, user and donor where both can request for Plasma. The proposedmethod helps the users to check the availability of donors. A donor has to register to thewebsite providing their details. The registered users can get the information about the donor count of each blood group. The database will have all the details such as name, email, phone number, infected status. Whenever a user requests for a particular blood group then the concerned blood group donors will receive the notification regarding therequirement. For instance, during COVID 19 crisis the requirement for plasma increaseddrastically as there were no vaccination found in order to treat the infected patients, with plasma therapy the recovery rates where high but the donor count was very low and in such situations it was very important to get the information about the plasma donors. Saving the donor information and notifying about the current donors would be ahelping hand as it can save time and help the users to track down the necessary information about the donors.

1.2 PURPOSE

The main aim of developing this system is to provide blood to the people who are inneed of plasma. The numbers of persons who are in need of plasma are increasing in large number day by day. Using this system user can search blood group available in thecity and he can also get contact number of the donor who has the same blood group he/she needs for plasma. In order to help people who are in need of plasma, this plasma donor application can be used effectively for getting the details of available plasma and user can also get contact number of the plasma donors having the same blood group and within the same city.

.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING PROBLEM

- **TITLE:** Instant Plasma Donor Recipient Connector web application.

AUTHOR: Kalpana Devi Guntoju[1], Swinish Jalli[2]

The world is suffering from the COVID 19 crisis and no vaccine has been found yet.. But there is another scientific way in which we can help reduce mortality or help people affected by COVID19 by donating plasma from recovered patients. In the absence of an approved antiviral treatment plan for a fatal COVID19 infection, plasma therapy is an experimental approach to treat COVID19-positive patients and help them faster recovery. Therapy is considered competent. In the recommendation system, the donor who wants to donate plasma can donate by uploading their COVID19 certificate and the blood bank can see the donors who have uploaded the certificate and they can make a request to the donor and the hospital can register/login and search for the necessary things. plasma from a blood bank and they can request a blood bank and obtain plasma from the blood bank. The main goal of our project is to design a user-friendly web application thatis like a scientific vehicle from which we can help reduce mortality or help those affected by COVID19 by donating plasma from patients who have recovered without approved anti-retro-viral therapy planning for a deadly COVID19 infection,plasma therapy is an experimental approach to treat those COVID-positive patientsand help them recover faster. Therapy, which is considered reliable and safe. If a particular person has fully recovered from COVID19, they are eligible to donate their plasma. As we all know, the traditional methods of finding plasma, one has to find out for oneself by looking at hospital records and contacting donors have beenrecovered, sometimes may not be available at home and move to other places. In this type of scenario, the health of those who are sick becomes disastrous. Therefore, it is not considered a rapid process to find plasma.

- **TITLE :** A Web Application to Manage All Blood Donation and Transfusion Processes.

AUTHOR: Rehab S. Ali[1], Tamer F. Hafez[2], Ali Badawey Ali[3].

Many lives could be lost due to the difficulty in obtaining a proper blood bag, Therefore, this work aims to help citizens fulfill their needs for a safe and reliable blood group by searching for and locating a specific blood group. In this paper, weillustrate the problem of the blood bags shortage which is represented in the uncontrolled blood banks and parallel markets, lack of awareness and confidence,disappearance of the rare blood groups, and the difficulty in finding a specific blood group. Hence, we proposed the Blood Bag web-based application that is connected to a centralized database to gather and organize the data from all bloodbanks and blood donation campaigns. The proposed application organizes and controls the whole critical processes related to blood donation, testing and storageof blood bags, and delivering it to the patient. One blood bag can save a life during surgeries or road accidents, etc. Usually patients or their families look for a specific blood group they indeed need in the blood banks but they normally cannotfind it due to the shortage of blood bags. This is because of the fear of donating blood and the misconception that donating blood is harmful and transmits diseases. This is one of the obstacles to provide the blood bags. The availability ofthe blood bags is critical because of the high proportion of patients with renal failure, some cases of birth, surgeries processes and incidents that need to get theblood as soon as possible to save these cases' lives. The blood bank is the pool ofdifferent blood groups where keeping a stockpile of blood to be distributed in case.The matched blood groups for a safe transfusion . Accidents (or any medical emergency and compensation of blood missing from the body), and keeping the blood in the freezer temperature.

- **TITLE :** Developing a plasma donor application using Function-as-a-service in AWS.

AUTHOR: Aishwarya R Gowri1

Plasma is a liquid portion of the blood, over 55% of human blood is plasma. Plasma is used to treat various infectious diseases and it is one of the oldest methods known as plasma therapy. Plasma therapy is a process where blood is donated by recovered patients in order to establish an antibody that fights the infection. In this project plasma donor application is being developed by using AWS services. The services used are AWS Lambda, API gateway, Dynamo DB, AWSElastic Compute Cloud with the help of these AWS services, it eliminates the need of configuring the servers and reduces the infrastructural costs associated with it and helps to achieve serverless computing. For instance, during COVID 19 crisis the requirement for plasma increased drastically as there were no vaccination found in order to treat the infected patients, with plasma therapy the recovery rateswhere high but the donor count was very low and in such situations it was very important to get the information about the plasma donors. Saving the donor information and notifying about the current donors would be a helping hand as it can save time and help the users to track down the necessary information about the donors. The proposed method helps the users to check the availability of donors. A donor has to register to the website providing their details. The registered users can get the information about the donor count of each blood group. The database will have all the details such as name, email, phone number, infected status. Whenever a user requests for a particular blood group then the concerned blood group donors will receive the notification regarding the requirement. A Json code is written to store the information, to fetch the requestedinformation in lambda.

- **TITLE :** Nearest Blood & Plasma Donor Finding: A Machine Learn in Approach.

AUTHOR: Nayan Das1, Asif Iqbal2.

The necessity of blood has become a significant concern in the present context all over the world. Due to a shortage of blood, people couldn't save themselves or their friends and family members. A bag of blood can save a precious life. Statistics show that a tremendous amount of blood is needed yearly because of major operations, road accidents, blood disorders, including Anemia, Hemophilia, and acute viral infections like Dengue, etc. Approximately 85 million people require single or multiple blood transfusions for treatment. Voluntary blood donors per 1,000 population of some countries are quite promising, such as Switzerland (113/1,000), Japan (70/1,000), while others have an unsatisfying resultlike India has 4/1,000, and Bangladesh has 5/1000. Recently a life-threatening virus, COVID-19, spreading throughout the globe, which is more vulnerable for older people and those with pre-existing medical conditions. For them, plasma is neededto recover their illness. Our Purpose is to build a platform with clustering algorithms which will jointly help to provide the quickest solution to find blood or plasma donor. Closest blood or plasma donors of the same group in a particular area can be explored within less time and more efficiently. Keywords—Blood donation, Plasma donation, K means clustering, Labeled Agglomerate clustering. Different methods have been used to solve this problem. This time, we have tried another way, a clustering approach, to solve the problem by grouping every user into small groups. This unsupervised machine learning approach is much faster and effective. In section II, we will discuss related work done previously to solve this problem. In section III, clustering algorithms relating to our project will explicitly be discussed. In section IV, our proposed method will be presented. In section V, we will analyze our experiment result.

- **TITLE :** Securing Information on a Web Application System to Facilitate Online Blood Donation Booking.

AUTHOR: Hrishitva Patel[1].

Blood donation has saved many lives in the past. According to the American Red Cross statistics, a patient needs a blood transfusion every two seconds. Many benefits arise from blood donation to both the donor and the blood recipients. With blood donation, cancer patients, people involved in accidents, or those battling diseases that require blood donation have access to enough blood to sustain their survival. There is a need to digitize the blood donation booking to facilitate blood donation across the United States, and ensure patients in need of blood, receive their donation from eligible donors on time. This report demonstrates the security measures implemented to secure patient and blood donor data on a blood donation booking web application. Blood is donated for different reasons in hospitals and other blood banks. It is essential to help blood recipients survive surgeries, cancer treatment, and chronic illnesses, among other illnesses. The World Health Organization describes blood as the most precious gift a person can give to a person in need of it. Blood donated comprises four components: platelets, plasma, white blood cells, and red blood cells. Cancer patients require a blood transfusion to enhance platelets back into the body after radiation therapy. With a developed web application to book a blood donation session, it is easier for hospitals to know which blood component they need most and thus inform the system administrator to prompt for more blood donors.

2.2 REFERENCES

1. Kalpana Devi Guntoju[1], Tejaswini Jalli[2], Instant Plasma Donor RecipientConnector web application, 2022.
2. Rehab S. Ali[1], Tamer F. Hafez[2], Ali Badawey Ali[3], A Web Application to Manage AllBlood Donation and Transfusion Processes, 2017.
3. Aishwarya R Gowri, Developing a plasma donor application using Function-as-a-service in AWS, 2020.
4. Nearest Blood & Plasma Donor Finding: A Machine Learning Approach, 2021
5. Hrishitva Patel, Securing Information on a Web Application System to FacilitateOnline Blood Donation Booking, 2022.

