PROJECT REPORT

TEAM ID : PNT2022TMID25930

PROJECT : PLASMA DONOR APPLICATION

DOMAIN : CLOUD APPLICATION DEVELOPMENT

TEAM MEMBERS:

1. NIVETHA.N
2. RUBETHA K
3. SANDHIYA.S
4. THIRUNILAINAYAGI.T

PROJECT REPORT - PLASMA DONOR APPLICATION

1.INTRODUCTION

1.1Project 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, IBM-48000-1662641884 Page No : 2 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

2. LITERATURE REVIEW

2.1 EXISTING PROBLEM:

TITLE: Instant Plasma Donor Recipient Connector web application. **AUTHOR:** Kalpana Devi Guntoju1, Swinish Jalli2 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, Page No : 3 IBM-48000-1662641884 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 antiretro-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 tofind 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. Ali1, Tamer F. Hafez2

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 Page No : 4 IBM-48000-1662641884 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 duringsurgeries 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 Go

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 Page No : 5 IBM-48000-166264188 4 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.

2.2 REFERENCES

1. Kalpana Devi Guntoju1, Tejaswini Jalli2, Instant Plasma Donor Recipient Connector web application, 2022.
2. Rehab S. Ali1, Tamer F. Hafez2, Ali Badawey Ali3, A Web Application to Manage All Blood 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 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 lotof 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.
In the emergency condition, sometimes it becomes very much difficult to look for the exact match of blood group of donor and acceptor. It may lead to delay in transaction of plasma within the specified amount of time. This application is providingeach entity the facility to approach nearby blood donors so that it will become much easier to search rare blood groups in the hour of need.

I am (USER) .
Donor has to upload their blood group details for plasma donation.

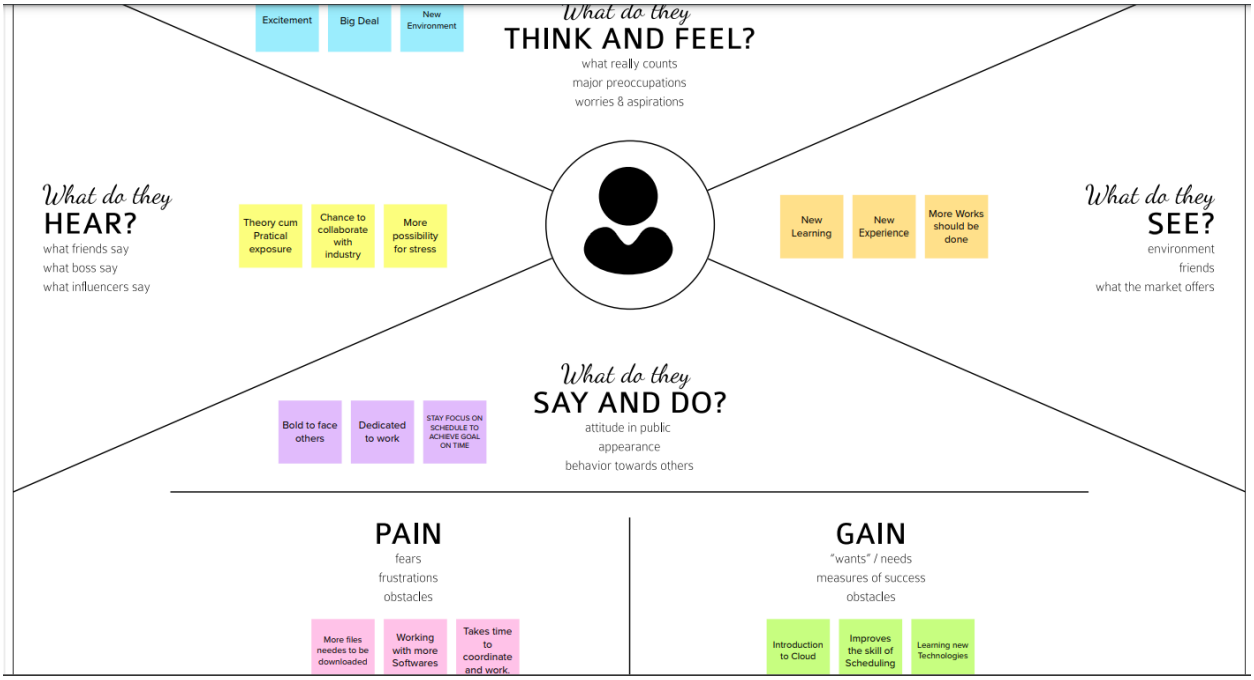
I am Trying to
The project blood bank management system is known to be a pilot project that is designed for the blood bank to gather blood from various sources and distribute it to theneedy people who have high requirements for it.

But
This might be the result of a human error, It leads to error prone results, It consumeslot of manpower to better results. Because It is hard, and there is a delay to know about the donor details.

Which makes me feel
If the blood group is not available in the blood bank user can request the donor to donate the plasma to him and save someone life. Using this system people can registerhimself or herself who want to donate plasma. To register in the system they have to enter their contact information like address mobile number etc.

