



PLASMA DONOR APPLICATION

NALAIYA THIRAN - PROJECT REPORT

PROJECT ID:PNT2022TMID00855

Submitted by

AUGUSTIN SHAM J [211419104032]

BABIN MON B [211419104033]

BADHRI KESAVA RAJA S M [211419104034]

GODSON RAJ R [211419104079]

In partial fulfillment for the award of the degree

Of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE, CHENNAI-600123.

(AN AUTONOMOUS INSTITUTION, AFFILIATED TO ANNA UNIVERSITY)

NOVEMBER 2022

PANIMALAR ENGINEERING COLLEGE, CHENNAI-600123.

(AN AUTONOMOUS INSTITUTION, AFFILIATED TO ANNA UNIVERSITY)

BONAFIDE CERTIFICATE

Certified that this project report

“PLASMA DONOR APPLICATION – PNT2022TMID00855”

is the bonafide work of

AUGUSTIN SHAM J (211419104032)

BABIN MON B (211419104033)

BADHRI KESAVA RAJA S M (211419104034)

GODSON RAJ R (211419104079)

who carried out the NALAIYA THIRAN project work under the supervision.

**Mrs. NAVYA
INDUSTRY MENTOR
IBM**

**Dr. N. PUGHAZENDHI
FACULTY MENTOR
Department of CSE
Panimalar Engineering College**

Table Of Contents

S.NO	TITLE	PAGE NO.
1.	INTRODUCTION	
	1.1 Project Overview	1
	1.2 Purpose	1
2.	LITERATURE SURVEY	
	2.1 Existing problem	2
	2.2 References	2
	2.3 Problem Statement Definition	3
3.	IDEATION & PROPOSED SOLUTION	
	3.1 Empathy Map Canvas	4
	3.2 Ideation & Brainstorming	5
	3.3 Proposed Solution	8
	3.4 Problem Solution fit	9
4.	REQUIREMENT ANALYSIS	
	4.1 Functional requirement	11
	4.2 Non-Functional requirements	12
5.	PROJECT DESIGN	
	5.1 Data Flow Diagrams	13
	5.2 Solution & Technical Architecture	14
	5.3 User Stories	15
6.	PROJECT PLANNING & SCHEDULING	
	6.1 Sprint Planning & Estimation	16
	6.2 Sprint Delivery Schedule	18
	6.3 Reports from JIRA	19

7.	CODING & SOLUTIONING	
	7.1 Feature 1	20
	7.2 Feature 2	20
	7.3 Database Schema	41
8.	TESTING	
	8.1 Test Cases	43
	8.2 User Acceptance Testing	46
9.	RESULTS	
	9.1 Performance Metrics	51
10.	ADVANTAGES & DISADVANTAGES	57
11.	CONCLUSION	58
12.	FUTURE SCOPE	59
13.	APPENDIX	
	13.1 Source Code	60
	13.2 GitHub & Project Demo Link	80

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 COVID-19 by donating plasma from patients who have recovered without approved antiretroviral therapy planning for a deadly COVID-19 infection, plasma therapy is an experimental approach to treat those COVID-positive patients and help them recover faster.

Therapy, which is considered reliable and safe. If a particular person has fully recovered from COVID-19, they are eligible to donate their plasma. As we all know, the traditional methods of finding plasma, one has to find out for oneself by looking at hospital records and contacting donors have been recovered, sometimes may not be available at home and move to other places. In this type of scenario, the health of those who are sick becomes disastrous. Therefore, it is not considered a rapid process to find plasma.

1.2. PURPOSE

During the COVID-19 crisis, the requirement of plasma became a high priority and the donor count has become low.

The Purpose of this Application is Saving the donor information and helping the needy by notifying the current donors list, would be a helping hand. In regard to the problem faced, This application is to be built which would take the donor details, store them and inform them upon a request.

2. LITERATURE SURVEY

2.1. EXISTING PROBLEM

- > Cannot Upload and Download the latest updates.
- > No use of Web Services and Remoting.
- > Risk of mismanagement and of data when the project is underdevelopment.
- > Less Security.
- > No proper coordination between different Applications and Users.
- > Fewer Users –Friendly

2.2. REFERENCE

1. Safe blood and blood products. Module 1: Safe blood donation. Geneva: World Health Organization; 2002. [17 August 2012]. http://www.who.int/bloodsafety/transfusion_services/bts_learning_materials/en/index.html.

2. Blood donor selection. Guidelines on assessing donor suitability for blood donation. Annex 3. Geneva: World Health Organization; 2012. [17 August 2012]. http://www.who.int/bloodsafety/voluntary_donation/blood_donor_selection_counselling/en/

3. WHO/IFRC. Towards 100% voluntary blood donation: A global framework for action. Geneva: World Health Organization; 2010. [17 August 2012]. <http://www.who.int/bloodsafety/publications/9789241599696/en/> [PubMed]

4. The Melbourne Declaration on 100% voluntary non-remunerated donation of blood and blood components. Geneva: World Health Organization; 2009. [17 August 2012]. http://www.who.int/worldblooddonorday/Melbourne_Declaration_VNRBD_2009.pdf.

5. WHO/CDC/IFRC. Blood donor counselling: Implementation guidelines. Geneva: World Health Organization; 2012. [17 August 2012]. http://www.who.int/bloodsafety/voluntary_donation

/blood_donor_selection_counselling/en/ [PubMed].

6.Screening donated blood for transfusion-transmissible infections. Geneva: World Health Organization; 2010. [17 August 2012]. http://www.who.int/bloodsafety/publications/bts_screendondbloodtransf/en/index.html. [PubMed].

2.3. PROBLEM STATEMENT DEFINITION

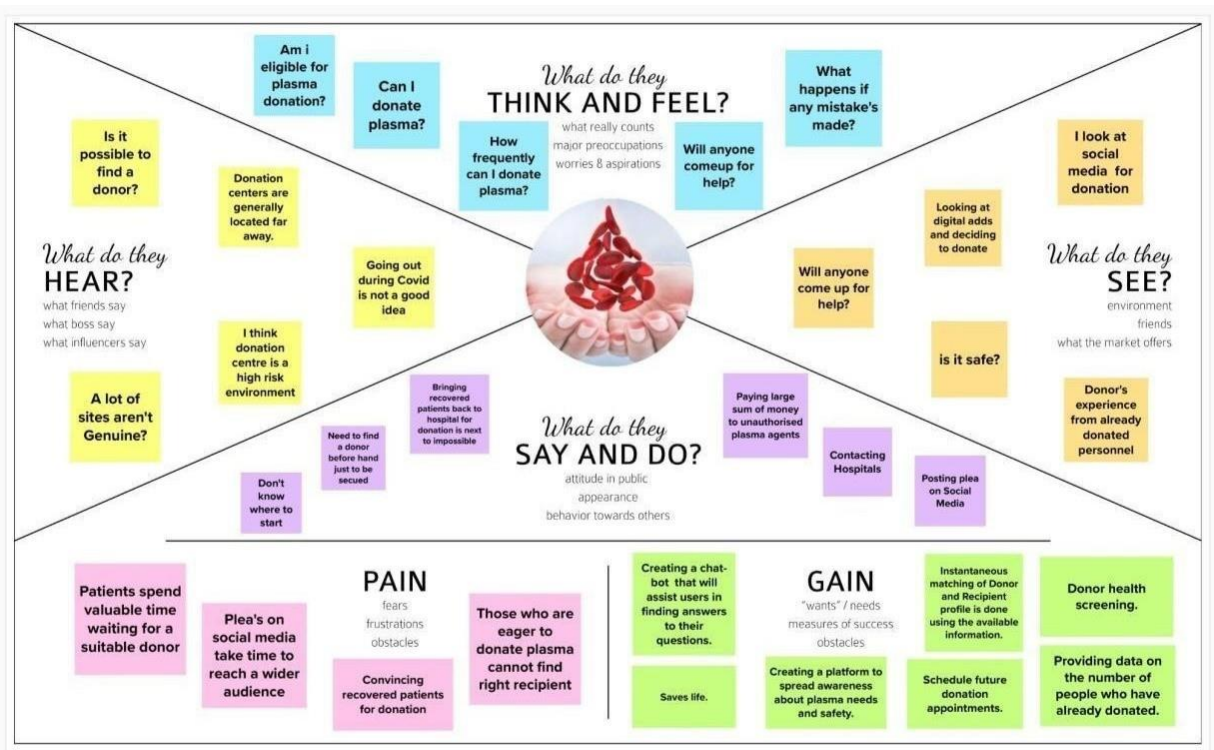
During the COVID 19 crisis, the requirement of plasma became a high priority and the donor count has become low. Saving the donor information and helping the needy by notifying the current donors list, would be a helping hand. In regard to the problem faced, an application is to be built which would take the donor details, store them and inform them upon a request.

Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	Recipient	To get plasma	It takes much time and hard to find	Finding Plasma donor's not easy	Frustrated
PS-2	Donor	To donate the plasma	It takes much time and hard to find	Finding the required blood group recipient is difficult	Irritated

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 behaviors 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

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



Brainstorm & idea prioritization

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

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

➔

Before you collaborate

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

🕒 10 minutes

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 [your problem statement]?



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.

Step-2: Brainstorm, Idea Listing and Grouping

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

AUGUSTIN SHAR J	BADHRISAYA RAJA S M	BABIN MOH B	GOUDON RAJ R
Aesthetic UI Design	User Friendly	Keeping Anonymity among users	Make the app reach rural places
Storing personal details like address in a secure manner	Responsive UI	Using 2FA in app	Posters and social media marketing
Email notification functionality	Collaboration with WHO	Collaboration with Google Calendar	Clear instructions
Dynamic database updation	Free of cost Platelets	Rewards for Donors	Provide platelets all over the world
Simple and direct buttons and instructions	Message and E mail notification of donors and receivers	Donation camps in Rural Areas	Coupon codes and Goodies to donors
Collaborating with government	User Security	Periodic Fundraising Sessions	User Feedback
Awareness page inside the app	Ease of access of Platelets	Section spreading the need for plasma donation	Arrange delivery facilities
Advertise and market the app	Keep track of users	Get specialists on-board for health support	Dark mode and light mode UI
Incentives to the Donor	Fast availability of Donors	Separate pages for Donor and Receiver	Fast fixing of bugs