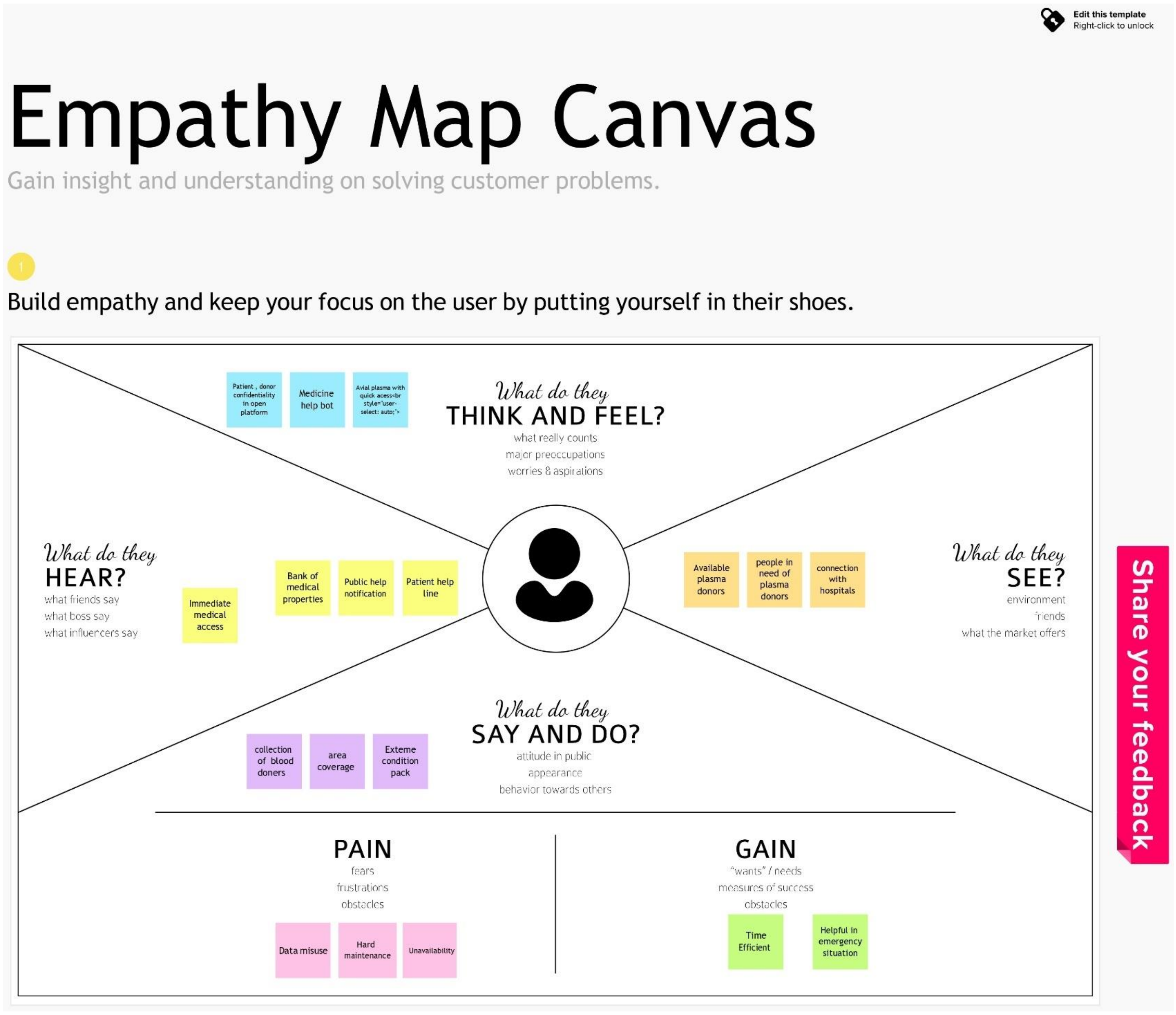
2.3 PROBLEM STATEMENT DEFINITION

In an effort to maintain the inventory, the safety of donors and staffs cannot be jeopa In critical or emergency situations where accident occurs or during on-going treatments and surgeries etc there is urgent need for specific blood group. It requires lot of time to make the blood available and it is inconvenient during emergency situation, some rare blood groups are time consuming and difficult to arrange which are O- , AB- etc. In our country there is less awareness of blood donation, near about 20% of Indian population donates blood. In existing system the blood bank management system exhibited at a lot of ineffectiveness and inefficiency that had fetched impact taken by management. The system which was manual that is based on paper card to collect blood donor data, keep record of blood donors and disseminate results to blood donors, had weakness that needed IT based solutions

CHAPTER 3

IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



3.2 IDEATION & BRAINSTORMING

Ideation Phase
Brainstorm & Idea Prioritization

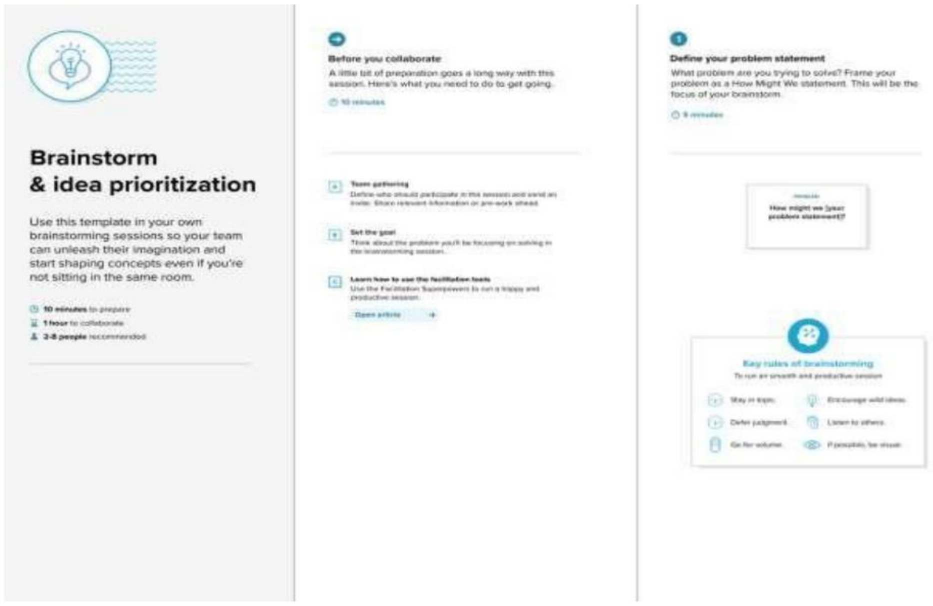
Date	19 September 2022
Team ID	PNT2022TMID00767
Project Name	Project – Plasma Donor Application
Maximum Marks	4 Marks

Brainstorm & Idea Prioritization Template:

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

Step-1: Team Gathering, Collaboration and Select the Problem Statement



Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

10 minutes to prepare
1 hour to collaborate
2-8 people recommended

Before you collaborate
A little bit of preparation goes a long way with this session. Here's what you need to do to get going.
10 minutes

Team gathering
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

Set the goal
Think about the problem you'll be focusing on solving in the brainstorming session.

Learn how to use the facilitation tools
Use the Facilitation Superpowers to run a happy and productive session.
Open article

Define your problem statement
What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.
5 minutes

How might we [your problem statement]?

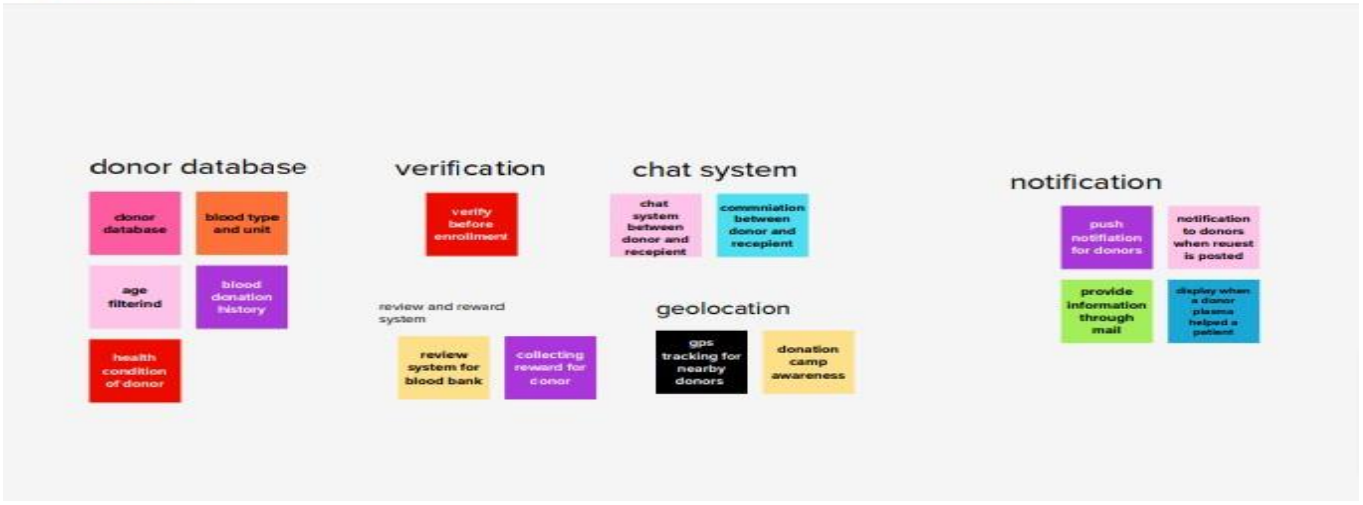
Key rules of brainstorming
To run an smooth and productive session

Stay in topic.	Encourage wild ideas.
Defer judgment.	Listen to others.
Go for volume.	If possible, be visual.