3. IDEATION & PROPOSED SOLUTION

3.1Empathy Map Canvas

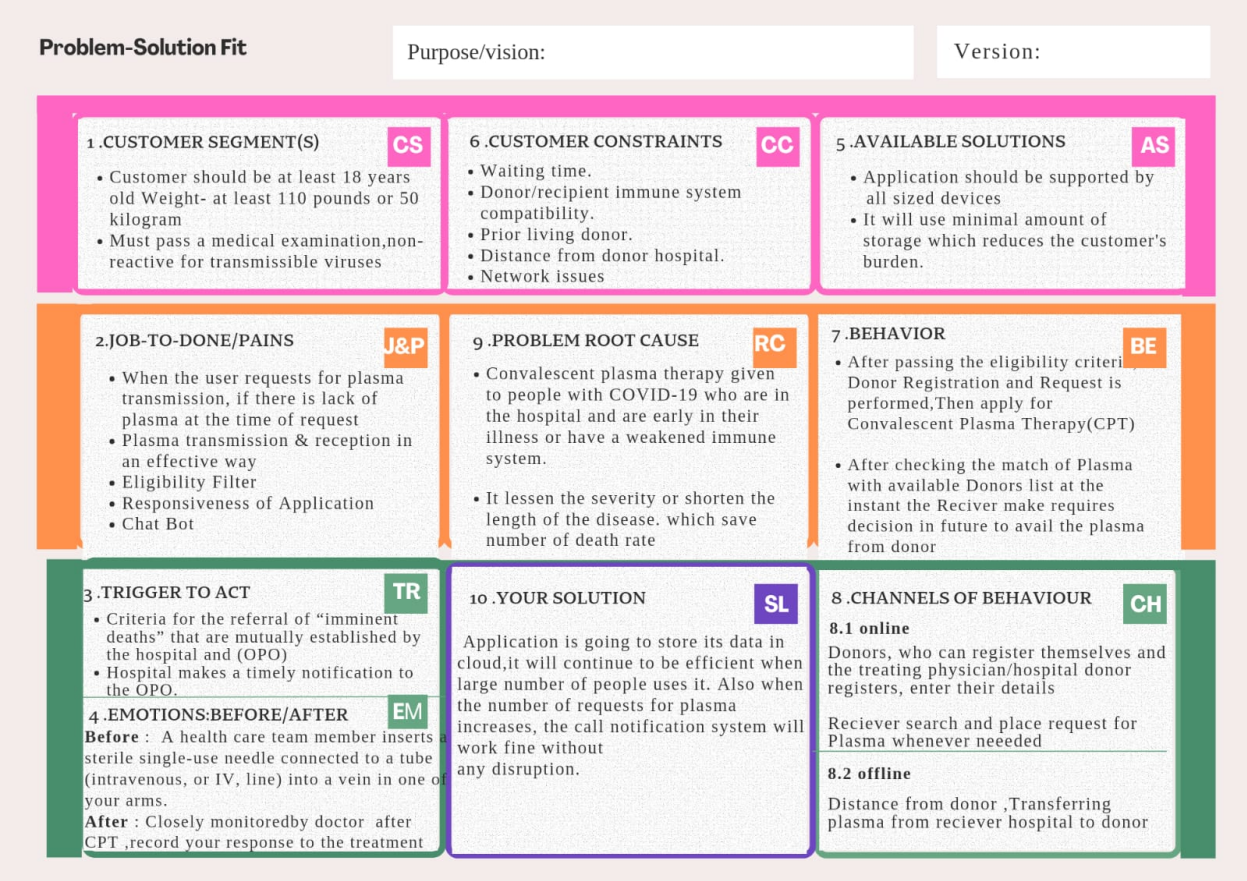


3.2 IDEATION AND BRAINSTORMING

3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To make Plasma transmission & reception in an effective way to save life of people in urgent circumstance.
2.	Idea / Solution description	Eligibility -Filter will be provided to check the eligibility of plasma donor and receiver Responsiveness -App should be supported by all sized devices
3.	Novelty / Uniqueness	When the user requests for plasma transmission, if there is lack of plasma at the time of request, automatically user will be marked in hold back list . Later when there is availability of plasma ,the receiver waiting in hold back list will be alerted via calling system
4.	Social Impact / Customer Satisfaction	Chat Bot - 24*7 support will be provided to tackle the issues of users which makes customer satisfied Storage - App will use minimal amount of storage which reduces the customer's burden.

3.4 Problem Solution Fit



4.REQUIREMENT ANALYSIS

4.1 Functional requirement

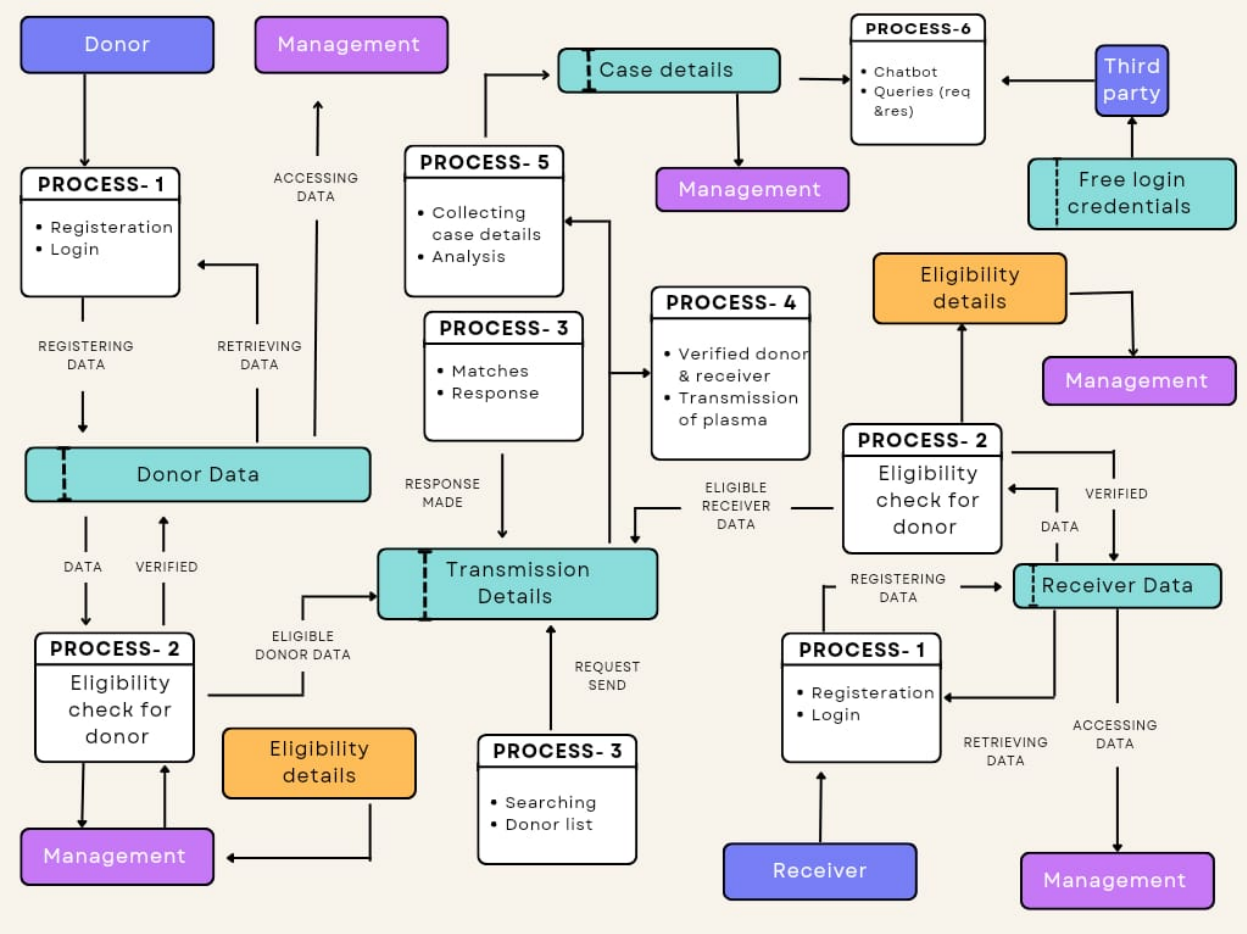
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	One brain dead donor can save up to eight lives of people suffering from end-stage organ failures It save patient from emergency situation
NFR-2	**Security**	It provide security to user personal data It avoid leakage of data
NFR-3	**Reliability**	When the user requests for plasma transmission, if there is lack of plasma at the time of request, automatically user will be marked in hold back list Later when there is availability of plasma ,the receiver waiting in hold back list will be alerted via calling system
NFR-4	**Performance**	It provides quick and accurate information. This can make a query from people into action, thus saving multiple lives