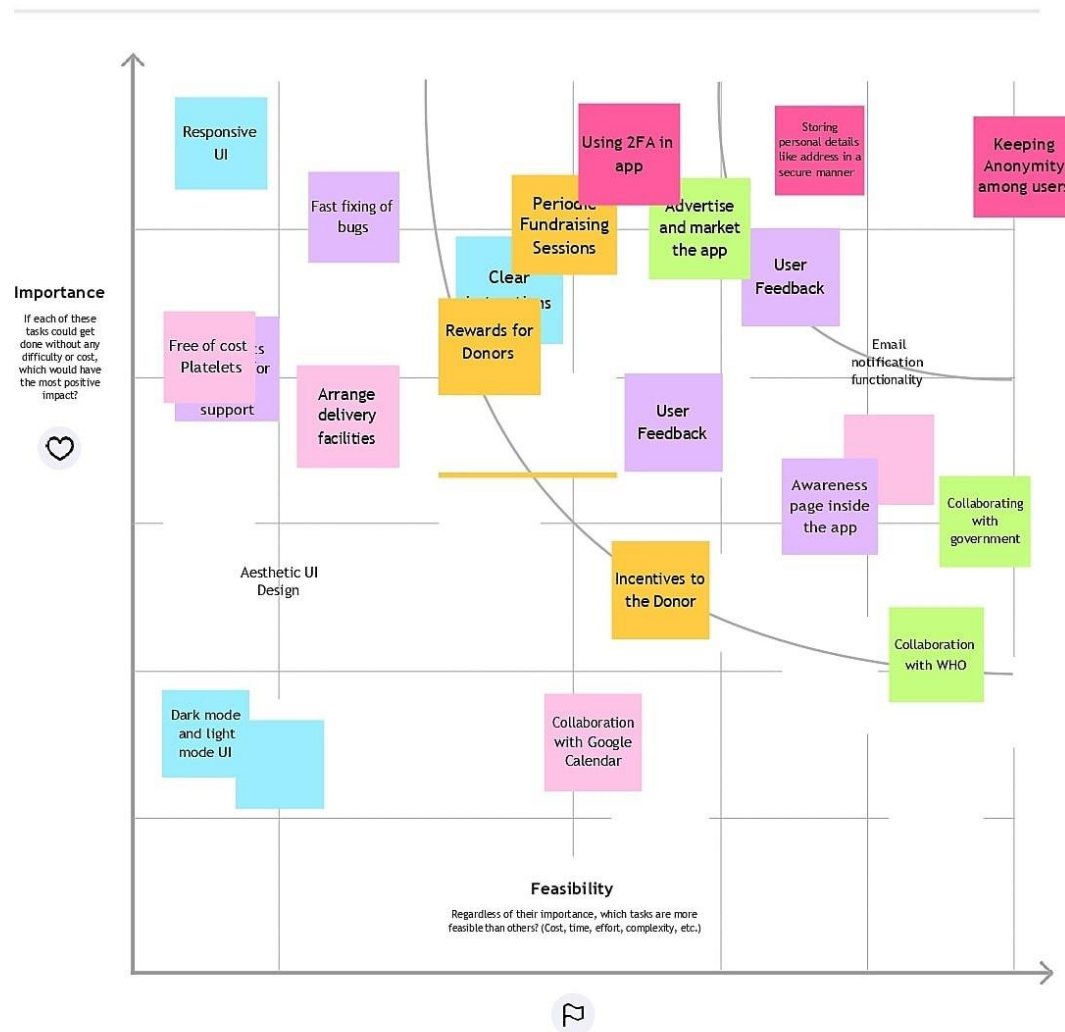
Step-3: Idea Prioritization

4

Prioritize

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

🕒 20 minutes



3.3. Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Plasma Donor identifying and providing it to the required peoples
2.	Idea / Solution description	A plasma is a liquid portion of the blood, over 55% of human blood is plasma. Plasma is used to treat various infectious diseases and it is one of the oldest methods known as plasma therapy. Plasma therapy is a process where blood is donated by recovered patients in order to establish antibodies that fights the infection. In this project plasma donor application is being developed by using AWS services.
3.	Novelty / Uniqueness	Plasma Donor Application by integrating it with IBM cloud
4.	Social Impact / Customer Satisfaction	Effect of donor motivation on donor satisfaction and loyalty is variable due to the influence of common donorship attitudes prevailing in donor population, impact of social marketing programs, focused on promotion of donor commitment and deliberate donorship. Thus, we have predicted that effect of donor motivation on donor relationship satisfaction and loyalty change.

5.	Business Model (RevenueModel)	<ul style="list-style-type: none"> • Can collaborate with plasma donor agencies. • Can collaborate with Hospitals.
6.	Scalability of the Solution	PLASMA DONOR APPLICATION The results are represented with the scalability of different logins in any application. It tells the systematic process of each user login scalability.

3.4. Problem Solution fit

Problem-solution fit is a term used to describe the point validating that the base problem resulting in a business idea really exists and the proposed solution solves that problem.

The problem-solution fit is when you-

- **Validate that the problem exists:** When you validate your problem hypothesis using real-world data and feedback. That is, you gather information from real users to determine whether they care about the pain point you're trying to solve.
- **Validate that your solution solves the problem:** When you validate that the target audience appreciates the value your solution delivers to them.

1. CUSTOMER SEGMENT(S) <ul style="list-style-type: none"> • Donors • Patient • Hospitals 	6. CUSTOMER CONSTRAINTS <ul style="list-style-type: none"> • Regular Internet connection • Donor health condition • Unavailability of plasma 	5. AVAILABLE SOLUTIONS <p>The existing application used only collecting details of donors but it does not notify them at the right time.</p> <p>Our solution is building a website that notifies the donors at the right time.</p>
2. JOBS-TO-BE-DONE/PROBLEMS <ul style="list-style-type: none"> • Difficult to find donors at the right time / at the time of emergency. • Donors not aware of plasma requirements. 	9. PROBLEM ROOT CAUSE <ul style="list-style-type: none"> • Not able to find the donors at the time of emergency. • Count of donors has been tremendously decreasing since hospital management couldn't contact them or get them notified at the right. 	7. BEHAVIOUR <p>The customer comes forward to</p> <ul style="list-style-type: none"> • Attend plasma donation camps. • Donate plasma • The hospital management/ patient is able to find plasma donors at the right time.
3. TRIGGERS <p>Blood donation improves or saves lives and enhances social solidarity. It is also influenced by increasing deaths due to unavailability of plasma at required times.</p>	10. YOUR SOLUTION <p>Creating website which will provide information about available donors and plasma. If not available, the customer will be notified when plasma is available.</p>	8. CHANNELS OF BEHAVIOUR <p>Online:</p> <p>Can use the website to find donors.</p> <p>Offline:</p> <p>Can use the record maintain by the hospital.</p>
4. EMOTIONS: BEFORE/AFTER <p>Before:</p> <p>Patient/ hospital find it hard to get a right resource to get plasma leaving them upset.</p> <p>After:</p> <p>The donors and customers have a feeling of satisfaction.</p>		

These forces for progress are the emotional forces that generate and shape the needs of consumers for a specific product. These forces can be used to better describe and understand the high demand from customers towards certain solutions with **JTBD (Jobs to-be-done, tasks to be completed)** or demand for some specific product.

4. REQUIREMENT ANALYSIS

Requirement analysis is significant and essential activity after elicitation. We analyze, refine, and scrutinize the gathered requirements to make consistent and unambiguous requirements. This activity reviews all requirements and may provide a graphical view of the entire system. After the completion of the analysis, it is expected that the understandability of the project may improve significantly.

4.1. Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story/ Sub-Task)
FR-1	User Registration	Registration through Form (WebApp)
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Certification	After the donor donates plasma, we will give them a certificate of appreciation and authentication.
FR-4	Statistical data	The availability of plasma is given in the page as stats, which will be helpful for the users.
FR-5	User Plasma Request	Users can request to donate plasma by filling out their quest form on the page. Once the request is submitted, they will get an email
FR-6	Searching/reporting requirements	Users can use the search bar to lookup information about camps and other topics.
FR-7	Virtual Assistants	A virtual assistant is a software agent that can carry out tasks or provide services on behalf of a person in response to commands or inquiries. When users enter their inquiries, the system will respond with pertinent information about plasma and details of plasma donation.

4.2. Non-Functional requirements

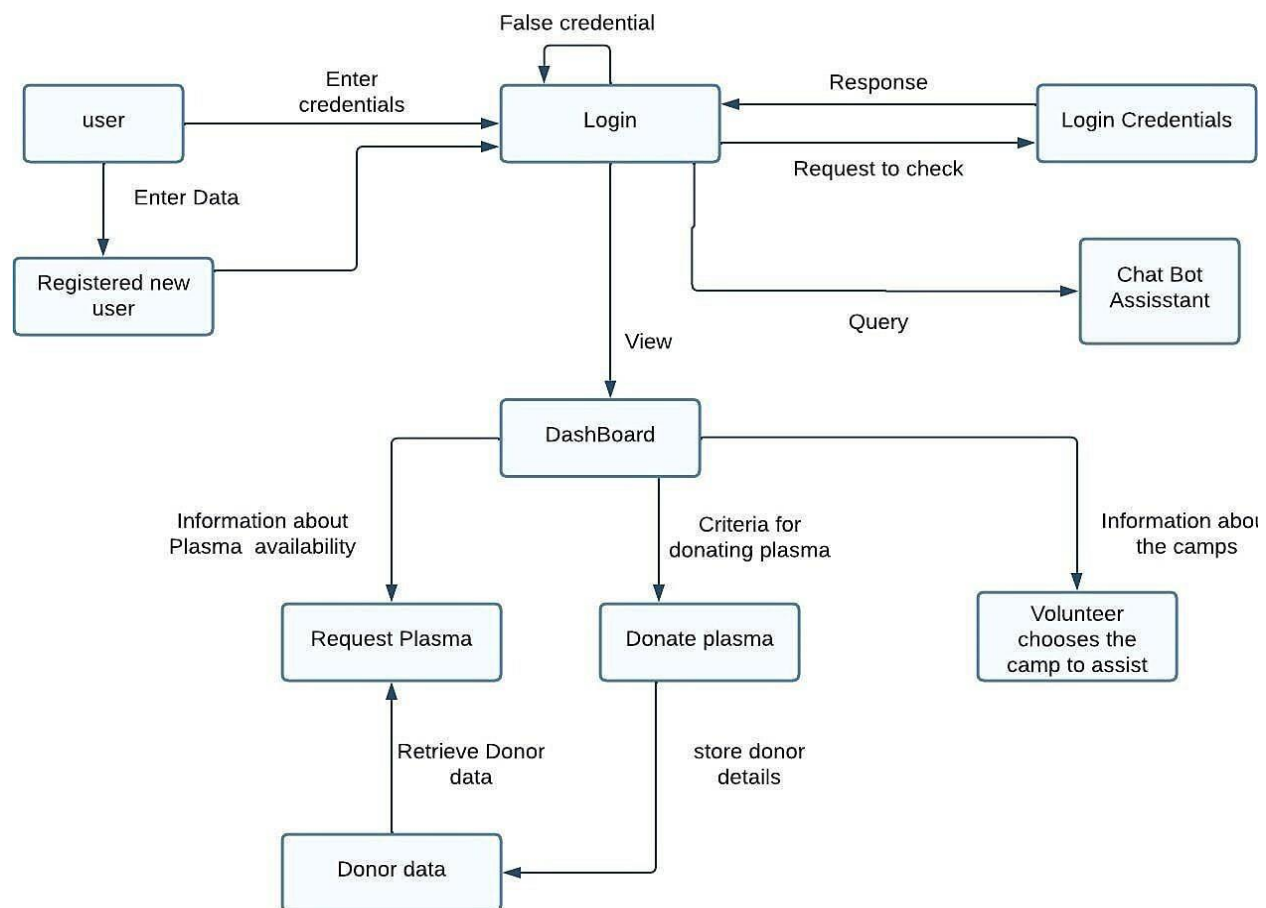
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Must have a good-looking User-friendly interface.
NFR-2	Security	It must be secured with the proper username and password.
NFR-3	Reliability	The system should be made in such a way that it is reliable in its operations and for securing the sensitive details.
NFR-4	Performance	Users should have a proper Internet Connection.
NFR-5	Availability	The system including the online and offline components should be available 24/7.
NFR-6	Scalability	The application can handle growing numbers of users and load without compromising on performance and causing disruptions to user experience.

5. PROJECT DESIGN

Project design is an early phase of the project lifecycle where ideas, processes, resources, and deliverables are planned out. A project design comes before a project plan as it's a broad overview whereas a project plan includes more detailed information.

5.1. Data Flow Diagrams

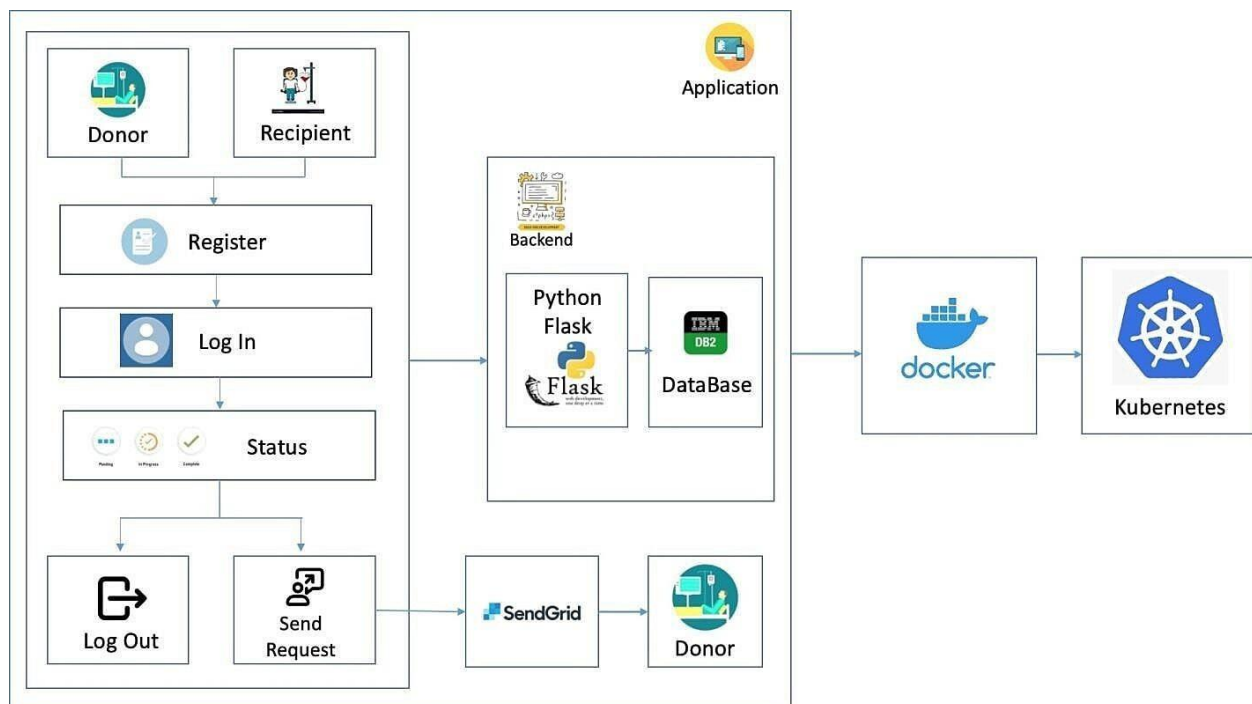
A data flow diagram (DFD) is a graphical or visual representation using a standardized set of symbols and notations to describe a business's operations through data movement. They are often elements of a formal methodology such as Structured Systems Analysis and Design Method (SSADM).



