CLOUD APPLICATION– PLASMA DONOR

PLASMA DONOR WEB APPLICATION USING CLOUD

PROJECT REPORT

Submitted by:

Anu K (950019104003)

Evanjalin R (950019104011)

Janani S (950019104015)

Karpagam B (950019104020)

TEAM ID:PNT2022TMID49558

ANNA UNIVERSITY REGIONAL CAMPUS – TIRUNELVELI

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AUGUST 2022 – NOVEMBER 2022

TABLE OF CONTENTS

1.INTRODUCTION

1.1 Project Overview:

The main goal of our project is to design a user-friendly web application that is 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 antiretroviral therapy planning for a deadly COVID19 infection, plasma therapy is an experimental approach to treat those COVID-positive patients and help them recover fasterTherapy, which is considered reliable and safe.

1.2 Purpose:

During COVID 19 crisis there requirement for plasma increased drastically as there were no vaccinations found in order to treat the infected patients. In such situation it was very difficult to find the plasma donor, check whether the donor was infected previously and was recovered, and which donor is eligible to donate plasma was a challenging task.

2.LITERATURE SURVEY

2.1Existing Problem:

1.PLASMA DONATION WEBSITE

The main goal of this application is to make it easier for the COVID-19 patients to get a plasma donor easily as well as donate plasma if they have recovered. The system target two types of users: the people who want to donate plasma and the people who need plasma. The user can also view the total active cases, nearby vaccine centers, hospitals address. The main objective of

developing the website is to make it easier for the COVID-19 patients to get a plasma donor easily and as soon as possible.

ADVANTAGES

- This website is fast and offers great accuracy as compared to manual registered keeping.
- It is very easy to use and understand. It is easily workable and accessible for everyone.

DISADVANTAGES

- It cannot automatically verify the genuine users.
- It requires active internet connection.

2. DELHI FIGHTS CORONA

The Delhi government has launched the country's first plasma bank. If you are a recovered corona patient, come forward and donate your plasma. In this time of emergency it becomes difficult to approach the right donor. To overcome this problem this application is proposed.

ADVANTAGES

- Recipients can get blood easily
- Donors can also donate plasma easily by checking the list of blood donation camps.

DISADVANTAGES

- It requires active internet connection.
- In some situation it was very difficult to find plasma donor.

2.2 References:

1. https://dev.to/nehasoni_/plasma-donation-website-using-mern-stack-26f5

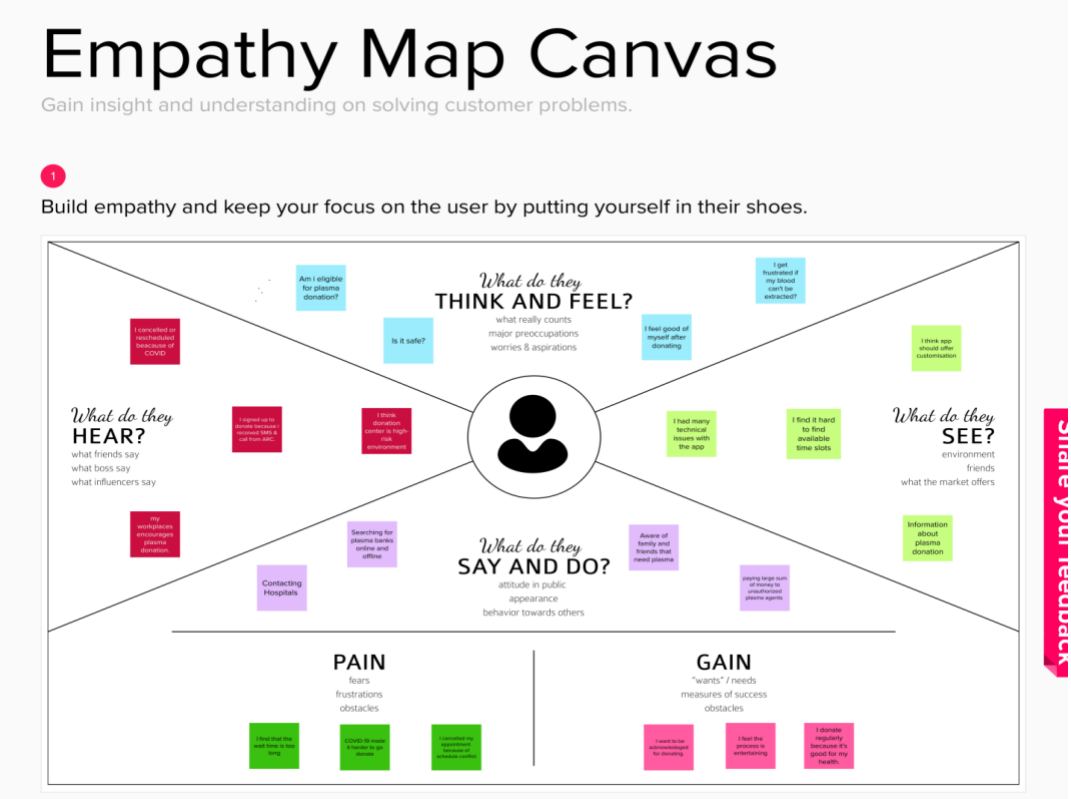
2. https://delhifightscorona.in/donateplasma/

2.3 Problem Statement Definition:

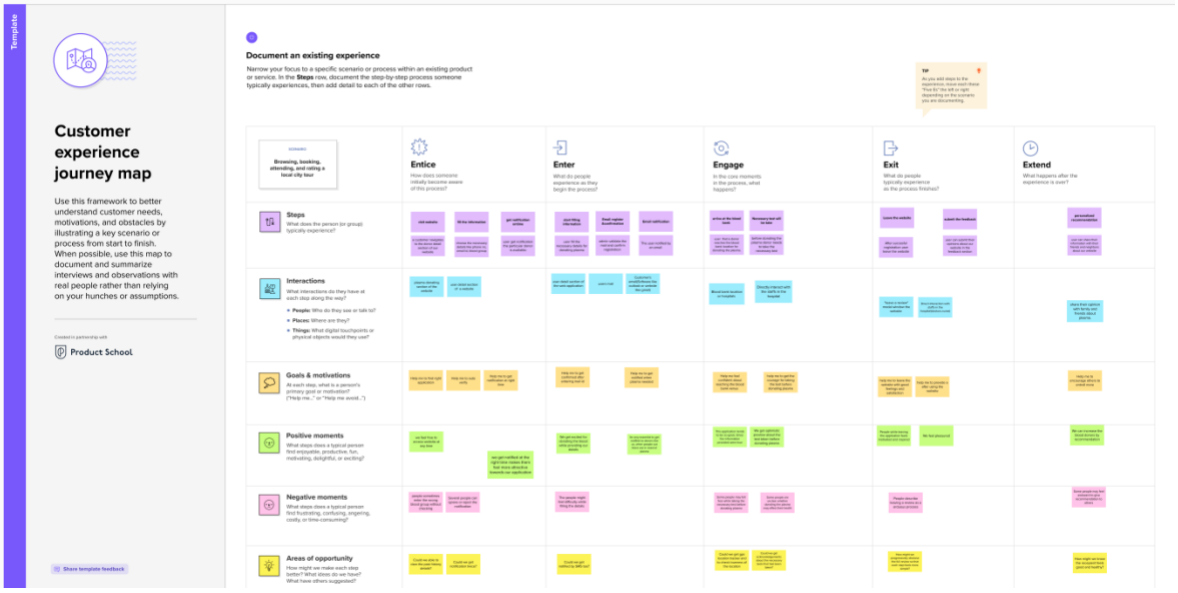
During COVID 19 crisis the requirement for plasma increased drastically as there were no vaccinations found in order to treat the infected patients. In such situation it was very difficult to find the plasma donor, check whether the donor was infected previously and was recovered, and which donor is eligible to donate plasma was a challenging task.