4.1 Non-Functional requirement

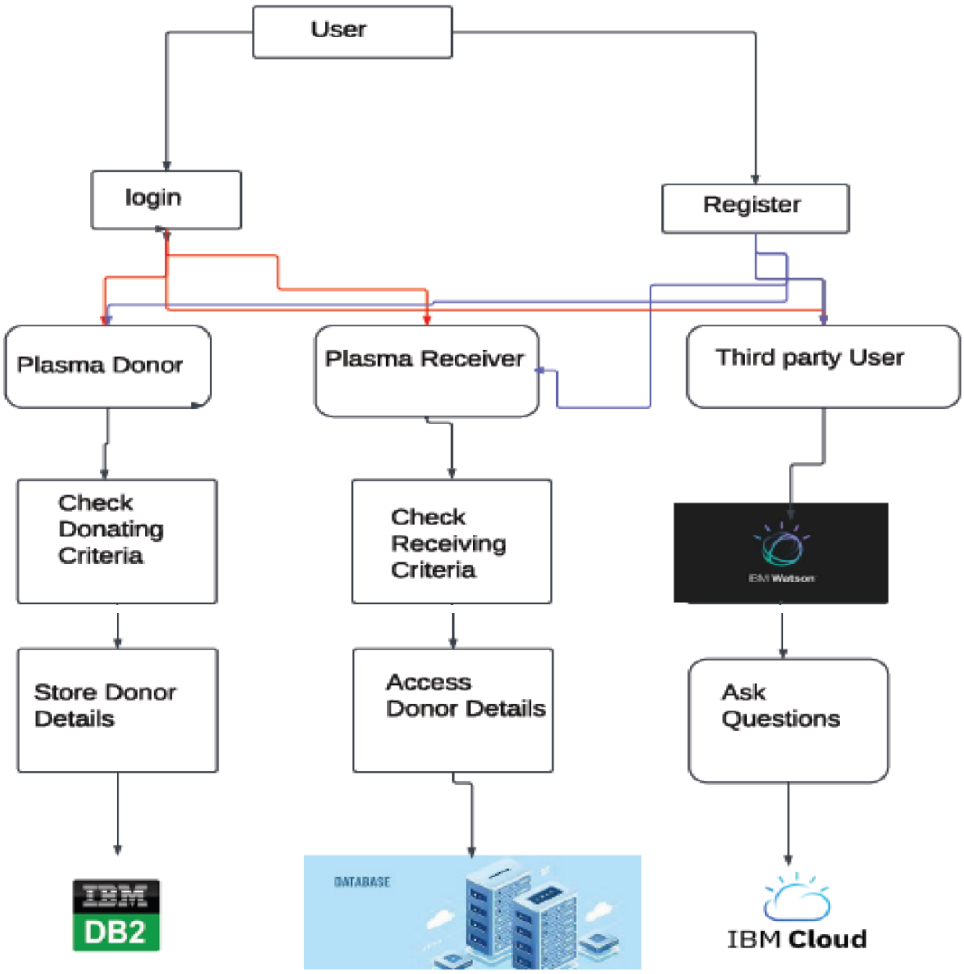
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	One brain dead donor can save up to eight lives of people suffering from end-stage organ failures It save patient from emergency situation
NFR-2	**Security**	It provide security to user personal data It avoid leakage of data
NFR-3	**Reliability**	When the user requests for plasma transmission, if there is lack of plasma at the time of request, automatically user will be marked in hold back list Later when there is availability of plasma ,the receiver waiting in hold back list will be alerted via calling system
NFR-4	**Performance**	It provides quick and accurate information. This can make a query from people into action, thus saving multiple lives
NFR-5	**Availability**	Responsiveness -App should be supported by all sized devices

5 . PROJECT DESIGN

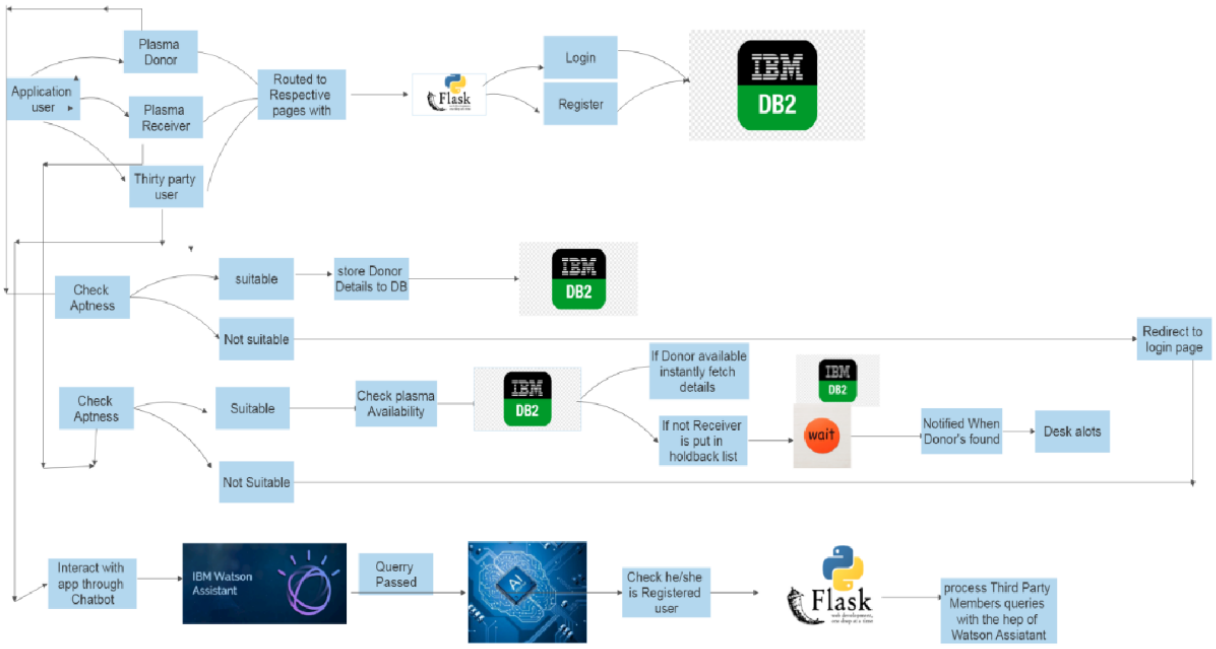
5.1 DATA FLOW DIAGRAM



5.2 SOLUTION ARCHITECTURE



Technical Architecture



5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Plasma Donor)	Registration	USN-1	As a donor, I can register for the application by entering my email /Phone number, password, and confirming my password.	I can create donor account	High	Sprint-1
	Login	USN-2	Registered donor can log into the application by entering donor email & password	I can access my account / dashboard	High	Sprint-1
	Verification	USN-3	As a donor, I can enter my details to check the donor eligibility criteria,	I can check my eligibility to donate plasma	Medium	Sprint-2
	Dashboard	USN-4	User can provide their personal details and location	I can complete my donor profile	Low	Sprint-3
	Acceptance	USN-5	User can accept their willingness to donate plasma	I can receive confirmation email & click confirm	Medium	Sprint-4