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

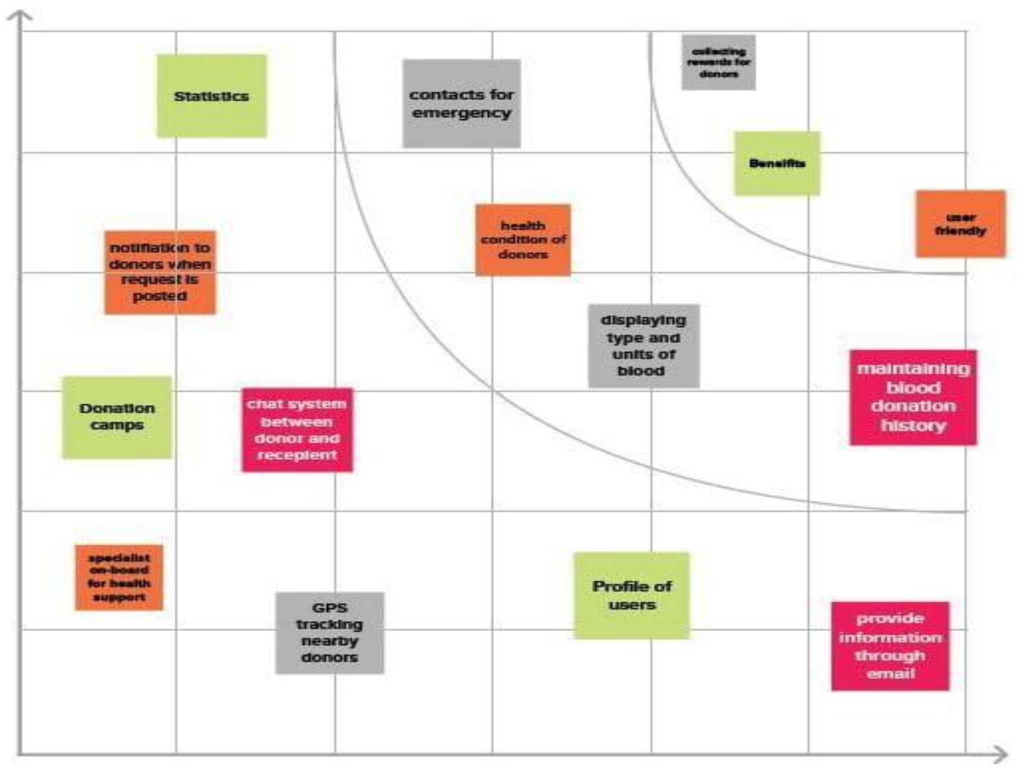
⏱ 20 minutes


donor database
donor database
blood type and unit
age filterind
blood donation history
health condition of donor
verification
verify before enrollment
review and reward system
review system for blood bank
collecting reward for donor
chat system
chat system between donor and recepient
commniation between donor and recepient
geolocation
gps tracking for nearby donors
donation camp awareness
notification
push notifiation for donors
notification to donors when reuest is posted
provide information through mail
display when a donor plasma helped a patient

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

⏱ 20 minutes



3.3 PROPOSED SOLUTION

The new idea will improve the existing system and it will move from conventional desktop system to mobile system. This paper introduces new features of improved system over existing system in many aspects. The proposed plasma donor application helps the people who are in need of plasma by giving them all details of plasma availability or regarding the donors with the same blood group. This is a web applicationallows you to access the whole information about plasma donor application, readily scalable and adaptable to meet the complex need of plasma Who are Key Facilitator forthe Healthcare Sector, it also supports all the functionalities of plasma donor application.

S. No.	Parameter	Description
1.	Problem Statement (Problemto be solved)	In existing system the blood bank management system exhibited at a lot of ineffectiveness and inefficiency that had fetched impact taken by management.
2.	Idea / Solution description	It saves time as he can search donorsonline without going anywhere.
3.	Novelty / Uniqueness	It is a user-friendly application.

4.	Social Impact / Customer Satisfaction	Accurate findings increase people's satisfaction.
5.	Business Model (RevenueModel)	-
6.	Scalability of the Solution	This system proposed here aims at connecting the donors & the patients by an online application.

3.4 PROBLEM SOLUTION FIT

1. CUSTOMER SEGMENTS Donor has to upload their blood group details for plasma donation.	**5. AVAILABLE SOLUTIONS** Get information about all the blood campaigns.	**8. CHANNELS OF BEHAVIOUR** It connects plasma donors and recipients through a Single and scalable platform.
2. JOBS TO BE DONE / PROBLEM Ineffectual to get the details systematically	**6 CUSTOMER CONSTRAINTS** Takes more time to get the information	**9. PROBLEM ROOT CAUSE** There isn't a systematic approach to gather donor information rapidly.
3. TRIGGERS It saves time as he can search donors online without going anywhere.	**7 BEHAVIOUR** Search donors of suitable blood groups and contact them if needed.	**10. YOUR SOLUTION** This system proposed here aims at connecting the donors & the patients by an online application. By usingthis application, the users can either raise a request for plasma donation or requirement. This system is used if anyone needs a Plasma Donor.

	4. EMOTIONS: BEFORE / AFTER Before, searching for donors took a lot of time. After, The people in need of plasma can search for the donors by giving their blood group and city name.			

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

1. Admin:

Admin can manage both donors and users. Admin has the only responsibility maintain and stored the record.

2. Users:

From this module user can create their account, when user create his account the user get a user id and password which identifies him uniquely. From this moduleuser can search donor for blood.

3. Donors Registration:

In this module, people who are interested in donating blood get registered in thissite and give his overall details related to donor. User details contain name, address,city, gender, blood group, location, contact number etc.

4. Donor Search:

The people who are in need of blood can search in our site for getting the detailsof donors having the same blood group and within the same city.

5. Notification:

In this module, notification sends to donors for emergency. SMS send to registered donors phone number.

4.2 NON FUNCTIONAL REQUIREMENTS

1.Usability:

The system shall allow the users to access the system with pc using web application. The system uses a web application as an interface. The system is user

friendly which makes the system easy

2.Availability:

The system is available 100% for the user and is used 24 hrs a day and 365 days a year. The system shall be operational 24 hours a day and 7 days a week.

3. Scalability :

Scalability is the measure of a system's ability to increase or decrease in performance and cost in response to changes in application and system processingdemands.

4. Security:

A security requirement is a statement of needed security functionality that ensures one of many different security properties of software is being satisfied.

5. Performance:

The information is refreshed depending upon whether some updates have occurred or not in the application. The system shall respond to the member in not less than two seconds from the time of the request submittal. The system shall be allowed to take more time when doing large processing jobs. Responses to view information shall take no longer than 5 seconds to appear on the screen.

6. Reliability:

The system has to be 100% reliable due to the importance of data and the damages that can be caused by incorrect or incomplete data. The system will run 7days a week. 24 hours a day

CHAPTER 5

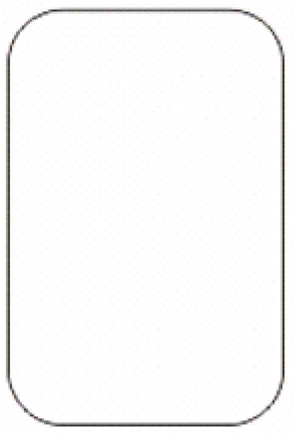
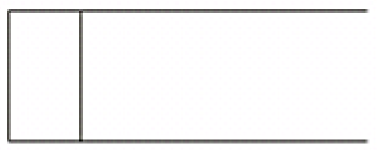
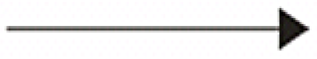
PROJECT DESIGN

5.1 DATA FLOW DIAGRAM

A data-flow diagram is a visual representation of how data moves through a system or a process (usually an information system). The DFD additionally gives details about each entity's inputs and outputs as well as the process itself. A data-flow diagram lacks control flow, loops, and decision-making processes. Using a flowchart, certain operations depending on the data may be depicted

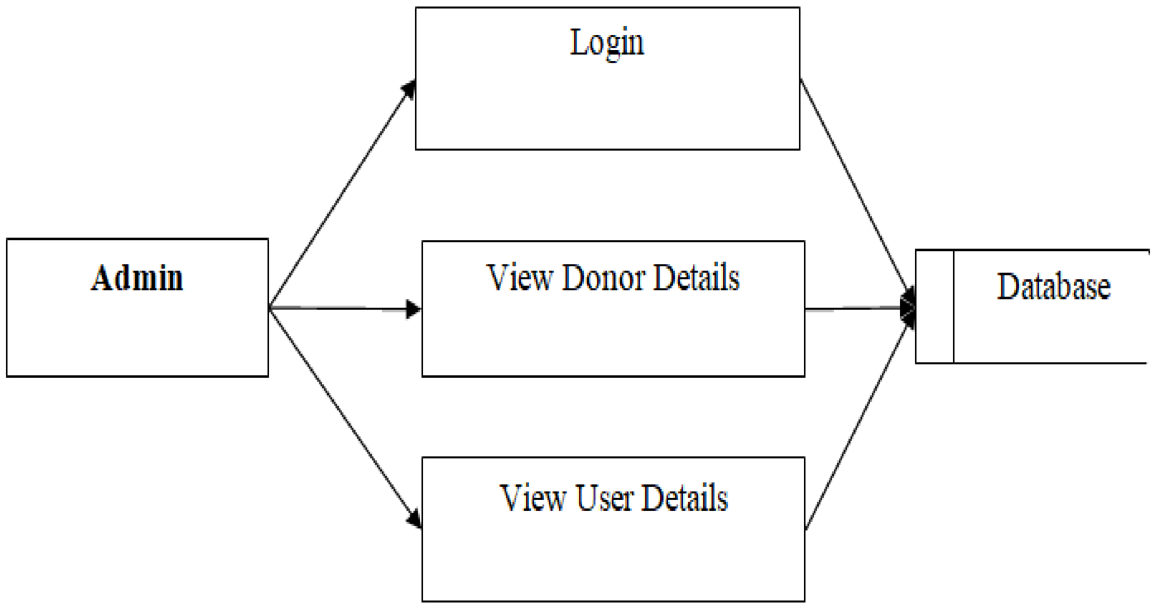
Data flow Symbols:

Symbol	Description
	An **entity**. A source of data or adestination for data.