3.IDEATION&PROPOSEDSOLUTION

3.1EmpathyMap



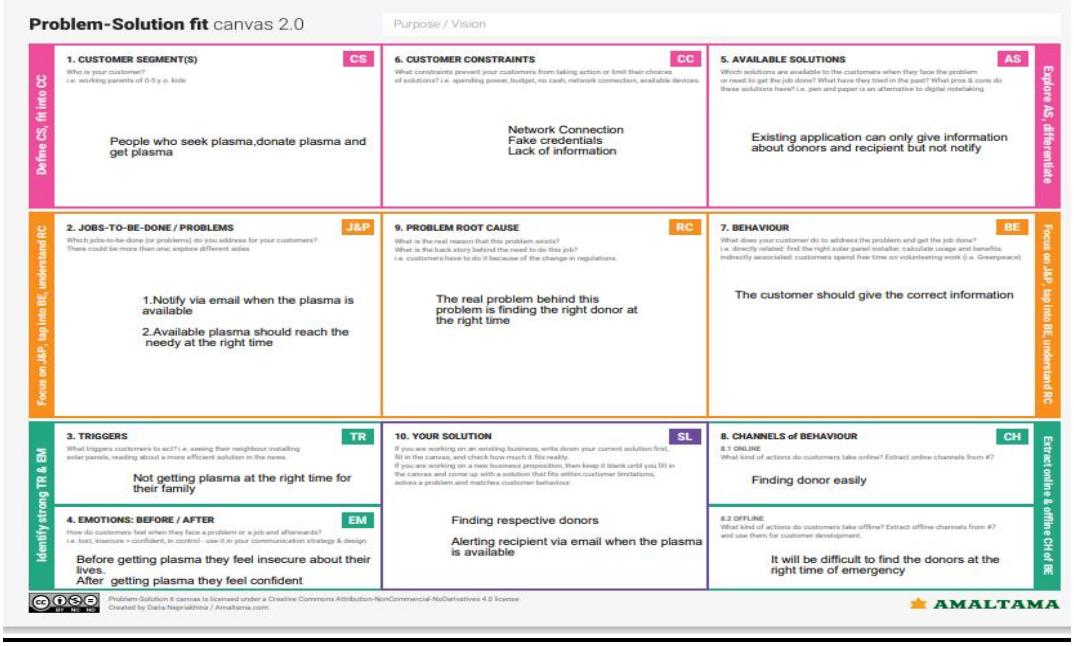
3.2 Ideation & Brainstorming:



3.3 Proposed Solution:

S.NO	PARAMETER	DESCRIPTION
1.	Problem Statement (Problem to be solved)	During COVID 19 crisis there requirement for plasma increased drastically as there were no vaccinations found in order to treat the infected patients. In such situation it was very difficult to find the plasma donor, check whether the donor was infected previously and was recovered, and which donor is eligible to donate plasma was a challenging task.
2.	Idea / Solution description	To maintain and update the plasma details in the form of web application. Users can get an Alert from email if the specific plasma is available.
3	Novelty / Uniqueness	Send notification to recipient if plasma is available
4	Social Impact / Customer Satisfaction	We can save life who are in need of plasma
5	Business Model (Revenue Model)	We can collaborate with government and it can utilize the app to help the people who are in need of plasma
6	Scalability of the Solution	Performance and speed do not slow down even if large number of users access the application at the same time.

3.4 Problem Solution fit



4. REQUIREMENT ANALYSIS

4.1 Functional & Non-Functional requirements:

Following are the functional requirements of the proposed solution

FR No	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through GmaiL
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Recipient Registration	Registration through Form Registration through Gmail
FR-4	Recipient Confirmation	Confirmation via Email

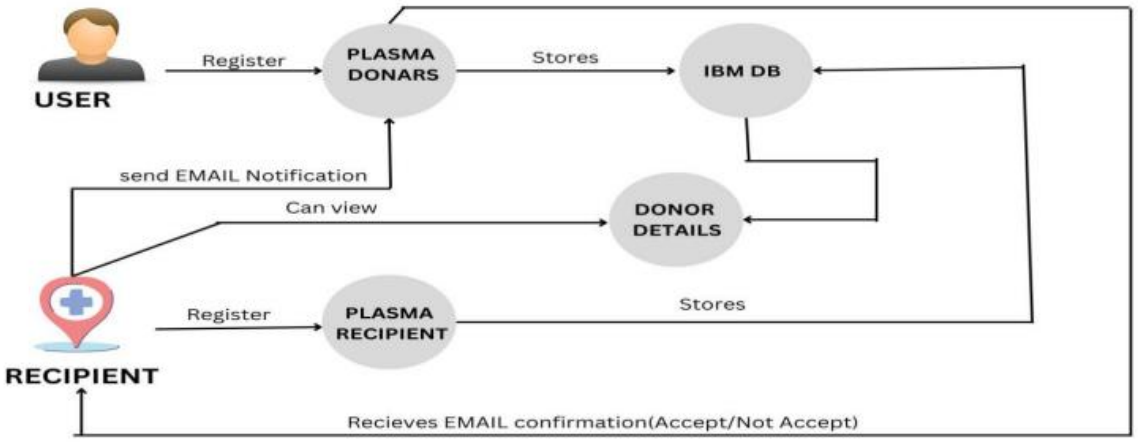
FR-5	Recipient Notify Donors	Notify Donors via Email

Following are the non-functional requirements of the proposed solution

NFR No	**Non-Functional Requirement**	**Description**
NFR-1	Usability	People can get plasma in a short period when they are in a critical situation.
NFR-2	Security	The informations provided by the users are securely stored in database.Admin can only access the database
NFR-3	Reliability	If any data provided by the user is wrong this app will intimate users to provide the correct information.
NFR-4	Performance	This application performs well ,it will help both the donors and recipient .
NFR-5	Availability	The donor availability time is much more than downtime.
NFR-6	Scalability	Many number of donors and recipient can get registered