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Plasma Receiver)	Registration	USN-1	As a receiver, I can register for the application by entering my email /Phone number, password, and confirming my password	I can create receiver account	High	Sprint-1
	Login	USN-2	Registered receiver can log into the application by entering receiver email & password	I can access my account / dashboard	High	Sprint-1
	Verification	USN-3	As a receiver, I can enter my details to check the receiver eligibility criteria	I can check my eligibility to receive plasma	Medium	Sprint-2
	Dashboard	USN-4	User can search the list of available donor	I can choose donor according to my convenience	Low	Sprint-3
	Access	USN-5	User can access the available donors list ,then they can choose the donor who is nearby to receiver	I can receive confirmation email & click confirm	Medium	Sprint-4
Customer(Third Party)	Registration	USN-1	Third Party user can register for the application by entering my email /Phone number, password, and confirming my password.	I can create Third Party account	High	Sprint-1
	Login	USN-2	Registered user can log into the application by entering user email & password	I can access my account	High	Sprint-1
	Query System	USN-3	User can ask their queries via Chabot which is available 24/7 to sort user issues	I can do interrogation	Medium	Sprint-3

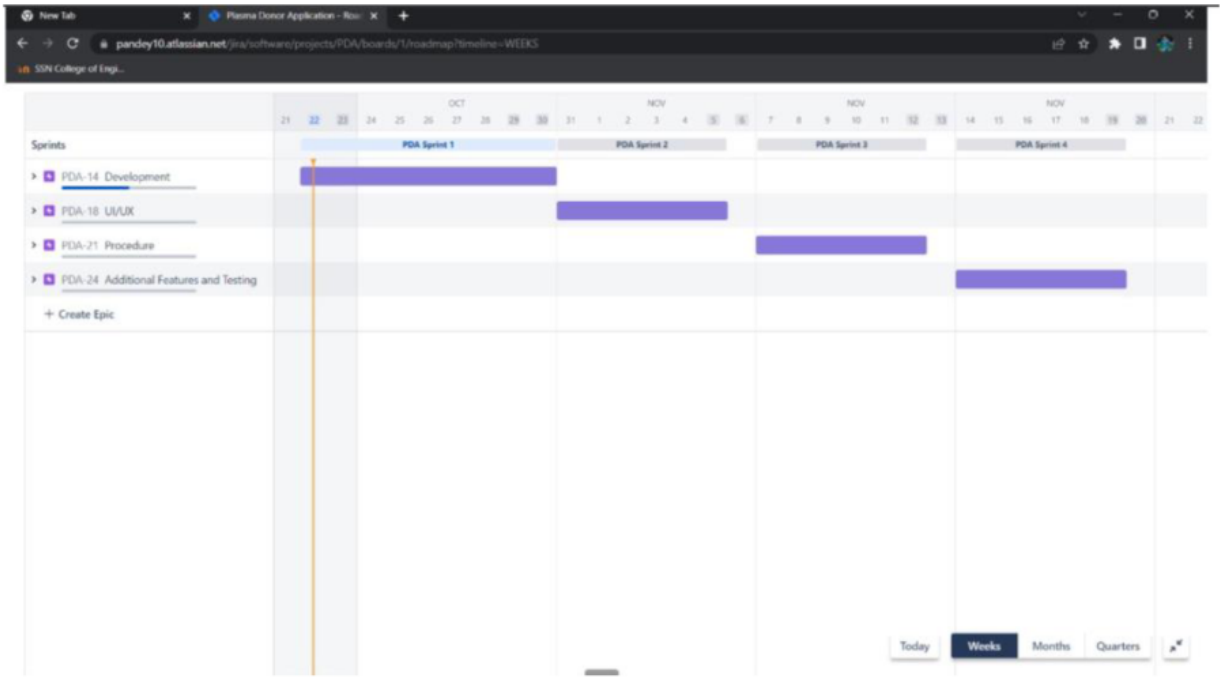
6.PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a donor, I can register for the application by entering my email /Phone number, password, and confirming my password.	4	High	S. Sandhiya, N. Nivetha,
Sprint-1	Login	USN-2	Registered donor can log into the application by entering donor email & password	3	High	T. Thirunilainayagi, S. Sandhiya
Sprint-2	Verification	USN-3	As a donor, I can enter my details to check the donor eligibility criteria,	10	Medium	T. Thirunilainayagi, K. Rubetha
Sprint-3	Dashboard	USN-4	User can provide their personal details and location	7	Low	K. Rubetha, N. Nivetha
Sprint-4	Acceptance	USN-5	User can accept their willingness to donate plasma	10	Medium	T. Thirunilainayagi, K. Rubetha, N. Nivetha, S. Sandhiya

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a receiver, I can register for the application by entering my email /Phone number, password, and confirming my password	4	High	S. Sandhiya, N. Nivetha
Sprint-1	Login	USN-2	Registered receiver can log into the application by entering receiver email & password	3	High	T. Thirunilainayagi, S. Sandhiya
Sprint-2	Verification	USN-3	As a receiver, I can enter my details to check the receiver eligibility criteria	10	Medium	T. Thirunilainayagi, K. Rubetha
Sprint-3	Dashboard	USN-4	User can search the list of available donor	7	Low	K. Rubetha, N. Nivetha
Sprint-4	Access	USN-5	User can access the available donors list ,then they can choose the donor who is nearby to receiver	10	Medium	T. Thirunilainayagi, K. Rubetha, N. Nivetha, S. Sandhiya
Sprint-1	Registration	USN-1	Third Party user can register for the application by entering my email /Phone number, password, and confirming my password.	3	High	S. Sandhiya, N. Nivetha
Sprint-1	Login	USN-2	Registered user can log into the application by entering user email & password	3	High	T. Thirunilainayagi, S. Sandhiya
Sprint-3	Query System	USN-3	User can ask their queries via Chabot which is available 24/7 to sort user issues	6	Medium	K. Rubetha, N. Nivetha

