

PLASMA DONOR APPLICATION

A Project Report

Submitted by

TEAM ID	PNT2022TM1D32947
SIVANANDHINI R	820419104068
SATHYA JAYASRI S	820419104064
VIJITHRA P	820419104084
SRIRAM M	8204191040302

1. INTRODUCTION

1.1 Project Overview

Although the government is carrying out Covid vaccination campaigns on a large scale, the number of vaccines produced is not enough for all the population to get vaccinated at present. And with the corona positive cases rising every day, saving lives has become the prime matter of concern. As per the data provided by WHO more than 3 million people have died due to the coronavirus .However, apart from vaccination, there is another scientific method by which a covid infected person can be treated and the death risk can be reduced.

This plasma therapy is an experimental approach to treat corona-positive patients and help them recover. This plasma therapy is considered to be safe & promising. A person who has recovered from Covid can donate his/her plasma to a person who is infected with the coronavirus.

1.2 Purpose

With rapid increase in the usage of social networks sites across the world, there is also a steady increase in blood donation requests as being noticed in the number of posts on these sites such as Facebook and twitter seeking blood donors. Finding blood donor is a challenging issue in almost every country. There are some blood donor finder applications in the market such as blood app by Red Cross and Blood Donor Finder application by Neologix. However, more reliable applications that meet the needs of users are prompted. So is the plasma donation also has need of application for donating plasma in the situations like covid.

2. LITERATURE SURVEY

2.1 Existing problem

Conventionally, when a patient needs plasma, he/she has to contact a blood bank or a compatible blood group of a donor in their circle, family, and friends. However, it is difficult to find suitable donor within a limited group of people in a given time. In addition, there is no guarantee that blood banks will have compatible blood group in stock. There is also steady increase in blood donation requests posts in social networking sites (like Facebook, twitter, Instagram, etc.) requesting for donation.

There are some plasma donor finder applications which allows the donor to book appointment with blood banks and also can find local blood drives and donation centers

quickly and easily . However, there is no direct communication between the donor and that clinic in need of a specific blood type. As a result, this app is more beneficial for donors but not for clinics to find needed blood type directly and promptly.

2.2 References

- [1] <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5682362/>
- [2] <https://nevonprojects.com/instant-plasma-donor-recipient-connector-android-app/h>
- [3] "Serverless computing:Economic and architectural impact",ESEC/FSE,2017. According to R. C. Gojko Adzic ,in this paper the author has carried out analysis based on the opportunities presented by serverless computing. They emphasize that serverless services are more affordable approach for many network services and it is more user friendly as serverless approach will relieve the customers from the intricacies of deployment. These services will help to improve the new business opportunities.
- [4] "Building a chatbot with severless computing",IBM watson research center,2016. According to C. P. C. a. V. I. M. Yan ,in this paper author conducted a survey of existing serverless platform in this paper from source projects, industry, academia, use cases, and key characteristics and has described the challenges and the open problems associated with it. Authors work presented a experience of serverless technologies using different services from different cloud provides such as Amazon, Google, IBM, Microsoft Azure.
- [5] "Cloud Event Programming Paradigms:Applications and Analysis", "9th IEEE International Conference on Cloud Computing(CLOUD),pp.pp.400 - 406,2017. According to S. E. a. B. J. J. Short , in this paper three demonstrators for IBM Bluemix OpenWhisk was presented. They exhibit even-based programming triggered by weather forecast data, speech utterances and Apple WatchOS2 application data. And also demonstrated a chatbot using IBM Bluemix OpenWhisk that calls on the IBM Watson services which include dates, weather, alarm services, news and music tutor.
- [6] "Making Serverless Computing More Serverless", IEEE 11th International Conference on Cloud Computing (CLOUD), pp. pp. 456-459, 2018., 2018. According to S. Z. Al-Ali , in this paper serverlessOS was designed. It comprises of components such as 1. desegregation model that leverages desegregation for abstraction but it will enable resources to move fluidly between servers for the performance. 2. The second key component is cloud orchestration layer which helps to manage fine-grained resource placement and allocation throughout the application lifetime with the help of global and local decision making 3. And the third component is an isolation capability which enforces data and resource isolation.

[7] “EMARS: Efficient Management and Allocation of Resources in Serverless”, IEEE 11th International Conference on Cloud Computing (CLOUD), pp. pp. 827-830, 2018. According to A. S. a. S. Jindal , in this paper an efficient resource management system for serverless computing framework was proposed which aims to enhance resource with a focus on memory allocation among the containers and the design which was added on top of an open-source serverless platform, openLambda and it is based on allocation workloads and serverless functions memory needs events are triggered.

2.3 Problem Statement Definition

Create a problem statement to understand your customer's point of view. The Customer Problem Statement template helps you focus on what matters to create experiences people will love.

A well-articulated customer problem statement allows you and your team to find the ideal solution for the challenges your customers face. Throughout the process, you'll also be able to empathize with your customers, which helps you better understand how they perceive your product or service.

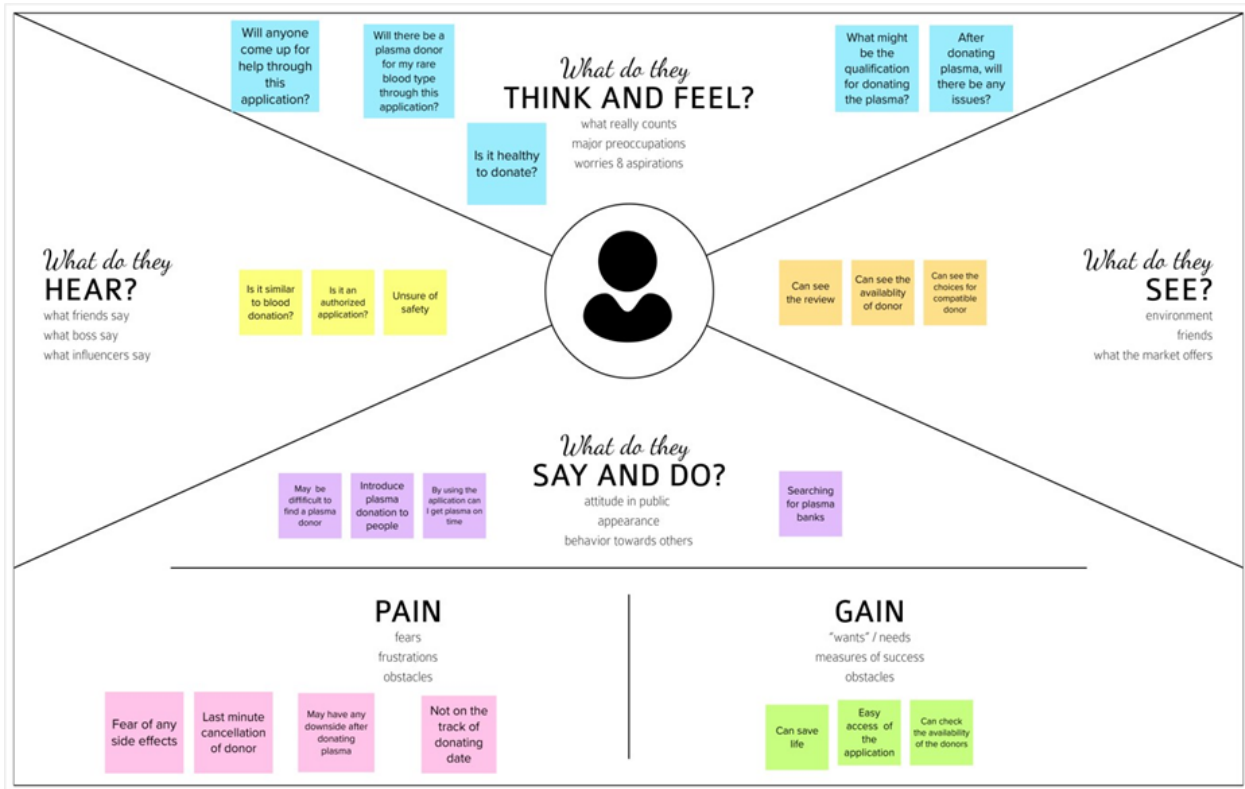
Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	A donor	Donate plasma	I'm unaware of the platforms	I don't know much about the plasma donation sites.	Dim-witted
PS-2	A recipient	Get plasma	I'm unable to find donors	I'm unaware of the donors availability	Depressed about my life

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes.

It is a useful tool to help teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.




3.2 Ideation & Brainstorming

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

[Share template feedback](#)

➔

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

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

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

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

[Open article](#) ➔


1

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

PROBLEM
How might we get the correct and necessary information on donor and recipient?



Key rules of brainstorming

To run a smooth and productive session

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

[illegible]

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, 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.

BLOOD GROUP INFORMATION

Donor blood group
A, B, AB, O
Rh factor: Rh+

Match with the patient
A, B, AB, O
Rh factor: Rh+

Is patient compatible with donor blood group?
Yes/No

CANCELLATION INFORMATION

Is the donor blood group
A, B, AB, O
Rh factor: Rh+

Is the patient blood group
A, B, AB, O
Rh factor: Rh+

Is the patient compatible with donor blood group?
Yes/No

DONOR INFORMATION

Is the donor blood group
A, B, AB, O
Rh factor: Rh+

Is the patient blood group
A, B, AB, O
Rh factor: Rh+

Is the patient compatible with donor blood group?
Yes/No

REGISTERED DONOR INFORMATION

Donor blood group
A, B, AB, O
Rh factor: Rh+

Is the patient blood group
A, B, AB, O
Rh factor: Rh+

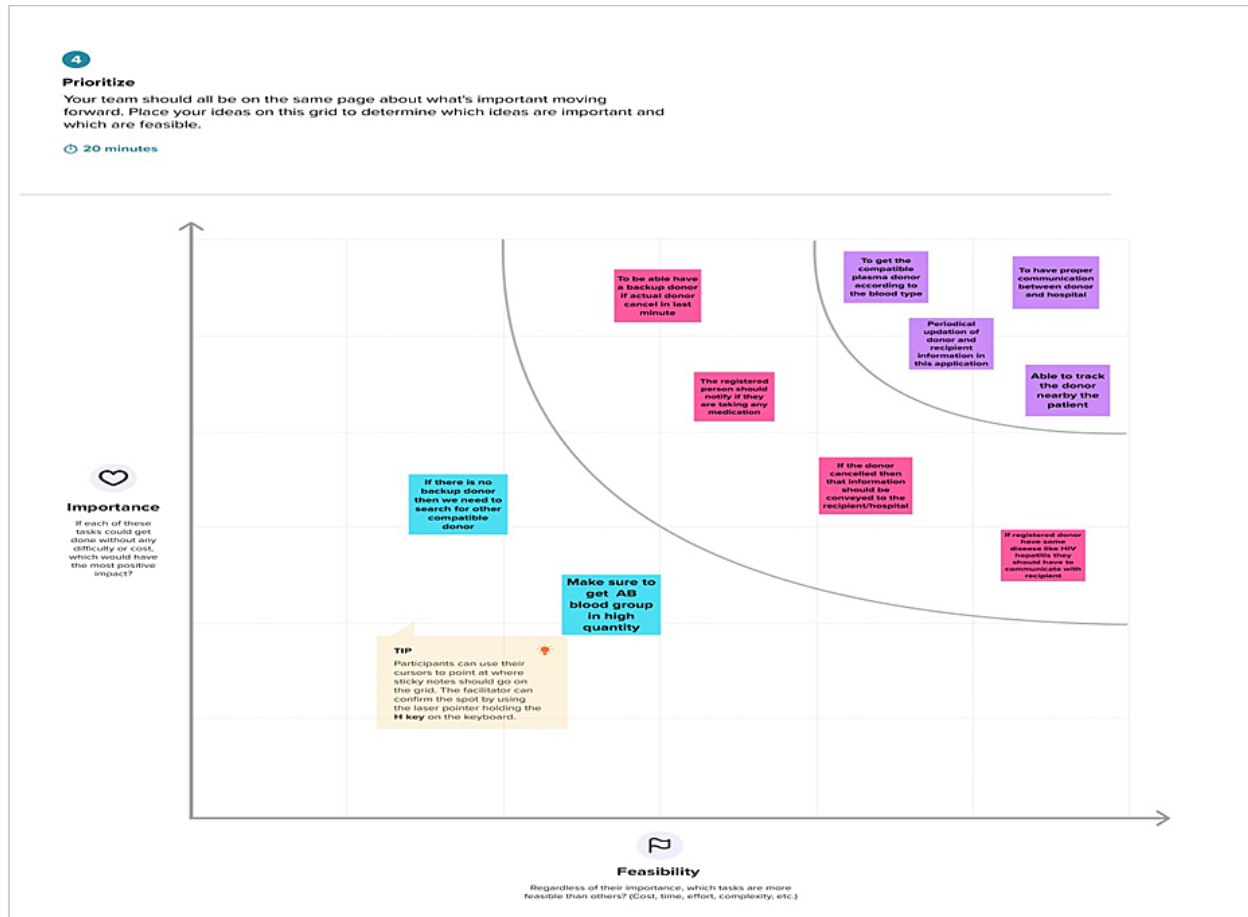
Is the patient compatible with donor blood group?
Yes/No

TP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mind.

Tip Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mind.

Step-3: Idea Prioritization





3.3 Proposed Solution

Project team shall fill the following information in proposed solution template.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Obtaining data about the availability of plasma in hospitals, blood banks and recipients blood type information is not quite easy.

