PLASMA DONOR APPLICATION CLOUD TECHNOLOGIES USING IBM

A Project report submitted in partial fulfillment of the requirements of the award of the degree of

Bachelor of Engineering

In

Computer Science and Engineering

By

DEEPA.S **(Reg.No.410119104007)**

MARIPRIYA .S **(Reg. No. 410119104030)**

SARATHA DEVI .G **(Reg.no.410119104048)**

SOWMIYA .S **(Reg. No. 410119104053)**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ADHI COLLEGE OF ENGINEERING AND TECHNOLOGY,

(Accredited by NACC.Affiliated by ANNA UNIVERSITY,chennai&Apprised by AICTE.)

KANCHIPURAM(Dt). CHENNAI- 600 025

2022.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ADHI COLLEGE OF ENGINEERING AND TECHNOLOGY,

CERTIFICATE

This is to certify that the project report titled"PLASMA DONOR APPLICATION" being submitted by

DEEPA.S	**(Reg.No.410119104007)**
MARIPRIYA .S	**(Reg. No. 410119104030)**
SARATHA DEVI .G	**(Reg.no.410119104048)**
SOWMIYA .S	**(Reg. No. 410119104053)**

in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering, to the Anna University, Chennai is a record of bonafied work carried out by them my guidance and supervision.

Faculty Mentor

Mrs. K..Banu Priya

Dept. of IBM

Industry Mentor

Dept. of IBM

TABLE OF CONTENT

1.INTRODUCTION

1.1. Project Overview

The Blood Donation Agent is to create an e-Information about the donor and organization that are related to donating the blood. Through this application any person who is interested in donating the blood can register himself in the same way if any organization wants to register itself with this site that can also register. Moreover if any general consumer wants to make request blood online he can also take the help of this site. Admin is the main authority who can do addition, deletion, and modification if required.

1.2. Project Description

This project is aimed to developing an online Blood Donation Information. The entire project has been developed keeping in view of the distributed client server computing technology, in mind. The Blood Donation Agent is to create an e-Information about the donor and organization that are related to donating the blood. Through this application any person who is interested in donating the blood can register himself in the same way if any organization wants to register itself with this site that can also register. Moreover if any general consumer wants to make request blood online he can also take the help of this site. Admin is the main authority who can do addition, deletion, and modification if required.The project has been planned to be having the view of distributed architecture, with centralized storage of the database. The application for the storage of the data has been planned. Using the constructs of MY-SQL Server and all the user interfaces have been designed using the ASP.Net technologies.The database connectivity is planned using the "SQL Connection" methodology. The application takes care of different modules and their associated reports, which are produced as per the applicable strategies and standards that are put forwarded by the administrative staff.

The entire project has been developed keeping in view of the distributed client server computing technology, in mind. The specification has been normalized up to 3NF to eliminate all the anomalies that may arise due to the database transaction that are executed by the general users and the organizational administration. The user interfaces are browser specific to give distributed accessibility for the overall system. The internal database has been selected as MY-SQL server 2000.

The basic constructs of table spaces, clusters and indexes have been exploited to provide higher consistency and reliability for the data storage. The MY-SQL server 2000 was a choice as it provides the constructs of high-level reliability and security. The total front end was dominated using the ASP.Net technologies. At all proper levels high care was taken to check that the system manages the data consistency with proper business rules or validations.The database connectivity was planned using the latest "SQL Connection" technology provided by Microsoft Corporation. The authentication and authorization was crosschecked at all the relevant stages. The user level accessibility has been restricted into two zones namely.

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 under development.
- Less Security.
- No proper coordination between different Applications and Users.
- Fewer Users – Friendly

2.2.References

1.Blood safety and availability, Fact Sheet July 2016, World Health Organization. Available online: http://www.who.int/mediacentre/factsheets/fs279/en/

2. Blood Facts and Statistics, American Red Cross, 2016. Available online: http://www.redcrossblood.org/learn-about-blood/blood-facts-and-statistics

3. Blood, organ and tissue donation -The need of blood donation in Canada, Available online: http://healthycanadians.gc.ca/diseases-conditions-maladies-affections/donation-contribution-eng.php

4.Blood App American Red Cross, 2016. Available online: http://www.redcrossblood.org/bloodapp

5.Blood Donor Finder, 2016 Google. Available online: https://play.google.com/store/apps/details?id=com.Neologix.BloodDonorFinder&hl=en

6. Rick Borup. An Introduction to Ruby and Rails. Southwest Fox conference in Gilbert, Arizona in October, 2010:1-50.

7. Devise by Plataformatec, 2017 GitHub, Inc., available online: https://github.com/plataformatec/devise

8. public_activity, 2017 GitHub, Inc., available online: https://github.com/chaps-io/public_activity#first-time-setup

9. jquery.timepicker, 2017 GitHub, Inc., available online: https://github.com/cover/jquery-timepicker-rails

10. Therubyracer. Available online: http://www.rubydoc.info/gems/therubyrace

11. Wiki ruby-pg. Available online: https://bitbucket.org/ged/ruby-pg/wiki/Home

12. Nifty Generators, 2017 GitHub, Inc., available online: https://github.com/ryanb/nifty-generators

13. Coffee-Rails. Available online: http://www.rubydoc.info/gems/coffee-rails/4.2.1

14. Rack, a modular Ruby webserver interface, 2017 GitHub, Inc., available online: https://github.com/rack/rack

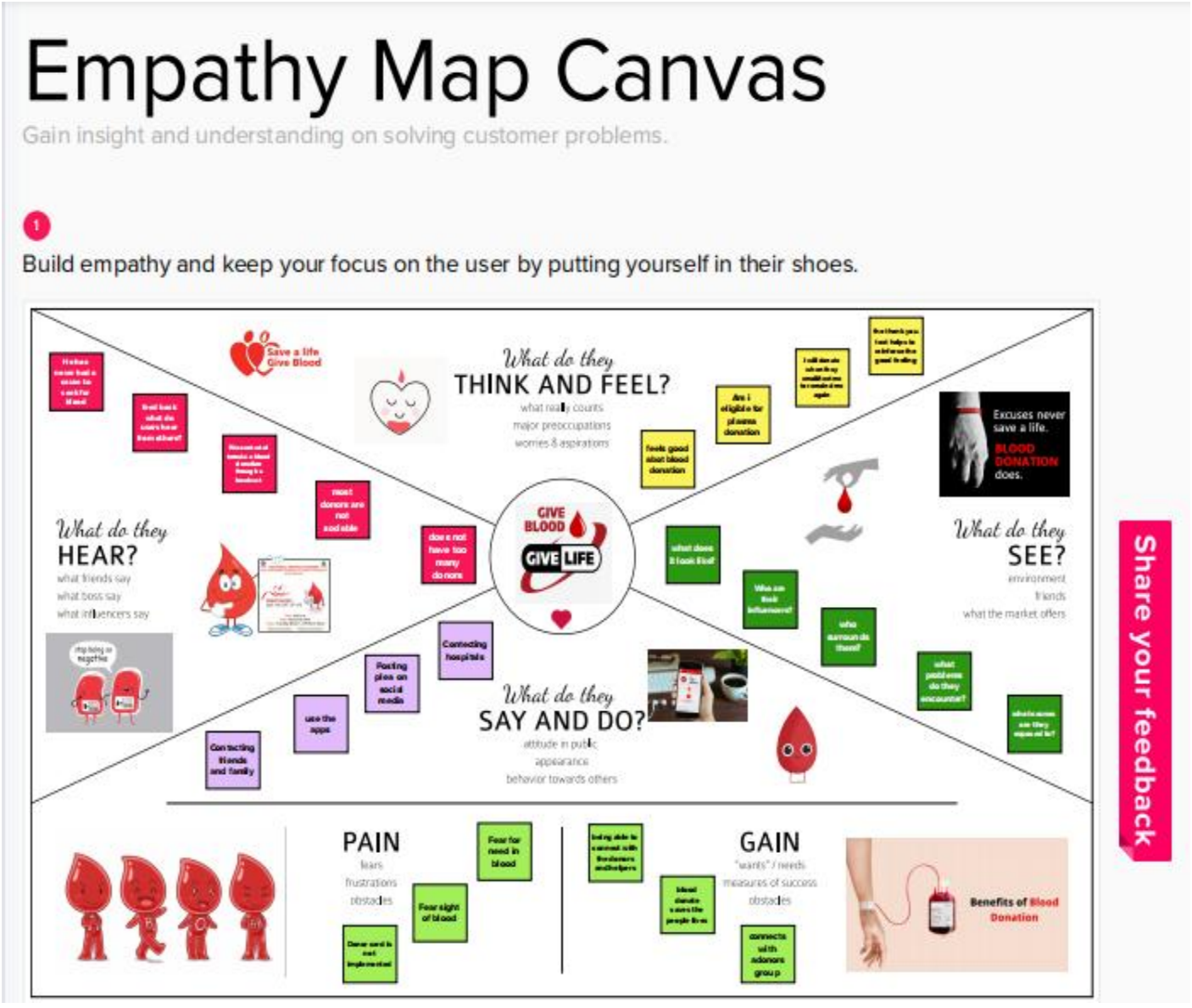
15. RAKE – Ruby Make. Available online: http://docs.seattlerb.org/rake/

2.3.Problem Statement Definition

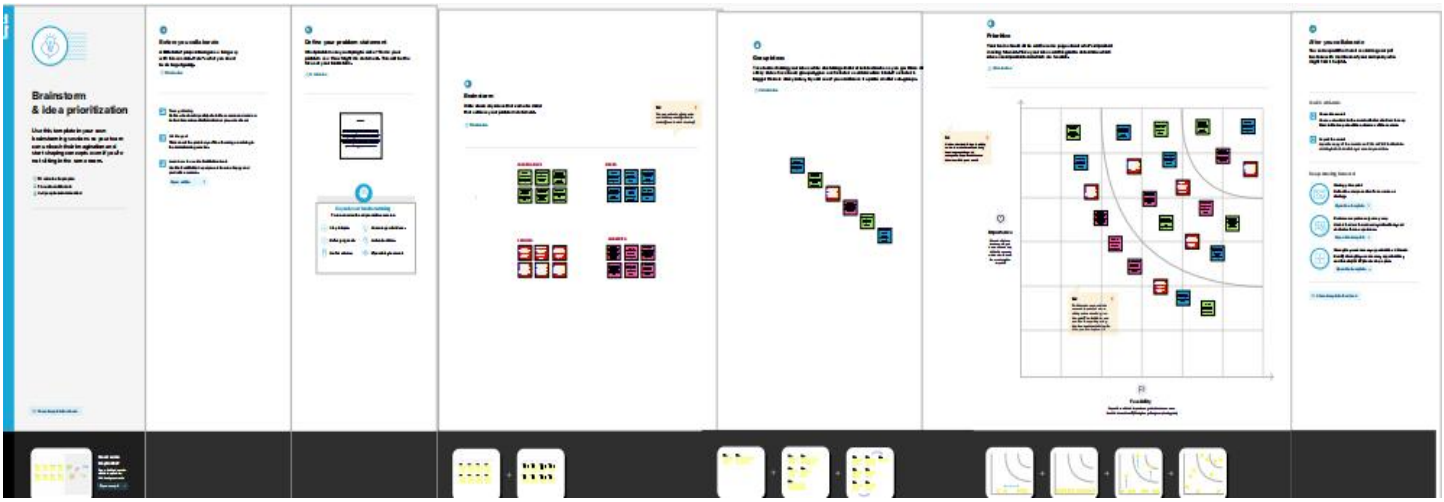
To debug the existing system, remove procedures those cause data redundancy, make navigational sequence proper. To provide information about audits on different level and also to reflect the current work status depending on organization/auditor or date. To build strong password mechanism.