6.2 Sprint Delivery Schedule



7.CODING & SOLUTIONING

7.1 Different perspective of using

As the application user, a user takes three perspective. The user can either be a plasma donor, plasma receiver or simply a third party user. In all three perpectives the user gets benifitted from the application

code:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
<style>
body
{
background-color:cyan;
}
h1
{
color:brown;
}
h2
{
color:coral;
}
p
{
color:blue;
}
.bottom
{
color:tomato;
}
button
{
background-color:pink;
foreground-color:red;
border: 7px solid green;
}
```

```
</style>
</head>
<body>
<header>
<h1>WELCOME PLASMA DONOR. KINDLY FILL OUT THE FOLLOWING DETAILS</h1><br>
</header>

<form action="/success" method="post" ><center>
Name:<input type="text" id="name"/><br><br>
Phone number: <input type="number" id="phno"/><br><br>
Blood Group :<input type="text" id="bgrp"/><br><br>
Address: <input type="text" id="address"/><br><br>
<button>
<input type="submit" id="submit" value="SUBMIT"/>
</button>

</center>
</FORM>
<br>
<br>
<br>
<p>
Patients all over the world rely on plasma protein therapies to treat rare, chronic diseases. These individuals rely on the generosity and commitment of plasma donors (and oftentimes the blood donor recruitment specialist). You may donate plasma in one of the many licensed and certified donor centers located in the U.S. Plasma often is referred to as the “gift of life”, because it is essential to helping thousands of people worldwide with rare, chronic diseases to live healthier, more productive, and more fulfilling lives
</p>
<footer>
<br><br><br>
<p class="bottom"><strong>We thank you donor for your willingness to donate blood(plasma) and your concern to save human lives who needs plasma for survival</strong></p>
</footer>
</body>
</html>
```

7.2 Dynamic Nature

As the plasma donor application lively interacts with the users and stores and retreive their data, the dynamic nature of application makes it versatile

code:

```
@app.route('/')

def logreg():
    global userid
    msg = ''


    if request.method == 'POST' :
        username = request.form['username']
        password = request.form['password']
        sql = "SELECT * FROM users WHERE username =? AND password=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.bind_param(stmt,2,password)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print (account)
        if account:
            session['loggedin'] = True
            session['id'] = account['USERNAME']
            userid=  account['USERNAME']
            session['username'] = account['USERNAME']
            msg = 'Logged in successfully !'

            msg = 'Logged in successfully !'
            return render_template('home.html', msg = msg)
        else:
            msg = 'Incorrect username / password !'
    return render_template('Login&Register', msg = msg)




@app.route('/register', methods =['GET', 'POST'])
def registet():
    msg = ''
    if request.method == 'POST' :
        username = request.form['username']
        email = request.form['email']
```

```
        password = request.form['password']
        sql = "SELECT * FROM users WHERE username =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            msg = 'Account already exists !'
        elif not re.match(r'[^@]+@[^@]+\.[^@]+', email):
            msg = 'Invalid email address !'
        elif not re.match(r'[A-Za-z0-9]+', username):
            msg = 'name must contain only characters and numbers !'
        else:
            insert_sql = "INSERT INTO  users VALUES (?, ?, ?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt, 1, username)
            ibm_db.bind_param(prep_stmt, 2, email)
            ibm_db.bind_param(prep_stmt, 3, password)
            ibm_db.execute(prep_stmt)
            msg = 'You have successfully registered !'
    elif request.method == 'POST':
        msg = 'Please fill out the form !'