2.	Idea / Solution description	The application will link all donors, control a plasma transfusion service and create a database to hold data on stocks of plasma in each area.
3.	Novelty / Uniqueness	The application make sure to have a constant availability AB type as its a universal plasma.
4.	Social Impact / Customer Satisfaction	By this application, it helps the donors to donate plasma for required recipients which helps in saving life's in covid situation and creating awareness on donating plasma.
5.	Business Model (Revenue Model)	The need of plasma increases day by day as there is covid so it increases the revenue.
6.	Scalability of the Solution	The demand of the plasma can be met through this application as it provides simplicity over complexity which helps the user to use them with ease.

3.4 Problem Solution fit

<p>1. CUSTOMER SEGMENT(S)</p>  <p>The customer who may be recipient or donor who should be above the age of 18 and below 65 would be able to donate and receive plasma and donor should have the weight of above 50 kg.</p>	<p>6. CUSTOMER CONSTRAINTS</p>  <p>The Donor should not be affected by any diseases such as HIV, Hepatitis etc.</p>	<p>5. AVAILABLE SOLUTIONS</p> <p>Can get the person directly to donate plasma. Getting plasma directly should be less time as a person donating plasma is nearby.</p>
--	---	---

Focus on J&P, tap into BE, understand RC	2. JOBS-TO-BE-DONE / PROBLEMS	J&P	9. PROBLEM ROOT CAUSE	RC	7. BEHAVIOUR	BE
	donor should be able to meet the requirements in order to donate the plasma.		Insufficient information about the plasma availability.		The Customer should give the correct details about the medication and health condition in order to prevent any future problem.	
	Inadequate information about the customer may cause vital problem. Even if the recipient get the hold of <u>donor</u> , the transfusion must not be delayed as it can destroy a life. Rumors may affect the count of the donor.		Any unfortunate situation may occur to the donor		The customer tries their best to donate plasma in the registered time.	

Identify strong TR & EM	3. TRIGGERS	TR	10. YOUR SOLUTION	SL	8. CHANNELS of BEHAVIOUR	CH	Identify Strong TR & EM
	<p>Good intention of the person may help in donating.</p> <p>The influence from other people who is donating plasma.</p> <p>The urge to save a life</p>		<p>Get in connection with the previous donors as they can donate in the regular basis.</p> <p>Get in connection with the recipient or an COVID patient after the transfusion.</p> <p>To be in contact with the hospital or patient to know the demand of plasma and to have check in with the blood bank.</p>		<p>8.1 ONLINE</p> <p>Creating Awareness.</p> <p>Finding help easily through social media platforms.</p> <p>8.2 OFFLINE</p> <p>People who are nearby donating center may be able to interact and deliver plasma in the regular interval of time.</p>		
	4. EMOTIONS: BEFORE / AFTER	EM					
	<p>The donor may have the fear of side effects.</p> <p>The recipient may <u>doubting</u> themselves after receiving plasma will they be able to live or not.</p>						

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story/ Sub-Task)
FR-1	User Registration	Registration throughForm Registration through Gmail Registration throughLinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User Login	Login viaGmail.
FR-4	User Credentials	Credentials of users is submitted.
FR-5	User Verification	User credentials are verified.
FR-6	Donor & Recipient Confirmation	Donor &Recipient are allocated to a certain time.

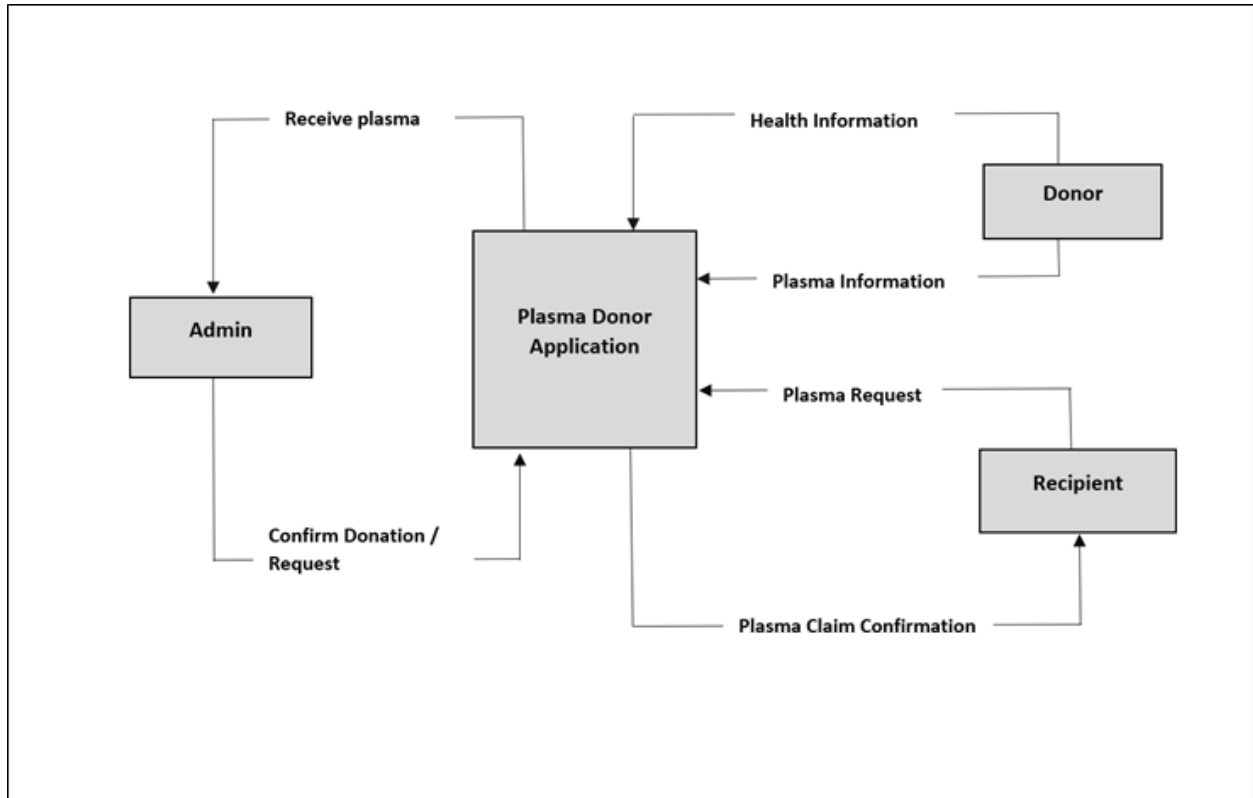
4.2 Non-Functional requirements

Following are the non-functional requirements of the proposedsolution.

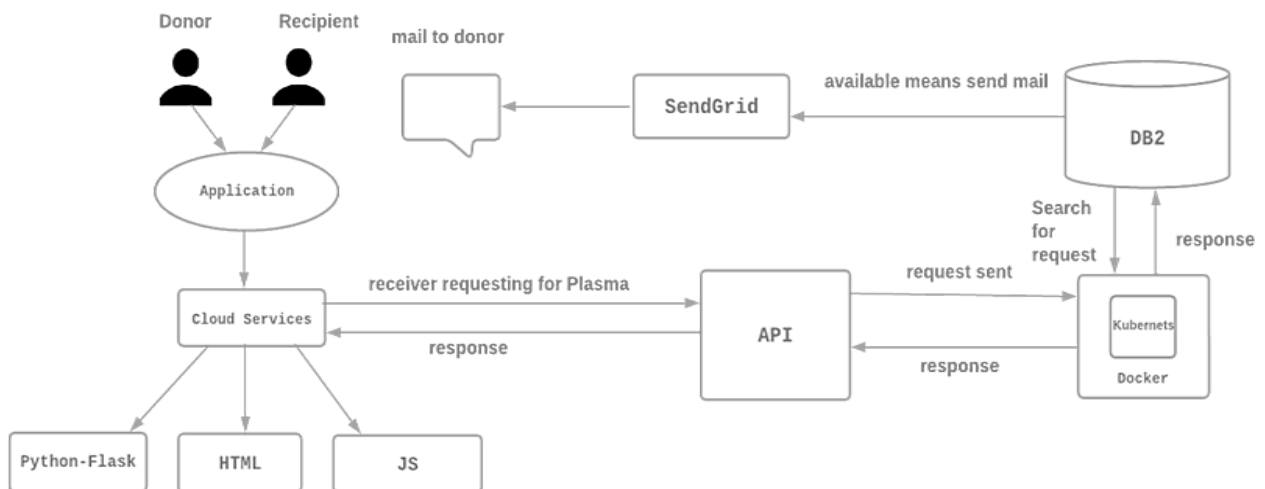
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	The quality of experience wheninteracting with the application willbe attained.
NFR-2	Security	The application prevents the donors and recipients data frombeing hijacked or misused.
NFR-3	Reliability	The application worksunder specific need of plasma in required time.
NFR-4	Performance	The application triesto provide quick responses to the recipients.
NFR-5	Availability	The application runs properly and meets the userrequirements.
NFR-6	Scalability	The application can handle more usersand evolve concurrently as per needs.

5. PROJECT DESIGN

5.1 Data Flow Diagrams



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 (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 confirmation 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 Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail	I can register and access dashboard with Gmail Login	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email &	I can receive confirmation email and register	High	Sprint-1

			password			
	Dashboard	USN-6	As a user, I can browse the dashboard by logging in before	I can access the resources in the dashboard	High	Sprint-1
Customer (Webuser)		USN-7	As a user, I can login to the application and see the categories of donor and recipient	I can see the information about the donor and recipient	Low	Sprint-2
Customer Care Executive		USN-8	As a user, I can see the feedback and comment	I can rectify the problem that are commented in the feedback session	High	Sprint-1
Administrator		USN-9	As a user, I can see the availability of donor information	I can update the donor information	Medium	Sprint-1
		USN-10	As a user, I can see the demand of the donor by the recipients request	I can fix the required donor for the requested recipients	Medium	Sprint-1

		USN-11	As a user, I can inform the donor and recipients of the date and time to donate	I can notify the donor and the recipients	Medium	Sprint-1
		USN-12	As a user, I have the information of previous donors and recovered covid patients	I can notify them to donate again	Low	Sprint-1

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 user, I can register for the application by entering my email, password, and confirming my password.	2	High	4
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	4
Sprint-2		USN-3	As a user, I can register for the application through Facebook	2	Low	4
Sprint-1		USN-4	As a user, I can register for the application through Gmail	2	Medium	4
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password	1	High	4

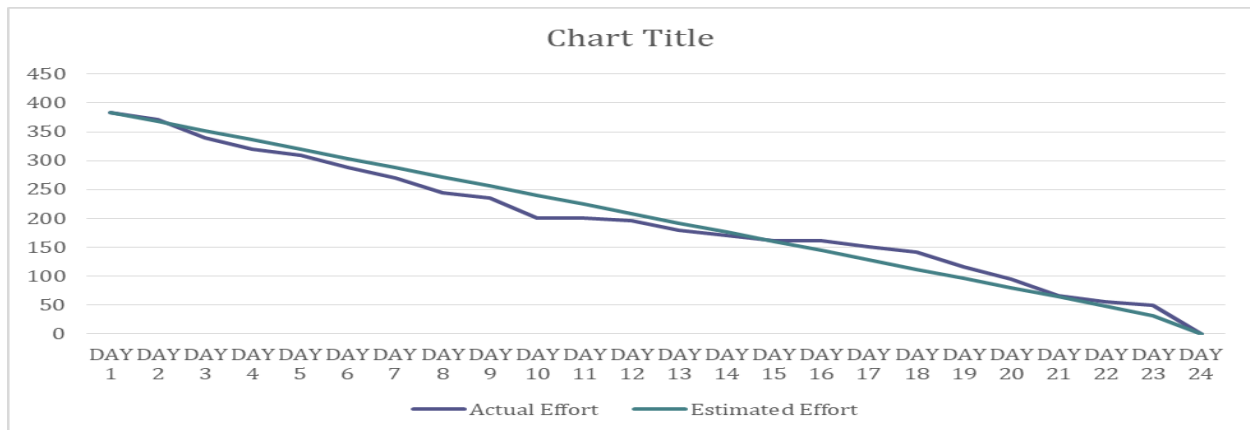
Sprint-3	Dashboard	USN-6	As a user, I can find the compatible donor by registering.	3	High	4
Sprint-3		USN-7	As a user, I can find the donor availability by logging in.	3	High	4
Sprint-2		USN-8	As a user, I can create a profile by registering.	2	Medium	4
Sprint-3		USN-9	As a user, I can see the demand of plasma.	3	Medium	4
Sprint-4	Database	USN-10	As a user, I can store the availability and need of plasma information value.	4	High	4

6.2 Sprint Delivery Schedule

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	6	6 Days	24 Oct 2022	29 Oct 2022		
Sprint-2	4	6 Days	31 Oct 2022	05 Nov 2022		
Sprint-3	9	6 Days	07 Nov 2022	12 Nov 2022		
Sprint-4	4	6 Days	14 Nov 2022	19 Nov 2022		

6.3 Reports from JIRA

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.