5.2. Solution & Technical Architecture

Solution architecture belongs to the list of most important practices executed before any tech solution development begins. In this article, we'll discuss what solution architecture is, describe the role of a solution architect, and explain how the adoption of this expertise can help solve business problems.

In a rapidly changing technology environment, organizations face the need to transform their processes and systems to meet emerging business requirements. This digital transformation demands specific expertise and a set of practices to align business focus with technology solutions.



5.3 User Stories

A user story is an informal, general explanation of a software feature written from the perspective of the end user. Its purpose is to articulate how a software feature will provide value to the customer.

USN-1: As a user, I can register for the application by entering my e-mail, password and confirming my password.

USN-2: As a user, I will receive confirmation e-mail once I have registered for the application

USN-3: As a user, I can register for the application through G-mail

USN-4: As a user, I can login to the application by entering e-mail & password

USN-5: As a user, I can send the proper requests to donate and obtain plasma

USN-6: As a user, I can register and log into the application by entering e-mail & password to view the profile

USN-7: As a user, I can send the proper requests to donate and obtain plasma.

USN-8: As a customer care executive, I can try to address user's concerns and questions.

USN-9: As an administrator I can help with user-facing aspects of a website, like its appearance.

USN-10: As an administrator, I can involve working with the technical side of websites

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	<i>Functional Requirement (Epic)</i>	User story number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	A User can register and create the user account.	6	High	Augustin Badhri Babin Godson
Sprint-1	Login	USN-2	A User can sign-in to the application by entering the registered e-mail id and password.	6	High	Augustin Badhri Babin
Sprint-1	Admin Register	USN-3	An admin can register through the admin registry.	4	Medium	Augustin Badhri
Sprint-1	Register Admin Via Script	USN-4	Creating an Admin Account using a python script. As for security reasons we should implement a separate python script.	4	High	Augustin Badhri
Sprint-2	Implementing Authentication System	USN-5	creating an authentication system for both admin and users using flask application	6	High	Augustin Badhri

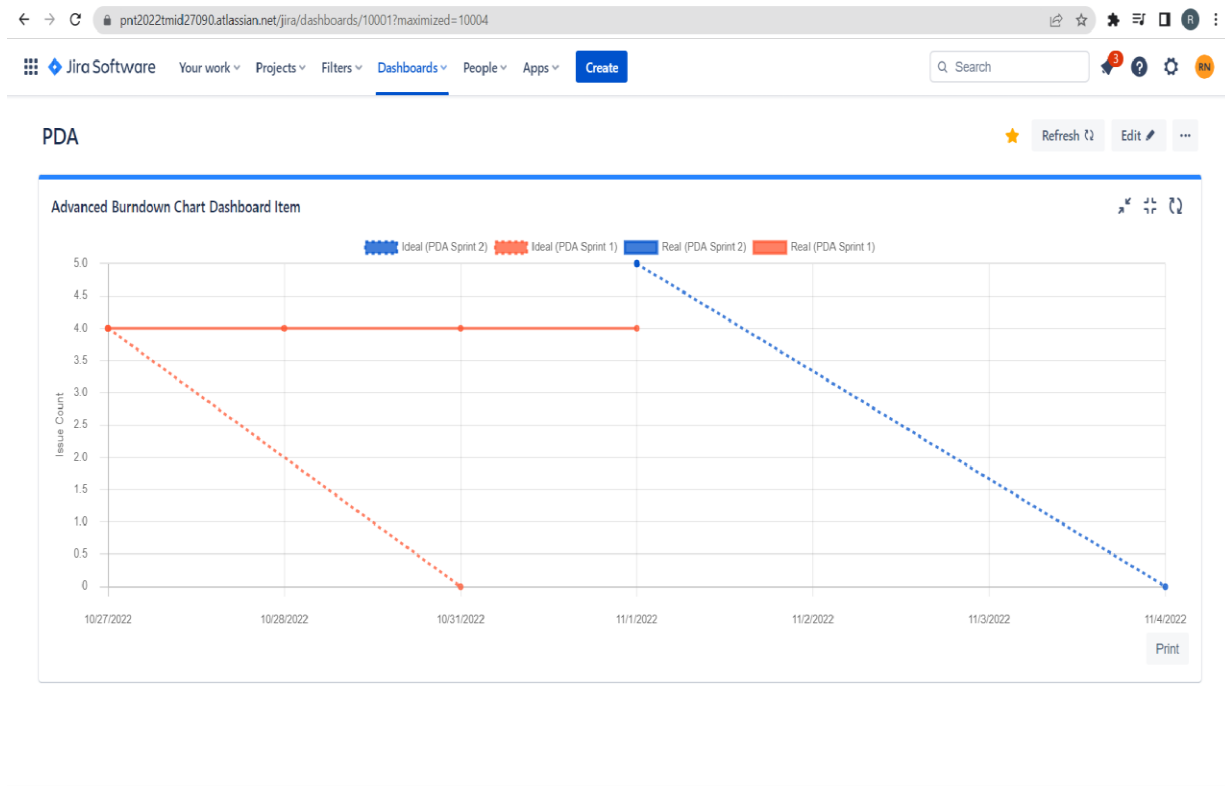
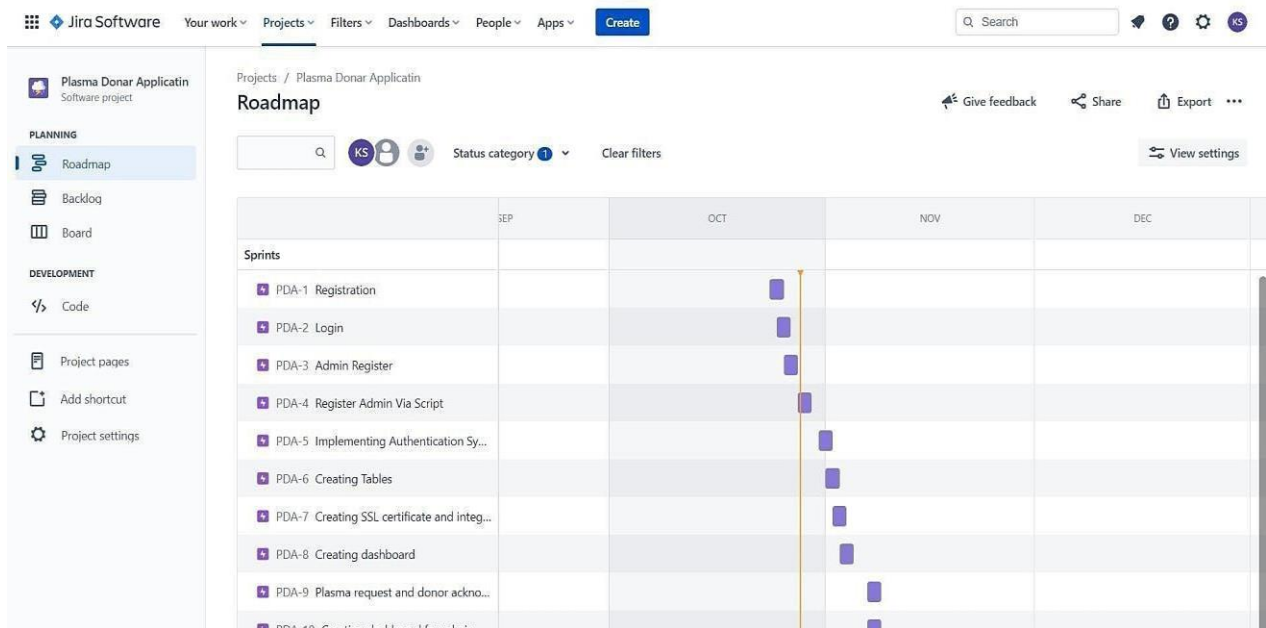
Sprint-2	Creating Tables	USN-6	Creating Db2 account and creating the tables in DB2 in IBM clouddb2	4	Medium	Babin Godson
Sprint-2	Creating SSL certificate and integrating with python code	USN-7	Creating the SSL certificate to connect db2 via python code.	6	High	Augustin Godson
Sprint-2	Creating dashboard	USN-8	Admin and Donor can interact with our application.	4	Medium	Babin
Sprint-3	Plasma request and donor acknowledge feature	USN-9	Admin can create plasma requests which will be shown in the user portal.	6	High	Badhri Augustin
Sprint-3	Creating dashboard for admin	USN-10	Admin dashboard, admin can view the total request has been requested for plasma by the recipient/user.	6	High	Godson Babin
Sprint-3	Integrating the Watson chat bot	USN-11	Users can use the chat bot for basic clarification using the chat bot	4	Medium	Badhri Augustin
Sprint-3	Integration with SendGrid	USN-12	The source/verification mail for both user (donor and recipient)	4	Medium	Badhri Babin
Sprint-4	Docker installation	USN-13	Installing Docker CLI	4	Low	Augustin

sprint-4	Creating Docker image	USN-14	Setting up the Docker environment and creating the Docker image file	6	High	Godson Babin
Sprint-4	Kubernetes	USN-15	Creating pods in Kubernetes and uploading it in IBM cloud	6	Medium	Badhri Augustin
Sprint-4	End-to-End Testing	USN-16	Implementing End-to-End testing	6	High	Godson Babin

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	8	5 Days	27 Oct 2022	31 Nov2022	8	03 Nov2022
Sprint-2	13	4 Days	01 Nov 2022	06 Nov 2022	12	07 Nov 2022
Sprint-3	11	5 Days	07 Nov 2022	12 Nov2022	11	09 Nov 2022
Sprint-4	9	5 Days	14 Nov 2022	19 Nov2022	8	15 Nov 2022

6.3 Reports from JIRA



7. CODING & SOLUTIONING

7.1 Feature 1

Python

It is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library. Guido van Rossum began working on Python in the late 1980s as a successor to the ABC programming language and first released it in 1991 as Python0.9.0., Python 2.0 was released in 2000 and introduced new features such as list comprehensions, cycle-detecting garbage collection, reference counting, and Unicode support. Python 3.0, released in 2008, was a major revision that is not completely backward-compatible with earlier versions. Python2 was discontinued with version 2.7.18 in 2020. Python consistently ranks as one of the most popular programming languages

7.2 Feature 2

Flask

Flask is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework-related tools.