3.IDEATION & PROPOSED SOLUTION

3.1.Empathy Map Canvas



3.2.Ideation & Brainstorming



3.3.Proposed Solution

S.No.	Parameter	Description
1	**Problem Statement (Problem to be solved)**	The corona virus disease 2019(COVID-19) affected millions of people worldwide and caused disruptions at the global level including in healthcare provision. N tries of the WHO African region have put in place measures for the COVID-19 pandemic containment that may adversely affect blood system activities and subsequently reduce the supply and demand of blood and blood components. This study aims to assess the impact of the COVID-19 pandemic on blood supply and demand in the WHO African Region and propose measures to address the challenges faced by countries.
2	**Idea/Solution description**	If anyone needs a Plasma Donor, this system comprises of Admin and User where both can request for a Plasma. In this system there is something called an active user, which means the user is an Active member of the App and has recovered from Covid 19,

		only such people are recommended here for Plasma Donation. Both parties can Accept or Reject the request. User has to Upload a Covid Negative report to be able to Donate Plasma.
3	**Novelty/Uniqueness**	This system proposed here aims at connecting the donors & the patients by an online application. By using this application, the users can eitherraise a request for plasma donation or requirement.
4	**Social impact/Customer Satisfaction**	The COVID-19 pandemic presents significant differences as it is an ongoing crisis that affects all individuals and has the potential to pose a direct health threat to anyone. Therefore, we propose that the pandemic may also negatively affect willingness to help, specifically blood donation intentions. It requires a high level of willingness to donate blood beyond the crisis outbreak, as more blood will be needed when postponed surgeries resume.While active donors show increased awareness of ability and eligibility to donate at the beginning of the pandemic compared to pre-pandemic, they feel significantly less able to donate as the pandemic progresses. Furthermore, inactive donors' perceived ability to donate significantly decreases in the pandemic phase compared to the pre-pandemic phase. Crucially, both active and inactive donors feel less responsible and less morally obliged to donate, resulting in an overall negative pandemic effect on blood donation intentions. The COVID-19 pandemic compromises blood donations endangering the life- saving blood supply.
5	**Business Modal(Revenue Model)**	**T**he plasma industry uses plasma to produce various blood products, Inclding clotting factors,intravenous immuno globulin, and albumin. Globally, the plasma industry is a big business. Companies that specialise in plasma have been regularly subjected to takeovers, and the industry is growing. It is generally accepted (and recommended by WHO) that the safest blood supplies

		are those freely given by carefully screened donors who have no vested interest.
6	**Scalability of the Solution**	Several blood collection organizations have explored the useof both behav-ioral nudges and economic incentives, with varying degrees of suc-cess. Economic incentives are often perceived as being in conflict with the core mission of blood banks, with donors' altruistic motivation, and, more generally, with moral values.Several blood collection organizations have explored the use of both behav-ioral nudges and economic incentives, with varying degrees of suc-cess. Economic incentives are often perceived as being in conflict with the core mission of blood banks, with donors' altruistic motivation, and, more generally, with moral values.

3.4.Problem Solution fit

Customer Problem Statement Template:

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.

I am	Describe customer with 3-4 key characteristics - *who are they?*	Describe the customer and their attributes here
I'm trying to	*List their outcome or "job" the care about - what are they trying to achieve?*	List the thing they are trying to achieve here
but	Describe what problems or barriers stand in the way – *what bothers them most?*	Describe the problems or barriers that get in the way here
because	Enter the "root cause" of why the problem or barrier exists – *what needs to be solved?*	Describe the reason the problems or barriers exist
which makes me feel	Describe the emotions from the customer's point of view – *how does it impact them emotionally?*	Describe the emotions the result from experiencing the problems or barriers

Example 1:

Example 2:

Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	a plasma donor	donate a plasma through mobile app	I can't donate a plasma immediately	I don't know the address and proper message from plasma acceptor & the COVID-19 pandemic lot of issues in the blood activities.	Guilty & helpless
PS-2	a plasma acceptor	get a plasma through mobile app	plasma doesn't get immediately	the COVID-19 pandemic reduce the supply and demand of plasma	fear

4.REQUIREMENT ANALYSIS

4.1.Functional requirement

Following are the functional requirements of the proposed solution.

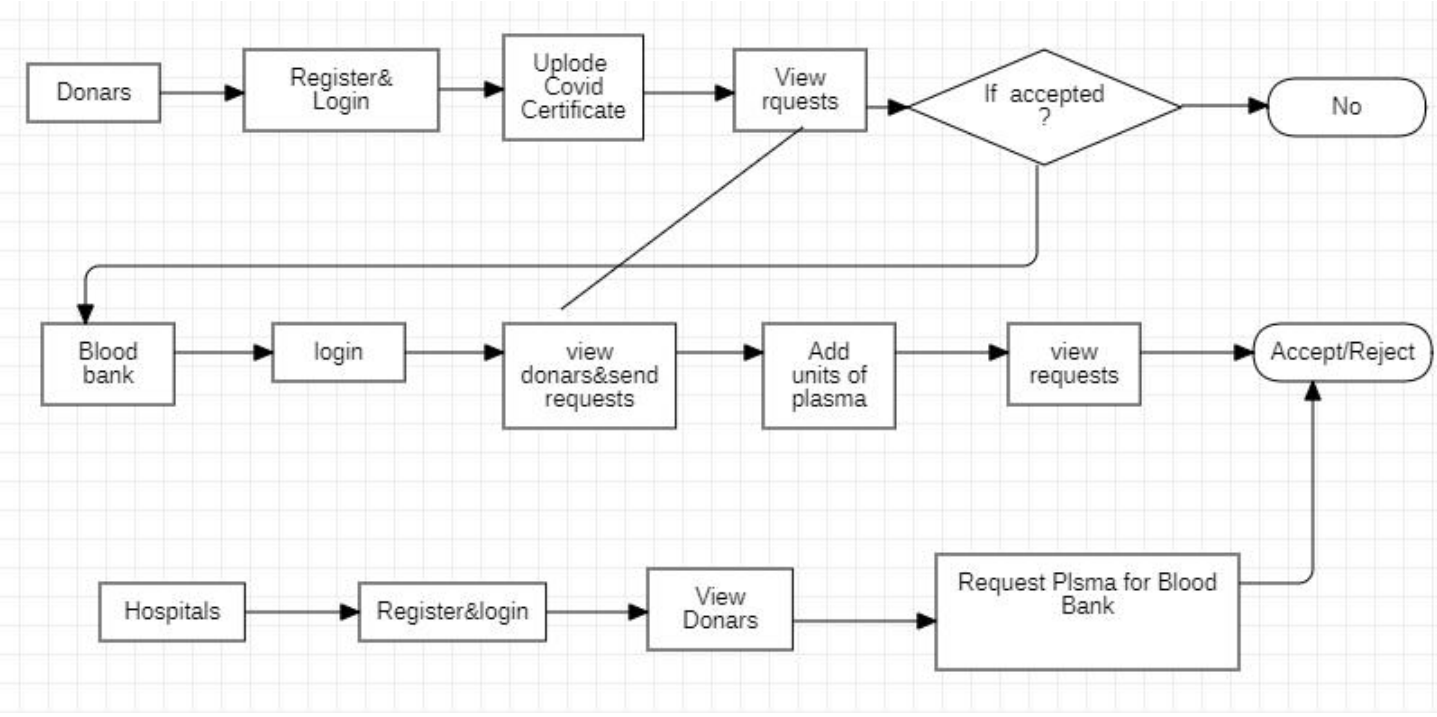
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through mobile number
FR-2	User Confirmation	Confirmation via mobile number
FR-3	Covid-19 certificate	Upload covid-19 through Form
FR-4	Blood group	Enter the blood group through form

4.2.Non-Functional requirement

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Donate the blood during emergency period
NFR-2	**Security**	Donor details should save in database
NFR-3	**Reliability**	User has to Upload a Covid Negative report to be able to Donate Plasma.
NFR-4	**Performance**	Donate the blood at correct time
NFR-5	**Availability**	All kind of blood available
NFR-6	**Scalability**	results of laboratory experiments to be applied to larger natural or artificial plasmas of interest

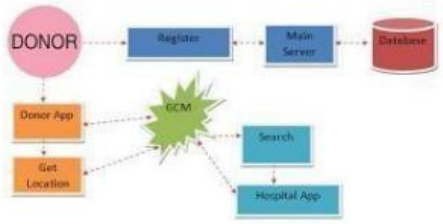
4.3.PROJECT DESIGN

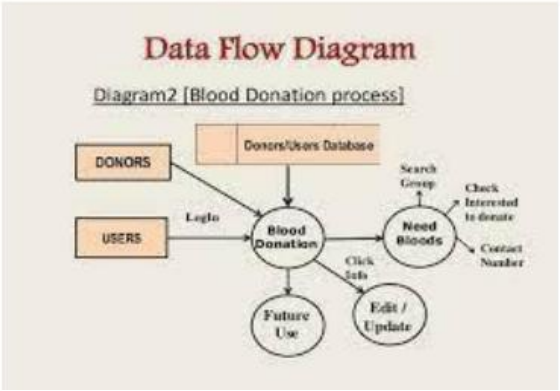


4.4.Data Flow Diagrams

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

Example:



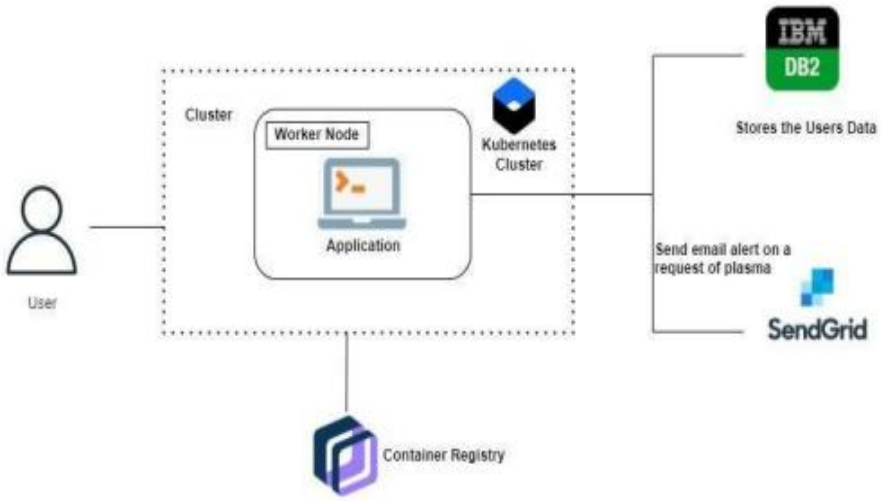


Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Installation	USN-1	As a user, I installed the app for day-to-day update and feeds.	My app will be installed on home screen		
Customer (Mobile user)		USN-2	As a user, I can register for the application by entering my email, password, confirming my password and phone number.	I can access my account / dashboard	High	Sprint-1
		USN-3	As a user, I will receive conformation email once I have registered for the application	I can receive conformation email & click confirm	High	Sprint-1
		USN-4	As a user, I can register for the application through Gmail	I can register & access the dashboard with Gmail account Login	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can login to the official page	High	Sprint-1
	Dashboard	USN-6	An open-source app that helps in connecting patients and plasma donors.This is a beginner friendly repository that helps.	I can access my dashboard	High	Sprint-1
Customer (Web user)	Browsing	USN-7	Enter the web site on the browser	I can even login through browser	Medium	Sprint-1

4.5.Solution & Technical Architecture.

Technical Architecture:



TECHNOLOGY STACK:

TABLE-1 : COMPONENTS AND TECHNOLOGIES:

S.No	COMPONENTS	DESCRIPTION	TECHNOLOGY
1.	User Interface	How user interacts with application. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript, Python, Flask
2.	Register to website	The user can able to register in website and fill their details. The user details are Stored in IBM DB2 securely.	Flask app using Kubernetes cluster, IBM DB2.
3.	Login to website	The user interact with the website to login into account. The user details are verified by comparing it with details stored in IBM DB2	Flask app using Kubernetes cluster, IBM DB2.
4.	Request for Donor/Register for donating	The user interact with the website to request for plasma Donor/register for willing to donate plasma.	Flask app using Kubernetes cluster, IBM DB2.
5.	Upload proof in website	The user can able to upload the vaccination certificate and other proofs.	Container registry,
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local Filesystem
8.	External API-1 (Email Alert)	To send email alerts to donor when a person requesting Plasma Donor.	SendGrid.
9.	Machine Learning Model	Machine Learning Model can be used for Chatbot.	IBM Watson.
10.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud	Local, Cloud Foundry, Kubernetes.