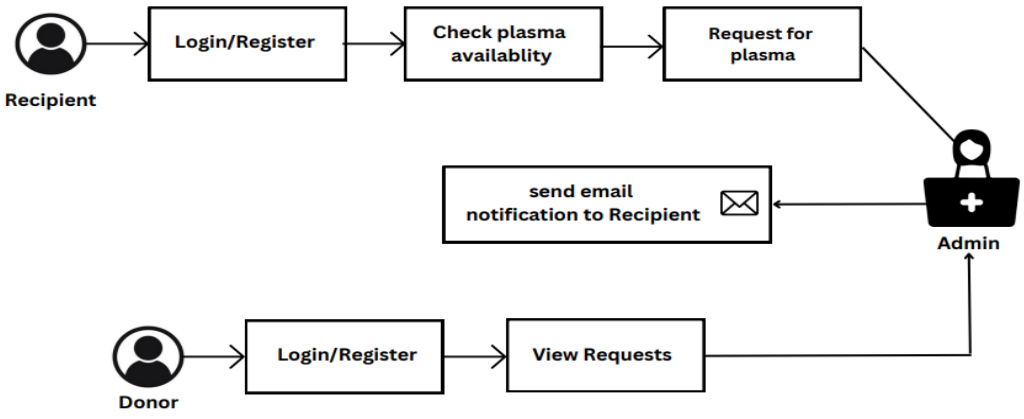
5. PROJECT DESIGN:

5.1 Data Flow Diagram:



A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

5.2 Solution & Technical Architecture:



5.3 User Stories:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Donor)	Registration	USN-1	As a DONOR, I can register for the application by entering my name, email, password, Blood group, Address, mobile number and confirming my password	I can access my account / dashboard	High	Sprint-1
		USN-2	As a DONOR, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1

		USN-3	As a DONOR, I can register in the application through Gmail	I can register & access the dashboard with Gmail Login	Low	Sprint-2
	LOGIN	USN-4	As a DONOR, I can log into the application by entering email & password	I can register & access the dashboard with Login	Medium	Sprint-1
					High	Sprint-1
	Dashboard	USN-5	As a DONOR, I can see the information about my application	I can see the dashboard with Register or login	Medium	Sprint-2
Customer (Recipient)	Registration	USN-6	As a RECIPIENT , I can register in the application by entering my name, email,	I can access my account / dashboard	High	Sprint-1

			password, Organizatio n name, Address, mobile number and confirming my password.			
	Login	USN-7	As a RECIPIENT , I can log into the application by entering email & password	I can register & access the dashboard with Login	Medium	Sprint-1
	Request Notification	USN-8	As a RECIPIENT , I can send Request to the donors by notifying them at the right time, i.e., when plasma needed	I can register & send the Request	High	Sprint-2
	Response from Donor	USN-9	As a RECIPIENT Ican Receive	I can Send Request &	High	Sprint-2

			reply from Donors	Get response from Donors		

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	Donor Registration	USN-1	As a donor, I can register for the application by entering my email, password, and confirming my password and plasma details.	1	High	Anu K
Sprint-2	Donor confirmation	USN-2	As a Donor, I will receive confirmation email once I have registered for the application	3	High	Evanjalin R
Sprint-1	Donor login	USN-3	As a Donor, I can log into the application by entering email and password	1	High	Janani S
Sprint-2	Recipient Registration	USN-4	As a Recipient, I can register for the application by entering required details	2	High	Karpagam B
Sprint-2	Recipient Login	USN-5	As a Recipient, I can log into the application by entering email & password	1	High	Evanjalin R
Sprint-3	Request Notification	USN-6	As a Recipient, I can send Request to the Donors by notifying them at the right time	3	High	Anu K
Sprint-4	Response from Donor	USN-7	As a Recipient, I can Receive response from Donors	3	High	Janani S
Sprint-4	Dashboard	USN-8	As a Donor, I can see the information about application	2	Medium	Karpagam B

6.2 Sprint Delivery Schedule:

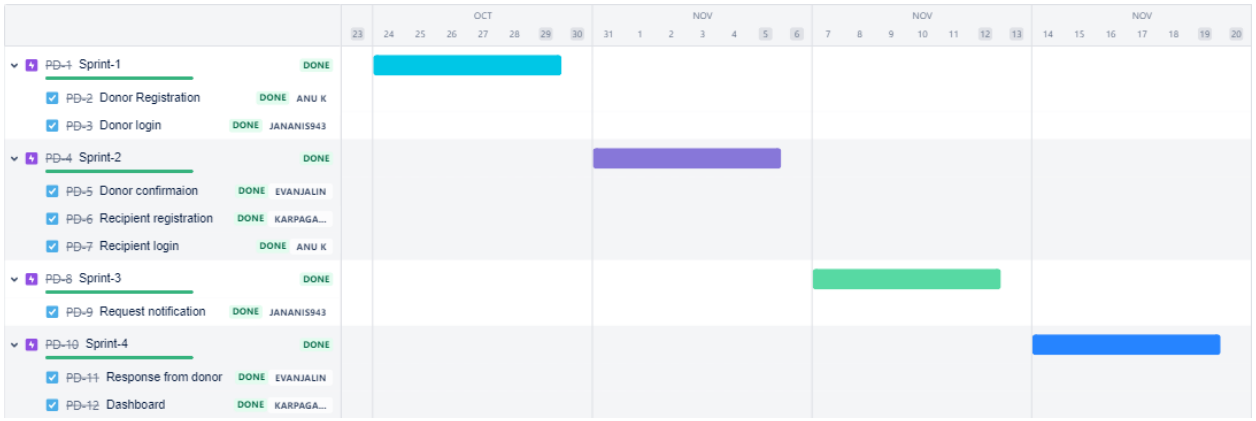
Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	12	6 Days	24 Oct 2022	29 Oct 2022	12	29 Oct 2022
Sprint-2	12	6 Days	31 Oct 2022	05 Nov 2022	12	05 Nov 2022
Sprint-3	12	6 Days	07 Nov 2022	12 Nov 2022	12	12 Nov 2022
Sprint-4	12	6 Days	14 Nov 2022	19 Nov 2022	12	19 Nov 2022

Velocity:
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

AV= sprint duration / velocity = 12/6 = 2

6.3 Reports from JIRA:



7.CODINIG & SOLUTIONING

7.1 FEATURE 1

- **Registration page**

  The volunteered donors can register in our website by filling the required information such as username, address, blood group, password etc.

  The recipient can register in our website in order to get plasma.

- **Login page**

  The login page is common for both the donor and recipient. Donors and Recipients can login in our website by giving username and password.