7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1

7.1.1 Login.html

```

<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
<meta charset="UTF-8">
<title>Plasma Donor App</title>
<link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet'
type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet'
type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet'
type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300'
rel='stylesheet' type='text/css'>
<link rel="stylesheet" href="style.css">
<style>
.login{
top: 20%;
}
</style>
</head>
<body>
<div class="header">
<div>Plasma Donor App</div>

```

```

<ul>
<li><a href="/registration">Register</a></li>
<li><a class="active" href="/login">Home</a></li>
</ul>
</div>
<div class="login" >
<div>
</div><!-- Main Input For Receiving Query to our ML -->
<form action="{{ url_for('loginpage')}}"method="post">
<input type="text" name="user" placeholder="Enter UserName"
required="required" style="color:black" />
<input type="password" name="passw" placeholder="Enter Password"
required="required" style="color:black" />
<button type="submit" class="btn btn-primary btn-block btn-
large">Login</button>
</form>
<br><br>
<div style="color:black">
{{ pred }}</div>
</div>
</body>
</html>

```



7.1.2 Register.html

```

<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
<meta charset="UTF-8">
<title>Plasma Donor App</title>
<link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet'

```

```

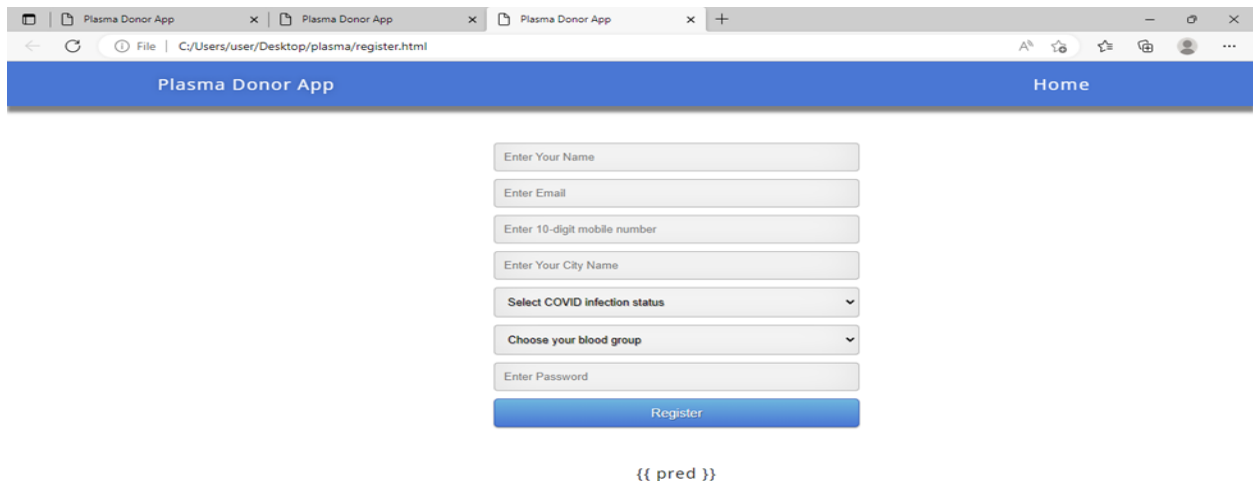
type='text/css'>
    <link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet'
type='text/css'>
    <link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet'
type='text/css'>
    <link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300'
rel='stylesheet' type='text/css'>
    <link rel="stylesheet" href="style.css">
    <style>
    .login{
    top: 20%;
    }
    </style>
    </head>
    <body>
    <div class="header">
    <div>Plasma Donor App</div>
    <ul>
    <li><a class="active" href="/login">Home</a></li>
    </ul>
    </div>
    <div class="login">
    <!-- Main Input For Receiving Query to our ML -->
    <form action="{ { url_for('register') } }" method="post">
    <input type="text" name="name" placeholder="Enter Your Name"
required="required" style="color:black"/>
    <input type="email" name="email" placeholder="Enter Email"
required="required" style="color:black"/>
    <input type="text" name="phone" placeholder="Enter 10-digit mobile number"
required="required" style="color:black"/>
    <input type="city" name="city" placeholder="Enter Your City Name"
required="required" style="color:black"/>
    <select name="infect">
    <option value="select" selected>Select COVID infection status</option>
    <option value="infected">Infected</option>
    <option value="uninfected">Uninfected</option>
    </select>
    <select name="blood">
    <option value="select" selected>Choose your blood group</option>
    <option value="O Positive">O Positive</option>
    <option value="A Positive">A Positive</option>

```

```

        <option value="B Positive">B Positive</option>
        <option value="AB Positive">AB Positive</option>
        <option value="O Negative">O Negative</option>
        <option value="A Negative">A Negative</option>
        <option value="B Negative">B Negative</option>
        <option value="AB Negative">AB Negative</option>
    </select>
    <input type="password" name="passw" placeholder="Enter Password"
required="required" style="color:black"/>
    <button type="submit" class="btn btn-primary btn-block btn-
large">Register</button>
</form>
<br><br>
<div style="color:black">
{{ pred }}</div>
</div>
</body>
</html>

```



7.1.3 Dashboard.html

```

<!DOCTYPE html>
<html lang="en">
<head>
<title>Plasma Donar App</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">

```

```

        <linkrel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
        <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
        <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></scrip
t>
        <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
        <link rel="stylesheet" href="style.css">
    </head>
    <style>
    .big{
    top:70;
    background-color:white;
    margin-top:80px;
    margin-left:550px;
    margin-right:550px;
    height:200px;
    border-radius: 25px;
    border: 3px solid #4a77d4;
    box-shadow: 6px 8px 4px grey;
    text-align:center;
    }
    .row{
    height:150px;
    }
    .col{
    margin:10px;
    margin-left:50px;
    margin-right:50px;
    border-radius: 25px;<!DOCTYPE html>
    <html lang="en">
    <head>
    <title>Plasma Donar App</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <linkrel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

```

```

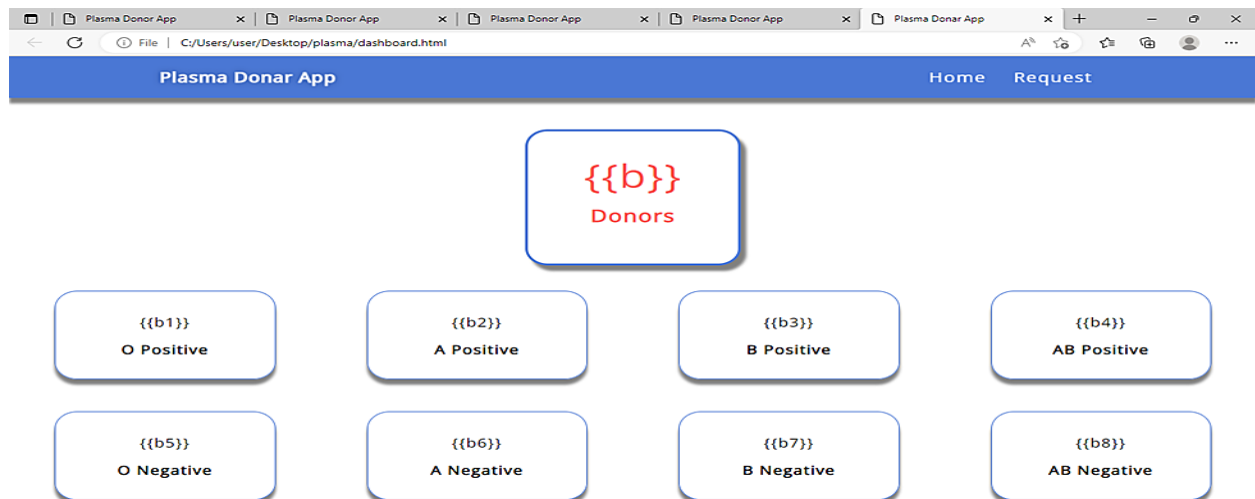
        <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></scrip
t>
        <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
        <link rel="stylesheet" href="style.css">
        </head>
        <style>
        .big{
        top:70;
        background-color:white;
        margin-top:80px;
        margin-left:550px;
        margin-right:550px;height:200px;
        border-radius: 25px;
        border: 3px solid #4a77d4;
        box-shadow: 6px 8px 4px grey;
        text-align:center;
        }
        .row{
        height:150px;
        }
        .col{
        margin:10px;
        margin-left:50px;
        margin-right:50px;
        border-radius: 25px;
        <div class="ext1"><font size="20px">{{b}}</font><br><b>Donors</b></div>
        </div>
        </div>
        <br>
        <div class="row">
        <div class="col" >
        <div class="ext">{{b1}}<br><b>O Positive</b></div>
        </div>
        <div class="col" >
        <div class="ext">{{b2}}<br><b>A Positive</b></div>
        </div>
        <div class="col" >
        <div class="ext">{{b3}}<br><b>B Positive</b></div>
        </div>

```

```

<div class="col" >
<div class="ext">{{b4}}<br><b>AB Positive</b></div>
</div>
</div> <br>
<div class="row">
<div class="col" >
<div class="ext">{{b5}}<br><b>O Negative</b></div>
</div>
<div class="col" >
<div class="ext">{{b6}}<br><b>A Negative</b></div>
</div>
<div class="col" >
<div class="ext">{{b7}}<br><b>B Negative</b></div>
</div>
<div class="col" >
<div class="ext">{{b8}}<br><b>AB Negative</b></div>
</div>
</div>
</body>
</html>

```



7.1.4 Request.html

```

<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>

```

```

    <meta charset="UTF-8">
    <title>Plasma Donor App</title>
    <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet'
type='text/css'>
    <link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet'
type='text/css'>
    <link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet'
type='text/css'>
    <link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300'
rel='stylesheet' type='text/css'>
    <link rel="stylesheet" href="style.css">
    <style>
    .login{
    top: 20%;
    }
    </style>
    </head>
    <body>
    <div class="header">
    <div>Plasma Donor App</div>
    <ul>
    <li><a class="active" href="/login">Home</a></li>
    </ul>
    </div>
    <div class="login">

    <!-- Main Input For Receiving Query to our ML -->
    <form action="{ { url_for('register') }}" method="post">
    <input type="text" name="name" placeholder="Enter Your Name"
required="required" style="color:black"/>
    <input type="email" name="email" placeholder="Enter Email"
required="required" style="color:black"/>
    <input type="text" name="phone" placeholder="Enter 10-digit mobile number"
required="required" style="color:black"/>
    <input type="city" name="city" placeholder="Enter Your City Name"
required="required" style="color:black"/>
    <select name="infect">
    <option value="select" selected>Select COVID infection status</option>
    <option value="infected">Infected</option>
    <option value="uninfected">Uninfected</option>
    </select>

```



```

<select name="blood">
  <option value="select" selected>Choose your blood group</option>
  <option value="O Positive">O Positive</option>
  <option value="A Positive">A Positive</option>
  <option value="B Positive">B Positive</option>
  <option value="AB Positive">AB Positive</option>
  <option value="O Negative">O Negative</option>
  <option value="A Negative">A Negative</option>
  <option value="B Negative">B Negative</option>
  <option value="AB Negative">AB Negative</option>
</select>
  <input type="password" name="passw" placeholder="Enter Password"
required="required" style="color:black"/>
  <button type="submit" class="btn btn-primary btn-block btn-
large">Register</button>
</form>
<br><br>
<div style="color:black">
  {{ pred }}</div>
</div>
</body>
</html>

```



7.1.5 Style sheet

```

@import url(https://fonts.googleapis.com/css?family=Open+Sans);
.btn {
  display: inline-block;
  *display: inline;
  *zoom: 1; padding:
  4px 10px 4px;

```

```

margin-bottom: 0;
font-size: 13px;
line-height: 18px;
color: #333333;
text-align: center;
text-shadow: 0 1px 1px rgba(255, 255, 255, 0.75);
vertical-align: middle;
background-color: #f5f5f5;
background-image: -moz-linear-gradient(top, #ffffff, #e6e6e6);
background-image: -ms-linear-gradient(top, #ffffff, #e6e6e6);
background-image: -webkit-gradient(linear, 0 0, 0 100%, from(#ffffff),
to(#e6e6e6));
background-image: -webkit-linear-gradient(top, #ffffff, #e6e6e6);
background-image: -o-linear-gradient(top, #ffffff, #e6e6e6);
background-image: linear-gradient(top, #ffffff, #e6e6e6);
background-repeat: repeat-x;
filter: progid:dximagetransform.microsoft.gradient(startColorstr=#ffffff,
endColorstr=#e6e6e6, GradientType=0);
border-color: #e6e6e6 #e6e6e6 #e6e6e6;
border-color: rgba(0, 0, 0, 0.1) rgba(0, 0, 0, 0.1) rgba(0, 0, 0, 0.25);
border: 1px solid #e6e6e6;
-webkit-border-radius: 4px;
-moz-border-radius: 4px;
border-radius: 4px;
-webkit-box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0,
0, 0.05);
-moz-box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0,
0.05);
box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05);
cursor: pointer; *margin-left: .3em;
}
.btn:hover, .btn:active, .btn.active, .btn.disabled, .btn[disabled] { background-
color: #e6e6e6; }
.btn-large {
padding: 9px 14px;
font-size: 15px;
line-height: normal;
-webkit-border-radius: 5px;
-moz-border-radius: 5px;
border-radius: 5px;
}