TABLE-2: APPLICATION CHARACTERISTICS:

S.No	CHARACTERISTICS	DESCRIPTION	TECHNOLOGY
1.	Open-Source Frameworks	Flask is an open source framework in python. Similarly Docker is also used.	Flask , Docker
2.	Security Implementations	Only registered users who have specific privileges has access to the website.	IBM DB2
3.	Scalable Architecture	3 – tier architecture, presentation tier, application tier, data tier	Python, IBM cloud services
4.	Availability	The application can be available for user at any time.	Kubernetes, Docker
5.	Performance	The application can handle multiple requests per second.	Kubernetes cluster, IBM

5.Project Planning & scheduling

5.1.Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Initial creation process	USN-1	Create template, Static and python flask app.	20	High	Sowmiya S Deepa S Maripriya S Saratha Devi G
Sprint-2	Cloud and database	USN-2	Connecting the python flask app with database, object storage created in Cloud and implementation of chatbot	20	High	Sowmiya S Deepa S Maripriya S Saratha Devi G
Sprint-3	Deployment in DevOps, Mailing	USN-3	Develop the project, create it as image with docker, containerize in container registry and deploy in Kubernetes, Add the mailing service	20	High	Sowmiya S Saratha Devi G Deepa S Maripriya S
Sprint-4	Testing, Deployment and user experience	USN-4	To do all the testing and to make sure the use of the software handy to user.	20	High	Sowmiya S Saratha Devi G Deepa S Maripriya S

5.2.Sprint Delivery Schedule

Project Tracker:

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	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

Velocity: Sprint - 1

Sprint duration = 6 days

Velocity of the team = 20 points

average velocity (AV) =

Velocity

Sprint duration

AV = 20/6 = 3.34

Average Velocity = 3.34

Velocity: Sprint 1 - 4

Sprint duration = 24 days

Velocity of the team = 80 points

average velocity (AV) = Velocity

Sprint duration

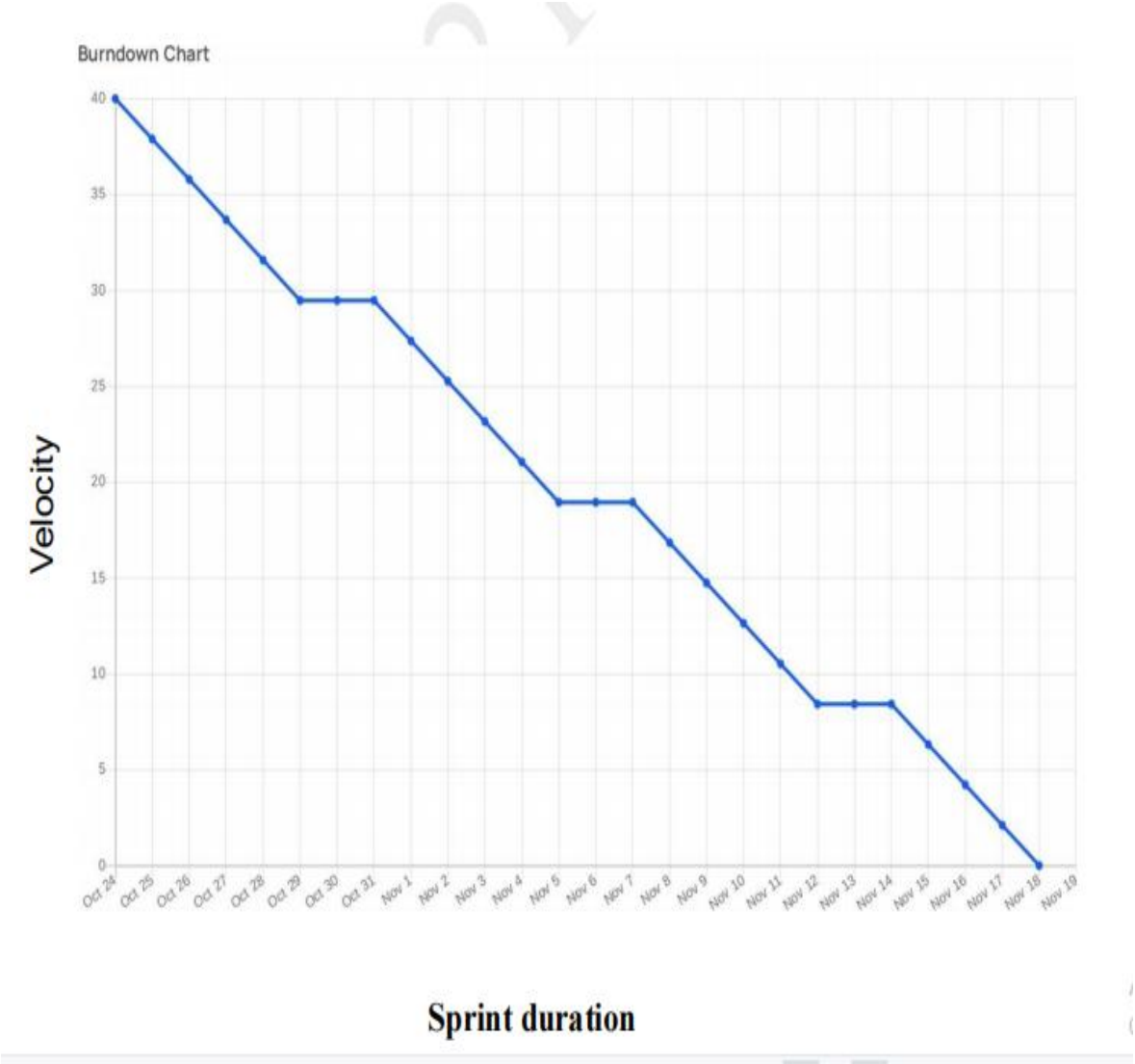
AV = 80/6 = 3.34

Total Average Velocity = 3.34

Burndown Chart:

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.

Sprint duration



6.CODING & SOLUTIONING

6.1.Feature 1

The common language runtime manages memory, thread execution, code execution, code safety verification, compilation, and other system services. These features are intrinsic to the managed code that runs on the common language runtime.With regards to security, managed components are awarded varying degrees of trust, depending on a number of factors that include their origin (such as the Internet, enterprise network, or local computer). This means that a managed component might or might not be able to perform file-access operations, registry-access operations, or other sensitive functions, even if it is being used in the same active application.The runtime enforces code access security. For example, users can trust that an executable embedded in a Web page can play an animation on screen or sing a song, but cannot access their personal data, file system, or network. The security features of the runtime thus enable legitimate Internet-deployed software to be exceptionally featuring rich.

6.2. Feature 2

The runtime also enforces code robustness by implementing a strict type- and code-verification infrastructure called the common type system (CTS). The CTS ensures that all managed code is self-describing. The various Microsoft and third-party language compilers Generate managed code that conforms to the CTS. This means that managed code can consume other managed types and instances, while strictly enforcing type fidelity and type safety.In addition, the managed environment of the runtime eliminates many common software issues. For example, the runtime automatically handles object layout and manages references to objects, releasing them when they are no longer being used. This automatic memory management resolves the two most common application errors, memory leaks and invalid memory references.The runtime also accelerates developer productivity. For example, programmers can write applications in their development language of choice, yet take full advantage of the runtime, the class library, and components written in other languages by other developers. Any compiler vendor who chooses to target the runtime can do so. Language compilers that target the .NET Framework make the features of the .NET Framework available to existing code written in that language, greatly easing the migration process for existing applications.

7.TESTING

7.1.Introductin

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive.A strategy for software testing integrates software test case design methods into a well-planned series of steps that result in the successful construction of software. Testing is the set of activities that can be planned in advance and conducted systematically. The underlying motivation of program testing is to affirm software quality with methods that can economically and effectively apply to both strategic to both large and small-scale systems.