LOGIN

First page of the blood plasma donor application is login page. The login page is common for admin, donor, and requester. A person can login with username and password with he/she registered. If the password of the person is correct, they will be taken to the dashboard. If the password or username is incorrect, it pops up a notification saying incorrect.

login.html

```
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
<meta charset="UTF-8">
<title>IBM 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="{ { url_for('static', filename='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="username" placeholder="Enter UserName"
required="required" style="color:black" />
<input type="password" name="password" 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">
{ { msg } }</div>
</div>
</body>
</html>

```

REGISTRATION

A new donor/requester can register themselves by providing their name, username, password, age, address, blood group and covid 19 infection status or any other medical history. By registering as a new user, they can login anytime using the username and password.

register.html

```
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
<meta charset="UTF-8">
<title>IBM PlasmaDonor 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="{ { url_for('static', filename='style.css') } }">
<style>
.login{ top: 20%;
}
</style>
</head>
<body>
<div class="header">
<div>Plasma Donor App</div>
<ul>
</div>
</ul>
<li><a class="active" href="/login">Home</a></li>
<div class="login">
<!-- Main Input For Receiving Query to our ML -->
<form action="{ { url_for('register') } }"method="post">
```

```

<input type="text" name="username" 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">
infection status</option>
</select>
<select name="blood"> group</option>
</select>
<option value="select" selected>Select COVID
<option value="infected">Infected</option>
<option value="uninfected">Uninfected</option>
<option value="select" selected>Choose your blood
<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>
<input type="password" name="password" 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">
{{ msg }}</div>
</div>
</body>
</html>
```

DASHBOARD

A dashboard is a visual display of all your data. While it can be used in all kinds of different ways, its primary intention is to provide information at-a-glance, such as KPIs. Here dashboard is used to dodge between pages for the ease of use for the users. dashboard.html

dashboard.html

```
<!DOCTYPE html>
<html>
<head>
<title>Plasma Donor Application</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Raleway">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
<style>
html,body,h1,h2,h3,h4,h5 {font-family: "Raleway", sans-serif}
body{
  position: relative;
}
.modal {
  visibility: hidden;
  opacity: 0;
```

```
position: absolute;
top: 0;
right: 0;
bottom: 0;
left: 0;
display: flex;
align-items: center;
justify-content: center;
background: rgba(77, 77, 77, .7);
transition: all .4s;
}
```

```
.modal:target {
  visibility: visible;
  opacity: 1;
}
```

```
.modal__content {
  border-radius: 4px;
  position: relative;
  width: 500px;
  max-width: 90%;
  background: #fff;
  padding: 1em 2em;
  border: 2px solid yellow;
}
```

```
/* .modal__footer {
  text-align: right;
  a {
    color: #585858;
```

```

}
i {
  color: #d02d2c;
}
} */
.modal__close {
  position: absolute;
  top: 10px;
  right: 10px;
  color: #585858;
  text-decoration: none;
  font-size: 2rem;
}

.request-button{

  height: auto;
  padding: .5em 1em;
  text-transform: uppercase;
  background-color: transparent;
  border: 1px solid #000;
  border-radius: 100vmax;
}

.request-button a{
  text-decoration: none;
}

form{
  padding: 1em;
  font-size: 1.2rem;

```

```
}
```

```
.form-control {  
  display: flex;  
  justify-content: space-between;  
  margin-bottom: 1.5em;
```

```
.form-control:last-  
child{  
  margin-bottom: 0;  
}
```

```
.form-control * {  
  width: 100%;  
}
```

```
.form-control button{  
  width: auto;  
  padding: .5em 2em;  
  background-color: transparent;  
  border: 1px solid #000;  
  border-radius: 100vmax;  
}
```

```
.gender-field {  
  display: grid;  
  grid-auto-flow: column;  
}
```

```
input{  
  outline: none;
```



```
}
```

```
input[type=submit]{  
  background-color: transparent;  
  border: none;  
  margin: 0;  
  padding: 0;  
}
```

```
.request-container{  
  display: flex;  
  justify-content: center;  
  align-items: center;  
  gap: 2em;  
  position: absolute;  
  top: 1em;  
  right: 2em;  
}
```

```
.request-container form{  
  margin: 0;  
  padding: 0;  
}
```

```
</style>
```

```
</head>
```

```
<body class="w3-pale-yellow">
```

```
<div class="request-container">
```

```
  <div class="request-button">
```

```
    <a href="#demo-modal">REQUEST</a>
```

```
</div>
```

```
  <div class="request-button">
```

```
<form action="/logout" method="POST">
  <input type="submit" value="LOG OUT">
</form>
</div>
</div>
```

```
<div id="demo-modal" class="modal">
  <div class="modal__content">
    <h1>Request Form</h1>
    <form action="/req" method="POST">
      <div class="form-control">
        <label for="username">Username:</label>
        <input
          type="text"
          id="username"
          name="username"
          maxlength="30"
          placeholder="username"
          required
        />
      </div>
      <div class="form-control">
        <label for="email">Email:</label>
        <input
          type="email"
          id="email"
          name="email"
          placeholder="email"
          required
        />
      </div>
    </form>
  </div>
</div>
```

```

<div class="form-control">
  <label for="mobile">Mobile:</label>
  <input
    type="number"
    id="mobile"
    name="mobile"
    placeholder="mobile"
    required
  />
</div>
<div class="form-control">
  <label for="bloodgroup">Blood Group:</label>
  <select name="blood" id="bloodgroup">
    <option value="" selected disabled hidden>
      Choose your blood group
    </option>
    <option value="O+">O+</option>
    <option value="A+">A+</option>
    <option value="B+">B+</option>
    <option value="AB+">AB+</option>
    <option value="O-">O-</option>
    <option value="A-">A-</option>
    <option value="B-">B-</option>
    <option value="AB-">AB-</option>
  </select>
</div>
<div class="form-control">
  <button type="submit">Submit</button>
  <button type="reset">Reset</button>
</div>
</form>

```

```

        <a href="#" class="modal__close">&times;</a>
    </div>
</div>
<!-- Top container -->
<div class="w3-bar w3-top w3-black w3-large" style="z-index:4">
    <button class="w3-bar-item w3-button w3-hide-large w3-hover-none w3-hover-text-light-grey" onclick="w3_open();"><i class="fa fa-bars"></i> Menu</button>
    <span class="w3-bar-item w3-right"></span>
</div>
<!-- Sidebar/menu -->
<nav class="w3-sidebar w3-collapse w3-amber w3-animate-left" style="z-index:3;width:300px;" id="mySidebar"><br>
    <div class="w3-container w3-row">
        <div class="w3-col s8 w3-bar">
            <span>Welcome, <strong>{ {un} }</strong></span><br>
            <a href="#" class="w3-bar-item w3-button"><i class="fa fa-user"></i></a>
        </div>
    </div>
    <hr>
    <div class="w3-container">
        <h5>Dashboard</h5>
    </div>
    <div class="w3-bar-block">
        <a href="#" class="w3-bar-item w3-button w3-padding-16 w3-hide-large w3-dark-grey w3-hover-black" onclick="w3_close()" title="close menu"><i class="fa fa-remove fa-fw"></i> Close Menu</a>
        <a href="#" class="w3-bar-item w3-button w3-padding w3-blue"><i class="fa fa-users fa-fw"></i> Overview</a>
        <a href="/index" class="w3-bar-item w3-button w3-padding"><i class="fa fa-home fa-fw"></i> Home </a>
    </div>

```

```

</nav>
<!-- Overlay effect when opening sidebar on small screens -->
<div class="w3-overlay w3-hide-large w3-animate-opacity" onclick="w3_close()"
style="cursor:pointer" title="close side menu" id="myOverlay"></div>
<!-- !PAGE CONTENT! -->
<div class="w3-main" style="margin-left:300px;margin-top:43px;">
  <!-- Header -->
  <header class="w3-container" style="padding-top:22px">
    <h5><b><i class="fa fa-dashboard"></i> My Dashboard</b></h5>
  </header>
  <div class="w3-row-padding w3-margin-bottom">
    <div class="w3-quarter">
      <div class="w3-container w3-amber w3-padding-16">
        <div class="w3-left"><i class="fa fa-tint w3-xxlarge"></i></div>
        <div class="w3-right">
          <h3>{ { dicrow } }</h3>
        </div>
      <div class="w3-clear"></div>
      <h4>Plasma Availability</h4>
    </div>
  </div>
  <div class="w3-quarter">
    <div class="w3-container w3-orange w3-text-white w3-padding-16">
      <div class="w3-left"><i class="fa fa-users w3-xxlarge"></i></div>
      <div class="w3-right">
        <h3>{ { tot } }</h3>
      </div>
    <div class="w3-clear"></div>
    <h4>Application Users</h4>
  </div>
</div>

```

```

</div>
<div class="w3-panel">
  <div class="w3-twothird">
    <h5>Plasma </h5>
    <table class="w3-table w3-striped w3-khaki">
      <tr>
        <td><i class="fa fa-tint"></i></td>
        <td><b>A+</b></td>
        <td><i>1</i></td>
      </tr>
      <tr>
        <td><i class="fa fa-tint"></i></td>
        <td><b>A-</b></td>
        <td><i>0</i></td>
      </tr>
      <tr>
        <td><i class="fa fa-tint"></i></td>
        <td><b>B+</b></td>
        <td><i>1</i></td>
      </tr>
      <tr>
        <td><i class="fa fa-tint"></i></td>
        <td><b>B-</b></td>
        <td><i>0</i></td>
      </tr>
      <tr>
        <td><i class="fa fa-tint"></i></td>
        <td><b>AB+</b></td>
        <td><i>2</i></td>
      </tr>
      <tr>

```

```

        <td><span class="fa fa-tint"></span></td>
        <td><b>AB-</td>
        <td><i>1</i></td>
    </tr>
    <tr>
        <td><i class="fa fa-tint"></i></td>
        <td><b>O+</td>
        <td><i>1</i></td>
    </tr>
    <tr>
        <td><i class="fa fa-tint"></i></td>
        <td><b>O-</td>
        <td><i>0</i></td>
    </tr>
</table>
</div>
</div>
</div>
<hr>
<br>
<div class="w3-container w3-khaki w3-padding-16 w3-margin-
top=100px"style="margin-left:300px;">
    <div class="w3-row">
        <div class="w3-container w3-third">
            <h5 class="w3-bottombar w3-border-green">Demographic</h5>
            <p>Language</p>
            <p>Country</p>
            <p>City</p>
        </div>
        <div class="w3-container w3-third">
            <h5 class="w3-bottombar w3-border-red">System</h5>

```

```

    <p>Browser</p>
    <p>OS</p>
    <p>More</p>
</div>
<div class="w3-container w3-third">
    <h5 class="w3-bottombar w3-border-orange">Target</h5>
    <p>Users</p>
    <p>Active</p>
    <p>Geo</p>
    <p>Interests</p>
</div>
</div>
</div>
<!-- Footer -->
<footer class="w3-container w3-padding-16 w3-pale-yellow">
    <h4>FOOTER</h4>
    <p>Powered by <a href="https://www.w3schools.com/w3css/default.asp"
target="_blank">w3.css</a></p>
</footer>
<!-- End page content -->
</div>
</body>
</html>

```

app.py

```

from turtle import st
from flask import Flask, render_template, request, redirect, url_for, session, jsonify
from markupsafe import escape
import os
from dotenv import load_dotenv
from sendgrid import sendgrid

```



```

from sendgrid.helpers.mail import Mail, Email, To, Content
import ibm_db
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=1bbf73c5-d84a-4bb0-
85b9-
ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud;PORT=32286;
SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=xnw033
76;PWD=CAip5VvCHctcirIL","")

app = Flask(__name__)
load_dotenv() #load keys from .env
sg = sendgrid.SendGridAPIClient(api_key =
os.environ.get('GmWQl2hzTKeUsU776_Xkyw')) #set SendGrid API Key
from_email = Email("babinmon2001@gmail.com")
@app.route('/')
def home():
    sql = "select * from register where age=21 "
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.execute(stmt)
    dicrow = ibm_db.fetch_assoc(stmt)
    print(dicrow)
    return render_template('index.html')

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

    return render_template('index.html')

@app.route('/logout',methods=['POST','GET'])
def logout():
    if request.method == 'POST':
        return render_template('logout.html')

```

```

else:
    return render_template('dash.html')

@app.route('/home',methods=['POST','GET'])
def homepage():
    if request.method == 'POST':
        return render_template('index.html')
    else:
        return render_template('logout.html')
@app.route('/req',methods=['POST','GET'])
def req():
    if request.method == 'POST':
        username = request.form['username']
        email = request.form['email']
        phone = request.form['mobile']
        blood=request.form['blood']
        insert_sql = "INSERT INTO request VALUES (?, ?, ?, ?)"
        prep_stmt = ibm_db.prepare(conn, insert_sql)
        ibm_db.bind_param(prepare_stmt, 1, username)
        ibm_db.bind_param(prepare_stmt, 2, email)
        ibm_db.bind_param(prepare_stmt, 3, phone)
        ibm_db.bind_param(prepare_stmt, 4, blood)
        ibm_db.execute(prepare_stmt)
        return render_template('index.html',msg="saved successfully")
@app.route('/access',methods=['POST','GET'])
def accesss():
    if request.method == 'POST':
        username = request.form['username']
        email = request.form['email']
        phone = request.form['mobile']
        address=request.form['address']