```

```

.btn:hover {
    color: #333333;
    text-decoration: none;
    background-color: #e6e6e6;
    background-position: 0 -15px;
    -webkit-transition: background-position 0.1s linear;
    -moz-transition: background-position 0.1s linear;
    -ms-transition: background-position 0.1s linear;
    -o-transition: background-position 0.1s linear;
    transition: background-position 0.1s linear;
}

.btn-primary, .btn-primary:hover {
    text-shadow: 0 -1px 0 rgba(0, 0, 0, 0.25);
    color: #ffffff;
}

.btn-primary.active { color: rgba(255, 255, 255, 0.75); }

.btn-primary {
    background-color: #4a77d4;
    background-image: -moz-linear-gradient(top, #6eb6de, #4a77d4);
    background-image: -ms-linear-gradient(top, #6eb6de, #4a77d4);
    background-image: -webkit-gradient(linear, 0 0, 0 100%, from(#6eb6de),
to(#4a77d4));
    background-image: -webkit-linear-gradient(top, #6eb6de, #4a77d4);
    background-image: -o-linear-gradient(top, #6eb6de, #4a77d4);
    background-image: linear-gradient(top, #6eb6de, #4a77d4);
    background-repeat: repeat-x;
    filter: progid:dximagetransform.microsoft.gradient(startColorstr=#6eb6de,
endColorstr=#4a77d4, GradientType=0);
    border: 1px solid #3762bc;
    text-shadow: 1px 1px 1px rgba(0,0,0,0.4);
    box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0,
0, 0.5);
}

.btn-primary:hover, .btn-primary.active, .btn-primary.active, .btn-
primary.disabled, .btn-primary[disabled] {
    filter: none;
    background-color: #4a77d4;
}

.btn-block { width: 100%; display: block; }
* { -webkit-box-sizing: border-box; -moz-box-sizing: border-box; -ms-box-

```

```

sizing:border-box; -o-box-sizing:border-box; box-sizing:border-box; }
html { width: 100%; height:100%; overflow:hidden; }
body {
    width: 100%;
    height:100%;
    font-family: 'Open Sans', sans-serif;
    background: #fffff;
    color: #000000;

    font-size: 18px;
    text-align:center;
    letter-spacing:1.2px;
}
.header {
    top:0;
    margin:0px;
    left: 0px;
    right: 0px;
    position: fixed;
    background: #4a77d4;
    color: white;
    box-shadow: 0px 8px 4px grey;
    overflow: hidden;
    padding: 15px;
    font-size: 1.5vw;
    width: 100%;
    text-align: center;
}
.login {
    position: absolute;
    top: 70%;
    left: 50%;
    margin: -25px 0 0 -150px;
    width:400px;
    height:400px;
}

.header div { color: #fff; text-shadow: 0 0 10px rgba(0,0,0,0.3); letter-
spacing:1px; text-align:center; float:left; padding-left:150px;}
ul {
    list-style-type: none;

```

```

margin: 0;
padding: 0;
padding-right: 150px;
overflow: hidden;
}
li {
float: right;
}

li a {
display: block;
color: white;
text-align: center;
padding: 0px 15px;
text-decoration: none;
}
input {
width: 100%;
margin-bottom: 10px;
background: rgba(255,255,255,255);
border: none;
outline: none;
padding: 10px;
font-size: 13px;
color: black;
text-shadow: black;
border: 1px solid rgba(0,0,0,0.3);
border-radius: 4px;
box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px
rgba(255,255,255,0.2);
-webkit-transition: box-shadow .5s ease;
-moz-transition: box-shadow .5s ease;
-o-transition: box-shadow .5s ease;
-ms-transition: box-shadow .5s ease;
transition: box-shadow .5s ease;
}
input:focus { box-shadow: inset 0 -5px 45px rgba(100,100,100,0.4), 0 1px 1px
rgba(255,255,255,0.2); }
select {
width: 100%;
margin-bottom: 10px;

```

```

        background: rgba(255,255,255,255);
        border: none;
        outline: none;
        padding: 10px;
        font-size: 13px;
        color: #000000;
        text-shadow: 1px 1px 1px rgba(0,0,0,0.3);
        border: 1px solid rgba(0,0,0,0.3);
        border-radius: 4px;
        box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px
        rgba(255,255,255,0.2);
        -webkit-transition: box-shadow .5s ease;
        -moz-transition: box-shadow .5s ease;
        -o-transition: box-shadow .5s ease;
        -ms-transition: box-shadow .5s ease;
        transition: box-shadow .5s ease;
    }

```

7.2 Feature 2

7.2.1 App.py

```

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

import ibm_db

import json

import requests

app = Flask(__name__)

conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=fbd88901-ebdb-
4a4f-a32e-
9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32731;SECURI
TY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=cts72740;PWD=dORK
SzIzhwxBmMvg","")

@app.route('/registration')

def home():

    return render_template('register.html')

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

```

```

def register():
    x = [x for x in request.form.values()]
    print(x)
    name=x[0]
    email=x[1]
    phone=x[2]
    city=x[3]
    infect=x[4]
    blood=x[5]
    password=x[6]

    sql = "SELECT * FROM plasmadonor WHERE email =?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,email)
    ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)
    print(account)
    if account:
        return render_template('register.html', pred="You are already a member, please
login using your details")
    else:
        insert_sql = "INSERT INTO plasmadonor VALUES (?, ?, ?, ?, ?, ?, ?)"
        prep_stmt = ibm_db.prepare(conn, insert_sql)
        ibm_db.bind_param(prepare_stmt, 1, name)
        ibm_db.bind_param(prepare_stmt, 2, email)
        ibm_db.bind_param(prepare_stmt, 3, phone)

```

```

        ibm_db.bind_param(prepare_stmt, 4, city)
        ibm_db.bind_param(prepare_stmt, 5, infect)
        ibm_db.bind_param(prepare_stmt, 6, blood)
        ibm_db.bind_param(prepare_stmt, 7, password)
        ibm_db.execute(prepare_stmt)

        return render_template('register.html', pred="Registration Successful, please
login using your details")

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

@app.route('/loginpage',methods=['POST'])
def loginpage():
    user = request.form['user']
    passw = request.form['passw']
    sql = "SELECT * FROM plasmadonor WHERE email =? AND password=?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,user)
    ibm_db.bind_param(stmt,2,passw)
    ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)
    print (account)
    print(user,passw)
    if account:
        return redirect(url_for('stats'))

```



```

else:

    return render_template('login.html', pred="Login unsuccessful. Incorrect
username / password !")

@app.route('/stats')
def stats():

    "sql = "SELECT blood FROM user group by blood"

    stmt = ibm_db.prepare(conn, sql)

    ibm_db.execute(stmt)

    count = ibm_db.fetch_assoc(stmt)

    print(count)"""
return render_template('stats.html',b=5,b1=2,b2=3,b3=4,b4=2,b5=1,b6=2,b7=1,b8=1)

@app.route('/requester')
def requester():

    return render_template('request.html')

@app.route('/requested',methods=['POST'])
def requested():

    bloodgrp = request.form['bloodgrp']

    address = request.form['address']

    print(address)

    sql = "SELECT * FROM plasmadonor WHERE blood=?"

    stmt = ibm_db.prepare(conn, sql)

    ibm_db.bind_param(stmt,1,bloodgrp)

    ibm_db.execute(stmt)

    data = ibm_db.fetch_assoc(stmt)

    msg = "Need Plasma of your blood group for: "+address

```

```

while data != False:

    print ("The Phone is : ", data["PHONE"])
    url="https://www.fast2sms.com/dev/bulk?authorization=xCXuwWTzyjOD2ARd1EngbH3a7tKI
q5PklJ8YSf0Lh4FQZecs9iNI1dSvuqprxFwCKYJXA5amQkBE36Rl&sender_id=FSTSMS&me
ssage="+msg+"&language=english&route=p&numbers="+str(data["PHONE"])

    result=requests.request("GET",url)

    print(result)

    data = ibm_db.fetch_assoc(stmt)

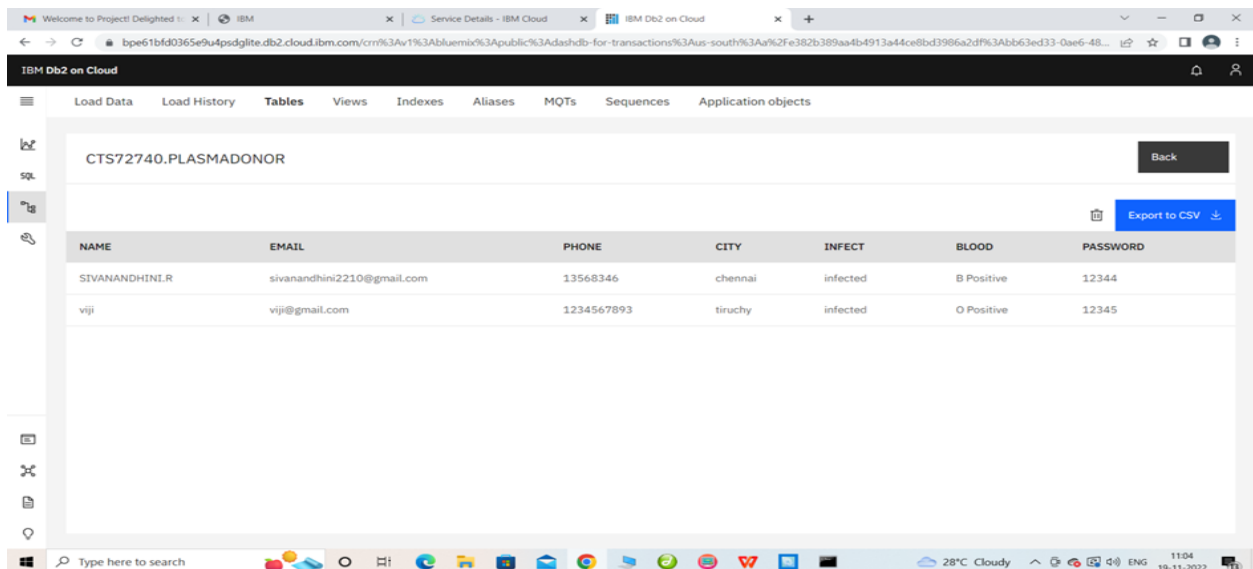
return render_template('request.html', pred="Your request is sent to the concerned
people.")

if __name__ == "__main__":

    app.run(host='0.0.0.0', port=8080)

```

Output :



The screenshot shows the IBM Db2 on Cloud web interface. The browser address bar displays a long URL. The interface includes a navigation menu with options like 'Load Data', 'Load History', 'Tables', 'Views', 'Indexes', 'Aliases', 'MQTs', 'Sequences', and 'Application objects'. The main content area shows a table with the title 'CTS72740.PLASMADONOR'. The table has columns: NAME, EMAIL, PHONE, CITY, INFECT, BLOOD, and PASSWORD. There are two rows of data. An 'Export to CSV' button is visible in the top right corner of the table area.

NAME	EMAIL	PHONE	CITY	INFECT	BLOOD	PASSWORD
SIVANANDHINI.R	sivanandhini2210@gmail.com	13568346	chennai	infected	B Positive	12344
viji	viji@gmail.com	1234567893	tiruchy	infected	O Positive	12345

7.2.2 Integrating sendgrid with python code

```

Senderemail.py
import smtplib
import sendgrid
import os
from sendgrid.helpers.mail import Mail, Email, To, Content
SUBJECT = "Interview Call"

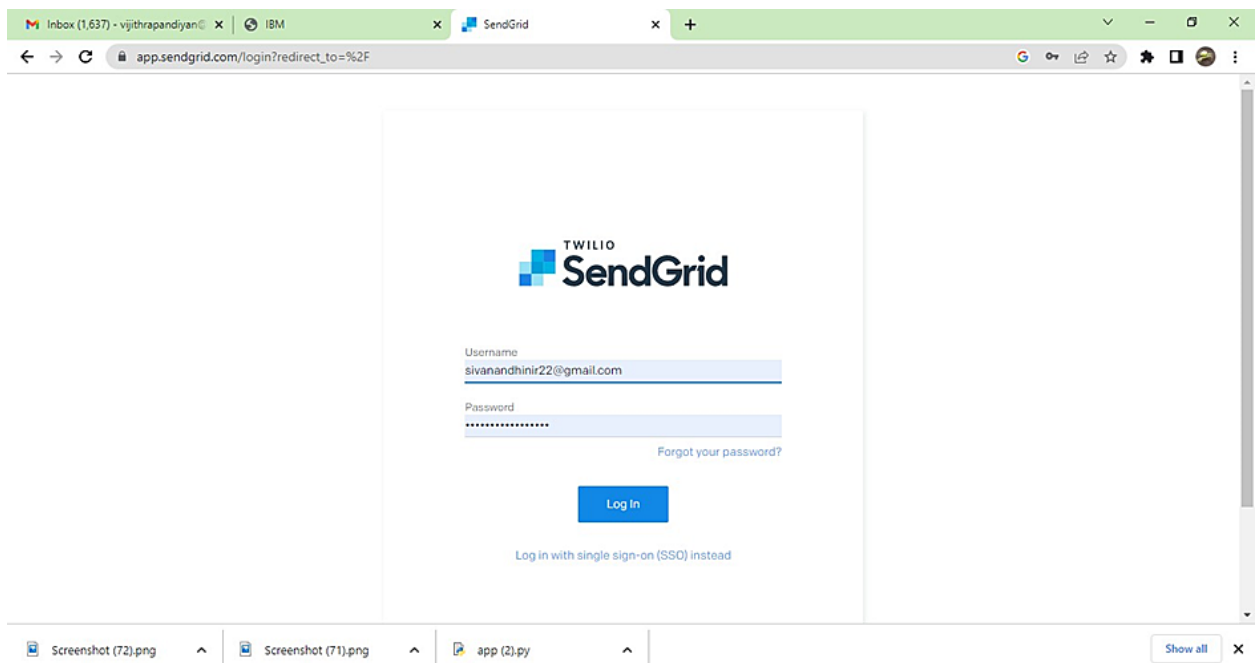
```

```

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.Jak_K6_OQRCHDWK9Cxxv64Q.8aziFLnMRH_
9P K5INqOcP7ylJmezx_qSKfB1iCeGL4o')
    from_email = Email("sivanandhini2210@gmail.com") # Change to your verified
sender
    to_email = To("vijithrap7@gmail.com") # 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)

```



SendGrid

app.sendgrid.com/guide

Review. Your account is under review and we'd like to know a little more about you and how you intend to use Twilio SendGrid. [Contact Us](#)

Sivanandhini R

Dashboard

Email API

Marketing

Design Library

Stats

Activity

Suppressions

Settings

Welcome, Sivanandhini !