7.2.User Acceptance Testing

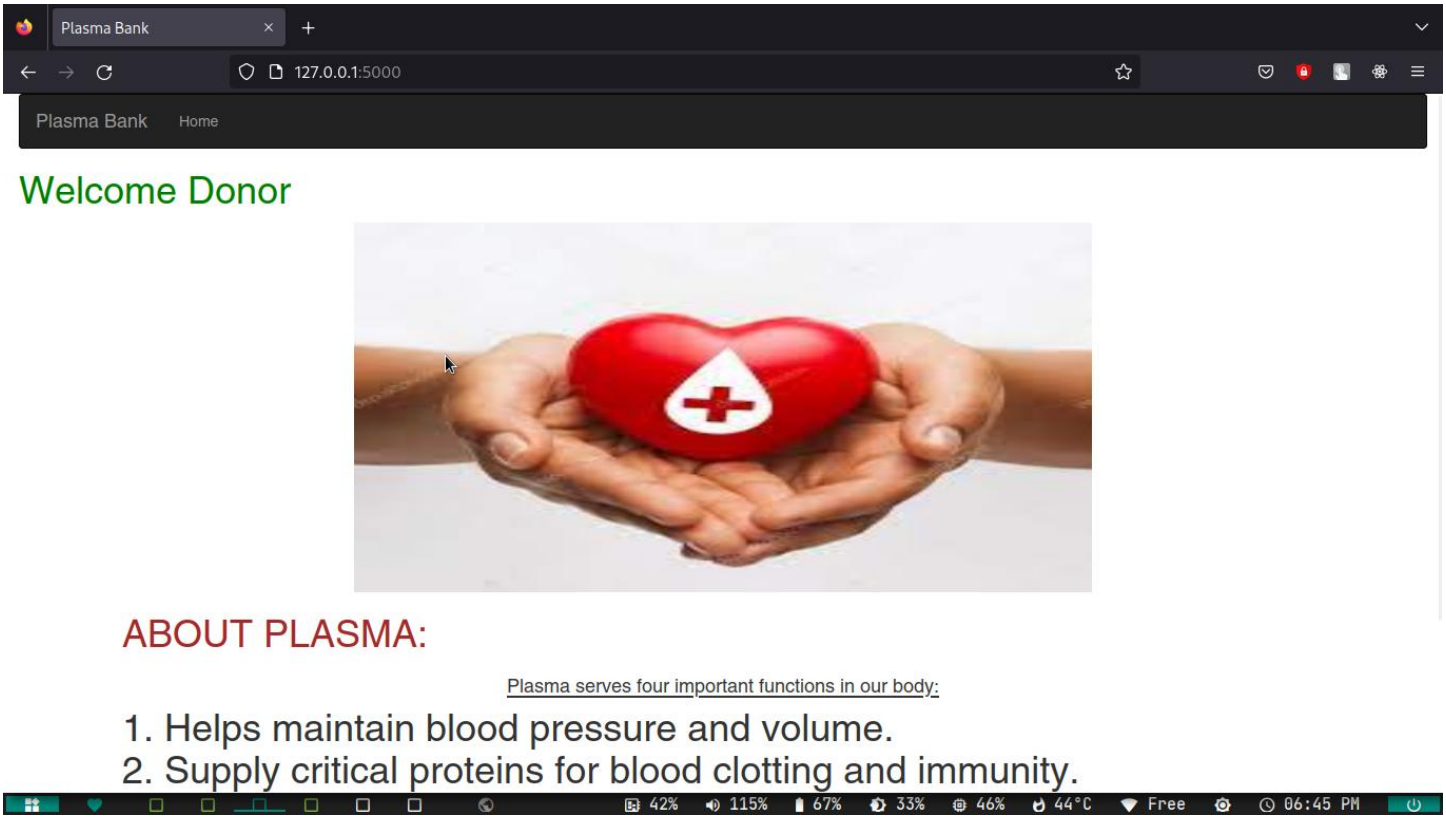
Test Case Analysis

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

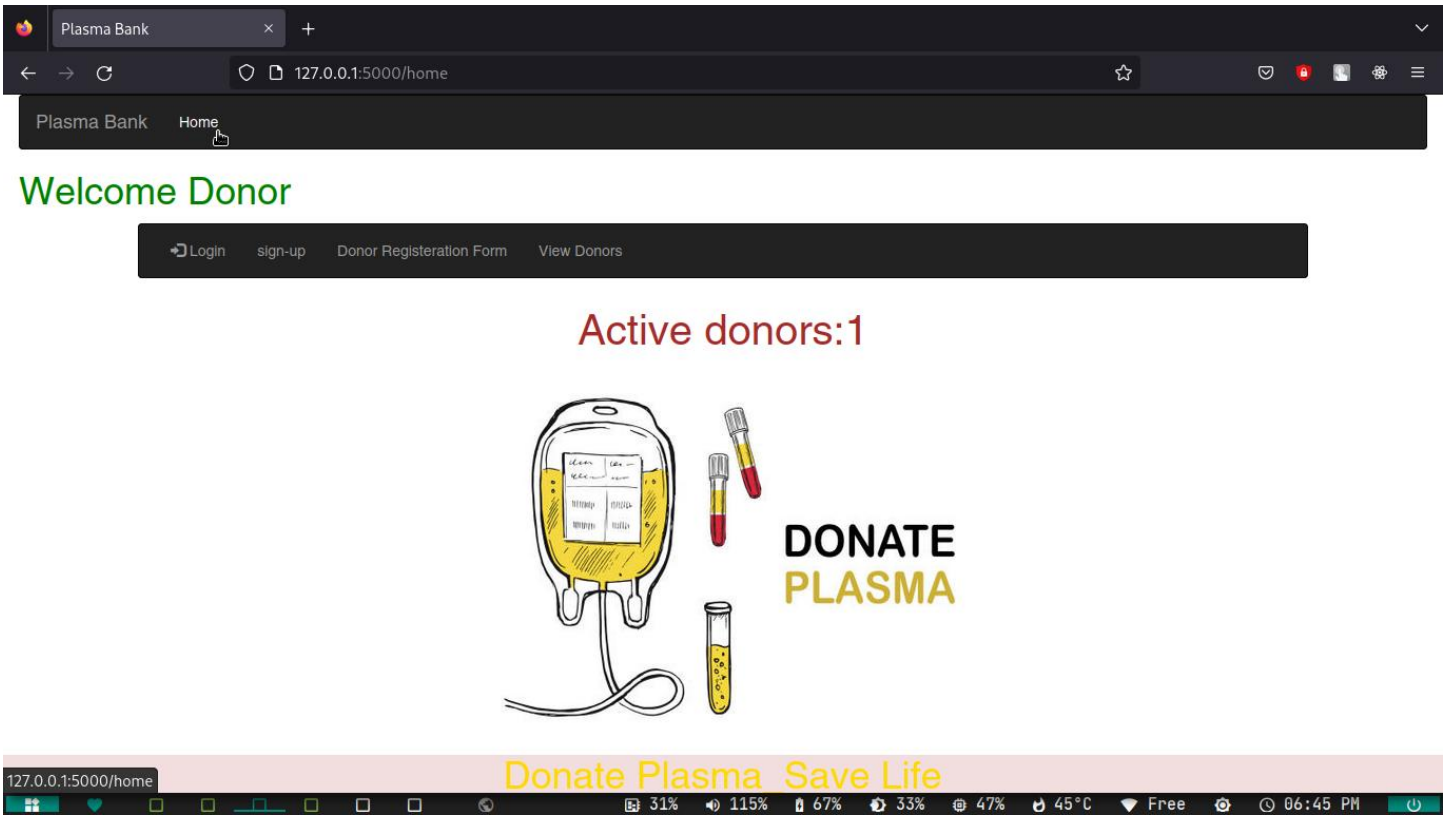
Section	Total Cases	Not Tested	Fail	Pass
Print Engine	5	0	0	5
Client Application	19	0	0	19
Security	1	0	0	1
Outsource Shipping	1	0	0	1
Exception Reporting	1	0	0	1
Final Report Output	1	0	0	1
Version Control	1	0	0	1