- **Request notification**

  The recipient posts a request then the concerned blood group donors will get notified about it.

- **Quick availability of plasma**

  The donors will provide required plasma at the correct time to save the life of the people.

7.2 Feature 2 (Coding)

1. Python flask:

We have used python flask to develop our project. Python flask is a web framework and a python module that let us to develop web application easily.

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

import re app = Flask(name)

2. IBM Cloud db2

When the donors and recipients registers in our website , the details get stored in the IBM Cloud db2. We have connected db2 with our project using python code.

```
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=9938aec0-8105-433e-8bf9-0fbb7e483086.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32459;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=hnl10309;PWD=4q2SvdOKme6RC7Zr","","")
stmt = ibm_db.prepare(conn,insert_sql)
ibm_db.bind_param(stmt, 1, username )
ibm_db.bind_param(stmt, 2, age)
ibm_db.bind_param(stmt, 3, bloodgroup)
ibm_db.bind_param(stmt, 4, gender )
ibm_db.bind_param(stmt, 5, phone )
ibm_db.bind_param(stmt, 6, address )
ibm_db.bind_param(stmt, 7, email )
ibm_db.bind_param(stmt, 8, password )
ibm_db.bind_param(stmt, 9, confirmpassword )
ibm_db.bind_param(stmt, 10, height )
ibm_db.bind_param(stmt, 11, weight ) ibm_db.execute(stmt)

msg = 'You have successfully registered !'
```

3. For receiving email:

The registered user will get the email using this python code.

```
# importing librariesfromflaskimportFlask
fromflask_mailimportMail,Messageapp=Flask(name)
mail=Mail(app)#instantiatethemailclass#configurationofmail
app.config['MAIL_SERVER']='smtp.gmail.com'app.config['MAIL_PORT']=465app.config['MAIL_USERNAME'] = 'karpagamboothanathan@gmail.com'app.config['MAIL_PASSWORD'] = 'nmydiidhksvvojid' app.config['MAIL_USE_TLS'] =
```

```
Falseapp.config['MAIL_USE_SSL']=True
mail=Mail(app)

#messageobjectmappedtoaparticularURL‘/’@app.route("/")
defindex():
msg=Message(
'youhaveregisteredsucessfully',
sender ='karpagamboothanathan@gmail.com',recipients=['siksha@gmail.com']
)
msg.body='Welcometoplasmadonorapplication!!Thanksforregistering'mail.send(msg)
return'Sent'

ifname          == 'main':app.run(debug=True)
```

4. Deploying flask app in Docker:

We have deployed our flask app in docker where all the code, libraries and dependencies together to make it possible for multiple containers to run in the same host and we can run our flask app using the dockerdesktop.

8.TESTING

8.1 Test cases

Section	Total cases	Non tested	Fail	Pass
Login	7	0	0	7
Registration	7	0	0	7
Performance	7	0	0	7
Final report output	7	0	0	7

8.2 User acceptance testing

1. Purpose of document

The purpose of document is to briefly explain the test coverage and open issues of the [Plasma Donor Application] project at the time of the release to User Acceptance Testing (UAT).

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	5	3	1	0	9
Duplicate	5	1	0	1	7
External	2	2	1	1	6
Fixed	3	1	0	0	4
Not reproduced	0	1	2	0	3
Skipped	3	2	1	1	7
Won’t fix	0	3	2	1	6
Total	18	13	7	4	42

3. Test case analysis

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

Section	Total cases	Non tested	Fail	Pass
Login	7	0	0	7
Registration	7	0	0	7
Performance	7	0	0	7
Final report output	7	0	0	7

9.RESULT

9.1 Performance metrics

S.No	Parameter	Values
1.	Model summary	During COVID -19 the requirement of plasma increased as there is no vaccination. Plasma from donors who have recovered from COVID-19 may contain antibodies that could help to recover. The website builds a platform which will help to provide quickest solution to find the plasma donors. The volunteered donor can register in this website by filling their details . The recipient can register in order to get plasma and they get verified email. The database will have all the details about the registered donor and recipient. Once they have registered, they can login to our page. If a recipient posts a request then the concerned blood group donors will get notified about it. The recipient will get the plasma at the right time.
2.	Accuracy	This is fast and offers great accuracy as compared to manual registered keeping.

10. ADVANTAGES & DISADVANTAGES

Advantages:

- It is very easy to use and understand. It is easy workable and accessible for everyone.
- It would help you to find the plasma donors easily depending on the availability.

Disadvantages:

- Forgot password option is unavailable.
- Internet speed

11. CONCLUSION

- Plasma is a liquid portion of blood; it is a mixture of water, proteins and salts. Antibodies are proteins made by the body in response to an infection. People fully rescued from COVID19 are encouraged to donate plasma, which can help to increase the lifespan of other patients because their plasma contains antigens which helps the affected person to recover faster.
- These immunoglobulin give your immune system a way to fight the virus when you are sick, so your plasma can be used to help others fight off illness. Individuals must fully resolve symptoms for at least 14 days prior are eligible to donate The efficient way of finding plasma donor for the infected people is implemented using the plasma donor application website. People fully rescued from COVID-19 are encouraged to donate plasma which can help to increase the lifespan of other patients.

12. FUTURE SCOPE

- Upgrading the UI that is more user-friendly which will help many users to access the website and also ensures that many plasma donors can be added into the community.
- One important future scope is availability of location based plasma donor details and extraction of location based donor’s details, which is very helpful to the acceptant people.

13. APPENDIX

SOURCE CODE

INDEX.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title></title>
    <link rel="stylesheet"href="{{url_for('static',filename='styles.css')}}">
<style>
    *{
        margin: 0;
        padding: 0;
    }
    .content{
        width: 1200px;
        height: auto;
```

```
    margin: auto;
    position: relative;
}
.content .par{
    padding-left: 25px;
    padding-bottom: 25px;
    font-family: Arial;
    color: beige;
    letter-spacing: 1.2px;
    line-height: 30px;
}

.content h1{
    font-family: 'Times New Roman';
    font-size: 50px;
    padding-left: 20px;
    margin-top: 9%;
    color: red;
    letter-spacing: 2px;
}.content .cn{
    width: 160px;
    height: 40px;
    background: #ff7200;
    border: none;
    margin-bottom: 10px;
```

```
        margin-left: 20px;
        font-size: 18px;
        border-radius: 10px;
        cursor: pointer;
        transition: .4s ease;
          }

    .content .cn a{
        text-decoration: none;
        color: #000;
        transition: .3s ease;
    }

    .content span{
        color: #ff7200;
        font-size: 65px
    }
</style>
</head>
<body>
    <div class="full-page">
        <div class="navbar">
            <div>
                <a href='index.html'></a>
            </div>
```