To get up and running quickly, please follow the steps outlined below.

Send your first emails with Twilio SendGrid

Create a sender identity

Before sending email, you'll need to create a sender identity. There are two ways to do this, but we recommend creating a Single Sender to get set up quickly and test your email integration.

Create a Single Sender

Authenticate a domain instead

Your sender identity is the "from" email address your recipients will see in their inbox. [Learn more about this.](#)

Your Business

<jane.doe@your_business.com>

SendGrid

sendgrid - vijithrapandian@gmail.com

app.sendgrid.com/settings/sender_auth/senders

Sivanandhini R

Dashboard

Email API

Marketing

Design Library

Stats

Activity

Suppressions

Settings

Account Details

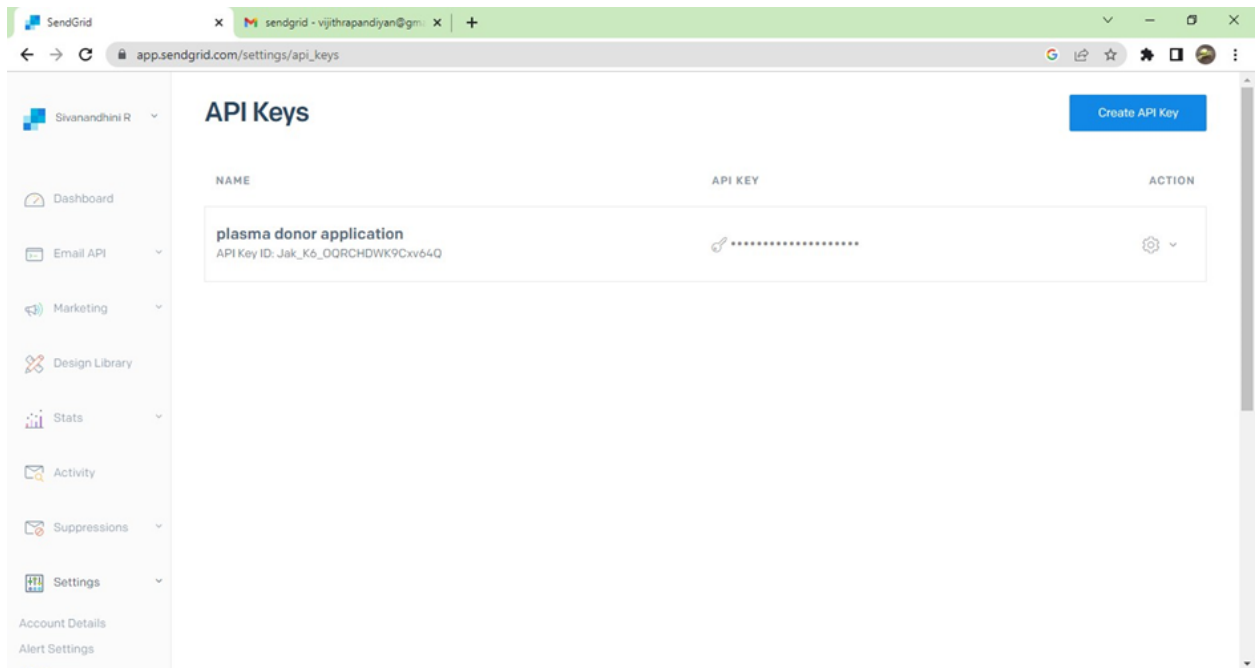
Alert Settings

Sender Authentication /

Single Sender Verification

Create New Sender

SENDERS	ADDRESS	NICKNAME	VERIFIED	ACTIONS
<div>sivanandhini</div> <div>FROM sivanandhini2210@gmail.com</div> <div>REPLY vijithrap7@gmail.com</div>	aamec kovilveni chennai, IND	sivanandhini	×	⋮



7.2.3 Deploy in kubernetes cluster

```
Command Prompt
LogIn Succeeded

C:\Users\sivan>cd C:\Users\sivan\Downloads\POA

C:\Users\sivan\Downloads\POA>kubectl apply -f deployment.yaml
deployment.apps/flask-node-deployment created

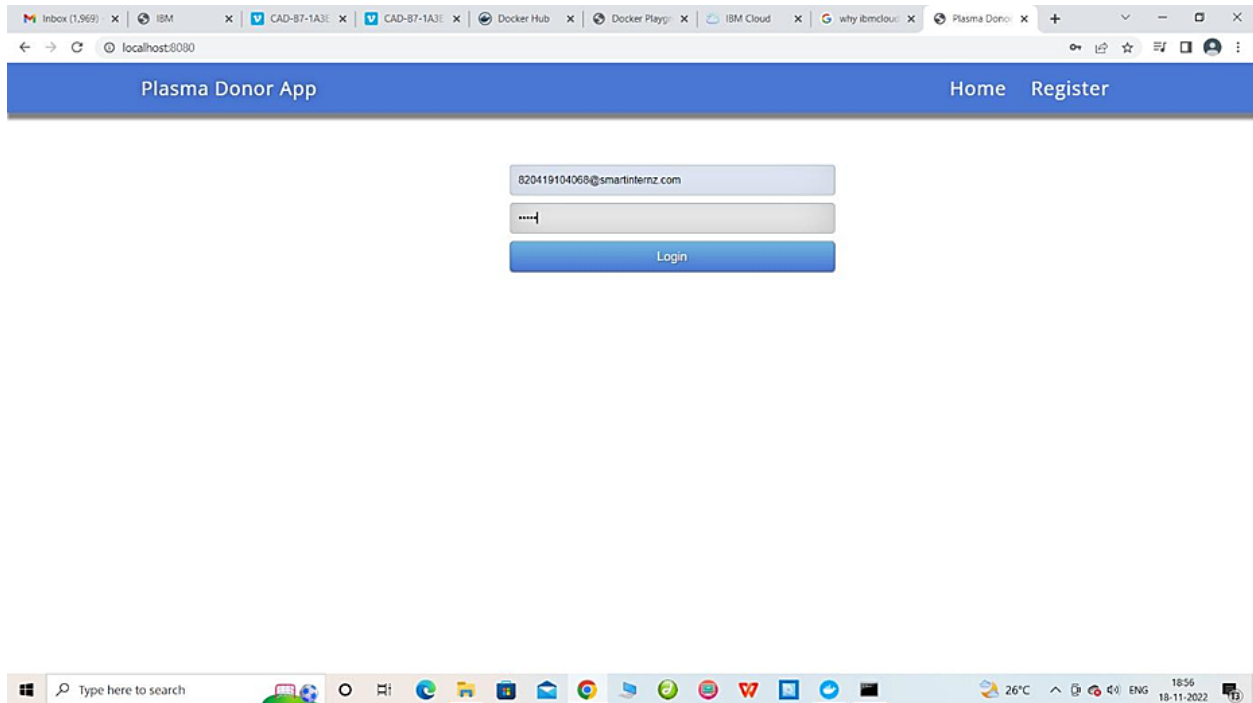
C:\Users\sivan\Downloads\POA>kubectl apply -f service.yaml
service/flask-node-deployment created

C:\Users\sivan\Downloads\POA>kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
flask-node-deployment-7f98f84fd-dcn7 0/1     ErrImagePull  0           74s

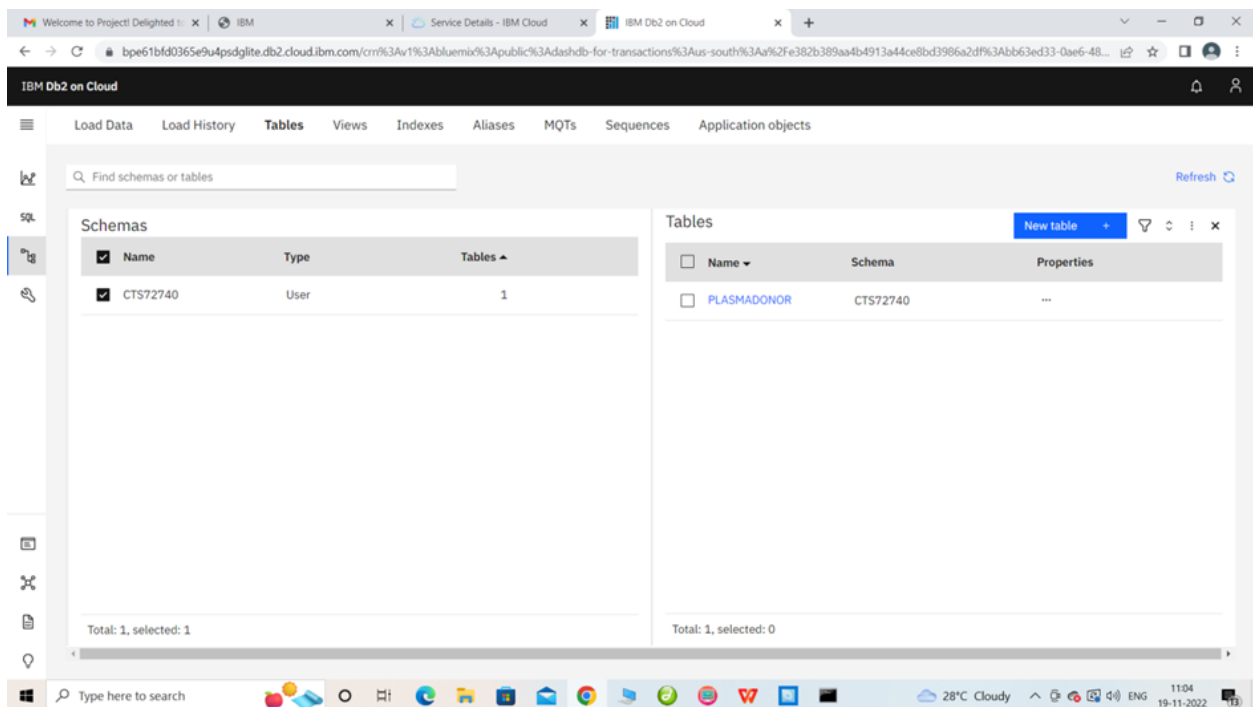
C:\Users\sivan\Downloads\POA>ibmcloud plugin install kubernetes-service
Looking up 'kubernetes-service' from repository 'IBM Cloud'...
Plug-in 'container-service[kubernetes-service/ks] 1.0.459' found in repository 'IBM Cloud'
Attempting to download the binary file...
26.86 MiB / 26.86 MiB [=====] 100.00% 37s
28168192 bytes downloaded
Installing binary...
Plug-in 'container-service 1.0.459' was successfully installed into C:\Users\sivan\bluemix\plugins\container-service. Use 'ibmcloud plugin show container-service' to show its details.

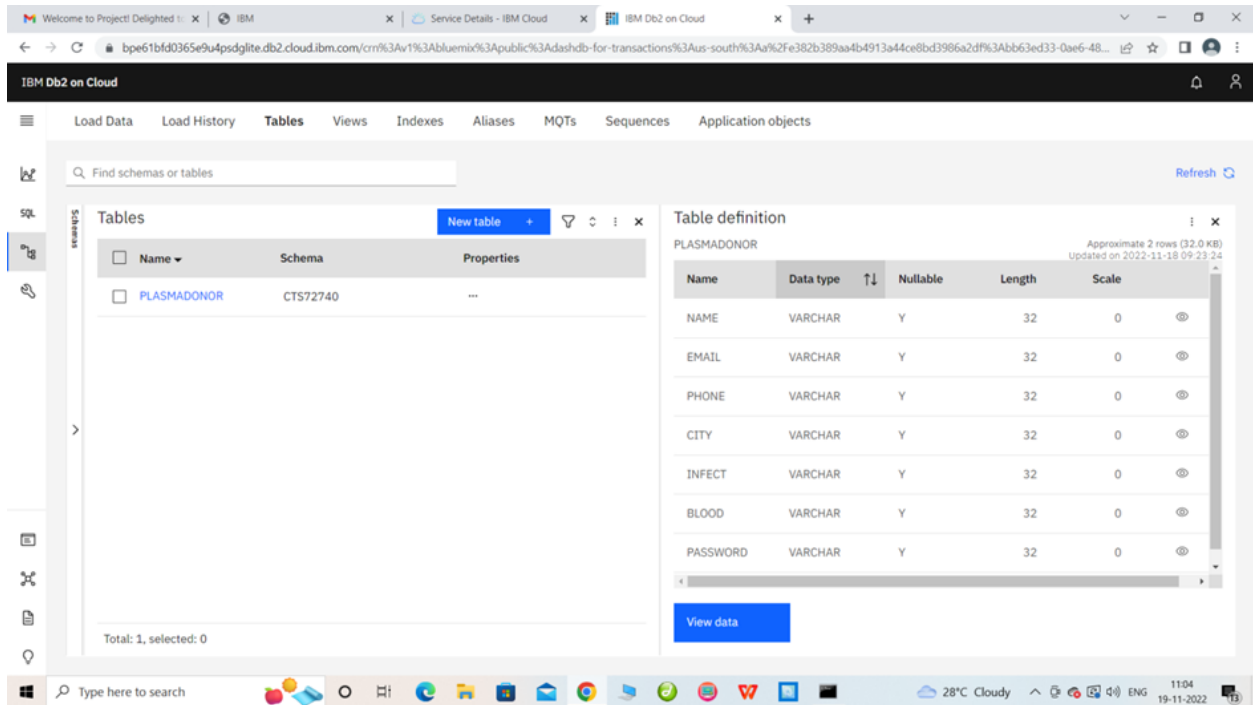
C:\Users\sivan\Downloads\POA>kubectl get service
NAME                                TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
flask-node-deployment              ClusterIP   10.102.34.236 <none>        5000/TCP    3m47s
kubernetes                         ClusterIP   10.96.0.1    <none>        443/TCP    13m

C:\Users\sivan\Downloads\POA>
```