	A **process** or task that is performed by the system.
	A **data store**, a place where data is held between processes.
	A **data flow**.

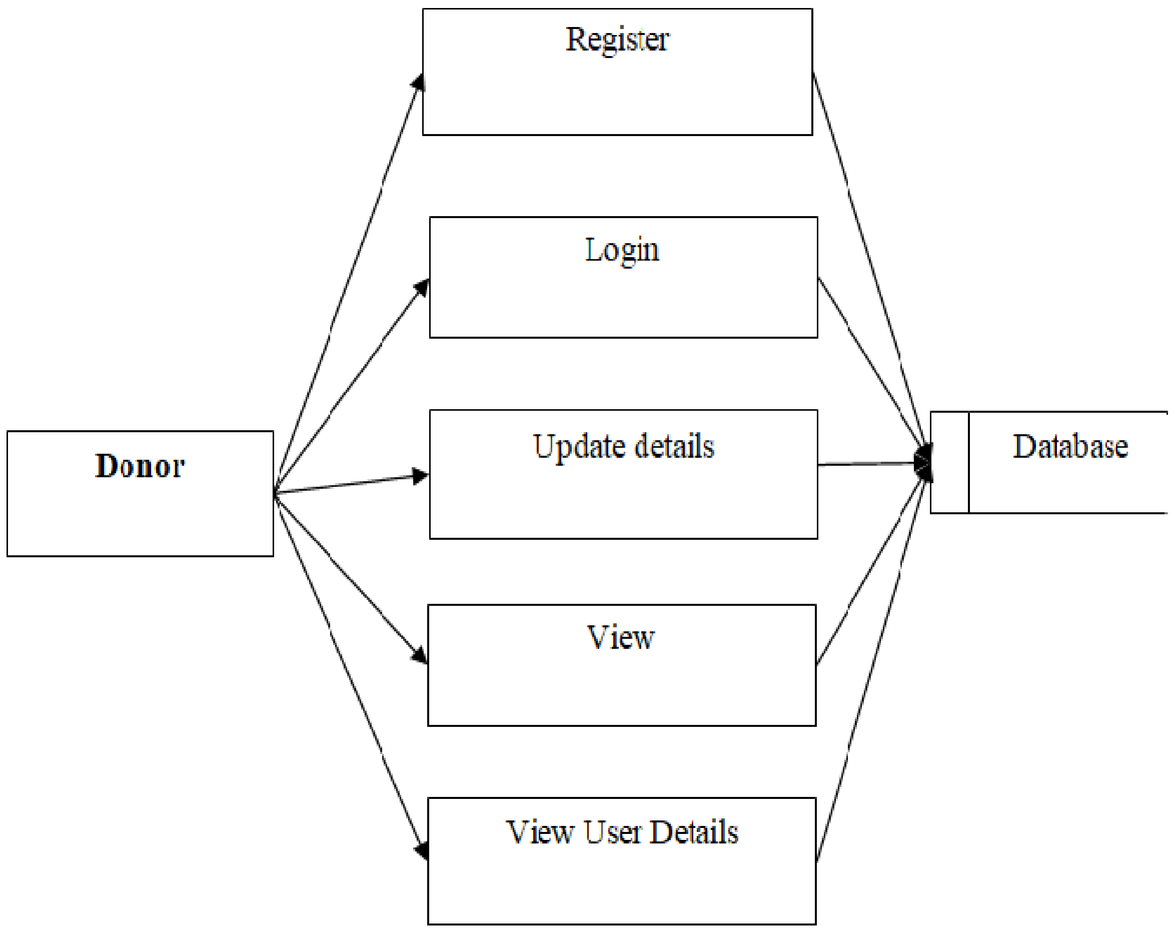
LEVEL 0:

The Level 0 DFD shows how the system is divided into 'sub-systems' (processes),each of which deals with one or more of the data flows to or from an external agent, andwhich together provide all of the functionality of the system as a whole. It also identifiesinternal data stores that must be present in order for the system to do its job, and shows the flow of data between the various parts of the system.



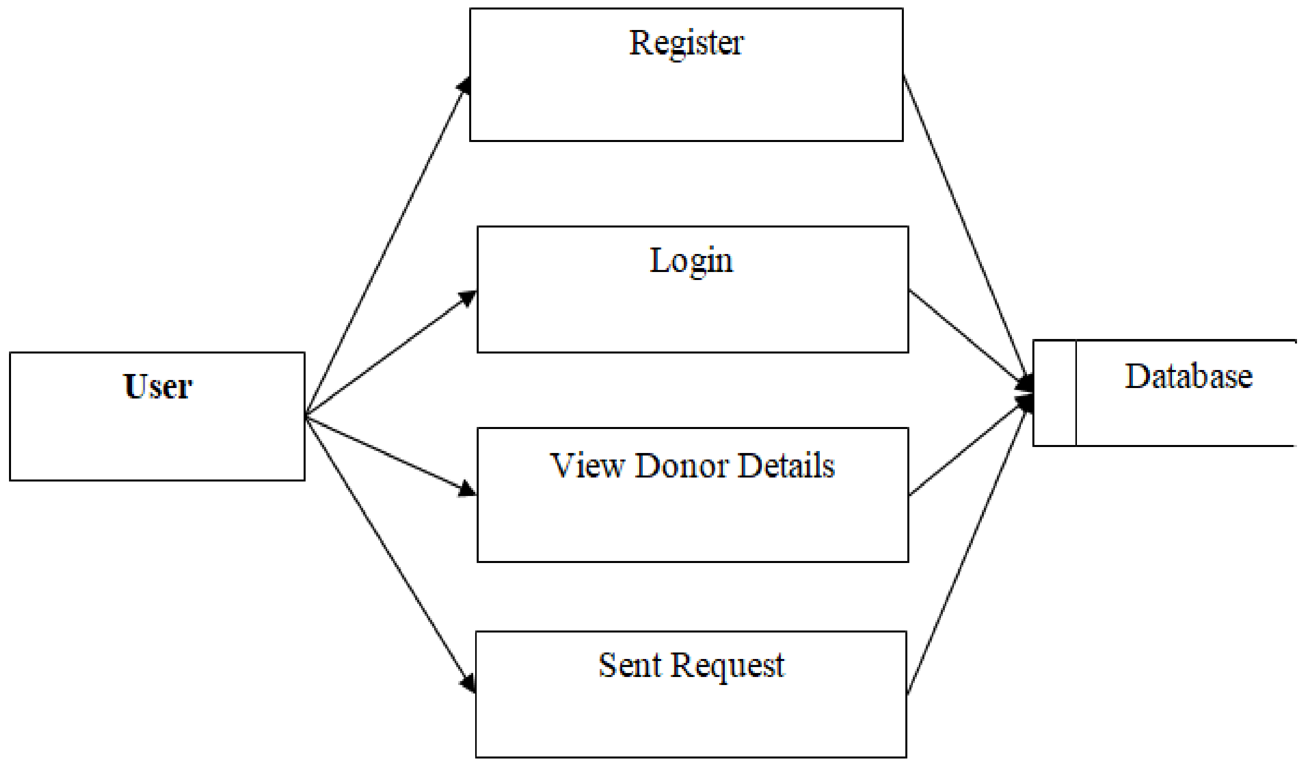
LEVEL 1:

The next stage is to create the Level 1 Data Flow Diagram. This highlights the main functions carried out by the system. As a rule, to describe the system was using between two and seven functions - two being a simple system and seven being a complicated system. This enables us to keep the model manageable on screen or paper.