```
        <nav>
          <ul id='MenuItems'>
            <li><a href='#'>HOME</a></li>
            <li><a href="{{'login'}}">LOGIN</a></li>
            <li><a href="{{'register'}}">DONATE PLASMA</a></li>
            <li><a href="{{'registers'}}">NEED PLASMA? </a></li></ul>
<nav>
      </div>
      <div class="content">
        <h1>PLASMA  DONOR<br>WEB APPLICATION</h1>
        <br><br>
        <p class="par"> “Blood Donation Costs You Nothing,<br> But It Can Mean The World To Someone In Need.”</p>
   </div>
</body>
</html>
```

DONOR REGISTER.html:

```
<Html><head>
    <title>Registration form</title>
    <link rel="stylesheet"href="{{url_for('static',filename='profile.css')}}">
 </head><center>
  <h1 class="word"></h1>
 </center>
 <body><div class="nav">
```

```
    <div class="logo">
      <a href="/userdashboard">  <i class="fas fa-arrow-left"></i></a>
       <center> <h3>Donor Registration</h3></center>
    </div>

</div>
<div class="cont" align="center" >

  <h3>
  <div class="profile">
  <h3>
    <form action="{{ url_for('register') }}" method="post">
        <table>
            <tr>
              <th>Name:</th>
              <td><input type="text" name="username" size="15" /></td>
            </tr>
            <tr>
               <th>Age:</th>
               <td><input type="text" name="age" size="15" /></td>
              </tr>
              <tr>
                <th>Bloodgroup:</th>
                <td><input type="text" name="bloodgroup" size="15" /></td>
              </tr>
```

```
<tr>
  <th>LastDonateDate:</th>
  <td><input type="text" name="lastdonatedate" size="15" /></td>
</tr>
<tr>
  <th>Gender:</th>
  <td><input type="text" name="gender" size="15" /></td>
</tr>
<tr>
  <th>Phone:</th>
  <td><input type="text" name="phone" size="15" /></td>
</tr>
<tr>
  <th>Address:</th>
  <td><input type="text" name="address" size="15" /></td>
</tr>
<tr>
  <th>Email:</th>
  <td><input type="text" name="email" size="15" /></td>
</tr>
<tr>
  <th>Password:</th>
  <td><input type="text" name="password" size="15" /></td>
</tr>
<tr>
```

```
                <th>Confirm Password:</th>
                <td><input type="text" name="confirmpassword" size="15" /></td>
              </tr>
              <tr>
                <th>Height:</th>
                <td><input type="text" name="height" placeholder="Please write your height in centimetres" size="15" /></td>
              </tr>
              <tr>
                <th>Weight:</th>
                <td><input type="text" name="weight" placeholder="Please write your weight in centimetres" size="15" /></td>
              </tr>
              </table>
            <div class="cont" align="center" >
              <button><input type="submit" value="REGISTER" /></button>
              </div>
            </form> </div>
      </div>
    </div>
    <span class="alert{{indicator}}">{{a}}</span>
  </body>
  </html>
```

RECEIPIENT REGISTER.html:

```
<Html>

    <head>
        <title>Registration form</title>
        <link rel="stylesheet"href="{{url_for('static',filename='profile.css')}}">
    </head>

    <center>

        <h1 class="word"></h1>
    </center>
    <body>

        <div class="nav">
            <div class="logo">
              <a href="/userdashboard">  <i class="fas fa-arrow-left"></i></a>
               <center> <h3>Recipient Registration</h3></center>
            </div>
           </div>
        <div class="cont" align="center" >
          <h3>
          <div class="profile">
          <h3>
            <form action="{{ url_for('registers') }}" method="post">
                <table>
```

```
<tr>
  <th>Name:</th>
  <td><input type="text" name="username" size="15" /></td>
</tr>
<tr>
    <th>Age:</th>
    <td><input type="text" name="age" size="15" /></td>
  </tr>
  <tr>
    <th>Bloodgroup:</th>
    <td><input type="text" name="bloodgroup" size="15" /></td>
  </tr>
  <tr>
    <th>Gender:</th>
    <td><input type="text" name="gender" size="15" /></td>
  </tr>
  <tr>
    <th>Phone:</th>
    <td><input type="text" name="phone" size="15" /></td>
  </tr>
  <tr>
    <th>Address:</th>
    <td><input type="text" name="address" size="15" /></td>
  </tr>
  <tr>
```

```
<th>Email:</th>
<td><input type="text" name="email" size="15" /></td>
</tr>
<tr>
<th>Password:</th>
<td><input type="text" name="password" size="15" /></td>
</tr>
<tr>
<th>Confirm Password:</th>
<td><input type="text" name="confirmpassword" size="15"
/></td>
</tr>
<tr>
<th>Height:</th>
<td><input type="text" name="height" placeholder="Please write
your height in centimetres" size="15" /></td>
</tr>
<tr>
<th>Weight:</th>
<td><input type="text" name="weight" placeholder="Please write
your weight in centimetres" size="15" /></td>
</tr>
</table>
<div class="cont" align="center" >
<button><input type="submit" value="REGISTER" /></button>
</div>
```

```
            </form>
          </div>
        </div>
      </div>
    </body>

    </html>
```

LOGIN.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Login</title>
    <link rel="stylesheet"href="{{url_for('static',filename='coding.css')}}">
    <link
href="https://fonts.googleapis.com/css2?family=Kaushan+Script&display=swap"
rel="stylesheet">
</head>
<body>
  <form action="/login" method="post">
    <div class="full-page">
        <div class="sub-page">
            <div class="navigation-bar">
                <div class="logo">
                    <a href='login.html'></a>
```

```
            </div>
          </div>
          <div class="row">
            <div class="col-1">
              <div class="form-box">
                <div class="form">
                  <form class="login-form">
                    <center><h1 class="main-heading">Login
Form</h1></center>
                    <br>
                    <input type="text" name="username" placeholder="Enter
Your Username" size="15" />
                    <br>
                    <input type="text" name="password" placeholder="Enter Your
password" size="15" />
                                        <button>
                      <input type="submit" value="LOGIN" />
                    </button>
                  </form>
                </div>
                </div>
            </div>
          </div>
        </div>
    </div>
    <script src='https://code.jquery.com/jquery-3.2.1.min.js'>
```

```
    </script>
    <script>
        $('.message a').click(function(){$('form').animate({height: "toggle",opacity: "toggle"},"slow");});
    </script>
  </form>
</body>
</html>
```

DASHBOARD.html:

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="UTF-8">
    <link href='https://unpkg.com/boxicons@2.0.7/css/boxicons.min.css' rel='stylesheet'>
     <meta name="viewport" content="width=device-width, initial-scale=1.0">
     <style>
       /* Googlefont Poppins CDN Link */
@import url('https://fonts.googleapis.com/css2?family=Poppins:wght@200;300;400;500;600;700&display=swap');
*{
  margin: 0;
  padding: 0;
  box-sizing: border-box;
```