    return render_template('Login&Register.html')
```

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

8.2USER 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.

2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	4	2	3	20
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	11	2	4	20	37
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	14	13	26	77

3. Test Case Analysis

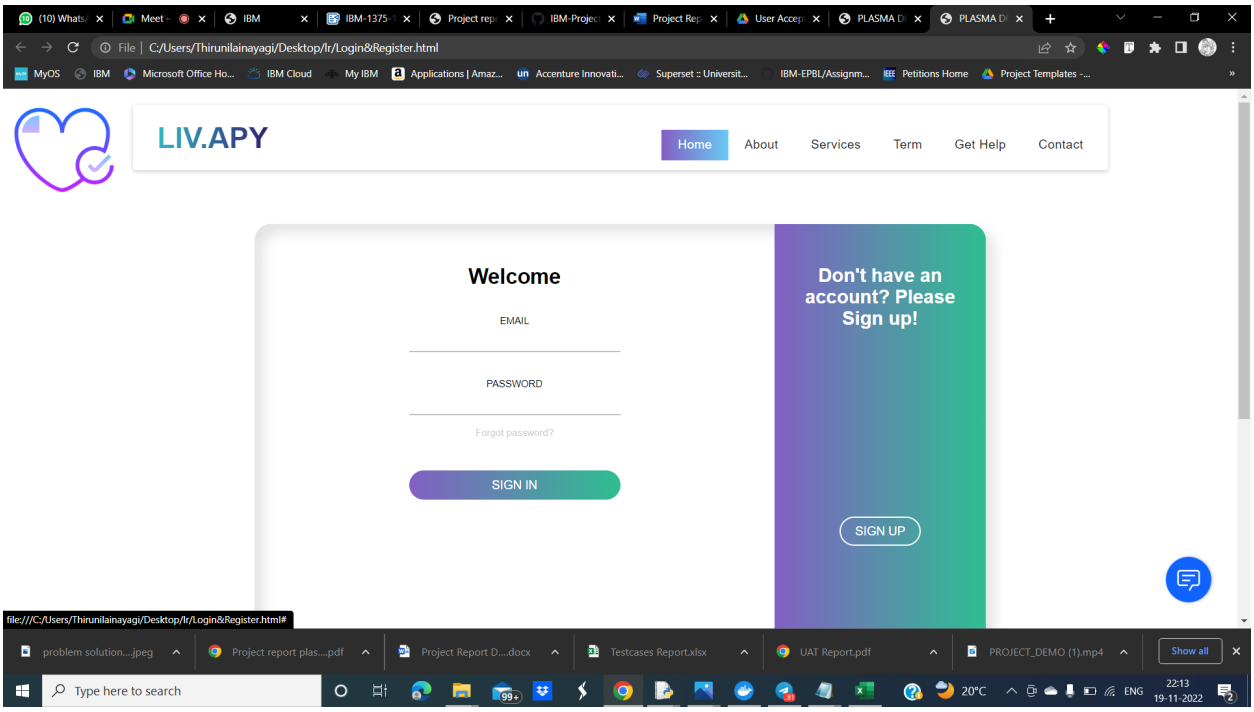
This report shows the number of test cases that have passed, failed, and untested

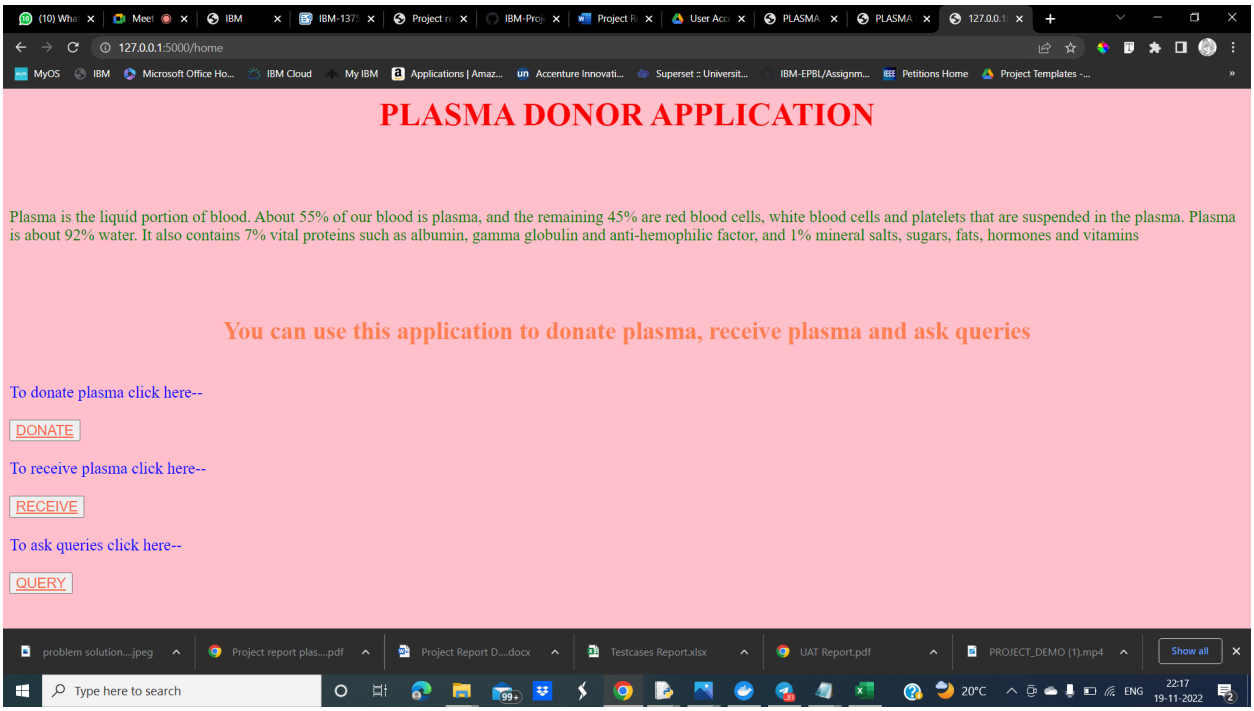
Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	51	0	0	51
Security	2	0	0	2
Outsource Shipping	3	0	0	3

9.RESULTS

9.1 PERFORMANCE METRICS:

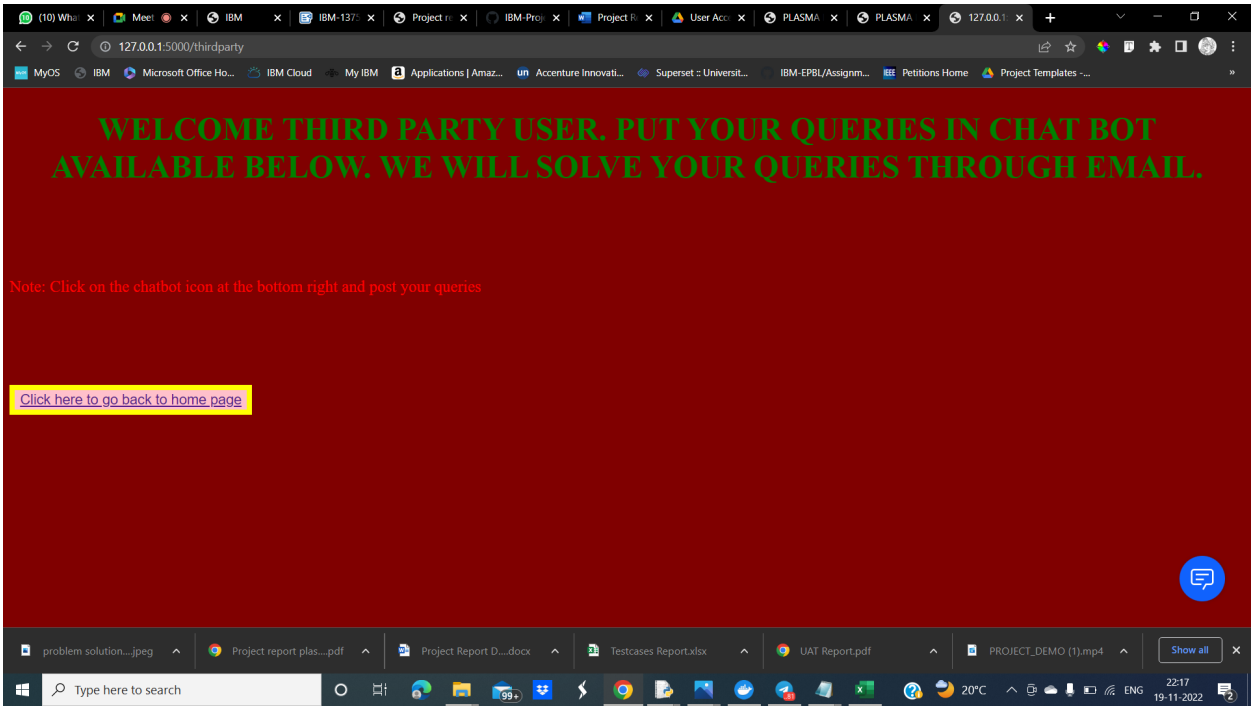
HOME PAGE





RECEIVER PAGE

THIRDPARTY PAGE



10.ADVANTAGES & DISADVANTAGES

ADVANTAGES

- User Friendly
- Instant fetch/upload of data
- Easy track of total blood donations
- Good look and feel
- Easy donor/receiver search

DISADVANTAGES

- Response time may be high when server is busy
- Donor details may be misused when app is hacked
- Receiver may not find apt donor at instant

11.CONCLUSION

Thus the web application - Plasma Donor Application is successfully built with all the relevent and necessary frontend , backend and connecting tools with proper configuration. Also this project is deployed in IBM cloud and in Docker container which gives public access to this project. The user gets his need (either donating or receiving plasma) satisfied when he/she uses our application.

12.FUTURE SCOPE

Plasma donation process involves an all time demand in today's modern society. This is especially increased in the past two years because of covid 19 disese. So our project will be a major help to all those who suffer to find donor for plasma donation. So this project has an all time scope among its users and it is very scalable also since it uses IBM DB2 and IBM Cloud

13.APPENDIX

SOURCE CODE

app.py