8. Results

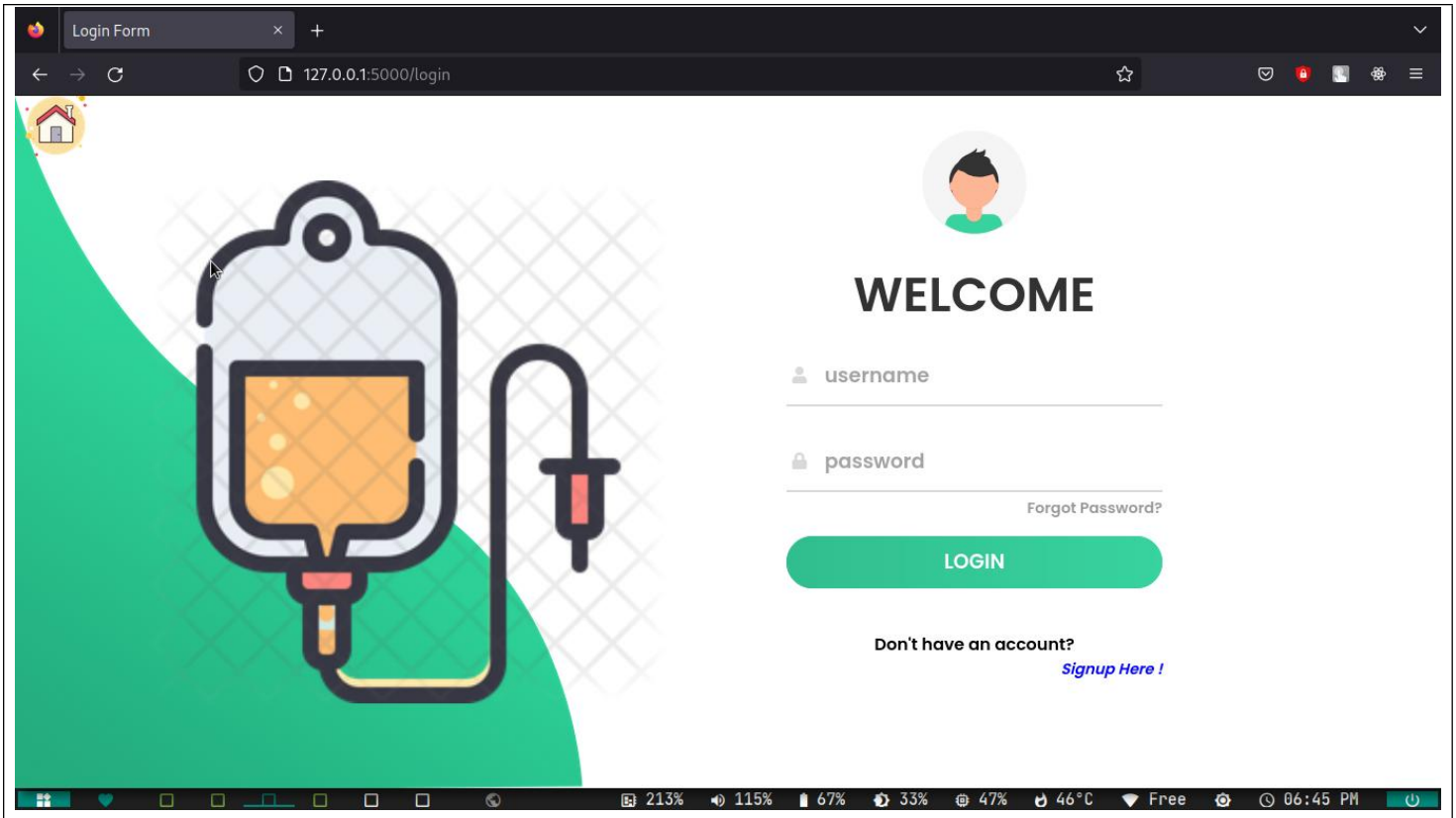
8.1.About Plasma Donor



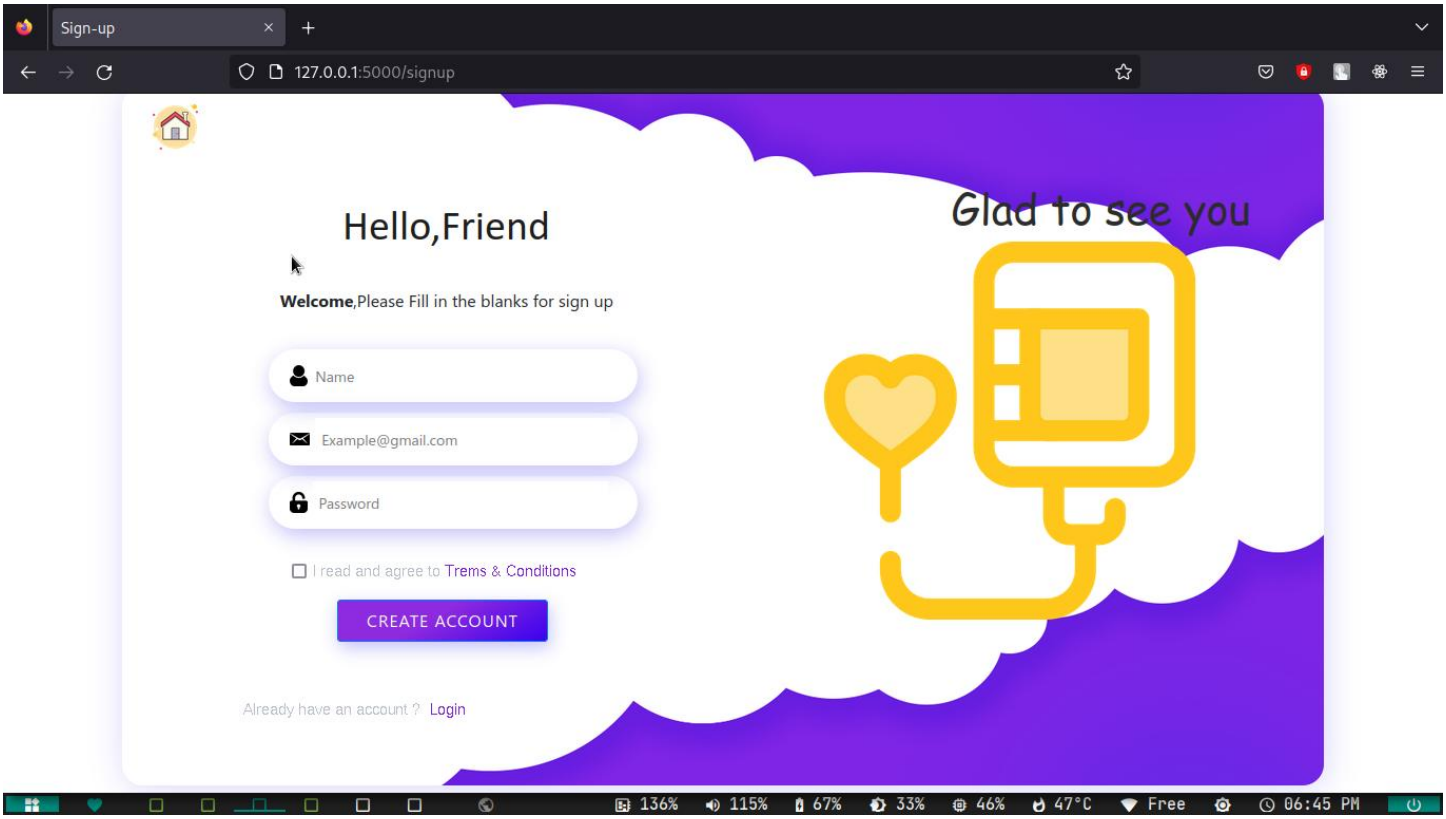
8.2.Donor Page



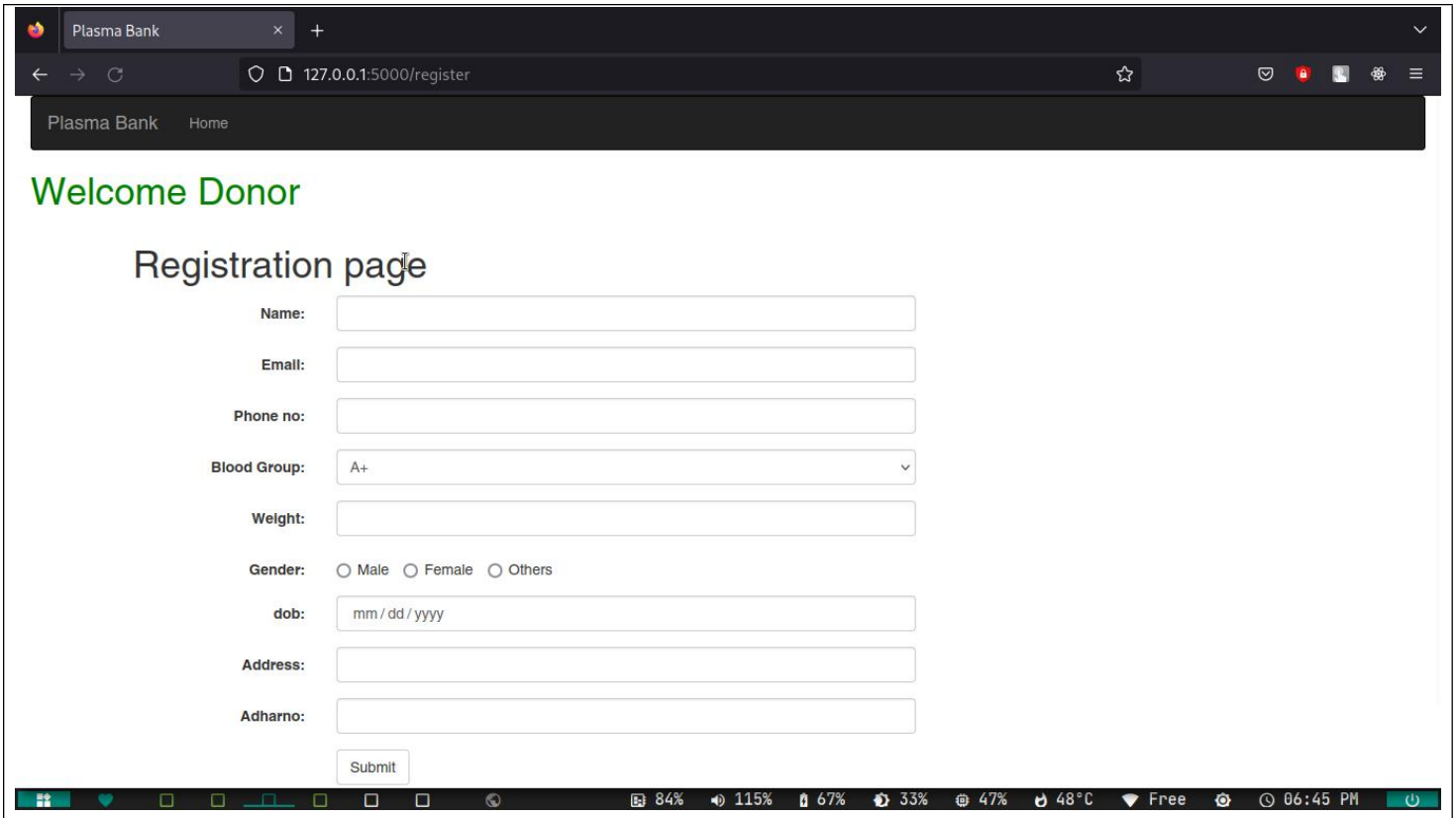
8.3.Login Page



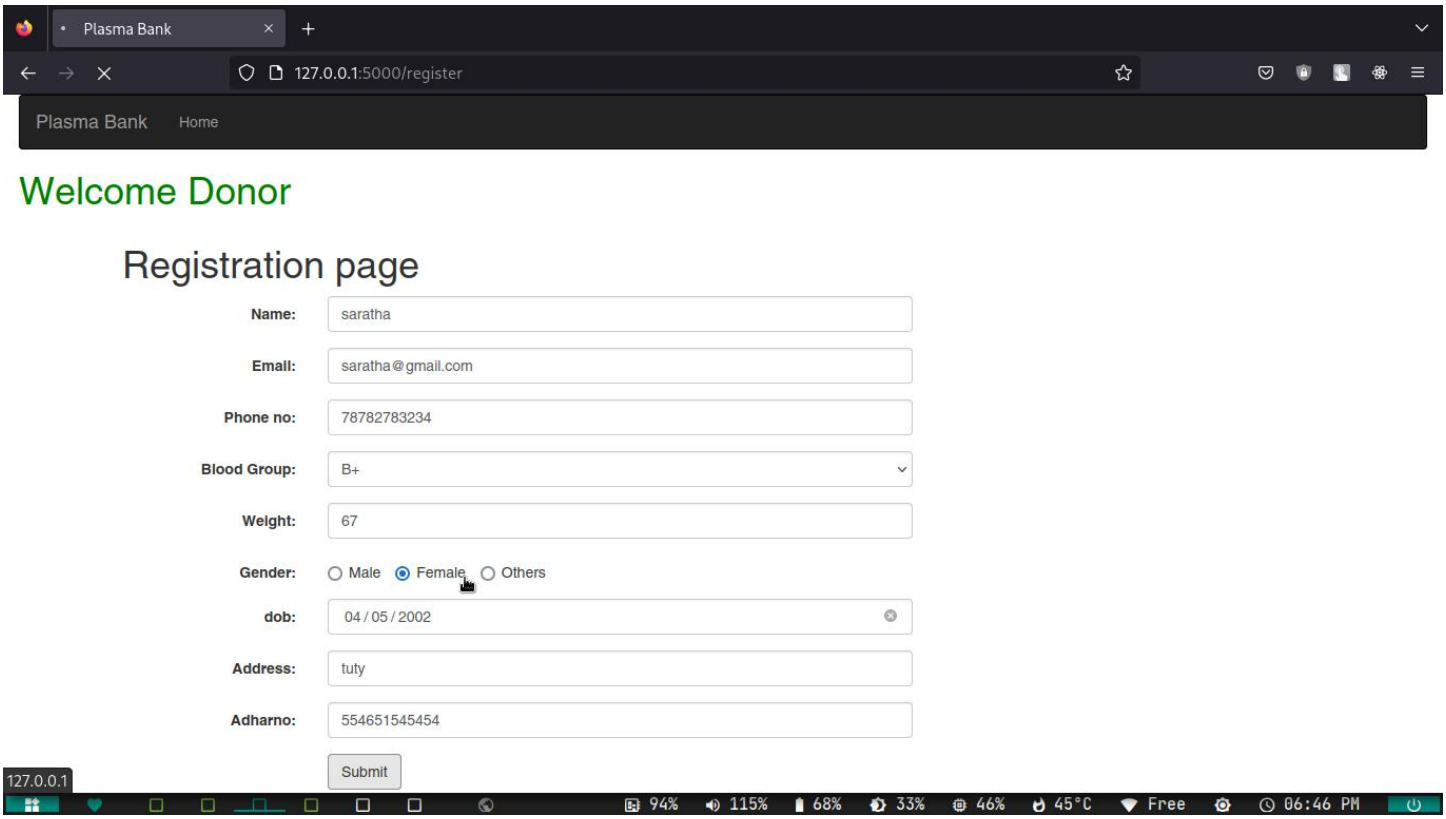
8.4.SignUp Page



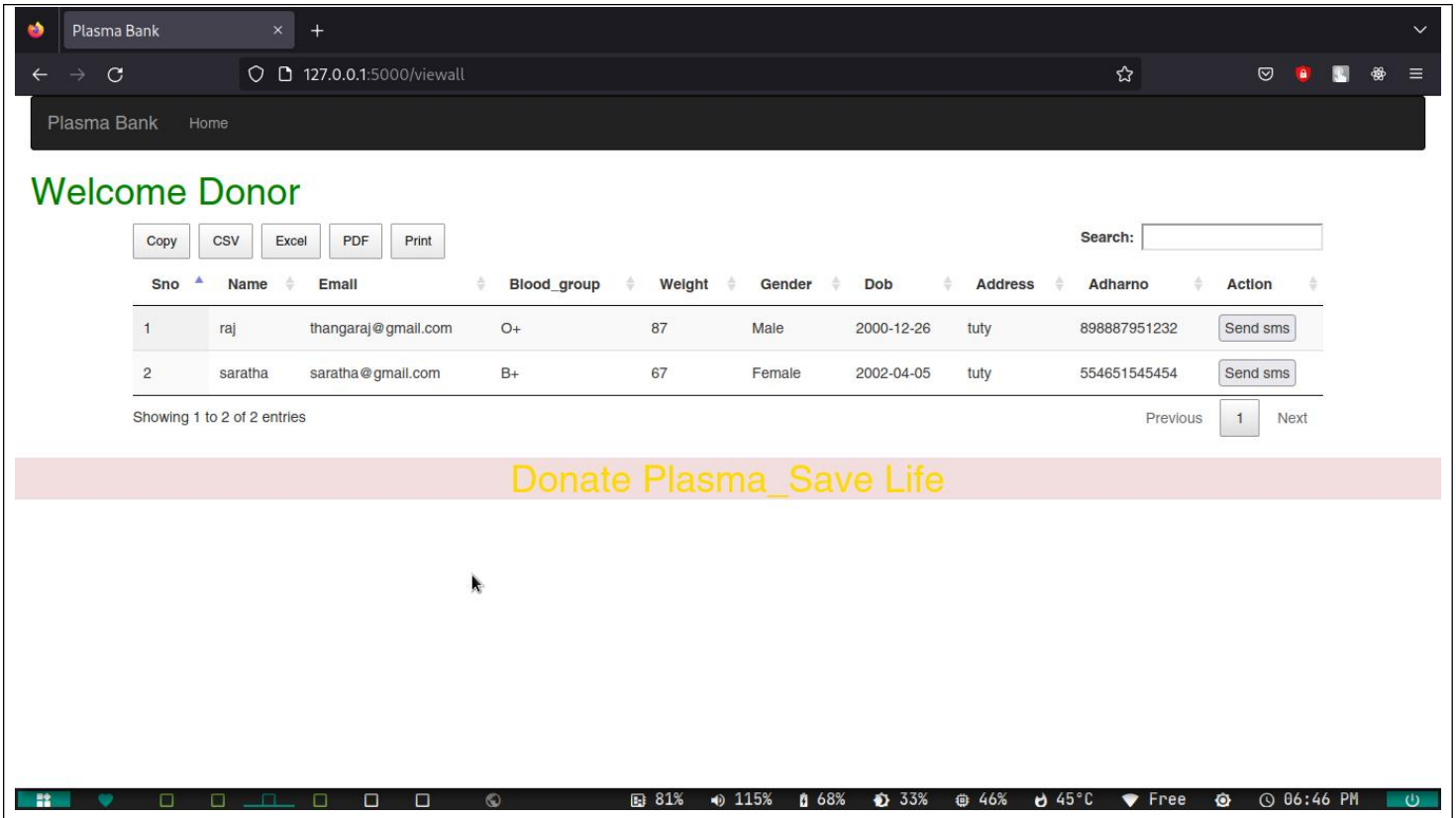
8.5.Donor Regisration Page



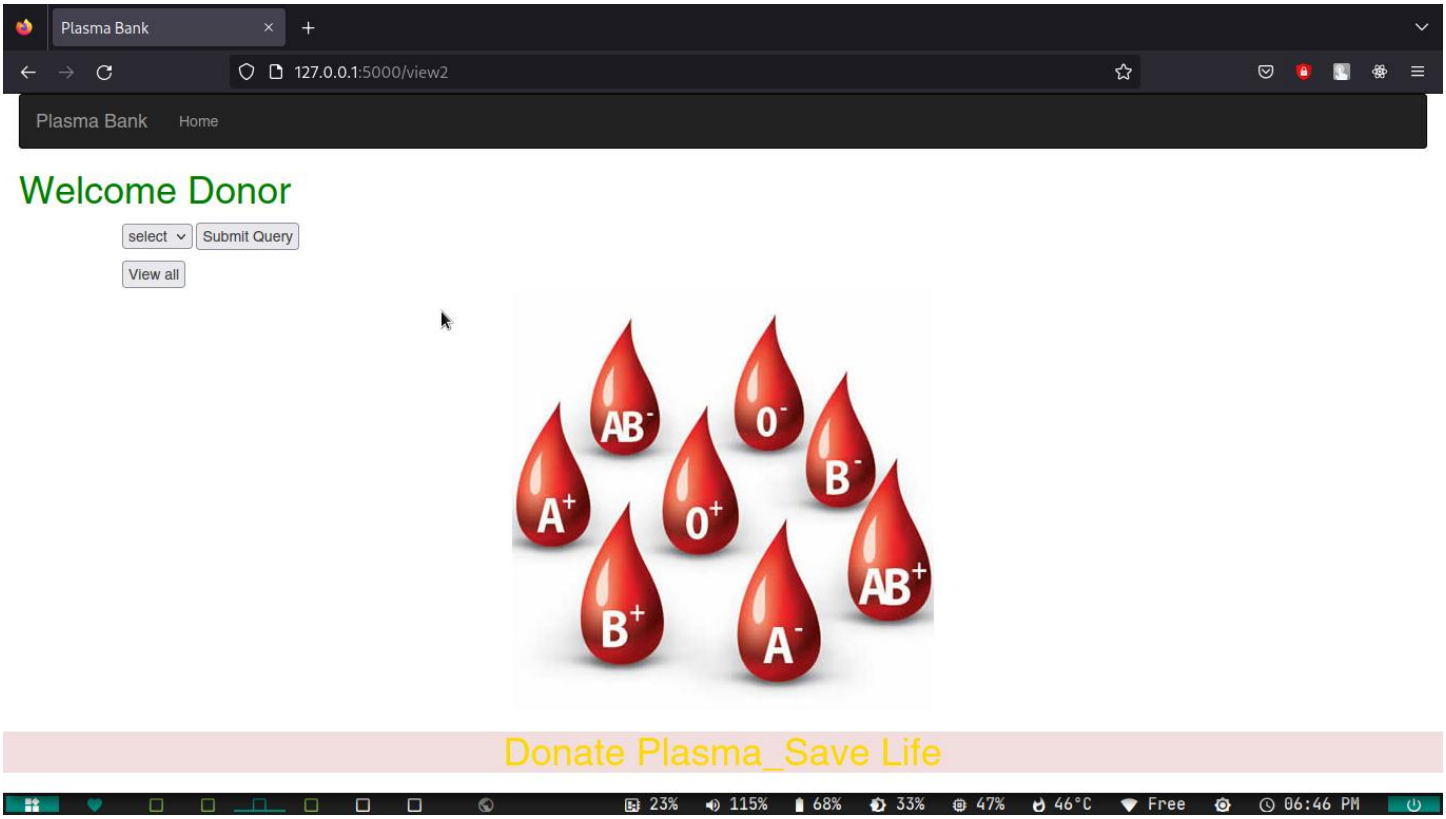
8.6.Donor Regisration Page



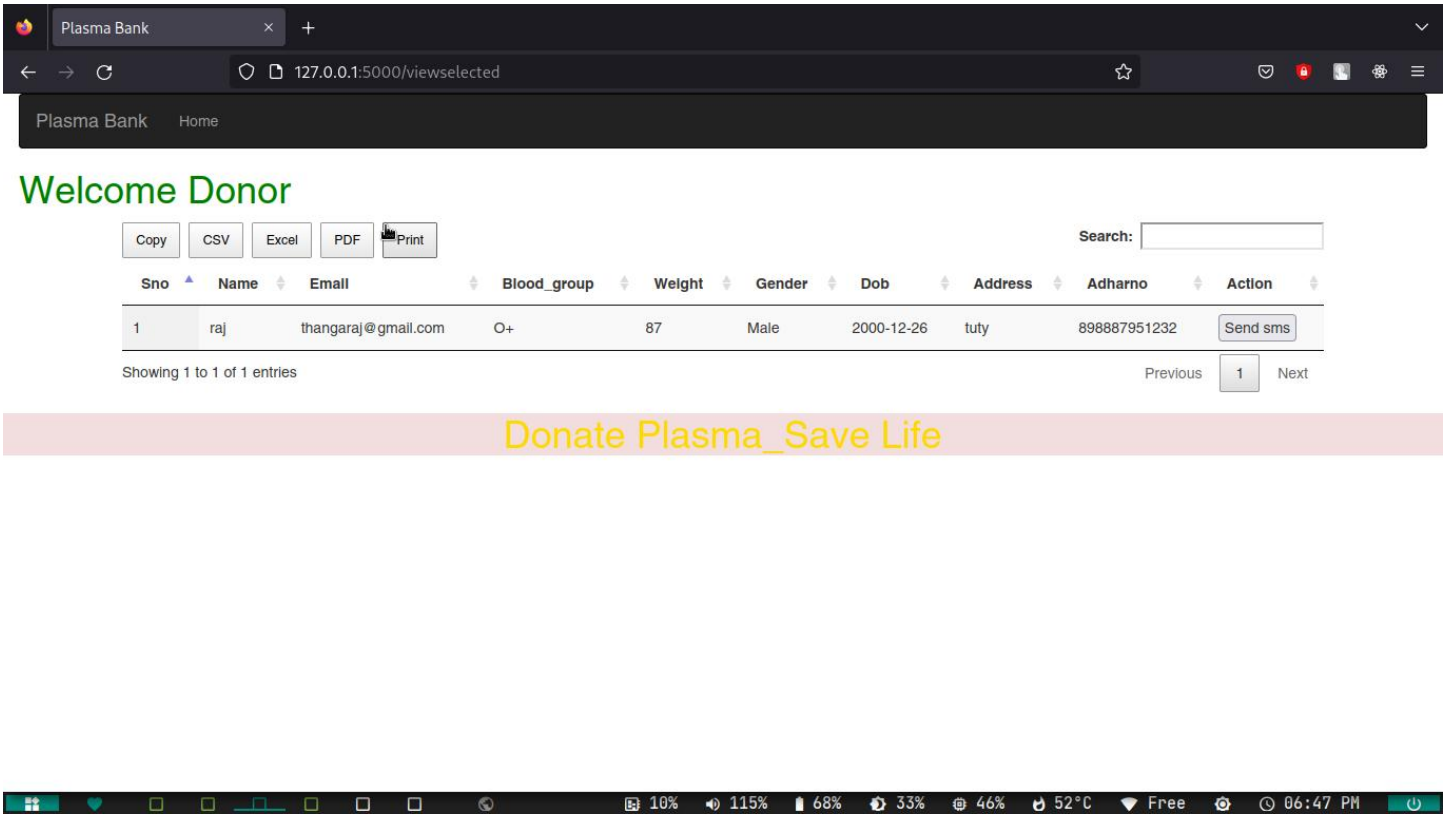
8.7 Donor Database Page



8.8. View Donor Page



8.9.Send Sms Page



9.ADVANTAGES & DISADVANTAGES

Advantages

At Software Things we focus on building real engagement with the help of apps. We want our products to be useful for users (a wordplay related to blood) application. It was especially important considering the major goal the app was aimed to achieve – increasing the number of blood donations. **We knew that donating blood saves lives, therefore we wanted to give users what they need to donate blood**.

The simplest way to do this was to ask donors what they expect from the app. Good relations with the client allowed us to reach the potential users, among whom we spread the information that we want to conduct workshops with blood donors on their needs regarding the potential app. **The users' response surprised us positively, people came to the workshops and contributed many useful insights. As a result, we have defined what elements had to be the core of the app**

Disadvantages

1.We couldn't introduce in MVP all the features that users needed, so the app will have to gradually evolve. Therefore, it was essential to deliver the app which will be able to develop, and additional functionalities could be implemented in the future.

2.The application holds the medical data which must be safely stored, so ensuring data security was a must.

3.Enabling the application to process a lot of data, and at the same time to work fast so the user will use it conveniently

4.We had to create a user registration process that must be developed in a way to ensure that only the authorized person – the blood donor – has access to their medical data.

10.CONCLUSION

It has been a great pleasure for me to work on this exciting and challenging project. This project proved good for me as it provided practical knowledge of not only programming in ASP.NET and VB.NET web based application and no some extent Windows Application and SQL Server, but also about all handling procedure related with Blood Bequeath Federal. It also provides knowledge about the latest technology used in developing web enabled application and client server technology that will be great demand in future. This will provide better opportunities and guidance in future in developing projects independently.

11.FUTURE SCOPE

The top two features rated as "useful" or "very useful" by the highest percentage of surveyed donors were the ability to ask questions when needed (70.8%), and the ability to locate the nearest blood center on the map and calculate the time needed to reach it (67.8%). Also, more than half of the respondents (52.9%) expressed interest in being notified about special recruitment events, such as blood drives and promotions and blood product shortages (52.2%). The ability to quickly confirm an appointment was found to be useful for 49.8% of respondents. Furthermore, similar percentages of respondents rated the ability to share donations on social media for donation promotion, and the option of requesting someone from the blood center to call the donor back (43.7% and 43.1%, respectively). The option of leaving a message for the blood center, including comments, suggestions, pictures or video, was considered useful by the lowest percentages of respondents (33.9%).