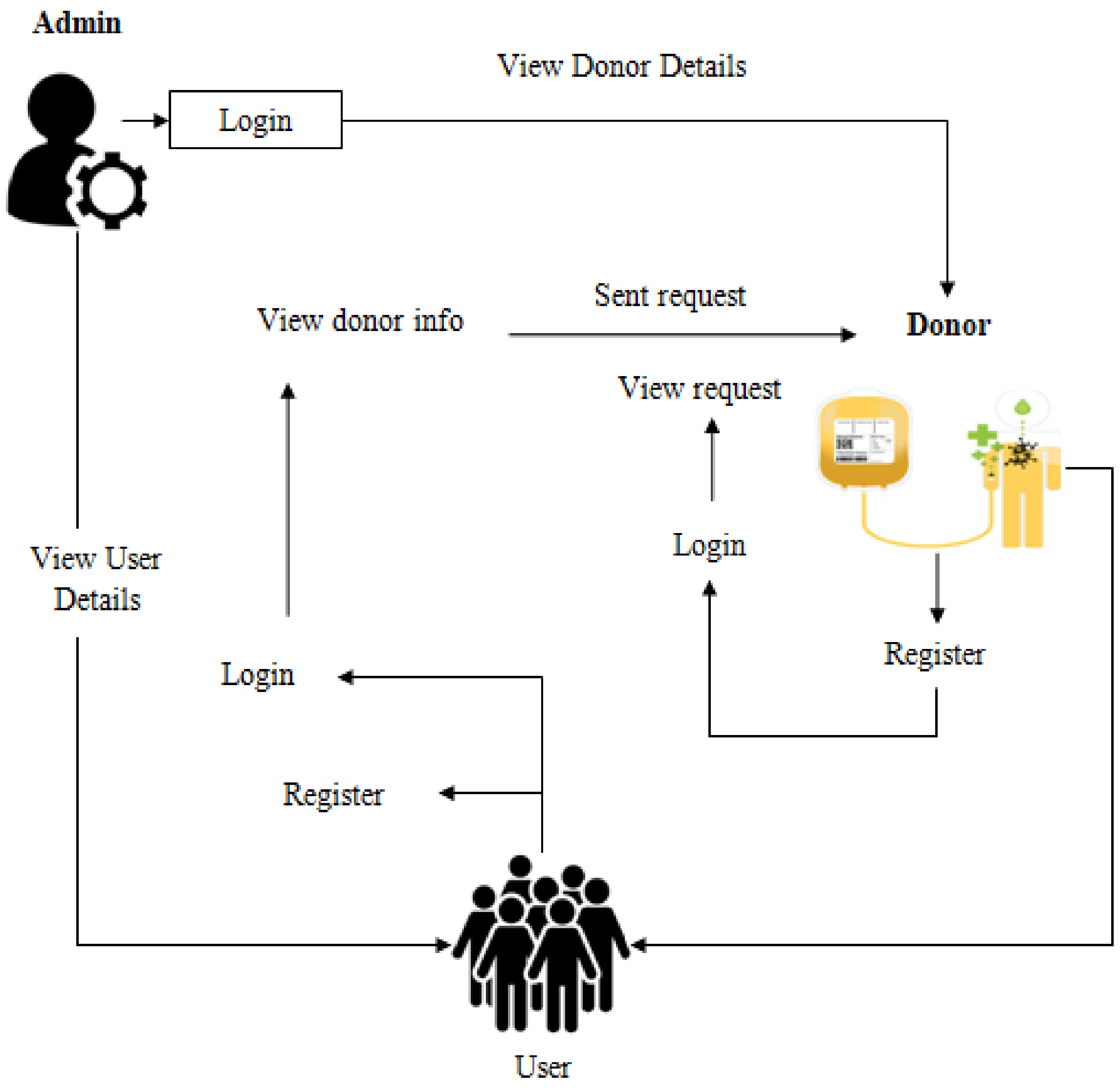
LEVEL 2:

A Data Flow Diagram (DFD) tracks processes and their data paths within the business or system boundary under investigation. A DFD defines each domain boundaryand illustrates the logical movement and transformation of data within the defined boundary. The diagram shows 'what' input data enters the domain, 'what' logical processes the domain applies to that data, and 'what' output data leaves the domain. Essentially, a DFD is a tool for process modeling and one of the oldest.



5.2 SOLUTION & TECHNICAL ARCHITECTURE

System architecture, also known as systems architecture, is the conceptual model that describes a system's structure, behavior, and additional viewpoints. An architecture description is a formal description and representation of a system that is organized in a way that allows for reasoning about the system's structures and behaviors. System architecture might include system components, their outwardly evident attributes, and the connections (e.g., behavior) between them. It can give a strategy for acquiring items and developing systems that will operate together to accomplish the entire system. There have been initiatives to codify languages for describing system architecture; they are referred to collectively as architecture description languages (ADLs).



5.3 USER STORIES

S.No	Component	Description	Technology
1.	User Interface	Web UI	HTML, CSS, JavaScript
2.	Application Logic-1	Logic for a process in theapplication	Python

3.	Application Logic-2	Logic for a process in the application	SMTP
4.	Application Logic-3	Logic for a process in the application	SMTP
5.	Database	Data Type, Configurations etc.	MySQL
6.	Cloud Database	Database Service on Cloud	Local host
7.	File Storage	File storage requirements	Local host
8.	External API-1	Purpose of External API used in the application	-
9.	External API-2	Purpose of External API used in the application	-
10.	Machine Learning Model	Purpose of Deep Learning Model	Classifies and detect the images with high accuracy
11.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	Local Server Configuration

APPLICATION CHARACTERISTICS:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	PYCHARM
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	-
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	Able to respond the changes in an application
4.	Availability	Justify the availability of application (e.g. use of load balancers, Distributed servers etc.)	The system must always be functional
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	Takes no longer time to response

5.4 USER STORIES:

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story/ Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmati on email once I have registered for the application	I can receive confirmation email &click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Gmail	I can receive confirmation notifications through Gmail	Medium	Sprint-1

	Login	USN-4	As a user, I canlogin to the application by entering email &password	I can accessinto my User profile and view details in dashboard	High	Sprint-1
	Dashboard	USN-5	As a user,I cansend the proper requests to donate andobtain plasma.	I can receive appropriate notificationsthrough email	High	Sprint-1
Customer (Web user)	Login	USN-6	As a user,Ican register and log into the application by entering email& password to view the profile	I can accessinto my User profile and view details in dashboard	High	Sprint-1
	Dashboard	USN-7	As a user,I cansend the proper requests to donate andobtain plasma.	I can receive appropriate notificationsthrough email	High	Sprint-1

Custome rCare Executive	Applicat ion	USN-8	As a customer care executive,Ican try to address user’s concerns and questions	I can view and address their concerns and questions	Medium	Sprint-2
Administr ator	Applicat ion	USN-9	As an administrator Ican help with user-facing aspects of a website, likeits appearance ,navigation and use ofmedia.	I can change the appearance and navigatio nin a userfriendly manner	Medium	Sprint-3
		USN -10	As an administrator, I can involve working withthe technical side ofwebsites.	I can helpwith such as troubleshoo ting issues, setting up web hosts,ensuri ng users have-access and programmi ng servers	Medium	Sprint-1

CHAPTER 6

PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

SPRINT	USER STORY / TASK	STORY POINTS	PRIORITY	TEAM MEMBERS
Sprint – I	Get the dataset	3	High	Lokesh Raj D
	Explore the data	2	Medium	Lokesh Raj D Dinesh Kumar E
	Data Pre-Processing	3	High	Nishanth R J Hari Haran S
	Prepare training and testing data	3	High	Nishanth R J Hari Haran S
Sprint – II	Create the model	3	High	Nishanth R J
	Train the model	3	High	Hari Haran S
	Test the model	3	High	Dinesh Kumar E
Sprint – III	Improve the model	2	Medium	Nishanth R J Hari Haran S
	Save the model	3	High	Lokesh Raj D
	Build the Home Page	3	High	Dinesh Kumar E Lokesh Raj D
	Setup a database to store input images	2	Medium	Nishanth R J

Sprint – IV	Build the results page	3	High	Dinesh Kumar E Lokesh Raj D
	Integrate the model with the application	3	High	Dinesh Kumar E Nishanth R J
	Test the application	3	High	Hari Haran S Dinesh Kumar E

6.2 SPRINT DELIVERY SCHEDULE

SPRINT	TOTAL STORY POINTS	DURATION	SPRINT START DATE	SPRINT END DATE (PLANNED)	STORY POINTS COMPLETED (AS ON PLANNED DATE)	SPRINT RELEASE DATE (ACTUAL)
Sprint – I	11	6 Days	24 Oct 2022	29 Oct 2022	11	29 Oct 2022
Sprint – II	9	6 Days	31 Oct 2022	05 Nov 2022	9	05 Nov 2022
Sprint – III	10	6 Days	07 Oct 2022	12 Nov 2022	10	12 Nov 2022
Sprint – IV	9	6 Days	14 Nov 2022	19 Nov 2022	9	19 Nov 2022

CHAPTER 7

CODING & SOLUTIONING

7.1 Feature 1

```
#main.py
# importing
libraries from flask
import Flask
from flask_mail import Mail,
Message app = Flask(_name_)
mail = Mail(app)  # instantiate the
mail class # configuration of mail
app.config['MAIL_SERVER'] =
'smtp.gmail.com'
app.config['MAIL_PORT'] = 465
app.config['MAIL_USERNAME'] =
'sampletest685@gmail.com'
app.config['MAIL_PASSWORD'] =
'hneucvnontsuwgpj' app.config['MAIL_USE_TLS'] =
False app.config['MAIL_USE_SSL'] = True
mail = Mail(app)
# message object mapped to a particular URL '/'
@app.route("/")
def index():
  msg =
    Message(
    'Hello',
    sender='sampletest685@gmail.co
    m',
    recipients=['sangeeth5535@gmail.c
    om']
  )
  msg.body = 'Hello Flask message sent from
  Flask-Mail' mail.send(msg)
  return 'Sent'
if___name___== '_main_':
  app.run(debug=True)
```

7.2 FEATURE-2:

```
-- phpMyAdmin SQL Dump
-- version 2.11.6
-- http://www.phpmyadmin.net
--
-- Host: localhost
-- Generation Time: Nov 05, 2022 at 04:58 AM
-- Server version: 5.0.51

-- PHP Version: 5.2.6
SET SQL_MODE="NO_AUTO_VALUE_ON_ZERO";
/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
--
-- Database: `1plasmadb`
--
--------------------------------

--
-- Table structure for table `admintb`
--
CREATE TABLE `admintb` (
  `UserName` varchar(250) NOT NULL,
  `Password` varchar(250) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
--
```

```
-- Dumping data for table `admintb`
--
INSERT INTO `admintb` (`UserName`, `Password`)
VALUES('admin', 'admin');
--------------------------------
--
-- Table structure for table `donortb`
--
CREATE TABLE `donortb` (
  `id` bigint(10) NOT NULL auto_increment,
  `Name` varchar(250) NOT NULL,
  `Mobile` varchar(250) NOT NULL,
  `Email` varchar(250) NOT NULL,
  `UserName` varchar(250) NOT NULL,
  `Password` varchar(250) NOT
 NULL,PRIMARY KEY  (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=2 ;
```

CHAPTER 8

TESTING

8.1 TEST CASES

A test case has components that describe input, action and an expected response, in order to determine if a feature of an application is working correctly.A test case is a set of instructions on “HOW” to validate a particular test objective/target, which when followed will tell us if the expected behavior of the system is satisfied or not.