```

```

password=request.form['password']
insert_sql = "INSERT INTO access VALUES (?, ?, ?, ?, ?)"
prep_stmt = ibm_db.prepare(conn, insert_sql)
ibm_db.bind_param(prepare_stmt, 1, username)
ibm_db.bind_param(prepare_stmt, 5, email)
ibm_db.bind_param(prepare_stmt, 3, phone)
ibm_db.bind_param(prepare_stmt, 4, address)
ibm_db.bind_param(prepare_stmt, 2, password)
ibm_db.execute(prepare_stmt)
return render_template('index.html',msg="saved successfully")
@app.route('/reg',methods = ['POST', 'GET'])
def reg():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        email = request.form['email']
        blood=request.form['blood-group']
        gender=request.form['gender']
        age=request.form['age']
        address=request.form['address']
        phone=request.form['mobile']
        insert_sql = "INSERT INTO register VALUES (?, ?, ?, ?, ?, ?, ?, ?)"
        prep_stmt = ibm_db.prepare(conn, insert_sql)
        ibm_db.bind_param(prepare_stmt, 1, username)
        ibm_db.bind_param(prepare_stmt, 2, password)
        ibm_db.bind_param(prepare_stmt, 3, email)
        ibm_db.bind_param(prepare_stmt, 4, blood)
        ibm_db.bind_param(prepare_stmt, 5, gender)
        ibm_db.bind_param(prepare_stmt, 6, age)
        ibm_db.bind_param(prepare_stmt, 7, address)
        ibm_db.bind_param(prepare_stmt, 8, phone)

```

```

ibm_db.execute(prepare_stmt)
to_email = To(email) # set user as recipient for confirmation email
subject = "Welcome to GetPlasma"
content = Content("text/html",
                  "<p>Hello " + username + ",</p><p>Thank you for registering to the
GetPlasma Application!</p><p>If this wasn't you, then immediately report to our
<a href=\"mailto:getplasmaproject@gmail.com\">admin</a> or just reply to this
email.</p>")
email = Mail(from_email, to_email, subject, content) # construct email format
email_json = email.get() # get JSON-ready representation of the mail object
return render_template('index.html',msg="saved successfully")
@app.route('/login',methods = ['POST', 'GET'])
def login():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        sql = "select * from access where username=? and password=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, username)
        ibm_db.bind_param(stmt, 2, password)
        ibm_db.execute(stmt)
        dic = ibm_db.fetch_assoc(stmt)
        print(dic)
        role = str()
        requests = []
        if dic:
            un=dic['USERNAME']
            print(un)
            sql = "select count(*) from register"
            stmt = ibm_db.prepare(conn, sql)
            ibm_db.execute(stmt)

```

```

dicrow = ibm_db.fetch_assoc(stmt)
sql1 = "select count(*) from request"
stmt1 = ibm_db.prepare(conn, sql1)
ibm_db.execute(stmt1)
dicrow1 = ibm_db.fetch_assoc(stmt1)
sql2 = "select count(*) from access"
stmt2 = ibm_db.prepare(conn, sql2)
ibm_db.execute(stmt2)
dicrow2 = ibm_db.fetch_assoc(stmt2)
tot=(dicrow['1']+dicrow1['1']+dicrow2['1'])
bloodA = ibm_db.fetch_assoc(stmt1)
print(bloodA)
return render_template('dash.html',un=un,dicrow=dicrow['1'],tot=tot)
else:
    return render_template('index.html')
return render_template('index.html')

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

```

7.3 Database Schema

IBM Db2

A hybrid ANSI-compliant data virtualization tool for accessing, querying and summarizing data across the enterprise which Provides a massively parallel processing (MPP) architecture Exploits Hive, HBase and Apache Spark concurrently for best-in-class analytic capabilities Requires only a single database connection or query to connect disparate sources such as HDFS, RDMS, NoSQL databases, object stores and Web HDFS, Provides low latency support for ad-hoc and complex queries, high performance, and federation capabilities Understands dialects from other vendors and various products from Oracle, IBM, Db2 and IBM Netezza Enables advanced row and column security.

IBM Db2 on Cloud

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

XNW03376.REQUEST

Back

Export to CSV

USERNAME	EMAIL	MOBILE	BLOOD
babin	ba@gmail.com	9632587415	AB+
babin	ba@gmail.com	9632587415	AB+
godson	bbabinmon@gmail.com	9874563210	O-
sham	hg@gmail.com	3698521475	AB-
sham	sham28@gmail.com	9654123875	A+
vijo	vijo@gmail.com	9512357846	AB-

Search

ENG IN 10:23 PM 18-11-2022

KUBERNETES

Kubernetes also known as “k8s” or “kube” is a container orchestration platform for scheduling and automating the deployment, management, and scaling of containerized applications. Kubernetes was first developed by engineers at Google before being open sourced in 2014. It is a descendant of Borg, a container orchestration platform used internally at Google. Kubernetes is Greek for helmsman or pilot, hence the helm in the Kubernetes logo (link resides outside IBM). Today, Kubernetes and the broader container ecosystem are maturing into a general-purpose computing platform and ecosystem that rivals — if not surpasses — virtual machines (VMs) as the basic building blocks of modern cloud infrastructure and applications. This ecosystem enables organizations to deliver a high- productivity Platform-as-a-Service (PaaS) that addresses multiple infrastructure-related and operations-related tasks and issues surrounding cloud-native development so that development teams can focus solely on coding and innovation.

8. TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functional of your components, sub- assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectation and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement

8.1 Test Cases

Sprint-1

Test caseID	Test Scenario	Expected Result	Status
Home_TC_OO1	Verify user is able to see theLogin , register and support button	Login, register and support button is displayed	Pass
Home_TC_OO2	Verify whether register button works	Redirected to registration page	Pass
Home_TC_OO3	Verify whether login button works	Redirected to login page	Pass
Home_TC_OO4	Verify whether support button works	Redirected to support page	Pass
Registration_TC_OO1	Verify whether registerbutton works	Redirects to Personal details page	Pass
Registration_TC_OO2	Verify whether the page will redirect to login page if account already registered	Redirects to Login page	Pass
Login_TC_OO1	Verify whether user can see email and password text box	User should navigate to user account homepage	Pass

Login_TC_OO2	Verify user is able to log into application with Valid credentials	Application redirects to dashboard	Pass
Login_TC_OO3	Verify user able to log into application with Invalid Credentials	Application should show 'Incorrect e-mail or password' validation message.	Pass

Sprint-2

Test caseID	Test Scenario	Expected Result	Status
Dashboard_TC_OO1	Verify user is able to see the dashboard after login in with correct credentials	Respective Dashboard is displayed	Pass
Dashboard_TC_OO2	Verify the user login as donor, it should donor dashboard	Redirected to donor dashboard	Pass
Dashboard_TC_OO3	Verify the user login as requester, it should donor dashboard	Redirected to requester dashboard	Pass
Dashboard_TC_OO4	Verify the user logins as admin, it should admin dashboard	Redirected to admin dashboard	Pass

Sprint-3

Test caseID	Test Scenario	Expected Result	Status
Request_TC_OO 1	Verify if the requester can view request form for blood plasma	Request form is displayed	Pass
Request_TC_OO 2	Verify if the admin can view request form for blood plasma	Request form is displayed	Pass
Request_TC_OO 3	Verify if the donor can view request form for blood plasma	Request form is displayed	Pass
Request_TC_OO 4	Verify if the requester can make request for blood plasma	Make plasma request	Pass
Request_TC_OO 5	Verify if the donor can make request for blood plasma	Make plasma request	Pass
Request_TC_OO 6	Verify if the donor can view the plasma requested	View plasma request	Pass
Request_TC_OO 7	Verify if the admin can view the plasma requested	View plasma request	Pass
Request_TC_OO 8	Verify if the admin can make request for blood plasma	Make plasma request	Pass
Request_TC_OO 9	Verify if the admin can update or reduce the plasma stock	Can update and view stock	Pass

Sprint-4

Test case ID	Test Scenario	Expected Result	Status
View_TC_OO1	Verify if the requester can view if their request has been approved or in waiting state	Request status is displayed	Pass
View_TC_OO2	Verify if the admin can view the approved plasma request	Request status is displayed	Pass
View_TC_OO3	Verify if the donor can view their donation history	Donation history is displayed	Pass
view_TC_OO4	Verify if the donor can accept a plasma request	Accept plasma request	Pass
Request_TC_OO5	Verify if the mail has been sent to donor	Send mail	Pass
Request_TC_OO6	Verify if the admin can view donor and requester details	View donor and requester details	Pass

8.2 USER ACCEPTANCE TESTING

The purpose of this 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).

SPRINT-1

DEFECT ANALYSIS

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

Resolution	Severity1	Severity2	Severity3	Severity4	Sub total
By Design	1	2	2	3	8
Duplicate	1	0	1	0	2
External	2	0	0	1	3
Fixed	4	2	4	1	11
Not Reproduced	0	0	0	0	0
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	16	6	7	14	24

TEST CASE ANALYSIS

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

Section	Total Cases	Not Tested	Fail	Pass
Login	5	0	0	5
Register	19	0	0	19

SPRINT-2

DEFECT ANALYSIS

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

Resolution	Severity1	Severity2	Severity3	Severity4	Subtotal
By Design	1	2	2	3	8
Duplicate	1	0	1	0	2
External	2	0	0	1	3
Fixed	4	2	4	6	16
Not Reproduced	0	0	0	0	0
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	16	6	7	14	29

TEST CASE ANALYSIS

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

Section	Total Cases	Not Tested	Fail	Pass
Login	5	0	0	5
Register	19	0	0	19
Dashboard	5	0	0	5

SPRINT-3

DEFECT ANALYSIS

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

Resolution	Severity1	Severity2	Severity3	Severity4	Subtotal
By Design	1	4	2	3	10
Duplicate	1	0	1	0	2
External	2	0	0	1	3
Fixed	8	2	4	6	20
Not Reproduced	0	0	0	0	0
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	16	6	7	14	35

TEST CASE ANALYSIS

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

Section	Total Cases	Not Tested	Fail	Pass
Login	5	0	0	5
Register	19	0	0	19
Dashboard	5	0	0	5
Request blood plasma	6	0	0	6

SPRINT-4

DEFECT ANALYSIS

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

Resolution	Severity1	Severity2	Severity3	Severity4	Subtotal
By Design	1	4	2	3	10
Duplicate	1	0	1	0	2
External	2	0	0	1	3
Fixed	12	2	4	10	28
Not Reproduced	0	0	0	0	0
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	16	6	7	14	43