```
  font-family: 'Poppins', sans-serif;
}
.sidebar{
  position: fixed;
  height: 100%;
  width: 240px;
  background: red;
  transition: all 0.5s ease;
}
.sidebar.active{
  width: 60px;
}
.sidebar .logo-details{
  height: 80px;
  display: flex;
  align-items: center;
}
.sidebar .logo-details i{
  font-size: 28px;
  font-weight: 500;
  color: #fff;
  min-width: 60px;
  text-align: center
}
.sidebar .logo-details .logo_name{
```

```
  color: #fff;
  font-size: 24px;
  font-weight: 500;
}
.sidebar .nav-links{
  margin-top: 10px;
}
.sidebar .nav-links li{
  position: relative;
  list-style: none;
  height: 50px;
}
.sidebar .nav-links li a{
  height: 100%;
  width: 100%;
  display: flex;
  align-items: center;
  text-decoration: none;
  transition: all 0.4s ease;
}
.sidebar .nav-links li a.active{
  background: #081D45;
}
.sidebar .nav-links li a:hover{
  background: #081D45;
```

```
}
.sidebar .nav-links li i{
  min-width: 60px;
  text-align: center;
  font-size: 18px;
  color: #fff;
}
.sidebar .nav-links li a .links_name{
  color: #fff;
  font-size: 15px;
  font-weight: 400;
  white-space: nowrap;
}
.sidebar .nav-links .log_out{
  position: absolute;
  bottom: 0;
  width: 100%;
}

nav .profile-details img{
  height: 40px;
  width: 40px;
  border-radius: 6px;
  object-fit: cover;
}
```

```
nav .profile-details .admin_name{
  font-size: 15px;
  font-weight: 500;
  color: #333;
  margin: 0 10px;
  white-space: nowrap;
}
nav .profile-details i{
  font-size: 25px;
  color: #333;
}

/* Responsive Media Query */
@media (max-width: 1240px) {
  .sidebar{
    width: 80px;
  }
  .sidebar.active{
    width: 250px;
  }
  .sidebar.active ~ .home-section{
    /* width: calc(100% - 220px); */
    overflow: hidden;
    left: 220px;
  }
```

```
  .sidebar.active ~ .home-section nav{
    width: calc(100% - 220px);
    left: 220px;
  }
}

@media (max-width: 1000px) {
  .overview-boxes .box{
    width: calc(100% / 2 - 15px);

    margin-bottom: 15px;
  }
}
@media (max-width: 700px) {
  nav .sidebar-button .dashboard,
  nav .profile-details .admin_name,
  nav .profile-details i{
    display: none;
  }

@media (max-width: 550px) {
  .overview-boxes .box{
    width: 100%;
    margin-bottom: 15px;
  }
```

```
  .sidebar.active ~ .home-section nav .profile-details{
    display: none;
  }
}
  @media (max-width: 400px) {
  .sidebar{
    width: 0;
  }
  .sidebar.active{
    width: 60px;
  }
  .sidebar.active ~ .home-section{
    left: 60px;
    width: calc(100% - 60px);
  }
  }
  .sidebar.active ~ .home-section nav{
    left: 60px;
    width: calc(100% - 60px);
  }
}
     </style>
   </head>
<body>
  <div class="sidebar">
```

```
    <div class="logo-details">
      <i class='bx bxl-c-plus-plus'></i>
      <span class="logo_name">PLASMA DONOR</span>
    </div>
      <ul class="nav-links">
        <li>
          <a href="/profile">
            <i class='bx bx-box' ></i>
            <span class="links_name">profile</span>
          </a>
        </li>
        <li>
          <a href="/display">
            <i class='bx bx-box' ></i>
            <span class="links_name">Request</span>
          </a>
        </li>
      </ul>
  </div>
  <section class="home-section">
  </script>
</body>
</html>
```

PROFILE.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Profile</title>
    <link rel="stylesheet"href="{{url_for('static',filename='profile.css')}}">
</head>
<body> <div class="nav">
        <div class="logo">
          <a href="/userdashboard">  <i class="fas fa-arrow-left"></i></a>
           <center> <h3>Profile</h3></center>
        </div>
       </div>
    <div class="cont" align="center">
      <h3>
      <div class="profile">
      <h3>
        <table>
          <tr>
            <th>Name:</th>
            <td>{{username}}</td>
          </tr>
          <tr>
            <th>Age:</th>
            <td>{{age}}</td>
```

```
</tr>
<tr>
  <th>BloodGroup:</th>
  <td>{{bloodgroup}}</td>
</tr>
<tr>
  <th>Gender:</th>
  <td>{{gender}}</td>
</tr>
<tr>
  <th>Phone:</th>
  <td>{{phone}}</td>
</tr>
<tr>
  <th>Address:</th>
  <td>{{address}}</td>
</tr><tr>
  <th>E-Mail:</th>
  <td>{{email}}</td>
</tr>
<tr>
  <th>Height:</th>
  <td>{{height}}</td>
</tr>
<tr>
```

```
          <th>Weight</th>
          <td>{{weight}}</td>
         </tr>
        </table>
        </div>
    </div>
  </div>
</body>
</html>
```

DISPLAY.html

```
<html><head>
                <link
rel="stylesheet"href="{{url_for('static',filename='coding1.css')}}">

        </head>
        <body>
<div>
        <center>

    <div class="name"><span></span>
     <div class="val">
          {% for val in data %}
          <div>{{val}}</div>
          {% endfor %}
```

```
            </div>
  {% if state == "SENT" %}
</center><center>
        <div class="content " >
                <h1>Request Email Sent Successfully...! </h1>
            </div>
            </center{% else %}
        <form action="/send_email" method="post">
                <div class="content " >
                    <h1>Requesting plasma!!!<h1></div>
                <br>

                <br>

                <label> Blood Group:</label>
              <input type="text" name="bloodgroup" value="{{bloodgroup}}"
placeholder="Enter BloodGroup"/>
                <br>
                <br>
                <br>
                <label> Email:</label>
              <input type="text" name="email1" placeholder="Enter Email"/>
                <br>

                <br>
          <button type="submit" class="button1">NOTIFY</button>
```

```
            </form>
            {% endif %}

</div>
</div>
</body>
</html>
```

STATIC (style) CODE

```
*{margin: 0;
  padding: 0;
box-sizing: border-box;
}
full-page
{ height: 100vh;
 width: 100%;
 background: azure;
 position: absolute;
}
.sub-page
{
  width: 1266px;
  height: 559px;
  position: absolute;
   background: url(bb.jpg);
```

```
  background-size: cover;
   background-position: center;
 left: 50px;
top: 50px;
}navigation-bar
{
```