```
from flask import Flask, render_template, request, redirect, url_for, session
import ibm_db
import re

app = Flask(__name__)

app.secret_key = 'a'

conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=54a2f15b-5c0f-46df-8954-
7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32733;SECURITY=SS
L;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=bsb19147;PWD=M8Q6aCGQuLHwiHkU",",'
')

@app.route('/')

def logreg():
    global userid
    msg = ''


    if request.method == 'POST' :
```

```
        username = request.form['username']
        password = request.form['password']
        sql = "SELECT * FROM users WHERE username =? AND password=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.bind_param(stmt,2,password)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print (account)
        if account:
            session['loggedin'] = True
            session['id'] = account['USERNAME']
            userid=  account['USERNAME']
            session['username'] = account['USERNAME']
            msg = 'Logged in successfully !'

            msg = 'Logged in successfully !'
            return render_template('home.html', msg = msg)
        else:
            msg = 'Incorrect username / password !'
    return render_template('Login&Register', msg = msg)




@app.route('/register', methods =['GET', 'POST'])
def registet():
    msg = ''
    if request.method == 'POST' :
        username = request.form['username']
        email = request.form['email']
        password = request.form['password']
        sql = "SELECT * FROM users WHERE username =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            msg = 'Account already exists !'
        elif not re.match(r'[^@]+@[^@]+\.[^@]+', email):
```

```
            msg = 'Invalid email address !'
        elif not re.match(r'[A-Za-z0-9]+', username):
            msg = 'name must contain only characters and numbers !'
        else:
            insert_sql = "INSERT INTO  users VALUES (?, ?, ?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt, 1, username)
            ibm_db.bind_param(prep_stmt, 2, email)
            ibm_db.bind_param(prep_stmt, 3, password)
            ibm_db.execute(prep_stmt)
            msg = 'You have successfully registered !'
    elif request.method == 'POST':
        msg = 'Please fill out the form !'

    return render_template('Login&Register.html')

@app.route('/home',methods =['GET', 'POST'])

def home():
    return render_template('home.html')


@app.route('/donor',methods =['GET', 'POST'])

def donor():
    msg = ''
     if request.method == 'POST' :
        username = request.form['username']
        email = request.form['email']

        phno= request.form['phno']
        addr = request.form['addr']
        bg = request.form['bg']
        sql = "SELECT * FROM users WHERE username =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            msg = ''
```

```
            return render_template('home.html', msg = msg)



        insert_sql = "INSERT INTO  PDA_DONOR VALUES (?, ?, ?, ?, ?)"
        prep_stmt = ibm_db.prepare(conn, insert_sql)
        ibm_db.bind_param(prep_stmt, 1, username)
        ibm_db.bind_param(prep_stmt, 2, email)
        ibm_db.bind_param(prep_stmt, 3, phno)
        ibm_db.bind_param(prep_stmt, 4, addr)
        ibm_db.bind_param(prep_stmt, 5, bg)
        ibm_db.execute(prep_stmt)
        msg = 'You have successfully filled your details !'
        session['loggedin'] = True
        TEXT = "Hello sandeep,a new appliaction for job position" +jobs+"is requested"


    return render_template('donor.html')

@app.route('/success',methods =['GET', 'POST'])

def success():
    return render_template('success.html')

@app.route('/receiver',methods =['GET', 'POST'])

def receiver():
    if request.method == 'POST' :
        username = request.form['username']
        email = request.form['email']

        phno= request.form['phno']
        addr = request.form['addr']
        bg = request.form['bg']
        sql = "SELECT * FROM users WHERE username =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
```

```
            msg = ''
            return render_template('home.html', msg = msg)
    return render_template('receiver.html')
#sendmail(TEXT,"alferedd34@gmail.com")
        sendgridmail("alferedd34@gmail.com",TEXT)

@app.route('/thirdparty',methods =['GET', 'POST'])

def thirdparty():
    return render_template('thirdparty.html')


if __name__ == '__main__':
   app.run(debug='True')
```

donor.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
<style>
body
{
background-color:cyan;
}
h1
{
color:brown;
}
h2
{
color:coral;
}
p
{
color:blue;
}
.bottom
```

```
{
color:tomato;
}
button
{
background-color:pink;
foreground-color:red;
border: 7px solid green;
}
</style>
</head>
<body>
<header>
<h1>WELCOME PLASMA DONOR. KINDLY FILL OUT THE FOLLOWING DETAILS</h1><br>
</header>

<form action="/success" method="post" ><center>
Name:<input type="text" id="name"/><br><br>
Phone number: <input type="number" id="phno"/><br><br>
Blood Group :<input type="text" id="bgrp"/><br><br>
Address: <input type="text" id="address"/><br><br>
<button>
<input type="submit" id="submit" value="SUBMIT"/>
</button>

</center>
</FORM>
<br>
<br>
<br>
<p>
Patients all over the world rely on plasma protein therapies to treat rare, chronic diseases. These individuals rely on the generosity and commitment of plasma donors (and oftentimes the blood donor recruitment specialist). You may donate plasma in one of the many licensed and certified donor centers located in the U.S. Plasma often is referred to as the “gift of life”, because it is essential to helping thousands of people worldwide with rare, chronic diseases to live healthier, more productive, and more fulfilling lives
</p>
<footer>
<br><br><br>
<p class="bottom"><strong>We thank you donor for your willingness to donate blood(plasma)
```

and your concern to save human lives who needs plasma for survival</p>
</footer>
</body>
</html>

receiver.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
<style>
body
{
background-color:coral;
}
h1
{
color:brown;
}
h3
{
color:green;
}
p
{
color:blue;
}
.bottom
{
color:tomato;
}
button
{
background-color:pink;
foreground-color:red;
border: 7px solid red;
}
</style>
```

```
</head>
<body><center>
<h1>
WELCOME PLASMA RECEIVER!!!!
</h1>
<p>

<h3>HELLO RECEIVER. WE KNOW THAT YOU ARE IN NEED OF PLASMA. PLEASE DO NOT WORRY. WE ARE HERE TO HELP YOU. KINDLY SEE THE BELOW DATABASE WHICH HAS THE LIST OF DONORS. SELECT ANYONE MEMBER FROM THE TABLE BELOW AND GET YOUR NEED SATISFIED.<h3>
</center>
</p>
<br />
<br />
<br />

<table border="2">

 <thead>
  {%- for column in columns %}
    <th>{{ column }}</th>
  {%- endfor %}
 </thead>
 <tbody>
 {%- for row in items %}
   <tr>
   {%- for column in columns %}
     <td>{{ row|attr(column) }}</td>
   {%- endfor %}
   </tr>
 {%- endfor %}
 </tbody>