Characteristics of a good test case:

1. Accurate: Exacts the purpose.
2. Economical: No unnecessary steps or words.
3. Traceable: Capable of being traced to requirements.
4. Repeatable: Can be used to perform the test over and over.
5. Reusable: Can be reused if necessary.

S.NO	Scenario	Input	Excepted output	Actual output
1	Admin Login Form	User name and password	Login	Login success.
2	Donor Registration Form	Donor basi cdetails	Registration	Donor registration details stored in database.

3	User Registration Form	User basi cdetails	Registration	User registration details stored in database.
4	User Login Form	User name and password	Login	Login success.

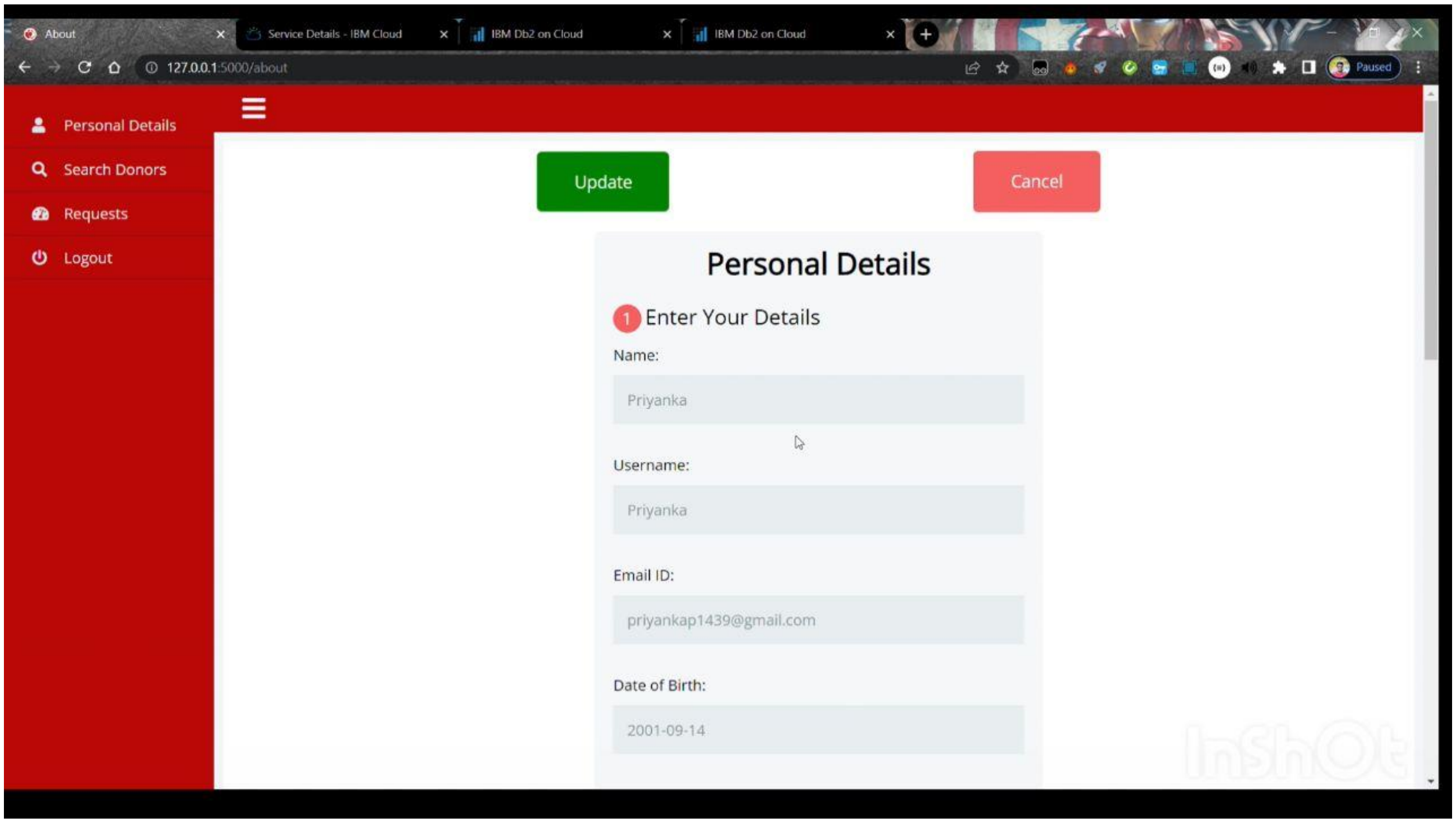
8.2 USER ACCEPTANCE TESTING

This is a type of testing done by users, customers, or other authorized entities to determine application/software needs and business processes. Acceptance testing is the most important phase of testing as this decides whether the client approves the application/software or not. It may involve functionality, usability, performance, and U.Iof the application. It is also known as user acceptance testing (UAT), operational acceptance testing (OAT), and end-user testing.