7.3 Database Schema (if Applicable)

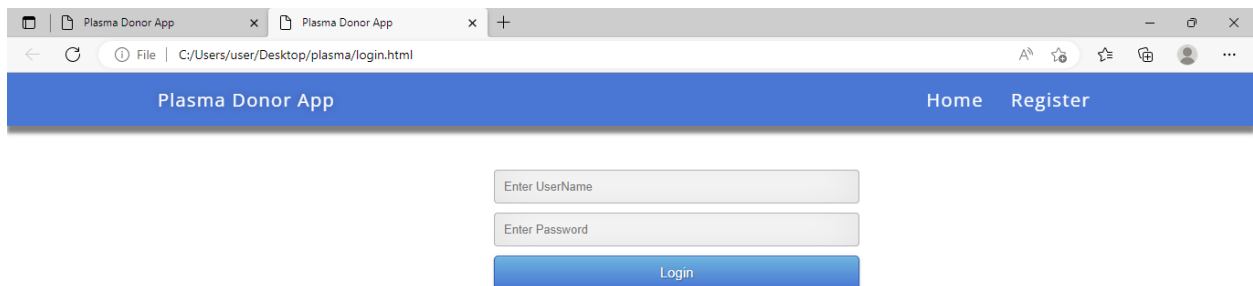




8. TESTING

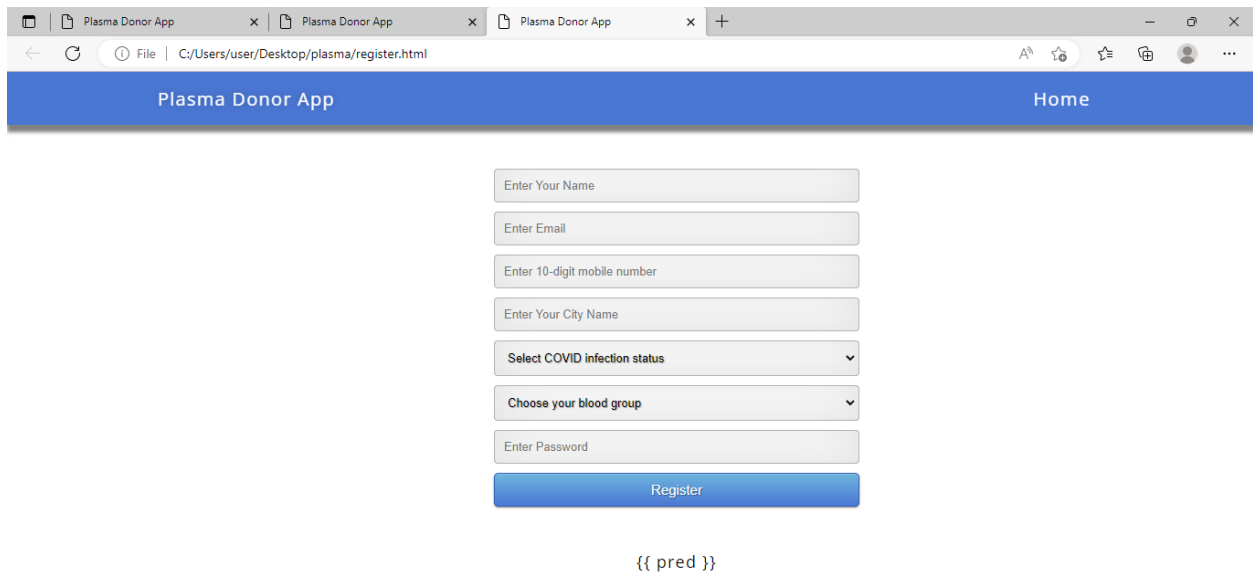
8.1 Test Cases:

8.1.1 Login page



{{ pred }}

8.1.2 Registration Page



The screenshot shows a web browser with three tabs, all titled "Plasma Donor App". The address bar shows the file path "C:/Users/user/Desktop/plasma/register.html". The page has a blue header with "Plasma Donor App" on the left and "Home" on the right. The main content area contains a registration form with the following fields: "Enter Your Name", "Enter Email", "Enter 10-digit mobile number", "Enter Your City Name", "Select COVID infection status" (a dropdown menu), "Choose your blood group" (a dropdown menu), and "Enter Password". A blue "Register" button is at the bottom of the form.

Enter Your Name

Enter Email

Enter 10-digit mobile number

Enter Your City Name

Select COVID infection status

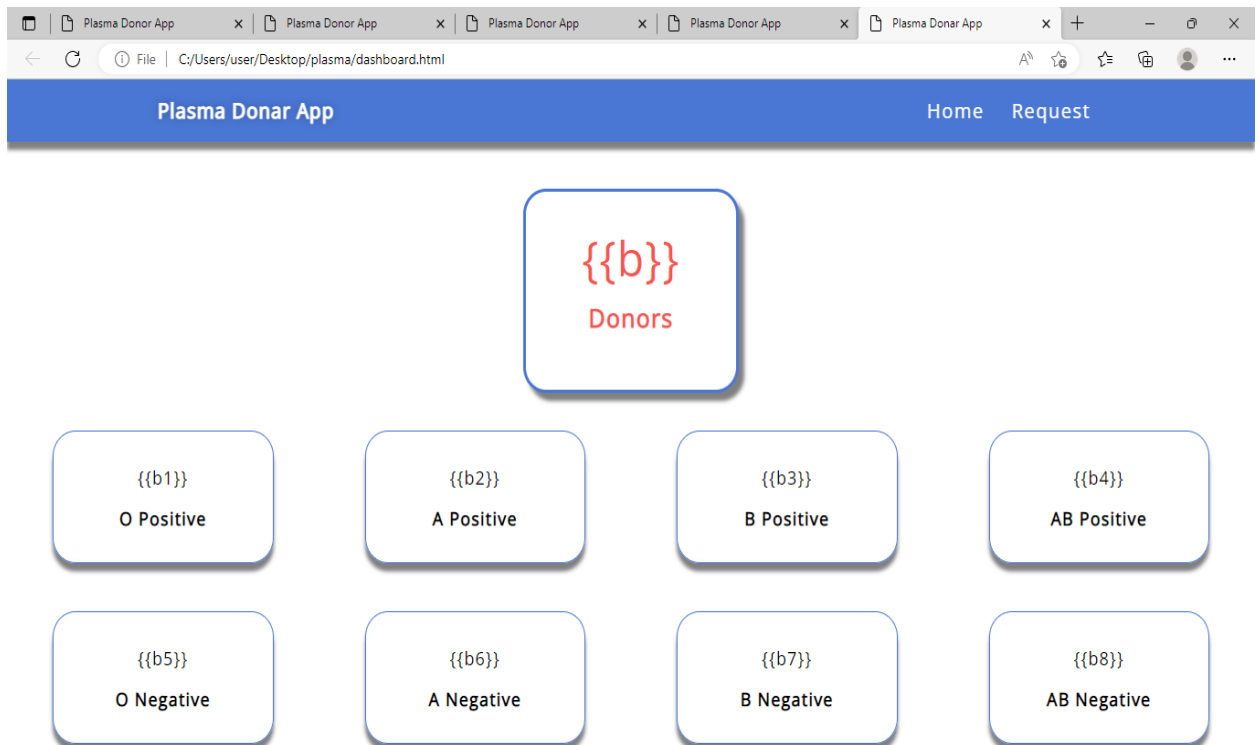
Choose your blood group

Enter Password

Register

{{ pred }}

8.1.3 Dashboard Page



The screenshot shows a web browser with five tabs, all titled "Plasma Donor App". The address bar shows the file path "C:/Users/user/Desktop/plasma/dashboard.html". The page has a blue header with "Plasma Donor App" on the left and "Home" and "Request" on the right. The main content area features a central box labeled "Donors" with a red placeholder "b". Below this are eight boxes arranged in two rows of four, each representing a blood type: "O Positive", "A Positive", "B Positive", "AB Positive" in the top row, and "O Negative", "A Negative", "B Negative", "AB Negative" in the bottom row. Each box contains a placeholder "b" above the blood type name.

Donors

O Positive

A Positive

B Positive

AB Positive

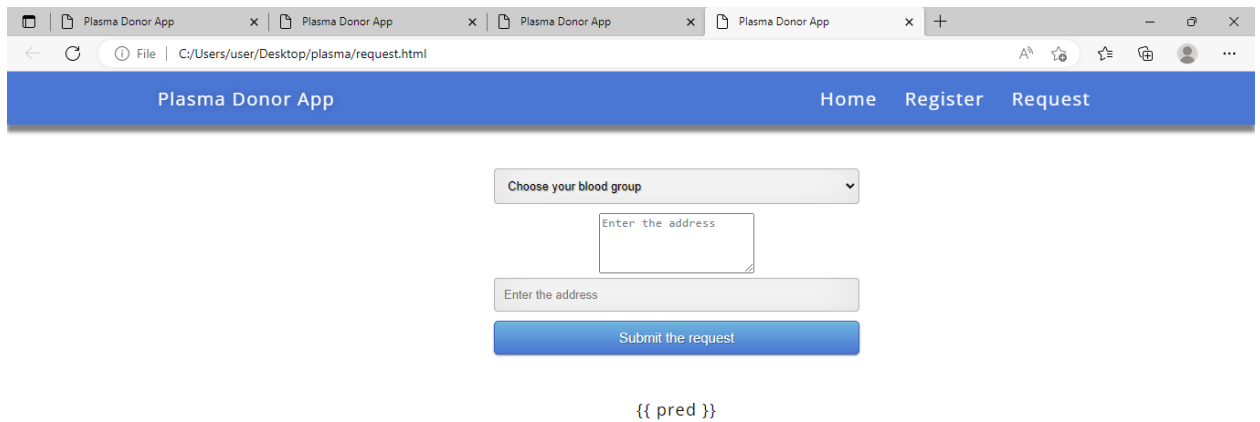
O Negative

A Negative

B Negative

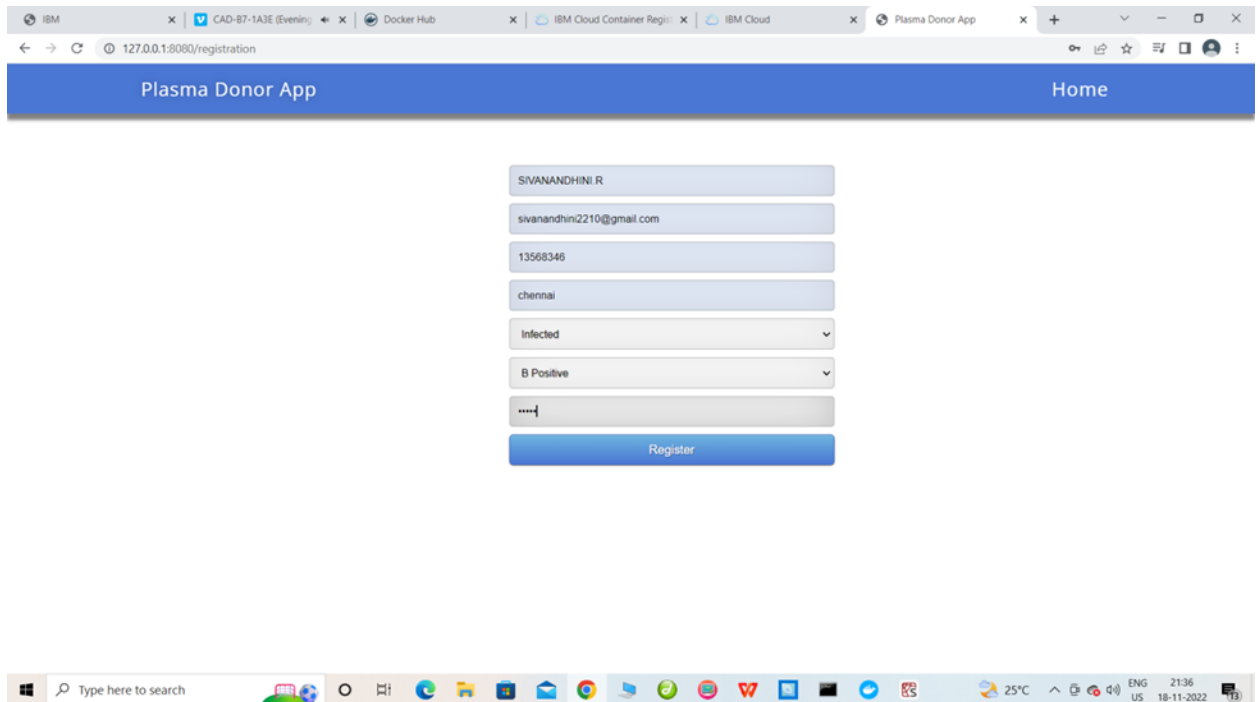
AB Negative

8.1.4 Request Page



The screenshot shows a web browser with four tabs, all titled "Plasma Donor App". The address bar shows the file path "C:/Users/user/Desktop/plasma/request.html". The page has a blue header with the text "Plasma Donor App" on the left and navigation links "Home", "Register", and "Request" on the right. The main content area contains a form with the following elements: a dropdown menu labeled "Choose your blood group", a text input field labeled "Enter the address", another text input field labeled "Enter the address", and a blue button labeled "Submit the request". Below the form, there is a placeholder text "{{ pred }}".