PROFILE.css

```
@import
url("https://fonts.googleapis.com/css2?family=Caesar+Dressing&family=Dancing
+Script:wght@500&family=Poppins:wght@400;500&display=swap");
@import
url("https://fonts.googleapis.com/css2?family=Gemunu+Libre&display=swap");
@import
url("https://fonts.googleapis.com/css2?family=Roboto+Mono&display=swap");
@import
url("https://fonts.googleapis.com/css2?family=Roboto+Mono&display=swap");
* {
  margin: 0;
  padding: 0;
}
body {
  width: 100%;
  height: 100%;
}
a {
```

```
  text-decoration: none;
}
.nav {
  background-color: red;
  padding: 10px;
  display: flex;
  justify-content: space-between;
}
.nav .logo {
  display: flex;
  gap: 10px;
}

.nav .logo i {
  text-align: center;
  color: bisque;
  font-size: 30px;

  padding: 8px;
}
.nav .logo h3 {
  color: bisque;
  font-size: 30px;
  font-family: "Poppins", cursive;
  text-align: center;
```

```
}
.nav .name {
  text-align: center;
  color: black;
  font-size: 20px;
  font-family: "Poppins", cursive;
}
.cont .logo {
  color: crimson;
  font-size: 90px;
}
.cont .logo i {
  border: 4px solid black;
  border-radius: 50%;
  padding: 4px;
}
.cont .profile {
  margin-top: 20px;
  width: 60%;
}
.cont .profile table {
  padding: 20px;
  width: auto;
  font-family: system-ui, -apple-system, BlinkMacSystemFont, "Segoe UI",
Roboto,
    Oxygen, Ubuntu, Cantarell, "Open Sans", "Helvetica Neue", sans-serif;
```

```
    margin-bottom: 20px;
}
.cont .profile tr {


  background-color: azure;
}
.cont .profile td {
  padding: 10px;
  text-align: center;
}
.cont .profile th {
  background-color: crimson;
  color: bisque;
  padding: 5px;
}
.cont .profile a {

  background-color: gold;
  padding: 10px;
  border-radius: 10px;
  font-family: "Roboto Mono", monospace;
}
```

STYLE.css

```
*
{
   margin: 0;
   padding: 0;
   box-sizing: border-box;
}
.full-page
{
   height: 100%;
        width: 100%;
        background-image: linear-
gradient(rgba(0,0,0,0.4),rgba(0,0,0,0.4)),url(bloodimage1.png);
        background-position: center;
        background-size: cover;
        position: absolute;
}
.navbar
{
   display: flex;
   align-items: center;
   padding: 20px;
   padding-left: 50px;
   padding-right: 30px;
   padding-top: 50px;
}
nav
```

```
{
    flex: 1;
    text-align: right;
}
nav ul
{
    display: inline-block;
    list-style: none;
}
nav ul li
{
    display: inline-block;
    margin-right: 70px;
}
nav ul li a
{
    text-decoration: none;
    font-size: 20px;
    color: white;
    font-family: sans-serif;
}
nav ul li button
{
    font-size: 20px;
    color: white;
```

```
    outline: none;
    border: none;
    background: transparent;
    cursor: pointer;
    font-family: sans-serif;
}
nav ul li button:hover
{
    color: aqua;
}
nav ul li a:hover
{
    color: aqua;
}
a
{
    text-decoration: none;
    color: palevioletred;
    font-size: 28px;
}
#login-form
{
    display: none;
}
.form-box
```

```
{
   width:380px;
        height:480px;
        position:relative;
        margin:2% auto;
        background:rgba(0,0,0,0.3);
        padding:10px;
   overflow: hidden;
}
.button-box
{
        width:220px;
        margin:35px auto;
        position:relative;
        box-shadow: 0 0 20px 9px #ff61241f;
        border-radius: 30px;
}
.toggle-btn
{
        padding:10px 30px;
        cursor:pointer;
        background:transparent;
        border:0;
        outline: none;
        position: relative;
```

```
}
#btn
{
        top: 0;
        left:0;
        position: absolute;
        width: 110px;
        height: 100%;
        background: #F3C693;
        border-radius: 30px;
        transition: .5s;
}
.input-group-login
{
        top: 150px;
        position:absolute;
        width:280px;
        transition:.5s;
}
.input-group-register
{
    top: 120px;
        position:absolute;
        width:280px;
        transition:.5s;
```

```
}
.input-field
{
	width: 100%;
	padding:10px 0;
	margin:5px 0;
	border-left:0;
	border-top:0;
	border-right:0;
	border-bottom: 1px solid #999;
	outline:none;
	background: transparent;
}
.submit-btn
{
	width: 85%;
	padding: 10px 30px;
	cursor: pointer;
	display: block;
	margin: auto;
	background: #F3C693;
	border: 0;
	outline: none;
	border-radius: 30px;
}
```

```
.check-box
{
	margin: 30px 10px 34px 0;
}
span
{
	color:#777;
	font-size:12px;
	bottom:68px;
	position:absolute;
}
#login
{
	left:50px;
}
#login input
{
	color:white;
	font-size:15;
}
#register
{
	left:450px;
}
#register input
```

```
{
        color:white;
        font-size: 15;
}
```