CHAPTER 9

RESULTS

9.1 PERFORMANCE METRICS



Personal Details

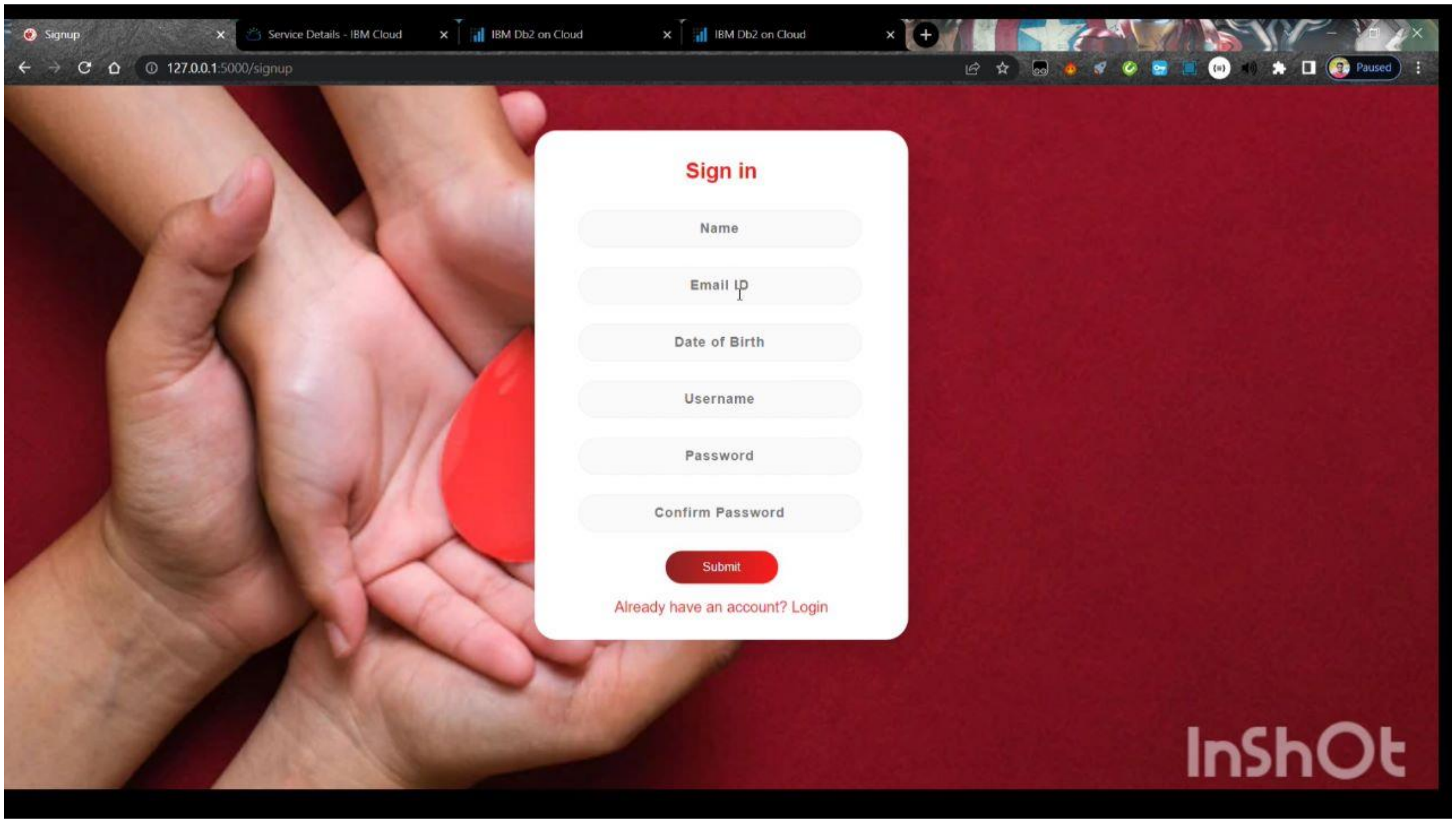
1 Enter Your Details

Name: Priyanka

Username: Priyanka

Email ID: priyankap1439@gmail.com

Date of Birth: 2001-09-14



Sign in

Name

Email ID

Date of Birth

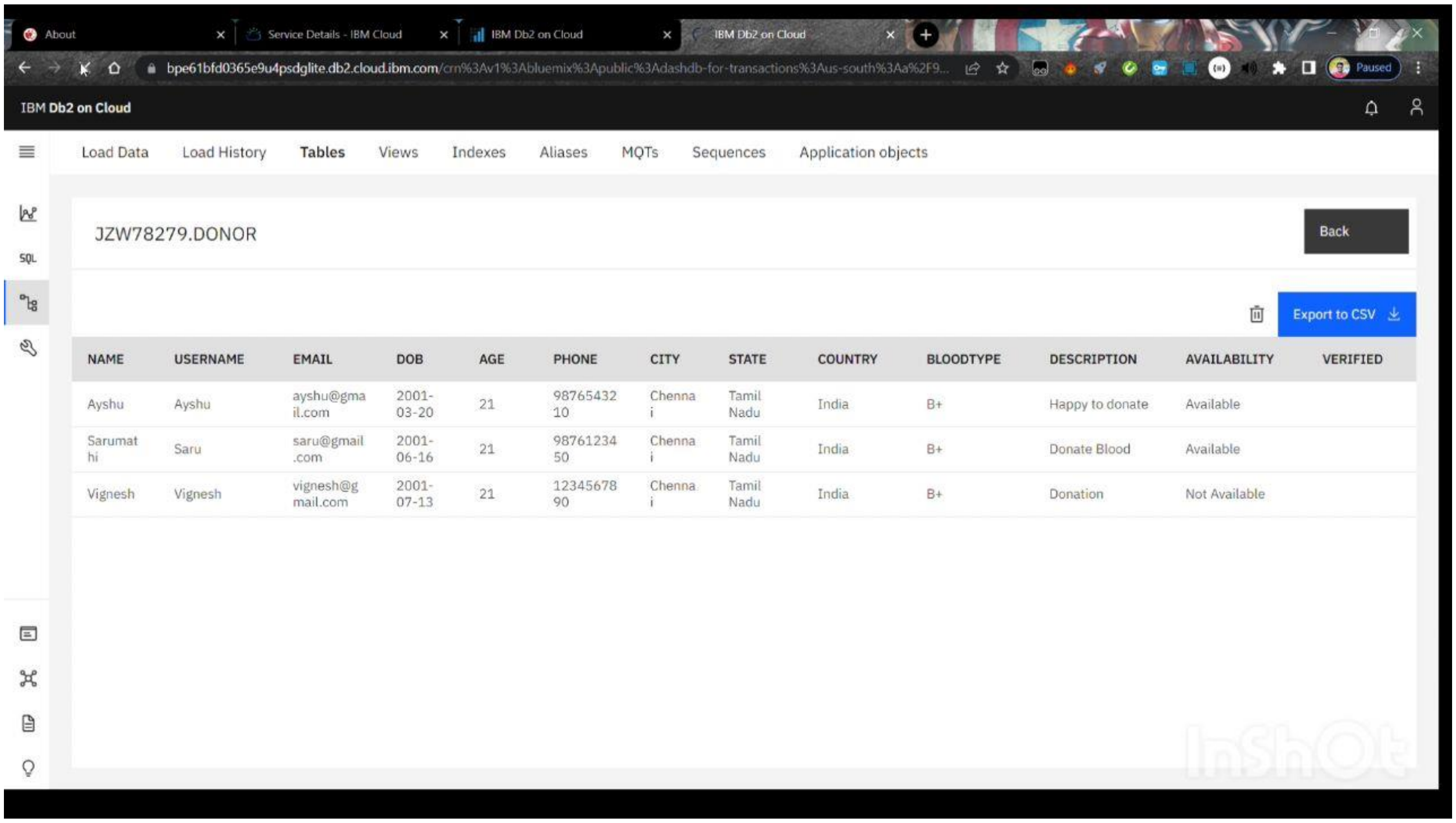
Username

Password

Confirm Password

Submit

Already have an account? Login


JZW78279.DONOR

NAME	USERNAME	EMAIL	DOB	AGE	PHONE	CITY	STATE	COUNTRY	BLOODTYPE	DESCRIPTION	AVAILABILITY	VERIFIED
Ayshu	Ayshu	ayshu@gmail.com	2001-03-20	21	9876543210	Chennai	Tamil Nadu	India	B+	Happy to donate	Available	
Sarumathi	Saru	saru@gmail.com	2001-06-16	21	9876123450	Chennai	Tamil Nadu	India	B+	Donate Blood	Available	
Vignesh	Vignesh	vignesh@gmail.com	2001-07-13	21	1234567890	Chennai	Tamil Nadu	India	B+	Donation	Not Available	

CHAPTER 10

ADVANTAGES & DISADVANTAGES

ADVANTAGES

1. It is a user-friendly application.

2. The people in need of plasma can search for the donors by giving their blood group and city name.

3. It saves time as he can search donors online without going anywhere.

4. Using this system user can get plasma in time and can save and here our system work, whenever a person needs plasma user get information of the person who has the same blood group needs.

DISADVANTAGES

1. It is time consuming.
2. It leads to error prone results.
3. It consumes lot of manpower to better results.
4. It lacks of data security.
5. Retrieval of data takes lot of time.

CHAPTER 11

CONCLUSION

This project is designed for successful completion of project on Plasma Donor Application system. The basic building aim is to provide plasma donation service to the city recently.Plasma Donor Application System is a Web based application that is designed to store, process, retrieve and analyze information concerned with the administrative and inventory management within a plasma. This project aims at maintaining all the information pertaining to plasma donors, different blood groups available in each plasma bank and helps them manage in a better way plasma donation system can collect plasma from many donors in short from various sources and distribute that plasma to needy people who require plasma. To do all this we require high quality Web Application to manage those jobs. Plasma application provides a reliable platform to connect local plasma donors with patients.

CHAPTER 12

FUTURE SCOPE

This system is developed such a way that additional enhancement can be done without much difficulty. The renovation of the project would increase the flexibility of the system. In future, we can develop this project in android platform. We will add extra features like donor location tracking system (GPS), Feedback form, and enable call option etc.