8.1.5 Database Connection Page



The screenshot shows a web browser with multiple tabs, including "IBM", "CAD-87-1A3E (Evenin)", "Docker Hub", "IBM Cloud Container Regi", "IBM Cloud", and "Plasma Donor App". The address bar shows the URL "127.0.0.1:8080/registration". The page has a blue header with the text "Plasma Donor App" on the left and a "Home" link on the right. The main content area contains a registration form with the following elements: text input fields for "SIVANANDHINI R", "sivanandhin2210@gmail.com", "13568346", and "chennai"; dropdown menus for "Infected" and "B Positive"; a password input field with masked characters "****"; and a blue button labeled "Register". At the bottom of the screen, there is a Windows taskbar with a search bar, several application icons, and system tray information including "25°C", "ENG US", and the date "21:36 18-11-2022".

Search results - sivanandhini2211 x IBM x Plasma Donor App x Service Details - IBM Cloud x IBM Db2 on Cloud

bpe61bfd0365e9u4psdglite.db2.cloud.ibm.com/crn%3Av1%3Abluemix%3Apublic%3Adashdb-for-transactions%3Aus-south%3Aa%2Fe382b389aa4b4913a44ce8bd3986a2df%3Aabb63ed33-0ae6-48...

IBM Db2 on Cloud

Load Data Load History **Tables** Views Indexes Aliases MQTs Sequences Application objects

CTS72740.PLASMADONOR Back

Export to CSV

NAME	EMAIL	PHONE	CITY	INFECT	BLOOD	PASSWORD
SIVANANDHINI.R	sivanandhini2210@gmail.com	13568346	chennai	infected	B Positive	12344
viji	viji@gmail.com	1234567893	tiruchy	infected	O Positive	12345

Type here to search 31°C 13:01 18-11-2022

8.2 User Acceptance Testing:

8.2.1 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	2	4	2	3	11
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	4	2	4	4	14
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	3	2	3	8
Totals	9	12	13	12	46

8.2.2 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	2	2	3
Client Application	51	10	5	36
Security	2	1	1	0
Outsource Shipping	3	2	0	1
Exception Reporting	9	3	3	3
Final Report Output	4	2	1	1
Version Control	2	1	0	1

8.2.3 Testcases Report

				Date	3-Nov-22								
				Team ID	PNT2022TMD32947								
				Project Name	Project - Plasma Donor Application								
				Maximum Marks	4 marks								
Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
LoginPage_TC_004	Functional	Register page	Verify user is able to log into application with invalid credentials	Laptop,IBM cloud account,IBM DB2	1.Enter URL(https://shopenzer.com/) and click go 2.Click on My Account dropdown button 3.Enter invalid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	name:user1 mail:user1@gmail.com phone:8976654645 city:chennai covid infection:infected blood:A+ password:55555	Application should show if the user has successfully registered or not.	Working as expected	Pass	Steps are clear to follow			
LoginPage_TC_004	Functional	Register page	Verify user is able to log into application with invalid credentials	Laptop,IBM cloud account,IBM DB2	1.Enter URL(https://shopenzer.com/) and click go 2.Click on My Account dropdown button 3.Enter Valid username/email in Email text box 4.Enter Invalid password in password text box 5.Click on login button	name:user2 mail:user2@gmail.com phone:8976654647 city:chennai covid infection:infected blood:A+ password:55555	Application should show incorrect details.	Working as expected	Pass	Steps are clear to follow			
LoginPage_TC_005	Functional	Register page	Verify user is able to log into application with invalid credentials	Laptop,IBM cloud account,IBM DB2	1.Enter URL(https://shopenzer.com/) and click go 2.Click on My Account dropdown button 3.Enter invalid username/email in Email text box 4.Enter Invalid password in password text box 5.Click on login button	name:user3 mail:user3@gmail.com phone:867946234 city:trichy covid infection: not infected blood:O+ password:34567	Application should show if the user already registered or not.	Working as expected	Pass	Steps are clear to follow			

9. RESULTS

9.1 Performance Metrics

NFT - Risk Assessment									
S.No	Project Name	Scope/feature	Functional Changes	Hardware Changes	Software Changes	Impact of Downtime	Load/Volumem Changes	Risk Score	Justification
1	Plasma Donor appli	New	Low	No Changes	Low	nil	>5 to 10%	RED	As we have seen the chnages
NFT - Detailed Test Plan									
S.No	Project Overview		NFT Test approach	umptions/Dependencies/R			Approvals/SignOff		
1	Plasma Donor Application								
End Of Test Report									
S.No	Project Overview	NFT Test approach	NFR - Met	Test Outcome	GO/NO-GO decision	Recommendations	Identified Defects (Detected/Closed/Open)	Approvals/SignOff	

10. ADVANTAGES & DISADVANTAGES

10.1 Advantages

- It is a user-friendly application.
- It will help people to find plasma easily

10.2 Disadvantages

- It cannot auto verify user genuineness.
- It requires an active internet connection.

11. CONCLUSION

Plasma donor application provides a reliable platform to connect local blood donors with patients. BLOODR creates a communication channel through authenticated clinics whenever a patient needs blood donation. It is a useful tool to find compatible blood donors who can receive blood request posts in their local area. Clinics can use this web application to maintain the blood donation activity. Future improvement of the BLOODR is explained.

12. FUTURE SCOPE

The discovery phase, can often times be chaotic. Besides appreciating that this is part of the process and to be okay with it, we recognised in retrospect that there were uncharted paths which we did not explore, but should have. As students, we have the goal to complete the project deliverables on time. Amidst the chaos, we found ourselves having conversations such as which path do we want to hunt down, and why. Eventually, we sat down as a team and collectively decided to move on after devoting an agreed set amount of time to the discovery phase.

In the ideal world, with more time, we should tease out these other paths. It is worthwhile hunting down all our options further, testing a variety of hypothesis, before moving on to the design process.

13. APPENDIX

1.INTRODUCTION

1.1 Project Overview

1.2 Purpose

2. LITERATURE SURVEY

2.1 Existing problem

2.2 References

2.3 Problem Statement Definition

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

3.3 Proposed Solution

3.4 Problem Solution fit

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

4.2 Non-Functional requirements

5. PROJECT DESIGN

5.1 Data Flow Diagrams

5.2 Solution & Technical Architecture

5.3 User Stories

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

6.3 Reports from JIRA

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1

7.1.1 Login.html

7.1.2 Register.html

7.1.3 Dashboard.html

7.1.4 Request.html

7.1.5 Style sheet

7.2 Feature 2

7.2.1 App.py

7.2.2 Intergrating sendgrid with python code

7.2.3 Deploy in kubernetes cluster

7.3 Database Schema (if Applicable)

8. TESTING

8.1 Test Cases

8.1.1 Login Page

8.1.2 Registration Page

8.1.3 Dashboard Page

8.1.4 Request Page

8.1.5 Database Connection Page

8.2 User Acceptance Testing

8.2.1 Defect Analysis

8.2.2 Test Case Analysis

8.2.3 TestCases Report

9. RESULTS

9.1 Performance Metrics

10. ADVANTAGES & DISADVANTAGES

10.1 Advantages

10.2 Disadvantages

11. CONCLUSION

12. FUTURE SCOPE

13. APPENDIX

Source Code

GitHub & Project Demo Link

Source Code

Login.html

```
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
<meta charset="UTF-8">
<title>Plasma Donor App</title>
<link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet'
type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet'
type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet'
type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300'
rel='stylesheet' type='text/css'>
<link rel="stylesheet" href="style.css">
<style>
.login{
top: 20%;
}
```

```

</style>
</head>
<body>
<div class="header">
<div>Plasma Donor App</div>
<ul>
<li><a href="/registration">Register</a></li>
<li><a class="active" href="/login">Home</a></li>
</ul>
</div>
<div class="login" >
<div>
</div><!-- Main Input For Receiving Query to our ML -->
<form action="{{ url_for('loginpage')}}"method="post">
<input type="text" name="user" placeholder="Enter UserName"
required="required" style="color:black" />
<input type="password" name="passw" placeholder="Enter Password"
required="required" style="color:black" />
<button type="submit" class="btn btn-primary btn-block btn-
large">Login</button>
</form>
<br><br>
<div style="color:black">
{{ pred }}</div>
</div>
</body>
</html>

```

Register.html

```

<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
<meta charset="UTF-8">
<title>Plasma Donor App</title>
<link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet'
type='text/css'>

```



```

        <link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet'
type='text/css'>
        <link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet'
type='text/css'>
        <link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300'
rel='stylesheet' type='text/css'>
        <link rel="stylesheet" href="style.css">
        <style>
        .login{
        top: 20%;
        }
        </style>
        </head>
        <body>
        <div class="header">
        <div>Plasma Donor App</div>
        <ul>
        <li><a class="active" href="/login">Home</a></li>
        </ul>
        </div>
        <div class="login">
        <!-- Main Input For Receiving Query to our ML -->
        <form action="{ { url_for('register') } }"method="post">
        <input type="text" name="name" placeholder="Enter Your Name"
required="required" style="color:black"/>
        <input type="email" name="email" placeholder="Enter Email"
required="required" style="color:black"/>
        <input type="text" name="phone" placeholder="Enter 10-digit mobile number"
required="required" style="color:black"/>
        <input type="city" name="city" placeholder="Enter Your City Name"
required="required" style="color:black"/>
        <select name="infect">
        <option value="select" selected>Select COVID infection status</option>
        <option value="infected">Infected</option>
        <option value="uninfected">Uninfected</option>
        </select>
        <select name="blood">
        <option value="select" selected>Choose your blood group</option>
        <option value="O Positive">O Positive</option>
        <option value="A Positive">A Positive</option>
        <option value="B Positive">B Positive</option>

```

```

        <option value="AB Positive">AB Positive</option>
        <option value="O Negative">O Negative</option>
        <option value="A Negative">A Negative</option>
        <option value="B Negative">B Negative</option>
        <option value="AB Negative">AB Negative</option>
    </select>
    <input type="password" name="passw" placeholder="Enter Password"
required="required" style="color:black"/>
    <button type="submit" class="btn btn-primary btn-block btn-
large">Register</button>
</form>
<br><br>
<div style="color:black">
{{ pred }}</div>
</div>
</body>
</html>

```

Dashboard.html

```

<!DOCTYPE html>
<html lang="en">
<head>
<title>Plasma Donar App</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<linkrel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></scrip
t>
<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
<link rel="stylesheet" href="style.css">
</head>
<style>
.big{
top:70;

```

```

background-color:white;
margin-top:80px;
margin-left:550px;
margin-right:550px;
height:200px;
border-radius: 25px;
border: 3px solid #4a77d4;
box-shadow: 6px 8px 4px grey;
text-align:center;
}
.row{
height:150px;
}
.col{
margin:10px;
margin-left:50px;
margin-right:50px;
border-radius: 25px;<!DOCTYPE html>
<html lang="en">
<head>
<title>Plasma Donar App</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<linkrel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></scrip
t>
<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
<link rel="stylesheet" href="style.css">
</head>
<style>
.big{
top:70;

```

```

background-color:white;
margin-top:80px;
margin-left:550px;
margin-right:550px;height:200px;
border-radius: 25px;
border: 3px solid #4a77d4;
box-shadow: 6px 8px 4px grey;
text-align:center;
}
.row{
height:150px;
}
.col{
margin:10px;
margin-left:50px;
margin-right:50px;
border-radius: 25px;
<div class="ext1"><font size="20px">{ {b}}</font><br><b>Donors</b></div>
</div>
<br>
<div class="row">
<div class="col" >
<div class="ext">{ {b1}}<br><b>O Positive</b></div>
</div>
<div class="col" >
<div class="ext">{ {b2}}<br><b>A Positive</b></div>
</div>
<div class="col" >
<div class="ext">{ {b3}}<br><b>B Positive</b></div>
</div>
<div class="col" >
<div class="ext">{ {b4}}<br><b>AB Positive</b></div>
</div>
</div> <br>
<div class="row">
<div class="col" >