APP.PY

```
from flask import Flask, render_template, request, redirect, url_for, session
import ibm_db
from flask_mail import Mail, Message
import re
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'] = 'karpagamboothanathan@gmail.com'
app.config['MAIL_PASSWORD'] = 'nmydiidhksvvojid'
app.config['MAIL_USE_TLS'] = False
app.config['MAIL_USE_SSL'] = True
mail = Mail(app)
app.secret_key = 'a'
```

```
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=9938aec0-8105-433e-8bf9-0fbb7e483086.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32459;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=hnl10309;PWD=4q2SvdOKme6RC7Zr","","")

@app.route('/')

def home():

    return render_template('index.html')


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

def login():

    msg = ''

    if request.method == 'POST'and 'username' in request.form and 'password' in request.form:

        username = request.form['username']

        password = request.form['password']

        stmt = ibm_db.prepare(conn,'SELECT * FROM donor WHERE username = ? AND password = ?')

        ibm_db.bind_param(stmt,1,username)

        ibm_db.bind_param(stmt,2,password)

        ibm_db.execute(stmt)

        account = ibm_db.fetch_tuple(stmt)

        stmt = ibm_db.prepare(conn,'SELECT * FROM recipient WHERE username = ? AND password = ?')

        ibm_db.bind_param(stmt,1,username)

        ibm_db.bind_param(stmt,2,password)

        ibm_db.execute(stmt)

        account = ibm_db.fetch_tuple(stmt)
```

```
        if account:
            session['loggedin'] = True
            session['username'] = account[1]
            msg = 'Logged in successfully !'
            return redirect(url_for('dashboard'))
        else:
            msg = 'Incorrect username / password !'
            return render_template('login.html', a = msg)
    return render_template('login.html')

@app.route('/dashboard')
def dashboard():
    return render_template('dashboard.html')

@app.route('/logout')
def logout():
    session.pop('loggedin', None)
    session.pop('id', None)
    session.pop('username', None)
    return redirect(url_for('login'))

@app.route('/profile',methods=["POST","GET"])
def profile():
    if 'username' in session:
        id = session['username']
```

```
        stmt = ibm_db.prepare(conn, 'SELECT * FROM donor WHERE username =
?')
        ibm_db.bind_param(stmt, 1, id)
        ibm_db.execute(stmt)
        acc = ibm_db.fetch_tuple(stmt)


        stmt = ibm_db.prepare(conn, 'SELECT * FROM recipient WHERE username
= ?')
        ibm_db.bind_param(stmt, 1, id)
        ibm_db.execute(stmt)
        acc = ibm_db.fetch_tuple(stmt)
        return render_template('profile.html',
username=acc[1],age=acc[2],bloodgroup=acc[3],gender=acc[4],phone=acc[5],addr
ess=acc[6],email=acc[7],passsword=acc[8],confirmpassword=acc[9],height=acc[1
0],weight=acc[11])
    return render_template('profile.html')




@app.route('/send_email', methods = ['POST'])
def send():
    if 'bloodgroup' in request.form:
        type = request.form['bloodgroup']
        remail=request.form['email1']
        stmt = ibm_db.prepare(conn, 'SELECT email FROM donor WHERE
bloodgroup = ?')
```

```
        ibm_db.bind_param(stmt,1,type)
        ibm_db.execute(stmt)
        acc = ibm_db.fetch_tuple(stmt)


        mails = []
        while acc != False:
            mails.append(acc[0])
            acc = ibm_db.fetch_tuple(stmt)
        msg = Message('Blood Request',
sender='karpagamboothanathan@gmail.com', recipients=mails)
        msg.body = "I need " + request.form['bloodgroup'] + " Blood(plasma).  "  +
remail + " - contact this email to donate"
        mail.send(msg)
        return render_template('display.html', state="SENT")


    return 'Please Provide Blood in Form'


@app.route('/display',methods=["POST","GET"])
def display():
    if request.method == 'POST' :
        alert = ''
        if 'name' in session:
            id = session['username']
            bloodgroup = request.form['bloodgroup']
            sqlselect = "SELECT * FROM recipient WHERE bloodgroup = ? order by
id asc"
```

```
            stmt = ibm_db.prepare(conn,sqlselect)
            ibm_db.bind_param(stmt, 1, bloodgroup)
            ibm_db.execute(stmt)
            acc = ibm_db.fetch_tuple(stmt)

        return render_template('display.html',m=alert)
    return render_template('display.html')



@app.route('/register', methods =['GET', 'POST'])
def register():
    msg = ''
    if request.method == 'POST':
        username = request.form['username']
        age = request.form['age']
        bloodgroup = request.form['bloodgroup']
        lastdonatedate= request.form['lastdonatedate']
        gender = request.form['gender']
        phone= request.form['phone']
        address= request.form['address']
        email = request.form['email']
        password = request.form['password']
        confirmpassword = request.form['confirmpassword']
        height= request.form['height']
        weight = request.form['weight']
```

```
        sql = "SELECT * FROM donor 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)
        msgg = Message(
              'Hello',
              sender ='karpagamboothanathangmail.com',
              recipients = [email]
             )
        msgg.body = ' Thank you...You are successfully registered in plasma donor application.'
        mail.send(msgg)


        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 = 'Username must contain only characters and numbers !'
        elif not username or not password or not username:
            msg = 'Please fill out the form !'
        else:
            insert_sql = "INSERT INTO donor(username,age,bloodgroup,lastdonatedate,gender,phone,address,email,password,confirmpassword,height,weight) VALUES(?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?,?)"
```

```
            stmt = ibm_db.prepare(conn,insert_sql)
            ibm_db.bind_param(stmt, 1, username )
            ibm_db.bind_param(stmt, 2, age)
            ibm_db.bind_param(stmt, 3, bloodgroup)
            ibm_db.bind_param(stmt, 4, lastdonatedate )
            ibm_db.bind_param(stmt, 5, gender )
            ibm_db.bind_param(stmt, 6, phone )
            ibm_db.bind_param(stmt, 7, address )
            ibm_db.bind_param(stmt, 8, email )
            ibm_db.bind_param(stmt, 9, password )
            ibm_db.bind_param(stmt, 10, confirmpassword )
            ibm_db.bind_param(stmt, 11, height)
            ibm_db.bind_param(stmt, 12, weight )
            ibm_db.execute(stmt)
            msg = 'You have successfully registered !'
    return render_template('donor.html', msg = msg)

@app.route('/registers', methods =['GET', 'POST'])
def registers():
    msg = ''
    if request.method == 'POST':
        username = request.form['username']
        age = request.form['age']
        bloodgroup = request.form['bloodgroup']
        gender = request.form['gender']
```

```
        phone= request.form['phone']
        address= request.form['address']
        email = request.form['email']
        password = request.form['password']
        confirmpassword = request.form['confirmpassword']
        height= request.form['height']
        weight = request.form['weight']
        sql = "SELECT * FROM recipient 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)
        msgg = Message(
                'Hello',
                sender ='karpagamboothanathangmail.com',
                recipients = [email]
               )
        msgg.body = ' Thank you...You are successfully registered in plasma donor application.  '
        mail.send(msgg)
        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 = 'Username must contain only characters and numbers !'
```

```
        elif not username or not password or not username:
            msg = 'Please fill out the form !'
        else:
            insert_sql = "INSERT INTO
recipient(username,age,bloodgroup,gender,phone,address,email,password,confirm
password,height,weight) VALUES(?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)"
            stmt = ibm_db.prepare(conn,insert_sql)
            ibm_db.bind_param(stmt, 1, username )
            ibm_db.bind_param(stmt, 2, age)
            ibm_db.bind_param(stmt, 3, bloodgroup)
            ibm_db.bind_param(stmt, 4, gender )
            ibm_db.bind_param(stmt, 5, phone )
            ibm_db.bind_param(stmt, 6, address )
            ibm_db.bind_param(stmt, 7, email )
            ibm_db.bind_param(stmt, 8, password )
            ibm_db.bind_param(stmt, 9, confirmpassword )
            ibm_db.bind_param(stmt, 10, height)
            ibm_db.bind_param(stmt, 11, weight )
            ibm_db.execute(stmt)
            msg = 'You have successfully registered !'
    return render_template('recipient.html', msg = msg)


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

GITHUB LINK:

https://github.com/IBM-EPBL/IBM-Project-25233-1659955267

DEMO LINK:

https://youtu.be/Zu-Y3dXnulE