When the surveyed donors were asked about potential areas of concerns associated with using the app, 67.3% of donors claimed that they are “concerned” or “very concerned” about using their personal information for other purposes. A similar percentage of respondents (65.8%) agreed that they are concern about getting too many alerts or messages. More than half of the respondents (54.1%) were concerned about being unable to get their questions answered immediately. Half of the respondents (50.2%) were concern about the app being too difficult to use, and 29.9% of respondents indicated concerns about the lack of personal touch, with it being the lowest percentage reported regarding areas of concern with plasma donor.

12.APPENDIX

12.1Source Code

About.html

```
    {% extends "layout.html" %}

{% block content %}

{% with messages = get_flashed_messages() %}
{% if messages %}

<script type="text/javascript">
        var m = {{ messages| safe }};
        for (var i = 0; i < m.length; i++) {
                alert(m[i]);
        }
</script>

{% endif %}
{% endwith %}

<div
        style="background-image:          url('https://www.freepik.com/premium-photo/vacuum-tubes-collection-blood-samples-yellow-background-transparent-with-purple-green-lid-label-identify-data-selective-focus_16488940.htm');">
        <h1>Registration page</h1>
        <form class="form-horizontal" action="{{url_for('register')}}" method="POST">

                <div class="form-group">
                        <label class="control-label col-sm-2" for="pwd">Name:</label>
                        <div class="col-sm-6">
                                <input      type="text"      class="form-control"      id="name"      name="name" placehloder="Enter Name">
                        </div>
                </div>
```

```
<div class="form-group">
<label class="control-label col-sm-2" for="pwd">Email:</label>
<div class="col-sm-6">
<input type="email" class="form-control" id="email" name="email" placehloder="Enter Email">
</div>
</div>

<div class="form-group">
<label class="control-label col-sm-2" for="pwd">Phone no:</label>
<div class="col-sm-6">
<input type="text" class="form-control" id="phno" name="phno" placehloder="Enter Phno">
</div>
</div>

<div class="form-group">
<label class="control-label col-sm-2" for="pwd">Blood Group:</label>
<div class="col-sm-6">
<select class="form-control" id="blood_group" name="blood_group">
<option value="O">O</option>
<option value="A+">A+</option>
<option value="B+">B+</option>
<option value="B-">B-</option>
<option value="AB+">AB+</option>
<option value="AB-">AB-</option>
<option value="A-">A-</option>
<option value="O-">O-</option>
</select>
</div>
</div>

<div class="form-group">
<label class="control-label col-sm-2" for="pwd">Gender:</label>
<div class="col-sm-6">
<label class="radio-inline"><input type="radio" name="gender" value="Male">Male</label>
<label class="radio-inline"><input type="radio" name="gender" value="Female">Female</label>
<label class="radio-inline"><input type="radio" name="gender" value="Others">Others</label>
</div>
</div>
```

```
            <div class="form-group">
                <label class="control-label col-sm-2" for="pwd">dob:</label>
                <div class="col-sm-6">
                    <input type="date" class="form-control" id="dob" name="dob" placehloder="Enter dob">
                </div>
            </div>


            <div class="form-group">
                <label class="control-label col-sm-2" for="Address">Address:</label>
                <div class="col-sm-6">
                    <input type="text" class="form-control" id="address" name="address" placehloder="Enter Address">
                </div>
            </div>


            <div class="form-group">
                <label class="control-label col-sm-2" for="adharno">Adharno:</label>
                <div class="col-sm-6">
                    <input type="text" class="form-control" id="adharno" name="adharno" placehloder="Enter Adharno">
                </div>
            </div>

            <div class="form-group">
                <div class="col-sm-offset-2 col-sm-10 ">
                    <button type="submit" class="btn btn-default">Submit</button>
                </div>
            </div>
        </form>
</div>
{% endblock %}
```