TEST CASE ANALYSIS

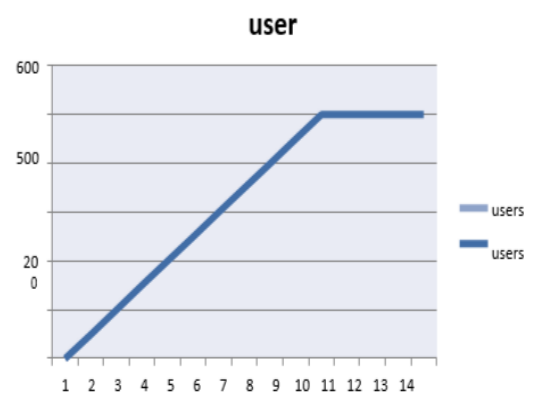
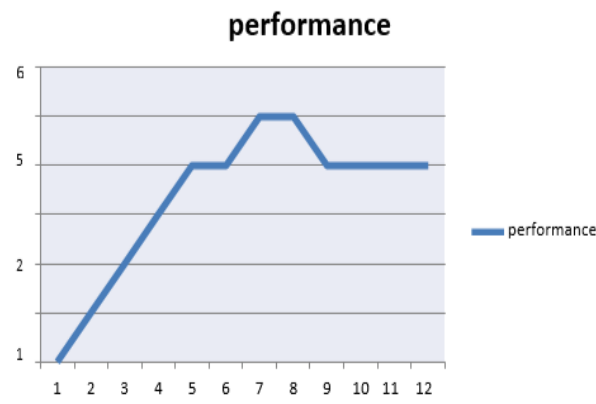
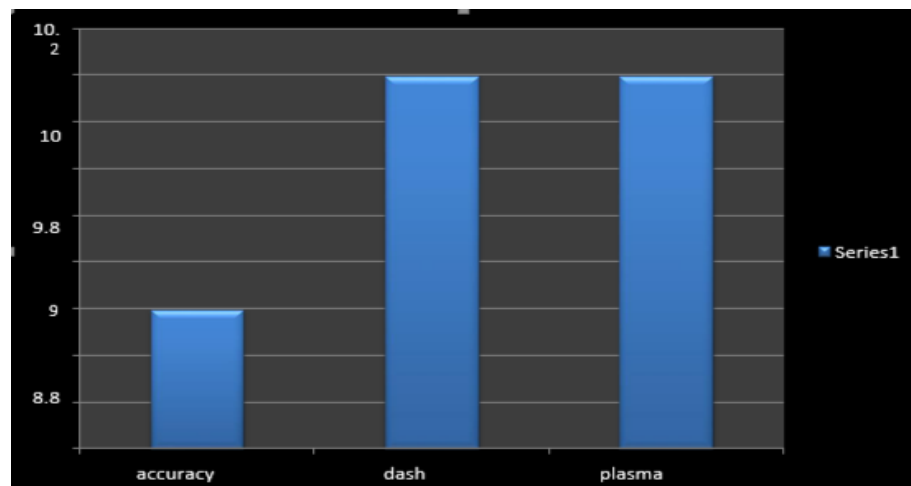
This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Login	5	0	0	5
Register	19	0	0	19
Dashboard	5	0	0	5
Request blood plasma	6	0	0	6
View request	8	0	0	8

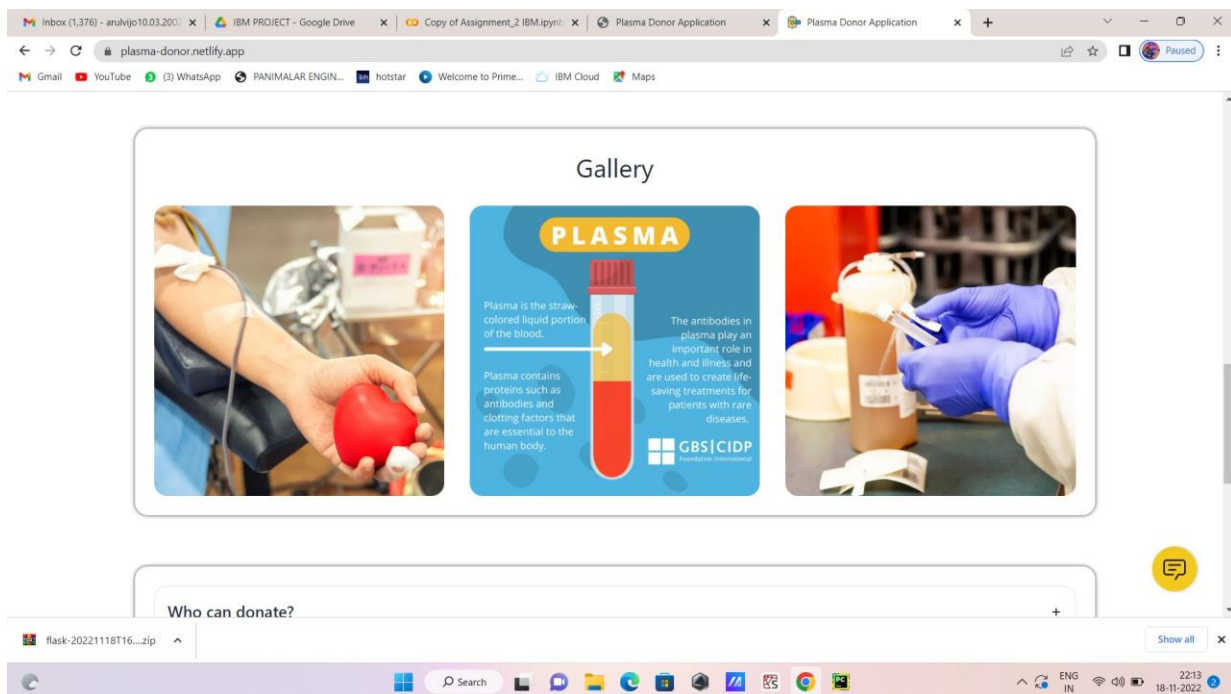
9. RESULTS

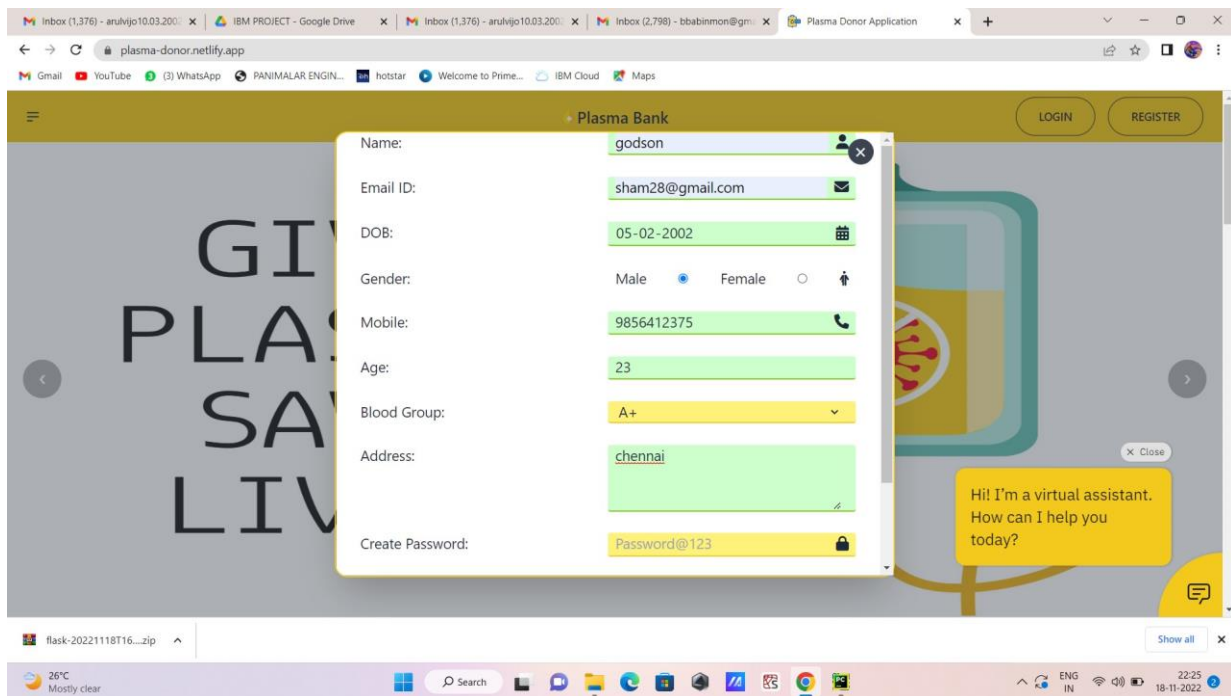
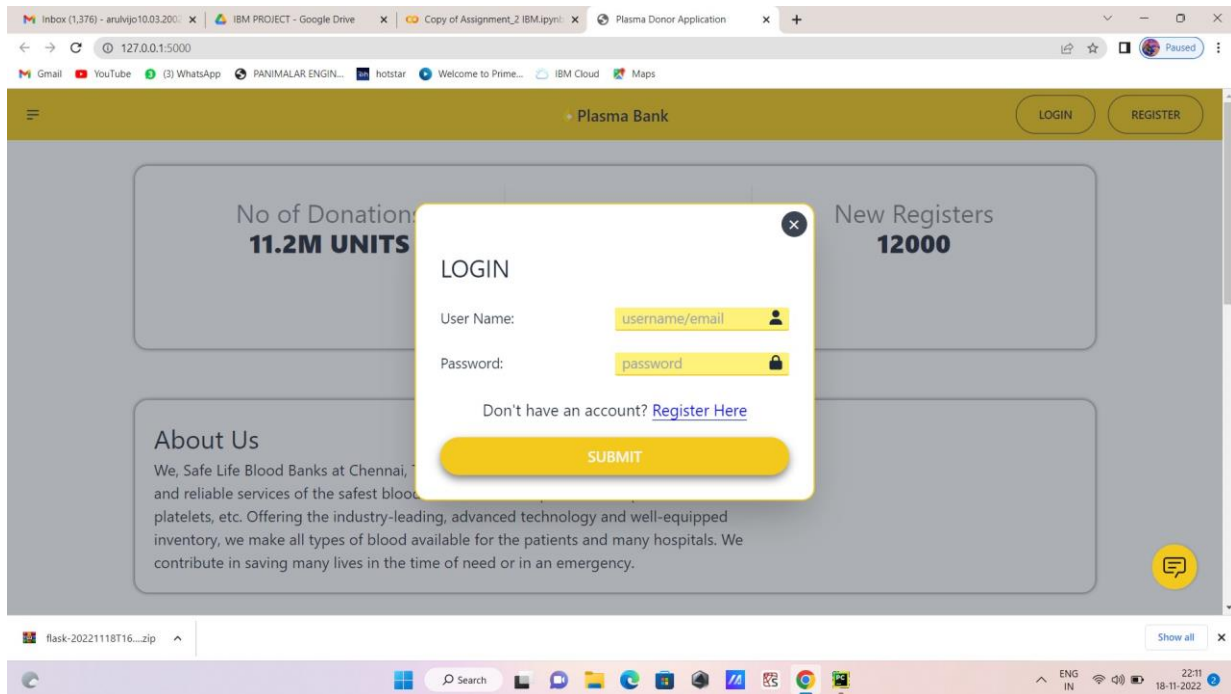
9.1 Performance Metrics

- Project metrics are used to track the progress and performance of a project.
- Monitoring parts of a project like productivity, scheduling, and scope make it easier for team leaders to see what's on track.
- As a project evolves, managers need access to changing deadlines or budgets to meet their client's expectations



Output Screenshots





Inbox (1,376) - arulvijo10.03.200... x IBM PROJECT - Google Drive x Copy of Assignment_2 IBM.ipynb x Plasma Donor Application x +

127.0.0.1:5000/login

Gmail YouTube WhatsApp PANIMALAR ENGIN... hotstar Welcome to Prime... IBM Cloud Maps

Welcome: badhri
Dashboard
Overview
Home

My Dashboard

Plasma Availability

6

Application Users

12

Plasma

A+	1
A-	0
B+	1
B-	0
AB+	2
AB-	1
O+	1
O-	0

REQUEST LOG OUT

flask-20221118T16...zip

ENG IN 22:11 18-11-2022

Inbox (1,376) - arulvijo10.03.200... x IBM PROJECT - Google Drive x Copy of Assignment_2 IBM.ipynb x Plasma Donor Application x +

127.0.0.1:5000/home

Gmail YouTube WhatsApp PANIMALAR ENGIN... hotstar Welcome to Prime... IBM Cloud Maps

Plasma Bank

LOGIN REGISTER

No of Donors 11.2M

Registers 2000

About Us
We, Safe Life Blood Banks... and reliable services of the platelets, etc. Offering the inventory, we make all types contribute in saving many lives.