```

```

<div class="ext">{{b5}}<br><b>O Negative</b></div>
</div>
<div class="col" >
<div class="ext">{{b6}}<br><b>A Negative</b></div>
</div>
<div class="col" >
<div class="ext">{{b7}}<br><b>B Negative</b></div>
</div>
<div class="col" >
<div class="ext">{{b8}}<br><b>AB Negative</b></div>
</div>
</div>
</div>
</body>
</html>

```

Request.html

```

<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
<meta charset="UTF-8">
<title>Plasma Donor App</title>
<link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet'
type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet'
type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet'
type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300'
rel='stylesheet' type='text/css'>
<link rel="stylesheet" href="style.css">
<style>
.login{
top: 20%;
}
</style>

```

```

</head>
<body>
<div class="header">
<div>Plasma Donor App</div>
<ul>
<li><a class="active" href="/login">Home</a></li>
</ul>
</div>
<div class="login">

<!-- Main Input For Receiving Query to our ML -->
<form action="{{ url_for('register')}}"method="post">
    <input type="text" name="name" placeholder="Enter Your Name"
required="required" style="color:black"/>
    <input type="email" name="email" placeholder="Enter Email"
required="required" style="color:black"/>
    <input type="text" name="phone" placeholder="Enter 10-digit mobile number"
required="required" style="color:black"/>
    <input type="text" name="city" placeholder="Enter Your City Name"
required="required" style="color:black"/>
    <select name="infect">
    <option value="select" selected>Select COVID infection status</option>
    <option value="infected">Infected</option>
    <option value="uninfected">Uninfected</option>
    </select>
    <select name="blood">
    <option value="select" selected>Choose your blood group</option>
    <option value="O Positive">O Positive</option>
    <option value="A Positive">A Positive</option>
    <option value="B Positive">B Positive</option>
    <option value="AB Positive">AB Positive</option>
    <option value="O Negative">O Negative</option>
    <option value="A Negative">A Negative</option>
    <option value="B Negative">B Negative</option>
    <option value="AB Negative">AB Negative</option>
    </select>
    <input type="password" name="passw" placeholder="Enter Password"

```

```

required="required" style="color:black"/>
    <button type="submit" class="btn btn-primary btn-block btn-
large">Register</button>
</form>
<br><br>
<div style="color:black">
{{ pred }}</div>
</div>
</body>
</html>

```

Style sheet

```

@import url(https://fonts.googleapis.com/css?family=Open+Sans);
.btn {
display: inline-block;
*display: inline;
*zoom: 1; padding:
4px 10px 4px;
margin-bottom: 0;
font-size: 13px;
line-height: 18px;
color: #333333;
text-align: center;
text-shadow: 0 1px 1px rgba(255, 255, 255, 0.75);
vertical-align: middle;
background-color: #f5f5f5;
background-image: -moz-linear-gradient(top, #ffffff, #e6e6e6);
background-image: -ms-linear-gradient(top, #ffffff, #e6e6e6);
background-image: -webkit-gradient(linear, 0 0, 0 100%, from(#ffffff),
to(#e6e6e6));
background-image: -webkit-linear-gradient(top, #ffffff, #e6e6e6);
background-image: -o-linear-gradient(top, #ffffff, #e6e6e6);
background-image: linear-gradient(top, #ffffff, #e6e6e6);
background-repeat: repeat-x;
filter: progid:dximagetransform.microsoft.gradient(startColorstr=#ffffff,
endColorstr=#e6e6e6, GradientType=0);
border-color: #e6e6e6 #e6e6e6 #e6e6e6;
border-color: rgba(0, 0, 0, 0.1) rgba(0, 0, 0, 0.1) rgba(0, 0, 0, 0.25);
border: 1px solid #e6e6e6;

```

```

    -webkit-border-radius: 4px;
    -moz-border-radius: 4px;
    border-radius: 4px;
    -webkit-box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05);
    -moz-box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05);
    box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05);
    cursor: pointer; *margin-left: .3em;
}
.btn:hover, .btn:active, .btn.active, .btn.disabled, .btn[disabled] { background-color: #e6e6e6; }
.btn-large {
    padding: 9px 14px;
    font-size: 15px;
    line-height: normal;
    -webkit-border-radius: 5px;
    -moz-border-radius: 5px;
    border-radius: 5px;
}
.btn:hover {
    color: #333333;
    text-decoration: none;
    background-color: #e6e6e6;
    background-position: 0 -15px;
    -webkit-transition: background-position 0.1s linear;
    -moz-transition: background-position 0.1s linear;
    -ms-transition: background-position 0.1s linear;
    -o-transition: background-position 0.1s linear;
    transition: background-position 0.1s linear;
}
.btn-primary, .btn-primary:hover {
    text-shadow: 0 -1px 0 rgba(0, 0, 0, 0.25);
    color: #ffffff;
}
.btn-primary.active { color: rgba(255, 255, 255, 0.75); }
.btn-primary {
    background-color: #4a77d4;
    background-image: -moz-linear-gradient(top, #6eb6de, #4a77d4);
    background-image: -ms-linear-gradient(top, #6eb6de, #4a77d4);
    background-image: -webkit-gradient(linear, 0 0, 0 100%, from(#6eb6de),

```



```

to(#4a77d4));
    background-image: -webkit-linear-gradient(top, #6eb6de, #4a77d4);
    background-image: -o-linear-gradient(top, #6eb6de, #4a77d4);
    background-image: linear-gradient(top, #6eb6de, #4a77d4);
    background-repeat: repeat-x;
    filter: progid:dximagetransform.microsoft.gradient(startColorstr=#6eb6de,
endColorstr=#4a77d4, GradientType=0);
    border: 1px solid #3762bc;
    text-shadow: 1px 1px 1px rgba(0,0,0,0.4);
    box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0,
0, 0.5);
    }
.btn-primary:hover,      .btn-primary:active,      .btn-primary.active,      .btn-
primary.disabled, .btn-primary[disabled] {
    filter: none;
    background-color: #4a77d4;
    }

.btn-block { width: 100%; display:block; }
* { -webkit-box-sizing:border-box; -moz-box-sizing:border-box; -ms-box-
sizing:border-box; -o-box-sizing:border-box; box-sizing:border-box; }
html { width: 100%; height:100%; overflow:hidden; }
body {
    width: 100%;
    height:100%;
    font-family: 'Open Sans', sans-serif;
    background: #ffffff;
    color: #000000;

    font-size: 18px;
    text-align:center;
    letter-spacing:1.2px;
}
.header {
    top:0;
    margin:0px;
    left: 0px;
    right: 0px;
    position: fixed;
    background: #4a77d4;
    color: white;

```

```

        box-shadow: 0px 8px 4px grey;
        overflow: hidden;
        padding: 15px;
        font-size: 1.5vw;
        width: 100%;
        text-align: center;
    }
    .login {
        position: absolute;
        top: 70%;
        left: 50%;
        margin: -25px 0 0 -150px;
        width: 400px;
        height: 400px;
    }

    .header div { color: #fff; text-shadow: 0 0 10px rgba(0,0,0,0.3); letter-
    spacing: 1px; text-align: center; float: left; padding-left: 150px;}
    ul {
        list-style-type: none;
        margin: 0;
        padding: 0;
        padding-right: 150px;
        overflow: hidden;
    }
    li {
        float: right;
    }

    li a {
        display: block;
        color: white;
        text-align: center;
        padding: 0px 15px;
        text-decoration: none;
    }
    input {
        width: 100%;
        margin-bottom: 10px;
        background: rgba(255,255,255,255);
        border: none;
    }

```

```

        outline: none;
        padding: 10px;
        font-size: 13px;
        color: black;
        text-shadow: black;
        border: 1px solid rgba(0,0,0,0.3);
        border-radius: 4px;
        box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px
        rgba(255,255,255,0.2);
        -webkit-transition: box-shadow .5s ease;
        -moz-transition: box-shadow .5s ease;
        -o-transition: box-shadow .5s ease;
        -ms-transition: box-shadow .5s ease;
        transition: box-shadow .5s ease;
    }
    input:focus { box-shadow: inset 0 -5px 45px rgba(100,100,100,0.4), 0 1px 1px
    rgba(255,255,255,0.2); }
    select {
        width: 100%;
        margin-bottom: 10px;
        background: rgba(255,255,255,255);
        border: none;
        outline: none;
        padding: 10px;
        font-size: 13px;
        color: #000000;
        text-shadow: 1px 1px 1px rgba(0,0,0,0.3);
        border: 1px solid rgba(0,0,0,0.3);
        border-radius: 4px;
        box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px
        rgba(255,255,255,0.2);
        -webkit-transition: box-shadow .5s ease;
        -moz-transition: box-shadow .5s ease;
        -o-transition: box-shadow .5s ease;
        -ms-transition: box-shadow .5s ease;
        transition: box-shadow .5s ease;
    }
}

```

App.py

```

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

import ibm_db

import json

import requests

app = Flask(__name__)

conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=fbd88901-ebdb-
4a4f-a32e-
9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32731;SECURI
TY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=cts72740;PWD=dORK
SzIzhwxBmMvg",",")

@app.route('/registration')

def home():

    return render_template('register.html')

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

def register():

    x = [x for x in request.form.values()]

    print(x)

    name=x[0]

    email=x[1]

    phone=x[2]

    city=x[3]

    infect=x[4]

    blood=x[5]

    password=x[6]

    sql = "SELECT * FROM plasmadonor WHERE email =?"

```

```

stmt = ibm_db.prepare(conn, sql)

ibm_db.bind_param(stmt,1,email)

ibm_db.execute(stmt)

account = ibm_db.fetch_assoc(stmt)

print(account)

if account:

    return render_template('register.html', pred="You are already a member, please
login using your details")

else:

    insert_sql = "INSERT INTO plasmadonor VALUES (?, ?, ?, ?, ?, ?, ?)"

    prep_stmt = ibm_db.prepare(conn, insert_sql)

    ibm_db.bind_param(prepare_stmt, 1, name)

    ibm_db.bind_param(prepare_stmt, 2, email)

    ibm_db.bind_param(prepare_stmt, 3, phone)

    ibm_db.bind_param(prepare_stmt, 4, city)

    ibm_db.bind_param(prepare_stmt, 5, infect)

    ibm_db.bind_param(prepare_stmt, 6, blood)

    ibm_db.bind_param(prepare_stmt, 7, password)

    ibm_db.execute(prepare_stmt)

    return render_template('register.html', pred="Registration Successful, please
login using your details")

@app.route('/')

@app.route('/login')

def login():

```

```

        return render_template('login.html')

@app.route('/loginpage',methods=['POST'])
def loginpage():

    user = request.form['user']

    passw = request.form['passw']

    sql = "SELECT * FROM plasmadonor WHERE email =? AND password=?"

    stmt = ibm_db.prepare(conn, sql)

    ibm_db.bind_param(stmt,1,user)

    ibm_db.bind_param(stmt,2,passw)

    ibm_db.execute(stmt)

    account = ibm_db.fetch_assoc(stmt)

    print (account)

    print(user,passw)

    if account:

        return redirect(url_for('stats'))

    else:

        return render_template('login.html', pred="Login unsuccessful. Incorrect
username / password !")

@app.route('/stats')
def stats():

    """sql = "SELECT blood FROM user group by blood"

    stmt = ibm_db.prepare(conn, sql)

    ibm_db.execute(stmt)

    count = ibm_db.fetch_assoc(stmt)

```

```

        print(count)'''

return render_template('stats.html',b=5,b1=2,b2=3,b3=4,b4=2,b5=1,b6=2,b7=1,b8=1)

@app.route('/requester')

def requester():

    return render_template('request.html')

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

def requested():

    bloodgrp = request.form['bloodgrp']

    address = request.form['address']

    print(address)

    sql = "SELECT * FROM plasmadonor WHERE blood=?"

    stmt = ibm_db.prepare(conn, sql)

    ibm_db.bind_param(stmt,1,bloodgrp)

    ibm_db.execute(stmt)

    data = ibm_db.fetch_assoc(stmt)

    msg = "Need Plasma of your blood group for: "+address

    while data != False:

        print      ("The      Phone      is      :      ",      data["PHONE"])
url="https://www.fast2sms.com/dev/bulk?authorization=xCXuwWTzyjOD2ARd1EngbH3a7tKI
q5PkIJ8YSf0Lh4FQZecs9iNI1dSvuqprxFwCKYJXA5amQkBE36Rl&sender_id=FSTSMS&me
ssage="+msg+"&language=english&route=p&numbers="+str(data["PHONE"])

        result=requests.request("GET",url)

        print(result)

        data = ibm_db.fetch_assoc(stmt)

    return render_template('request.html', pred="Your request is sent to the concerned

```

```
people.")
```

```
if __name__ == "__main__":
```

```
    app.run(host='0.0.0.0', port=8080)
```

Senderemail.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.Jak_K6_OQRCHDWK9C xv64Q.8aziFLnM  
RH_9P K5lNqOcP7ylJmez x_qSKfB1iCeGL4o')
```

```
    from_email = Email("sivanandhini2210@gmail.com") # Change to your verified  
sender
```

```
    to_email = To("vijithrap7@gmail.com") # 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)
```


GitHub & Project Demo Link

GitHub

<https://github.com/IBM-EPBL/IBM-Project-45911-1665990333.git>

Project Demo Link

<https://drive.google.com/file/d/1b5HEZNCOX2pDXYMtsuvQ8GRR1zN8pNv7/view?usp=sharing>