Bloodbank.html

```
    {% extends "layout.html" %}


{% block content %}


<center><img
src="
GBgYGBgYGBgaGRgYGBgcIS4lHB4rIRgYJjgmKy8xNTU1GiQ7QDs0Py40NTEBDAwMEA8QHhISHzQrJCs0NjQ0
NDExNDQ0NDY0NDQ0MTQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NP/AABEIAN8A4g
MBIgACEQEDEQH/xAAcAAACAgMBAQAAAAAAAAAAAAAAAAQUGAgMEBwj/xABAEAACAQIDBQUFBQY
EBwAAAAAAQIDEQQFIRIxQVFhBhMicYEykaGxwUJS0eHwBxQVYnKSI4LC8RYkM0NTotL/xAAaAQEAAwE
BAQAAAAAAAAAAAAAAAAQIDBAUG/8QAKBEAAwACAgICAgIBBQAAAAAAAECAxESIQQxQVEFIhNxYT
```

KRobHh/9oADAMBAAIRAxEAPwD10xYxpEgSQwBADQhgCBCGJgkBoBAGQWEmMAEDAGABjcYyAJIZiMkD
MbGSAAEgAQAMaAAAAAAAQxNgABjtAAZDEAAwAQAwAS6gAMABBiA2IEgFwBkAdxmCMkyQMGFxMAGI
AIA0xtmLBdSQMaEMAAAAAAAYAMVgQwDGwDAAAEMAAFcEANDEMALgwECAEO5i2CRgJAQBgILgDAQ
XAGFxXGAAANEgEMAAAAE2AMAEAArjEAABYCAMTHYLEgQACZAGgEBIGAmxABe40BAdpO0cMLG20tt8
N7Xpz/XQrVKVtmmHDeW1MLbZOTmo6tpebscs8zoxdnUin1Z5ZjO11erLwtq/W794qFOrV1lO/nqc7z7/ANKPcn8HUz
vNWv6PWqOJhP2Zxl5NM2nltDDV6TThJ6a6P6FryXtKpeCv4ZL7T0T8+RaMyfVdHH5P4141vHW1/wA/7FnA4MXm
1GmryqR8otNv0RV8f2wqTk40IKK+9LxP3bi9ZZRz4PCz5vU9fb6ReEB5pLHYuT2nOfo3Fe5Hfg+086WlRSa/md38Sqz
J+0dV/icqX6tN/SL4jJEVledU6/sySfIkKuIjDe9eXE1VJrezzbxXFcaWmbkBxfxBcIs2QxUW9dCFcv5K8aXwdG8YJiZcq
MQ7iAEACIBlcDGzAAzuFyq0c7nxszsp5y+KRks8s2eCkTrMWRtPN4vejojjoPiXVy/TKOKXtHVcwnNRV27I0PFx5nF
WrOo/5Vu/EirSQmG2dM8Y+C95rWNmuCZriKoZcqfezZRP0YZ3nkcPh5T3StaK68+tvnY8axuMlWnKc3dt8726Fj7f5g
5VVST0ilfzaUn84/2lQUjmzZHXR9V+H8acOPnrt/8ARI5dbb1LVhaiKXQq7LuTmDxyfErD6PQ8qHXZbcJW1NmOy1
VVdaMh8Lidd5N4XFGstPpni5YrG+UkJ/BJOdty4k/gcshTWkVfmSGNw8qUYyl9rf0e+3u+TOKWJL8VLMH5N5p6fX
+Dqko8kReZ4eMovRDq4wjsVjupFUi+HDapNFcqYiWHqXi2rMumR5oq8b313/ivr6lBzbEKUjq7LY3u6y5b/o/gZc9PR6
nmeKsuDk12keoUzdsmqEjKU7aGx8m57OnDV9l2e75HfchmEsbKmua5cvI0nJpaZlWJvtEy2FyHjm6fA2rM4miyS/kye
Ol8Em2JHCsxgH8ThyY5z9j+OvokLgR/8Vh1Ac5+x/Hf0USOINsMT1IP946gsV1PNR6z0WSni7Gbx9uJX4YrZV5fq1jl
qY9zklfS9i+2Z6RcsDXdTXhw69SWhF2IHKp6IsNF3RaHsztaFJOOppnXVrjxdSy3lcxmY7LkvL3v/YvVaJiHRSe11bbx
U31fwdvoQ0ZHTnNXarzfX56/U40zCls+p8euMJL6NyZthUaOZSMkyva9HZNkthcwcXqej9hcI63+PJeCLtH+aS3vyXz8
jzXIstni68KMN8nq+EYrWUn5K/rZcT3vAYONClGlBWjCKjFdFz5vizq8edvkzwfzXlTErHPt+/8AC/8ATHNcL3tKUVv
tdea1X4ep5ZXzVRum7NaNPenxTPX2eL/tEyt0Ma5RXgrLbjy2t00ut9f85pnWlyOH8NcvI8VfPaOWvnPJnBWzCT5nLG
Kgry38Pj7zTUquTu/TojidH1URCfSHOdzbgq2xUT9DluEZaq2+5XbbL5KXFnreFx/+HFv7sX57jupV0yk/vv8Ay0OEopJ
p+5plgwFa9jVU96PkcmNdssUHxObFT0Oih7JxZg9Ga11Jyyv2IKtiNiduBuhiupDZlNpt8vx/P4HLRxfU59tG/FMtCxI3iS
DhiupvjiC3IcSU74CM/eRkchxKf35sVZQSber5PXdw4cd5yTqxp7r3+Wunpp6pnG6jerZsoMnWzvq4pybbZnhJtyuR8CW
yynz+98l+Ya0gntl0yJOybLRRehXcqklEm4ztEpjZORbOHNJ2vYpmPrWqK75v13L6lxxjvFu5R45fPE15NPZhBtbW9t8d
leofdG0PUlazKpF1ZcXxfWxy3JvPsFTpWUVx1b3u/EgnoQz1/FybnTM7jTMEzfhKPeVIQ+/OMP7ml9Sujrd6Wz1z9luS
d1QeJmvFV0hdaqmn/qav5KJO9s85qYLCOtTUXPahFKabjq9dE09yfEi+3eYVcBg6Kwr2LTjDSKklBQl4bNNJaL3Hm2
bdrMTjacaNaUXFTjLSKi7pNK7XDxPhyOyqWOeK96PnMXj35eX+etOW+186R6r2Fz+rjqE6lWMIuM9hbCkk/DGTvt
N/eOb9pGB7zCd6leVGSkrb9ltKS8tzf9JQ+z3bCWX0nShSjNSm5vaupXairOztuSPR8jzJ5pgJynTVPbVSnsp7StbZvdpc/
gTNK44t96KZsN+Nn/mlanfXfweHTm27sxua9oVzg0fVqzY5GzC1LVIf1L5nM5GmFX/EXQvMbOXyvIUQ/wDPR6jm
eCjicO5Q8M1HRr7VuEuY+z1Ryim9/H00IrKsfJQ36WOzJKjlUkordJ3fBX1IZ4nw0XejLQ5cfNbJvoaJETnGISVjWn+pz
TO6ILNIXjJfyyfui38yswrWJ/E4m9Ocv5JL3p/kVVzKSto0b0yVhiup108SV6NRnXh6m07X+vw4hySqJj94A4u8p/efx/8
AkZXiTyK5smcInRGkbY0TqOTZhSgd+Glsr1uaIwMqsXsSXS/u1+VyrW0TL0y3ZPjVZXZZaVZNaM8pwmPdOzZbMi
zV1NIxlLyRkk5Z0PVLZYcfRlsS2d9npzIXCQlCmla3F+b3liw1dVItbmtGn9Su5zha22+7nFRa3O+/iRS+UTL+GU/tOm7
3ZW6dTgTmcYSvd7bj6O5AzobOppKTnRrGd47T+DpRnSquE4yW+MlJead0c1KpdG1MzaaZ7UXOSNr5PpPLMdDEU
YVYO8ZxUl68H1TuvQ8//bDNKGHikk26ktyvaKiv9RXOxfbKWBfdzi5UJO7Sfipye+UL6NPjH1539Twuf4HFRVq9GX
HZm4qSvzjKzXuOzksk632fNvDk8POr02l6IDsFkGGq5dSnVw1KcpbbcpQjKVu8klq1fcixZjUo5dg6koQjCEIycYpWTl
Lcl1cmjVje0mCwkLOvTiluhBqT8lGO75Hk3bPtfPMJqKThRi7xi34pPdtTtpe17JbrveRVTE6XsYsOXycvKtqW99+v6R
V7mLZjcwlM5Utn0d5VK2xznwNlDDvhqa8NS2nqWnKcFG6vFGjamTxcuWs17+PgwyfGSh4dl35f7l9yWjeKeyk5avzO

OFCMKbkopWi9y6ErgKijT2uS0+hlOnRWtqTsxFXYT14FSzjHPX9ehY8yy+Uqd3Nqcldck7brHnlWpOU5U5e1GTi/Ti
WuXszipSMsTXfdf1N+5P8AIiZEji1uit0VY49kskUbNEhKVjbNGqSJGx94Br2QGkTskYwNsInc6ERd0X0YnNCBmom
5UzJU3yA2Qma4XZ9nc7O3TdoXDspXVOnGNtXq+v6sQOaUH3d/utP0ej+S95tyitLZjJO+y9lq+qUn4X5XbXquZFei8d
vTPRJ01tRqxej8M16aP4HI5RlUcZbrP5ojJ5s9nZT8K3debOfLcZ3lV8lp6s56rs6Zh8ezZnvZ/vYN02lLhyfRlEzLKZU09u
6a4WsetQntbiK7QZf3lN7ttey+vJvh9C0vXoo/26Z5FPCunFO1uPvCjUUkWyWUSnHxrVLRciv5pk84WnFOy324F+U10
b4Ml4fXa+jQBrUJxV2m10F3y5lHLPUjyIteza2a5MxlWXB3MJNkqWyuXyYhezJu5hbU2Qg3wNtLDOTtf3GqSk8jNnrI
9fB0YCn8y95Jg7pNlbwWXKKW++nEt2DrbKjFK75GGStstE6RIYunenKK4xa9baHLkuYx2VGXr6G/ExnsNvTTcUuW
O7uo4ydru6fAjT9olaaaZ6VmmLhKmrPd8NCh4mK7+cvvSXwST+KZ1QxbaVndcr7+hrwmHnUlKSi5tLabjd22rpLTdu
NU6o56Ux6I2vTd3damiVElK9CSfiTXmrGhwLaKJkZOkaZUyWdJMxeGQ0TsidgCU/dUA0NnoUuzOGf2JLymxLsvh+
U/7iaSMkjr4r6OHk/siafZnDL7Mn5zf0OqnkmHjupL1bfzZIIaHFEcn9kVm2TQq4edOEIxco+GyS8S1j8UjyrCVJ0ZvZ0kr
qUXxW5prme2FL7Udk5VKjr0PalrOGmr4yjzvxRS432jTHeumUjG5tdWhCSfW2nuOrI67gt+r1fmZVMApPZnHYmuaa
XqnuNNWDpJximnzv521T13s5qxpo7IzNey4YXFLS7uStKamjznAZk09XxLPl2ap21MnuXpm3VLaJevl9pXXuOCvg47
VmtGiXpYnbWiNOMw8pfZ3cU/gQ5+UTNNPTKficujBytF7Nm92iIzKMopzfeTinfVJ7kuB6IqcVEhsVhI7TlBbMr3atp
L8yOTSLrTeyOllMGvBTVv6SLzfIZOm5Rp6rXTfbjoWzBYqLXXl1JJQcl7K94mnvaFva00eV0sLsxuySyHL9q8mv1vJ
ntFl2zJSUbRlK0vPf8AE58LN04TjGN5N2SSu9S9VtGcz2dUKK/XInMqoqKv9p739ER2AyyrJXl4fPf7idw2CjTV222US
bey9UktDx0bwsefZ1l+1PQveLqrdcrGYVltbML35rn6by+++jP1JryfK5zexThtNLV7ktdHJ8FdfPfoeg5JlUcNT2E9qTe1O
fGUvwW5GPZ3Lf3egk145eKfm90fRae8lbHbjjS2/Z52XJyel6NFXDwnpKEZeaTI3EdnsPP/ALez/S2vhuJcC7SZmqa9FY
rdkoP2akl5pP8AA459kJ8KsfWLRcwK8JLLJX2Ub/hGt/5If+34AXmwDhJP8tGpGSCxki5mCGCGgAQxDAOLNMthiK
bjJK/2ZW1i/wADzPN8FVozcJQ0XB7mucWetI1YnCwqrZnBSXC+9eT3opUbLzfE8NnJKXLg1x/M7KDlDxLVF17Udj
Yzht0V4lvjvuuaPOKkKtGbSbVtHF6p+jMqj4Z0xk+UX3JsxulqWahVUkeW5VmF5W9mXLg/IumVZmmrNnNpyzq2rW
17JnE09l7VjXGEJatIzniU47yJU2p/yt/HkytNJ9FpltdnRisvjfbhpLiuEvz6nVg6t0jbRaktEaJWU2lpz8yPXaJ3taZnmmGVS
Dhvb+HJnHhMr2NdE+LtqSkGrXOHF5jGCepdpe2UTfpG51FDezixOYLmQGPzq7siLqYqT1bshpv0S+K9kjmuaWukya
7G5HKVsTWWm+nF8eU305e8iex2TxxdZzqLap07O3CU2/CnzWjdvI9NSOnDhS/ZnJnzt/qgAAOg4xMQxMAABIYAg
GABihmJkAAxDBA0AACQRkhAAZFfz7srTxXjjaE+dvDLpJfVFgQyGk/ZKbXo8fz7slVpLbUHpxjqvNNEZluY929iq3
Fr2Z8Gvuz+jPcpRTVmrp709xTu0nYaFdSnRahPeov2W+SfAyrH10bRm0+ysfxmMV/1E/iWLJId7Te19r5cCl4PJ+6qbN
WDi07NPgy+YCUYRWyzkqUq0eir3O/k3YdOm9lvdx5ogMZmDp1pRb3ttdUyWzWu1ByjvWvmVHG1410tVfg+KIS30
i0vrkySxucS2NJpLm9xWsbm79m8pPnaxrrYS+rblwV2+qfC6MKOF11NVMpdmNU36NEa9SWqil8WbsFgalerCGs5yd
kr/pJcSYwOWTqy2YQcn0Xz5F+7MdmVhG6k2pVZK2m6C4qL4t8X+npCb9ejnyUl/ZJZHlccJRjTjq98pfek978uHoSDG
JnSlro429vYgAAAAAAFYBhYAQBYAAEMAQCGFgAAAAEgMQ0AMyQhgDAQyQUv9oWKVHuJqmpNzltaaygo2
cb/AOZNdUV+hiXFJxntQfsS425SXCS3Houa5XTxUFCpG9neLWjT6FXzPsS1BuhUd001BpK9nr4r8uhhkh0dWHKpWi
KeMcvDzNNfL8Nh4rwpy689H81c5nl9SnOzpz2+qbd+mmu9q5Y8i7LSqS7zExaX2YN6vq+S6GM43vSOm8spbZVsNh
KmKqOFGm5W9p7lHltN6ItOU9h9m0q811jD5OTLjhcJClHYhBQjvslvfN82bjecSXs4rzVXro04bCQpR2IQUYrgl83xN
oxGqMRCY2ABjYDITQAgAAAALAAABYAAAAAAAAAAxirAGSMjEyAGMxRkAAAgAAYhkgLAAwBCGABi
DBiSsQBiGAAgGAAhDEwAYrDAAQDAA//2Q==" width="700" height="350">