ACCESS FORM

Name: godson

Email ID: 211419104032@smartinternz.com

Mobile: 9856321475

Address: chennai

Password:

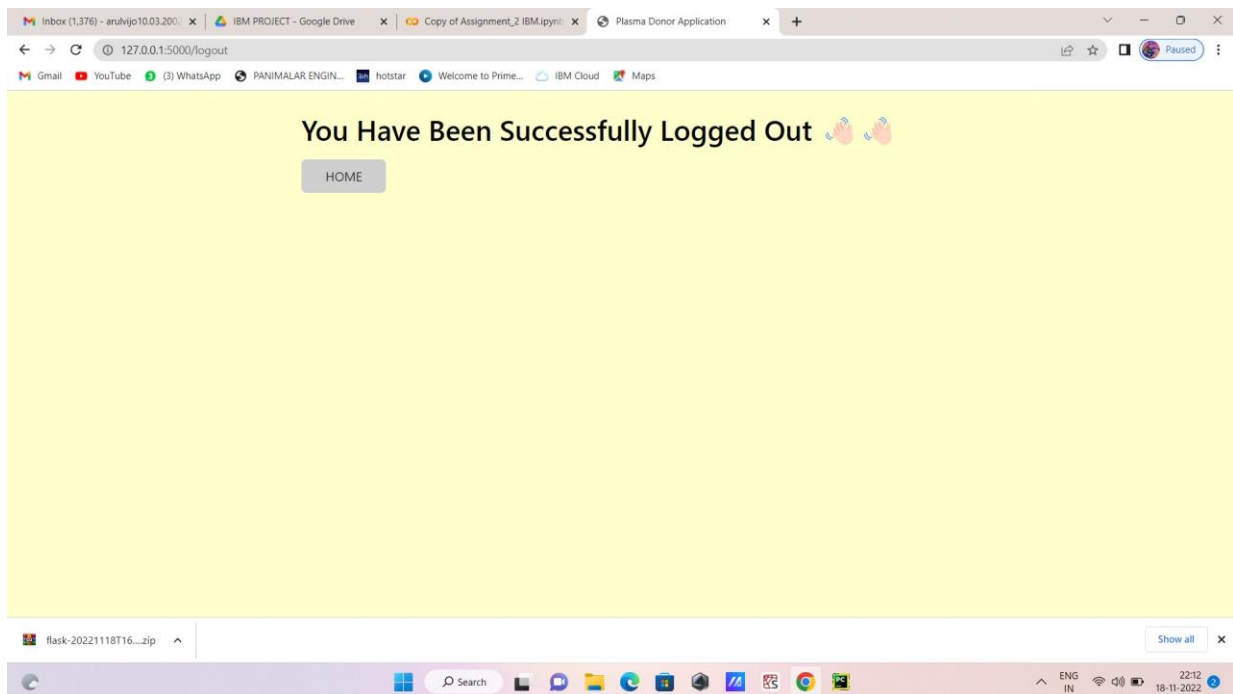
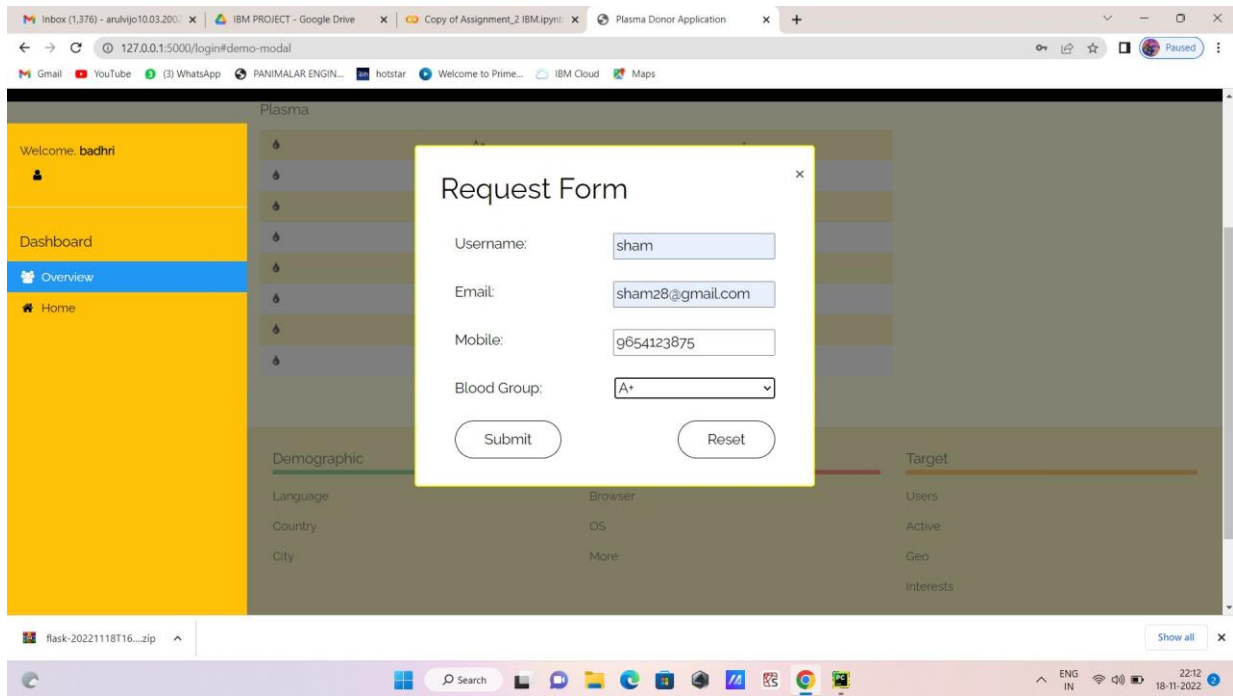
SUBMIT RESET

Close

Hi! I'm a virtual assistant. How can I help you today?

flask-20221118T16...zip

ENG IN 22:13 18-11-2022



IBM Db2 on Cloud

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

XNW03376.REQUEST

Back

Export to CSV

USERNAME	EMAIL	MOBILE	BLOOD
babin	ba@gmail.com	9632587415	AB+
babin	ba@gmail.com	9632587415	AB+
godson	bbabinmon@gmail.com	9874563210	O-
sham	hg@gmail.com	3698521475	AB-
sham	sham28@gmail.com	9654123875	A+
vijo	vijo@gmail.com	9512357846	AB-

ENG IN 10:23 PM 18-11-2022

IBM Db2 on Cloud

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

XNW03376.ACCESS

Back

Export to CSV

USERNAME	PASSWORD	MOBILE	ADDRESS	EMAIL
badhri	Badhri@123	9874563215	avadi	badhri@gmail.com

ENG IN 10:24 PM 18-11-2022

10. ADVANTAGES & DISADVANTAGES

ADVANTAGES

- > Plasma donor application is a user friendly and easy to use application.
- > It motivates many people to donate plasma which is in high demand as blood.
- > The need of blood plasma increased during covid 19 crisis, so this application meets the need of many patients.
- > The patients who are in need can register and can get blood plasma.
- > If a donor needs other blood group's plasma, he/she can login in the donor page to request for any other blood group's plasma.

DISADVANTAGES

- > It cannot auto verify user genuineness whether he/she is providing the correct information about their health history.
- > It requires an active internet connection.
- > Due to some wrong location the application will get confused

11. CONCLUSION

In recent days, it is noticed the increase in plasma request posts on social media such as Facebook, Twitter, and Instagram. Interestingly there are many people across the world interested in donating plasma when there is a need, but those donors don't have an access to know about the plasma donation requests in their local area. This is because that there is no platform to connect local plasma donors with patients. plasma solves the problem and creates a communication channel through authorized clinics whenever a patient needs plasma donation. It is a useful tool to find compatible plasma donors who can receive plasma request posts in their local area. Clinics can use this web application to maintain the plasma donation activity. Collected data through this application can be used to analyze donations to requests rates in a local area to increase the awareness of people by conducting donations camps.

Plasma Application can be developed to further improve user accessibility via integrating this application with various social networks application program interfaces (APIs).

Consequently, users can login and sign up using various social networks. This would increase number of donors and enhances the process of plasma donation.

User interface (UI) can be improved in future to accommodate global audience by supporting different languages across countries. Data scraping can be done from different social networks and can be shown in the plasma Request Feeds. Appointments can be synchronized with Google and Outlook calendars for the ease of users. Plasma application provides a reliable platform to connect local blood and plasma donors with patients. plasma creates a communication channel through authenticated clinics whenever a patient needs blood and as well as plasma donation. It is a useful tool to find compatible plasma donors who can receive plasma request posts in their local area. Clinics can use this web application to maintain the plasma donation activity. Future improvement of the plasma is explained

12. FUTURE SCOPE

The scope of a system means that which modules are being covered by the system. The scope clearly defines the boundaries of the proposed system. The functional areas of this application that lies under the scope of the proposed system are the management of the availability of donors, hospitals, plasma banks to the user or member at any time. The system calculates the estimated locations of the donors, hospitals and plasma banks and also provides online chat service between donors and consumers.

The client can also go through from the guidelines section to view the useful precautions needed before and after plasma transfusion. To be a member of the Android plasma Bank must fill the registration form and provide the necessary information.

Future iterations of this project may add more features, such as a native application for the healthcare sector or another business. It is easy to make additional enhancements to this system because of the way it was designed. The modification of the project would increase the system's adaptability. Furthermore, the functionalities are provided in a way that will improve the system's performance.

13. APPENDIX

13.1 Source Code

Login.html

```
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
<meta charset="UTF-8">
<title>IBM 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="{ { url_for('static', filename='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="username" placeholder="Enter UserName"
required="required" style="color:black" />
<input type="password" name="password" 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">
{{ msg }}</div>
</div>
</body>
</html>

```

register.html

```

<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
<meta charset="UTF-8">
<title>IBM PlasmaDonor 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="{ { url_for('static', filename='style.css') } }">
<style>
.login{ top: 20%;
}
</style>
</head>
<body>
<div class="header">
<div>Plasma Donor App</div>
<ul>
</div>
</ul>
<li><a class="active" href="/login">Home</a></li>
<div class="login">
<!-- Main Input For Receiving Query to our ML -->
<form action="{ { url_for('register') } }" method="post">
<input type="text" name="username" 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">
infection status</option>
</select>
<select name="blood"> group</option>

```

```

</select>
<option value="select" selected>Select COVID
<option value="infected">Infected</option>
<option value="uninfected">Uninfected</option>
<option value="select" selected>Choose your blood
<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>
<input type="password" name="password" 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">
{{ msg }}</div>
</div>
</body>
</html>

```

dashboard.html

```

<!DOCTYPE html>
<html>
<head>
<title>Plasma Donor Application</title>
<meta charset="UTF-8">

```

```

<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Raleway">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
<style>
html,body,h1,h2,h3,h4,h5 {font-family: "Raleway", sans-serif}
body{
  position: relative;
}
.modal {
  visibility: hidden;
  opacity: 0;
  position: absolute;
  top: 0;
  right: 0;
  bottom: 0;
  left: 0;
  display: flex;
  align-items: center;
  justify-content: center;
  background: rgba(77, 77, 77, .7);
  transition: all .4s;
}

.modal:target {
  visibility: visible;
  opacity: 1;
}

.modal__content {

```

```
border-radius: 4px;
position: relative;
width: 500px;
max-width: 90%;
background: #fff;
padding: 1em 2em;
border: 2px solid yellow;
}
```

```
/* .modal__footer {
  text-align: right;
  a {
    color: #585858;
  }
  i {
    color: #d02d2c;
  }
} */
.modal__close {
  position: absolute;
  top: 10px;
  right: 10px;
  color: #585858;
  text-decoration: none;
  font-size: 2rem;
}
```

```
.request-button{

  height: auto;
  padding: .5em 1em;
```

```
text-transform: uppercase;
background-color: transparent;
border: 1px solid #000;
border-radius: 100vmax;
}
```

```
.request-button a{
  text-decoration: none;
}
```

```
form{
  padding: 1em;
  font-size: 1.2rem;
}
```

```
.form-control {
  display: flex;
  justify-content: space-between;
  margin-bottom: 1.5em;
```

```
.form-control:last-}
child{
  margin-bottom: 0;
}
```

```
.form-control * {
  width: 100%;
}
```

```
.form-control button{
  width: auto;
```

```
padding: .5em 2em;
background-color: transparent;
border: 1px solid #000;
border-radius: 100vmax;
}
```

```
.gender-field {
  display: grid;
  grid-auto-flow: column;
}
```

```
input{
  outline: none;
}
```

```
input[type=submit]{
  background-color: transparent;
  border: none;
  margin: 0;
  padding: 0;
}
```

```
.request-container{
  display: flex;
  justify-content: center;
  align-items: center;
  gap: 2em;
  position: absolute;
  top: 1em;
  right: 2em;
}
```

```

.request-container form{
  margin: 0;
  padding: 0;
}
</style>
</head>
<body class="w3-pale-yellow">
<div class="request-container">
  <div class="request-button">
    <a href="#demo-modal">REQUEST</a>
  </div>
  <div class="request-button">
    <form action="/logout" method="POST">
      <input type="submit" value="LOG OUT">
    </form>
  </div>
</div>

<div id="demo-modal" class="modal">
  <div class="modal__content">
    <h1>Request Form</h1>
    <form action="/req" method="POST">
      <div class="form-control">
        <label for="username">Username:</label>
        <input
          type="text"
          id="username"
          name="username"
          maxlength="30"
          placeholder="username"