APPENDIX

SOURCE CODE

```
from flask import Flask, render_template, flash, request,
session,send_filefrom flask import render_template, redirect,
url_for, request
#from wtforms import Form, TextField, TextAreaField, validators,
StringField, SubmitFieldfrom werkzeug.utils import secure_filename
import datetime

from flask_mail import Mail,
Messageimport
mysql.connector
import sys

app = Flask(_name_
)
app.config['DEBUG'
]
app.config['SECRET_KEY'] =
'7d441f27d441f27567d441f2b6176a'@app.route("/")
def homepage():

   return
render_template('index.html')
@app.route("/AdminLogin")
def AdminLogin():

   return
render_template('AdminLogin.html')
@app.route("/DonorLogin")
def DonorLogin():

   return render_template('DonorLogin.html')
```

```
@app.route("/NewDon
or")def NewDonor():
   return
render_template('NewDonor.html')
@app.route("/UserLogin")
def UserLogin():

   return
render_template('UserLogin.html')
@app.route("/PersonalInfo")
def PersonalInfo():

   return
render_template('DonorPersonal.html')
@app.route("/NewUser")
def NewUser():

   return
render_template('NewUser.html')
@app.route("/AdminHome")
def AdminHome():

   conn = mysql.connector.connect(user='root', password='', host='localhost',
   database='1Plasmadb')cur = conn.cursor()
   cur.execute("SELECT * FROM
   regtb ")data = cur.fetchall()
   return
render_template('AdminHome.html',data=data
)@app.route("/AdminDonorInfo")
def AdminDonorInfo():

   conn = mysql.connector.connect(user='root', password='', host='localhost',
   database='1Plasmadb')
```

```
    cur = conn.cursor()

    cur.execute("SELECT * FROM
    personltb  ")data = cur.fetchall()
    return render_template('AdminDonorInfo.html',
data=data)@app.route("/UserHome")
def UserHome():

    user = session['uname']

    conn = mysql.connector.connect(user='root', password='', host='localhost',
    database='1Plasmadb')# cursor = conn.cursor()
    cur = conn.cursor()

    cur.execute("SELECT * FROM regtb where username='" +
    user + "'")data = cur.fetchall()
    return
render_template('UserHome.html',data=data
)@app.route("/DonorHome")
def DonorHome():

    cuname = session['cname']

    conn = mysql.connector.connect(user='root', password='', host='localhost',
    database='1Plasmadb')# cursor = conn.cursor()
    cur = conn.cursor()

    cur.execute("SELECT * FROM companytb where username='" +
    cuname + "'")data = cur.fetchall()
    return render_template('DonorHome.html', data=data)
```

```
@app.route("/adminlogin", methods=['GET',
'POST'])def adminlogin():
   error = None

   if request.method == 'POST':

     if request.form['uname'] == 'admin' or request.form['password'] == 'admin':

        conn = mysql.connector.connect(user='root', password='', host='localhost',
        database='1Plasmadb')# cursor = conn.cursor()
        cur = conn.cursor()
        cur.execute("SELECT * FROM
        regtb ")data = cur.fetchall()
        return render_template('AdminHome.html' ,
     data=data)else:
      return       render_template('index.html',
error=error)         @app.route("/donorlogin",
methods=['GET', 'POST'])def donorlogin():
   error = None

   if request.method == 'POST':
      username =
      request.form['uname']
      password =
      request.form['password']
      session['dname'] = request.form['uname']

      conn = mysql.connector.connect(user='root', password='', host='localhost',
      database='1Plasmadb')cursor = conn.cursor()
      cursor.execute("SELECT * from donortb where username='" + username + "' and
Password='" +password + "'")
      data =
      cursor.fetchone()if
      data is None:
         alert = 'Username or Password is wrong'

         return render_template('goback.html',
      data=alert)else:
         print(data[0])
         session['uid'] =
         data[0]
         conn = mysql.connector.connect(user='root', password='', host='localhost',
         database='1Plasmadb')# cursor = conn.cursor()
         cur = conn.cursor()

         cur.execute("SELECT * FROM donortb where username='" + username + "' and
Password='" +password + "'")
         data = cur.fetchall()

         return render_template('DonorHome.html',
data=data)@app.route("/userlogin",
methods=['GET', 'POST'])
def userlogin():

   if request.method == 'POST':
      username =
      request.form['uname']
      password =
      request.form['password']
      session['uname'] = request.form['uname']
```

```
        conn = mysql.connector.connect(user='root', password='', host='localhost',
        database='1Plasmadb')cursor = conn.cursor()
        cursor.execute("SELECT * from regtb where username='" + username + "'
and Password='" +password + "'")
        data =
        cursor.fetchone()if
        data is None:
          alert = 'Username or Password is wrong'

          return render_template('goback.html',
        data=alert)else:
          print(data[0])
          session['uid'] =
          data[0]
          conn = mysql.connector.connect(user='root', password='', host='localhost',
          database='1Plasmadb')# cursor = conn.cursor()
          cur = conn.cursor()

          cur.execute("SELECT * FROM regtb where username='" + username + "' and
Password='" +password + "'")
          data = cur.fetchall()

          return render_template('UserHome.html',
data=data )@app.route("/newuser",
methods=['GET', 'POST'])
def newuser():

  if request.method ==
        'POST': name1 =
        request.form['name']
        gender1 =
        request.form['gender']Age
        = request.form['age']
        email =
        request.form['email']
        pnumber =
        request.form['phone']
        address =
        request.form['address']



        uname =
        request.form['uname']
        password =
        request.form['psw']
        conn = mysql.connector.connect(user='root', password='', host='localhost',
        database='1Plasmadb')cursor = conn.cursor()
        cursor.execute(

          "INSERT INTO regtb VALUES ('" + name1 + "','" + gender1 + "','" + Age + "','" +
          email + "','" +

pnumber + "','" + address + "','" + uname + "','" +
        password + "')")conn.commit()
        conn.close()

        # return 'file register
    successfully' return
    render_template('UserLogin.html')
@app.route("/personal", methods=['GET',
'POST'])def personal():
    if request.method ==
        'POST': name1 =
        request.form['name']
        gender1 =
        request.form['gender']
        Age =
        request.form['age']
        email =
        request.form['email']
        pnumber =
        request.form['phone']
        address =
        request.form['address']
        blood =
        request.form['blood']
        health =
        request.form['health']
        dname = session['dname']
        conn = mysql.connector.connect(user='root', password='', host='localhost',
        database='1Plasmadb')cursor = conn.cursor()
        cursor.execute(

          "INSERT INTO personltb VALUES ('','" + name1 + "','" + gender1 + "','" + Age +
"','" + email +"','" + pnumber + "','" + address + "','" + blood + "','" + health + "','"+
dname+"')")
        conn.commi
        t()
        conn.close()
        alert = 'Record Saved'

        return render_template('goback.html',
data=alert)@app.route("/appr")
def appr():

    cid =
    request.args.get('cid')
    dname =
    session['dname']
    conn = mysql.connector.connect(user='root', password='', host='localhost',
    database='1Plasmadb')cursor = conn.cursor()
```

```
   cursor.execute(

      "delete from  personltb where id='" +
   str(cid) + "' ")conn.commit()
   conn.close()

   conn = mysql.connector.connect(user='root', password='', host='localhost',
   database='1Plasmadb')cur = conn.cursor()
   cur.execute("SELECT * FROM personltb where Username='"+
   dname +"' ")data = cur.fetchall()
   return render_template('DonorPersonalInfo.html',
data=data)@app.route("/DonorPersonalInfo")
def
   DonorPersonalInfo():
   dname =
   session['dname']
   conn = mysql.connector.connect(user='root', password='', host='localhost',
   database='1Plasmadb')cur = conn.cursor()
   cur.execute("SELECT * FROM personltb where Username='" +
   dname + "' ")data = cur.fetchall()
   return render_template('DonorPersonalInfo.html',
data=data)@app.route("/newdonor",
methods=['GET', 'POST'])
def newdonor():

   if request.method ==
      'POST': name1 =
      request.form['name']
      phone =
      request.form['phone']
      email =
      request.form['email']
      uname =
      request.form['uname']
      password =
      request.form['psw']
      conn = mysql.connector.connect(user='root', password='', host='localhost',
      database='1Plasmadb')cursor = conn.cursor()
      cursor.execute(

        "INSERT INTO donortb VALUES ('','" + name1 + "','" + phone + "','" + email +
"','" + uname +"','" + password + "')")
      conn.commi
      t()
      conn.close()
   return
render_template('DonorLogin.html')
@app.route("/Search")
def Search():

   return render_template('Search.html')
@app.route("/dsearch", methods=['GET',
'POST'])def dsearch():
   if request.form["submit"] ==
      "Submit":blood =
      request.form['blood']
      conn = mysql.connector.connect(user='root', password='', host='localhost',
      database='1Plasmadb')cur = conn.cursor()
      cur.execute("SELECT * FROM personltb  where blood ='" +
      blood + "'")data = cur.fetchall()
      return render_template('Search.html',
    data=data)elif request.form["submit"] ==
    "SendMail":
      blood =
      request.form['blood']
      info =
      request.form['info']
      conn = mysql.connector.connect(user='root', password='', host='localhost',
      database='1Plasmadb')cursor = conn.cursor()
      cursor.execute("SELECT  *  FROM  personltb where  Blood like '%" +
      blood + "%'")data = cursor.fetchall()
      for item in data:
        sendmsg(item[4],
        info)print(item[4])
      alert = 'Send Notication'

      return render_template('goback.html',
data=alert)@app.route("/SendRequest")
def SendRequest():

    session['cid'] =
    request.args.get('cid') return
    render_template('Notification.html'
    )
@app.route("/noti", methods=['GET',
'POST'])def noti():
    info =
    request.form['info']
    did = session['cid']
    print(did)
```

```
  conn = mysql.connector.connect(user='root', password='', host='localhost',
  database='1Plasmadb')cursor = conn.cursor()
  cursor.execute("SELECT  *  FROM  personltb where  id='" + did
  + "'")data = cursor.fetchone()
  if data:

    bloo
    =data[7]
    print(bloo)
  else:

    return 'Incorrect username / password !'

  conn = mysql.connector.connect(user='root', password='', host='localhost',
  database='1Plasmadb')cursor = conn.cursor()
  cursor.execute("SELECT  *  FROM  personltb where  Blood like '%" +
  bloo + "%'")data = cursor.fetchall()
  for item in data:
    sendmsg(item[4],
    info)print(item[4])
  alert = 'Send Notication'

  return render_template('goback.html',
data=alert)def sendmsg(Mailid,message):
  import smtplib

  from email.mime.multipart import
  MIMEMultipartfrom email.mime.text
  import MIMEText
```

```
from email.mime.base import
MIMEBase from email import
encoders
fromaddr =
"sampletest685@gmail.com"
toaddr = Mailid
# instance of
MIMEMultipart msg =
MIMEMultipart()
# storing the senders email
address msg['From'] =
fromaddr
# storing the receivers email
address msg['To'] = toaddr
# storing the
subject
msg['Subject'] =
"Alert"
# string to store the body of
the mail body = message
# attach the body with the msg
instance
msg.attach(MIMEText(body,
'plain'))
# creates SMTP session

s =
smtplib.SMTP('smtp.gmail.com',
587)# start TLS for security
s.starttls()

# Authentication

s.login(fromaddr, "hneucvnontsuwgpj")
```

```
    # Converts the Multipart msg into
    a stringtext = msg.as_string()
    # sending the mail
    s.sendmail(fromaddr,
    toaddr, text)# terminating
    the session
    s.quit()

if __name_== '_main_':
    app.run(debug=True,
    use_reloader=True)
```

GITHUB

https://github.com/IBM-EPBL/IBM-Project-4339-1658728862

PROJECT DEMO

https://drive.google.com/file/d/1xJDmLsdi93owP22Bcrjv-

MBwkLlRDttn/view?usp=share_link