</table>
<br />
<br />
<br />
<br />
<br />
<br />
```

```
<br />
<br />
<br />
<br />
<br />
<center>
<button>
<a href="home">
Click here to go back to home page
</a>

</button></center>
</body>
</html>
```

thirdparty.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
<style>
body
{
background-color:maroon;
}
h1
{
color:green;
}
button
{
background-color:pink;
foreground-color:red;
border: 7px solid yellow;
}
p
{
color:red;
```

```
}
</style>
<script>
  window.watsonAssistantChatOptions = {
    integrationID: "63bd42fa-1d3b-41c6-8ddc-a5d05bb49893", // The ID of this integration.
    region: "au-syd", // The region your integration is hosted in.
    serviceInstanceID: "aa08694c-20a5-43af-8818-0cc67e75b527", // The ID of your service
instance.
    onLoad: function(instance) { instance.render(); }
  };
  setTimeout(function(){
    const t=document.createElement('script');
    t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +
(window.watsonAssistantChatOptions.clientVersion || 'latest') + "/WatsonAssistantChatEntry.js";
    document.head.appendChild(t);
  });
</script>
</head>
<body>
<header><center>
<h1>
WELCOME THIRD PARTY USER. PUT YOUR QUERIES IN CHAT BOT AVAILABLE BELOW. WE WILL
SOLVE YOUR QUERIES THROUGH EMAIL.</h1></center>
</header>
<br>
<br>
<br>
<p>Note: Click on the chatbot icon at the bottom right and post your queries</p>

<br>
<br>
<br>
<br>
<button>
<a href="home">
Click here to go back to home page
</a>
</button>
</body>
</html>
```

Login&Register.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
<style>
body
{
background-color:maroon;
}
h1
{
color:green;
}
button
{
background-color:pink;
foreground-color:red;
border: 7px solid yellow;
}
p
{
color:red;
}
</style>
<script>
  window.watsonAssistantChatOptions = {
   integrationID: "63bd42fa-1d3b-41c6-8ddc-a5d05bb49893", // The ID of this integration.
   region: "au-syd", // The region your integration is hosted in.
   serviceInstanceID: "aa08694c-20a5-43af-8818-0cc67e75b527", // The ID of your service
instance.
   onLoad: function(instance) { instance.render(); }
  };
  setTimeout(function(){
   const t=document.createElement('script');
   t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +
(window.watsonAssistantChatOptions.clientVersion || 'latest') + "/WatsonAssistantChatEntry.js";
   document.head.appendChild(t);
```

```
  });
</script>
</head>
<body>
<header><center>
<h1>
WELCOME THIRD PARTY USER. PUT YOUR QUERIES IN CHAT BOT AVAILABLE BELOW. WE WILL SOLVE YOUR QUERIES THROUGH EMAIL.</h1></center>
</header>
<br>
<br>
<br>
<p>Note: Click on the chatbot icon at the bottom right and post your queries</p>

<br>
<br>
<br>
<br>
<button>
<a href="home">
Click here to go back to home page
</a>
</button>
</body>
</html>
```

home.html

```
<html>
<head>
<style>
body
{
background-color:pink;
}
h1
{
color:red;
}
h2
{
```

```
color:coral;
}
p
{
color:green;
}
div
{
color:blue;
}
a
{
color:tomato;
}
</style>
</head>
<body>

<h1><center>PLASMA DONOR APPLICATION</center></h1>
<br><br>
<p>Plasma is the liquid portion of blood. About 55% of our blood is plasma, and the remaining 45% are red blood cells, white blood cells and platelets that are suspended in the plasma.

Plasma is about 92% water. It also contains 7% vital proteins such as albumin, gamma globulin and anti-hemophilic factor, and 1% mineral salts, sugars, fats, hormones and vitamins</p>
<br><br>
<h2><center>You can use this application to donate plasma, receive plasma and ask queries
</center></h2>
<div ><br>
To donate plasma click here--<br><br>
<BUTTON>
<a type="button" id="donor" href="donor">DONATE</a>
</BUTTON>
</div>

<div ><br>
To receive plasma click here--<br><br>
<BUTTON>
<a type="button" id="receiver" href="receiver">RECEIVE</a><br>
</button>
</div>
```

```
<div ><br>
To ask queries click here--<br><br>
<BUTTON>
<a type="button" id="thirdparty" href="thirdparty">QUERY</a>
</button>
</div>


</body>
</html>
```

success.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
<style>
body
{
background-color:coral;
}
p
{
color:green;
}

a
{
color:tomato;
}
button
{
background-color:cyan;
foreground-color:red;
border: 7px solid red;
}
</style>
```

```
</head>
<body>
<form action="/home" method="post">
<br><bR>
<p>
<strong>
Thank you for submitting your details donor. You will be contacted by any registered receiver whenever they need yourhelp. Kindly co operate with them and save their life
</strong>
</p>
<center><button>
<input type="submit" value="Click here to go back to home page"/>
</button>
</center>
</form>
</body>
</html>
```

sendemail.py

```
import smtplib
import sendgrid
import os
from sendgrid.helpers.mail import Mail, Email, To, Content
SUBJECT = "Interview Call"
s = smtplib.SMTP('smtp.gmail.com', 587)

def sendmail(TEXT,email):
    print("sorry we cant process your candidature")
    s = smtplib.SMTP('smtp.gmail.com', 587)
    s.starttls()
    s.login("il.pradeepthi@gmail.com", "oms@1Ram")
    message  = 'Subject: {}\n\n{}'.format(SUBJECT, TEXT)
    s.sendmail("il.pradeepthi@gmail.com", email, message)
    s.quit()
def sendgridmail(user,TEXT):
    sg =
sendgrid.SendGridAPIClient('SG.nouVVZMwQTSYtih73r1TxQ.3H0kajWkEYpo0RV1iarxSVKbqvtjy
Z_nhPbKi3zeZnc')
    from_email = Email("sandeep@thesmartbridge.com")  # Change to your verified sender
    to_email = To(user)  # Change to your recipient
```

```
    subject = "Sending with SendGrid is Fun"
    content = Content("text/plain",TEXT)
    mail = Mail(from_email, to_email, subject, content)

    # Get a JSON-ready representation of the Mail object
    mail_json = mail.get()
    # Send an HTTP POST request to /mail/send
    response = sg.client.mail.send.post(request_body=mail_json)
    print(response.status_code)
    print(response.headers)
```

requirments.txt

```
Flask
ibm_db
sendgrid
```

Github Repository linK

https://github.com/IBM-EPBL/IBM-Project-1375-1658386309

Project Demo link

https://drive.google.com/file/d/16QPUxPee7fGX3Tp7IpX81obdUhpjKHyd/view