```



```

        required
    />
</div>
<div class="form-control">
    <label for="email">Email:</label>
    <input
        type="email"
        id="email"
        name="email"
        placeholder="email"
        required
    />
</div>
<div class="form-control">
    <label for="mobile">Mobile:</label>
    <input
        type="number"
        id="mobile"
        name="mobile"
        placeholder="mobile"
        required
    />
</div>
<div class="form-control">
    <label for="bloodgroup">Blood Group:</label>
    <select name="blood" id="bloodgroup">
        <option value="" selected disabled hidden>
            Choose your blood group
        </option>
        <option value="O+">O+</option>
        <option value="A+">A+</option>
    </select>

```

```

        <option value="B+">B+</option>
        <option value="AB+">AB+</option>
        <option value="O-">O-</option>
        <option value="A-">A-</option>
        <option value="B-">B-</option>
        <option value="AB-">AB-</option>
    </select>
</div>
<div class="form-control">
    <button type="submit">Submit</button>
    <button type="reset">Reset</button>
</div>
</form>
<a href="#" class="modal__close">&times;</a>
</div>
</div>
<!-- Top container -->
<div class="w3-bar w3-top w3-black w3-large" style="z-index:4">
    <button class="w3-bar-item w3-button w3-hide-large w3-hover-none w3-hover-text-light-grey" onclick="w3_open();"><i class="fa fa-bars"></i> Menu</button>
    <span class="w3-bar-item w3-right"></span>
</div>
<!-- Sidebar/menu -->
<nav class="w3-sidebar w3-collapse w3-amber w3-animate-left" style="z-index:3;width:300px;" id="mySidebar"><br>
    <div class="w3-container w3-row">
        <div class="w3-col s8 w3-bar">
            <span>Welcome, <strong>{ {un} }</strong></span><br>
            <a href="#" class="w3-bar-item w3-button"><i class="fa fa-user"></i></a>
        </div>
    </div>
</div>

```

```

<hr>
<div class="w3-container">
  <h5>Dashboard</h5>
</div>
<div class="w3-bar-block">
  <a href="#" class="w3-bar-item w3-button w3-padding-16 w3-hide-large w3-
dark-grey w3-hover-black" onclick="w3_close()" title="close menu"><i class="fa
fa-remove fa-fw"></i> Close Menu</a>
  <a href="#" class="w3-bar-item w3-button w3-padding w3-blue"><i class="fa
fa-users fa-fw"></i> Overview</a>
  <a href="/index" class="w3-bar-item w3-button w3-padding"><i class="fa fa-
home fa-fw"></i> Home </a>
</div>
</nav>
<!-- Overlay effect when opening sidebar on small screens -->
<div class="w3-overlay w3-hide-large w3-animate-opacity" onclick="w3_close()"
style="cursor:pointer" title="close side menu" id="myOverlay"></div>
<!-- !PAGE CONTENT! -->
<div class="w3-main" style="margin-left:300px;margin-top:43px;">
  <!-- Header -->
  <header class="w3-container" style="padding-top:22px">
    <h5><b><i class="fa fa-dashboard"></i> My Dashboard</b></h5>
  </header>
  <div class="w3-row-padding w3-margin-bottom">
    <div class="w3-quarter">
      <div class="w3-container w3-amber w3-padding-16">
        <div class="w3-left"><i class="fa fa-tint w3-xxlarge"></i></div>
        <div class="w3-right">
          <h3>{ { dicrow } }</h3>
        </div>
      <div class="w3-clear"></div>

```

```

    <h4>Plasma Availability</h4>
  </div>
</div>
<div class="w3-quarter">
  <div class="w3-container w3-orange w3-text-white w3-padding-16">
    <div class="w3-left"><i class="fa fa-users w3-xxlarge"></i></div>
    <div class="w3-right">
      <h3>{ { tot} }</h3>
    </div>
    <div class="w3-clear"></div>
    <h4>Application Users</h4>
  </div>
</div>
</div>
<div class="w3-panel">
  <div class="w3-twothird">
    <h5>Plasma </h5>
    <table class="w3-table w3-striped w3-khaki">
      <tr>
        <td><i class="fa fa-tint"></i></td>
        <td><b>A+</b></td>
        <td><i>1</i></td>
      </tr>
      <tr>
        <td><i class="fa fa-tint"></i></td>
        <td><b>A-</b></td>
        <td><i>0</i></td>
      </tr>
      <tr>
        <td><i class="fa fa-tint"></i></td>
        <td><b>B+</b></td>

```

```

        <td><i>1</i></td>
    </tr>
    <tr>
        <td><i class="fa fa-tint"></i></td>
        <td><b>B-</td>
        <td><i>0</i></td>
    </tr>
    <tr>
        <td><i class="fa fa-tint"></i></td>
        <td><b>AB+</td>
        <td><i>2</i></td>
    </tr>
    <tr>
        <td><span class="fa fa-tint"></span></td>
        <td><b>AB-</td>
        <td><i>1</i></td>
    </tr>
    <tr>
        <td><i class="fa fa-tint"></i></td>
        <td><b>O+</td>
        <td><i>1</i></td>
    </tr>
    <tr>
        <td><i class="fa fa-tint"></i></td>
        <td><b>O-</td>
        <td><i>0</i></td>
    </tr>
</table>
</div>
</div>
</div>

```

```

<hr>
<br>
<div class="w3-container w3-khaki w3-padding-16 w3-margin-
top=100px"style="margin-left:300px;">
  <div class="w3-row">
    <div class="w3-container w3-third">
      <h5 class="w3-bottombar w3-border-green">Demographic</h5>
      <p>Language</p>
      <p>Country</p>
      <p>City</p>
    </div>
    <div class="w3-container w3-third">
      <h5 class="w3-bottombar w3-border-red">System</h5>
      <p>Browser</p>
      <p>OS</p>
      <p>More</p>
    </div>
    <div class="w3-container w3-third">
      <h5 class="w3-bottombar w3-border-orange">Target</h5>
      <p>Users</p>
      <p>Active</p>
      <p>Geo</p>
      <p>Interests</p>
    </div>
  </div>
</div>
<!-- Footer -->
<footer class="w3-container w3-padding-16 w3-pale-yellow">
  <h4>FOOTER</h4>
  <p>Powered by <a href="https://www.w3schools.com/w3css/default.asp"
target="_blank">w3.css</a></p>

```

```

</footer>
<!-- End page content -->
</div>
</body>
</html>

```

app.py

```

from turtle import st
from flask import Flask, render_template, request, redirect, url_for, session, jsonify
from markupsafe import escape
import os
from dotenv import load_dotenv
from sendgrid import sendgrid
from sendgrid.helpers.mail import Mail, Email, To, Content
import ibm_db
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud;PORT=32286;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=xnw03376;PWD=CAip5VvCHctcirIL","")

app = Flask(__name__)
load_dotenv() #load keys from .env
sg = sendgrid.SendGridAPIClient(api_key = os.environ.get('GmWQl2hzTKeUsU776_Xkyw')) #set SendGrid API Key
from_email = Email("babinmon2001@gmail.com")
@app.route('/')
def home():
    sql = "select * from register where age=21 "
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.execute(stmt)

```

```

dicrow = ibm_db.fetch_assoc(stmt)
print(dicrow)
return render_template('index.html')
@app.route('/index')
def index():
    return render_template('index.html')
@app.route('/logout',methods=['POST','GET'])
def logout():
    if request.method == 'POST':
        return render_template('logout.html')
    else:
        return render_template('dash.html')
@app.route('/home',methods=['POST','GET'])
def homepage():
    if request.method == 'POST':
        return render_template('index.html')
    else:
        return render_template('logout.html')
@app.route('/req',methods=['POST','GET'])
def req():
    if request.method == 'POST':
        username = request.form['username']
        email = request.form['email']
        phone = request.form['mobile']
        blood=request.form['blood']
        insert_sql = "INSERT INTO request VALUES (?,?,,?)"
        prep_stmt = ibm_db.prepare(conn, insert_sql)
        ibm_db.bind_param(prepare_stmt, 1, username)
        ibm_db.bind_param(prepare_stmt, 2, email)
        ibm_db.bind_param(prepare_stmt, 3, phone)
        ibm_db.bind_param(prepare_stmt, 4, blood)

```



```

    ibm_db.execute(prepare_stmt)
    return render_template('index.html',msg="saved successfully")
@app.route('/access',methods=['POST','GET'])
def access():
    if request.method == 'POST':
        username = request.form['username']
        email = request.form['email']
        phone = request.form['mobile']
        address=request.form['address']
        password=request.form['password']
        insert_sql = "INSERT INTO access VALUES (?, ?, ?, ?, ?)"
        prepare_stmt = ibm_db.prepare(conn, insert_sql)
        ibm_db.bind_param(prepare_stmt, 1, username)
        ibm_db.bind_param(prepare_stmt, 5, email)
        ibm_db.bind_param(prepare_stmt, 3, phone)
        ibm_db.bind_param(prepare_stmt, 4, address)
        ibm_db.bind_param(prepare_stmt, 2, password)
        ibm_db.execute(prepare_stmt)
        return render_template('index.html',msg="saved successfully")
@app.route('/reg',methods = ['POST', 'GET'])
def reg():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        email = request.form['email']
        blood=request.form['blood-group']
        gender=request.form['gender']
        age=request.form['age']
        address=request.form['address']
        phone=request.form['mobile']
        insert_sql = "INSERT INTO register VALUES (?, ?, ?, ?, ?, ?, ?, ?)"

```

```

prep_stmt = ibm_db.prepare(conn, insert_sql)
ibm_db.bind_param(prepare_stmt, 1, username)
ibm_db.bind_param(prepare_stmt, 2, password)
ibm_db.bind_param(prepare_stmt, 3, email)
ibm_db.bind_param(prepare_stmt, 4, blood)
ibm_db.bind_param(prepare_stmt, 5, gender)
ibm_db.bind_param(prepare_stmt, 6, age)
ibm_db.bind_param(prepare_stmt, 7, address)
ibm_db.bind_param(prepare_stmt, 8, phone)
ibm_db.execute(prepare_stmt)
to_email = To(email) # set user as recipient for confirmation email
subject = "Welcome to GetPlasma"
content = Content("text/html",
    "<p>Hello " + username + ",</p><p>Thank you for registering to the
GetPlasma Application!</p><p>If this wasn't you, then immediately report to our
<a href=\"mailto:getplasmaproject@gmail.com\">admin</a> or just reply to this
email.</p>")
email = Mail(from_email, to_email, subject, content) # construct email format
email_json = email.get() # get JSON-ready representation of the mail object
return render_template('index.html',msg="saved successfully")
@app.route('/login',methods = ['POST', 'GET'])
def login():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        sql = "select * from access where username=? and password=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, username)
        ibm_db.bind_param(stmt, 2, password)
        ibm_db.execute(stmt)
        dic = ibm_db.fetch_assoc(stmt)

```

```

print(dic)
role = str()
requests = []
if dic:
    un=dic['USERNAME']
    print(un)
    sql = "select count(*) from register"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.execute(stmt)
    dicrow = ibm_db.fetch_assoc(stmt)
    sql1 = "select count(*) from request"
    stmt1 = ibm_db.prepare(conn, sql1)
    ibm_db.execute(stmt1)
    dicrow1 = ibm_db.fetch_assoc(stmt1)
    sql2 = "select count(*) from access"
    stmt2 = ibm_db.prepare(conn, sql2)
    ibm_db.execute(stmt2)
    dicrow2 = ibm_db.fetch_assoc(stmt2)
    tot=(dicrow['1']+dicrow1['1']+dicrow2['1'])
    bloodA = ibm_db.fetch_assoc(stmt1)
    print(bloodA)
    return render_template('dash.html',un=un,dicrow=dicrow['1'],tot=tot)
else:
    return render_template('index.html')
return render_template('index.html')

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

```

13.2 GITHUB LINK

<https://github.com/IBM-EPBL/IBM-Project-3900-1658670225>

Project Demo Link

<https://vimeo.com/772802569>