<h1 style="color:brown" ;>ABOUT PLASMA: <h1>

```
   <U>   <center>   <h4>Plasma serves four important functions in our body:</h4></center></U>
1. Helps maintain blood pressure and volume.<br>

2. Supply critical proteins for blood clotting and immunity.<br>

3. Carries electrolytes such as sodium and potassium to our muscles.<br>

4. Helps to maintain a proper pH balance in the body, which supports cell function.<br>



                {% endblock %}
```

Edit .html

```
    {% extends "layout.html" %}

{% block content %}

<h1>Edit page</h1>
<form class="form-horizontal" action="{{url_for('update')}}" method="POST">

<div class="form-group">
        <label class="control-label col-sm-2" for="pwd">Name:</label>
        <div class="col-sm-6">
                <input type="text" class="form-control" id ="name" name="name" value="{{data[0][1]}}"
placehloder="Enter Name">
   </div>
</div>

<div class="form-group">
  <label class="control-label col-sm-2" for="pwd">Email:</label>
  <div class="col-sm-6">
        <input type="email" class="form-control" id ="email" name="email" value="{{data[0][2]}}" placehloder="Enter
Email">
  </div>
</div>

<div class="form-group">
```

```
 <label class="control-label col-sm-2" for="pwd">Phno:</label>
 <div class="col-sm-6">
    <input type="text" class="form-control" id ="phno" name="phno" value="{{data[0][3]}}" placehloder="Enter Phno">
   </div>
</div>



<div class="form-group">
 <label class="control-label col-sm-2" for="pwd">Blood Group:</label>
 <div class="col-sm-6">
        <select class="form-control" id="blood_group" name="blood_group">
                <option value="A+" {% if data[0][4]=="A+"%} selected {% endif %}>A+</option>
                <option value="A-" {% if data[0][4]=="A-"%} selected {% endif %}>A-</option>
                <option value="B+" {% if data[0][4]=="B+"%} selected {% endif %}>B+</option>
                <option value="B-" {% if data[0][4]=="B+"%} selected {% endif %}>B-</option>
                <option value="AB+"{% if data[0][4]=="AB+"%} selected {% endif %}>AB+ </option>
                <option value="AB-"{% if data[0][4]=="AB-"%} selected {% endif %}>AB-</option>
                <option value="O+" {% if data[0][4]=="O+"%} selected {% endif %}>O+</option>
                <option value="O-" {% if data[0][4]=="O-"%} selected {% endif %}>O-</option>
                <option value="Don't know" {% if data[0][4]=="A+"%} selected {% endif %}>Don't Know</option>
        </select>
 </div>
 </div>



 <div class="form-group">
 <label class="control-label col-sm-2" for="pwd">Weight:</label>
 <div class="col-sm-6">
        <input type="text" class="form-control" id ="weight" name="weight" value="{{data[0][5]}}" placehloder="Enter weight">
 </div>
</div>
```

```
<div class="form-group">
                <label class="control-label col-sm-2" for="pwd">Gender:</label>
                <div class="col-sm-6">
                <label class="radio-inline"><input type="radio" name="gender" value="Male" {% if data[0][6]=="Male"%} checked {% endif %}>Male</label>
                <label class="radio-inline"><input type="radio" name="gender" value="Female" {% if data[0][6]=="Female"%} checked {% endif %}>Female</label>
                <label class="radio-inline"><input type="radio" name="gender" value="Others" {% if data[0][6]=="Others"%} checkeded {% endif %}>Others</label>
                </div>
</div>


<div class="form-group">
                <label class="control-label col-sm-2" for="pwd">dob:</label>
                <div class="col-sm-6">
                <input type="date" class="form-control" id ="dob" name="dob" value="{{data[0][7]}}" placehloder="Enter dob">
                </div>
</div>


<div class="form-group">
                <label class="control-label col-sm-2" for="Address">Address:</label>
                <div class="col-sm-6">
                <input type="text" class="form-control" id ="address" name="address" value="{{data[0][8]}}" placehloder="Enter Address">
                </div>
</div>


<div class="form-group">
    <label class="control-label col-sm-2" for="adharno">Adharno:</label>
    <div class="col-sm-6">
    <input type="text" class="form-control" id ="adharno" name="adharno" value="{{data[0][10]}}" placehloder="Enter Adharno">
    </div>
```

```
</div>

<div class="form-group">
                    <div class="col-sm-offset-2 col-sm-10 ">
                    <button type="submit" class="btn btn-default" value={{data[0][0]}} name='id'>Submit</button>
                    </div>
</div>
</form>



{% endblock %}
```

Inactive .html

```
{% extends "layout.html" %}

{% block content %}

<table id="example" class="display" style="width:100%">
      <thead>
         <tr>
            <th>Sno</th>
            <th>Name</th>
            <th>Email</th>
            <th>Phno</th>
            <th>Blood_group</th>
            <th>Weight</th>
            <th>Gender</th>
            <th>Dob</th>
            <th>Address</th>
            <th>Adharno</th>
            <th>Action</th>
            <th></th>
            <th></th>

         </tr>
```

```
</thead>
<tbody>
   {% for d in data %}
   <tr>
      <td>{{loop.index}}
      <td>{{d[1]}}</td>
      <td>{{d[2]}}</td>
      <td>{{d[3]}}</td>
      <td>{{d[4]}}</td>
      <td>{{d[5]}}</td>
      <td>{{d[6]}}</td>
      <td>{{d[7]}}</td>
      <td>{{d[8]}}</td>
      <td>{{d[9]}}</td>

      <td>
        <form action="{{url_for('edit')}}" method="POST">
        <button value={{d[0]}} name="edit">Edit</button>
        </form>
      </td>

      <td>
        <form action="{{url_for('activate')}}" method="POST">
        <button value={{d[0]}} name="hold">Activate</button>
        </form>
      </td>



      <td>
        <form action="{{url_for('delete')}}" method="POST">
        <button value={{d[0]}} name="delete">Delete</button>
        </form>
      </td>

   </tr>
   {% endfor %}
</tbody>
```

```
</table>

<script type="text/javascript">
$(document).ready(function() {
    $('#example').DataTable( {
        dom: 'Bfrtip',
        buttons: [
            'copy', 'csv', 'excel', 'pdf', 'print'
        ]
    } );
} );
</script>

<div class="container-fluid"> <!-- Header content -->
    <nav class="navbar navbar-inverse">
  <div class="container-fluid">
    <div class="navbar-header">
      <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#myNavbar">
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </button>

      <div>
    <ul class="nav navbar-nav navbar-right">
        <li><a href="{{url_for('view')}}"><span class="glyphicon glyphicon-log-out"></span> Active</a></li>

  </ul>
  </div>



    <div>
    <ul class="nav navbar-nav navbar-right">
        <li><a href="{{url_for('logout')}}"><span class="glyphicon glyphicon-log-out"></span> Logout</a></li>

  </ul>
  </div>
```

```
</div>
</div>
</nav>
</div>

{% endblock %}
```

Index.html

```
 {% extends "layout.html" %}

{% block content %}



<div class="container-fluid">
 <!-- Header content -->
 <nav class="navbar navbar-inverse">
  <div class="container-fluid">
   <div class="navbar-header">

    <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#myNavbar">
     <span class="icon-bar"></span>
     <span class="icon-bar"></span>
     <span class="icon-bar"></span>
    </button>
    <div>
     <ul class="nav navbar-nav navbar-right">
      <li><a href="{{url_for('login')}}"><span class="glyphicon glyphicon-log-in"></span> Login</a></li>
      <li><a href="{{ url_for('signup') }}">sign-up</a></li>
      <li><a href="{{ url_for('register') }}">Donor Registeration Form</a></li>
      <li><a href="{{ url_for('view2') }}">View Donors</a></li>
```

```
        </ul>
       </div>

     </div>
    </div>
  </nav>
</div>

<center>
  <div>
    <P style="color:brown;font-size:40px" ;>Active donors:{% for i in data %}{{ i[0] }} {% endfor %}</P>
</center>

<center><img
    src="https://media.istockphoto.com/id/1225770765/vector/donate-plasma-hand-drawn-
illustrations.jpg?s=612x612&w=0&k=20&c=gW7G7KzcpmhaCFIai9EX8aw_SUvEKlIKrIeeIB6U1PI="
    width="550" height="345"></center>

{% endblock %}
```

Info.html

```
{% extends "layout.html" %}

{% block content %}

{% with messages = get_flashed_messages() %}
  {% if messages %}

  <script type="text/javascript">
    var m={{ messages|safe }};
    for(var i=0;i<m.length;i++)
    {
      alert(m[i]);
    }
  </script>
  <style>
```

```
   body {
     background-image:  url('https://png.pngtree.com/background/20210715/original/pngtree-beautiful-noble-yellow-white-abstract-gold-line-watercolor-background-picture-image_1325754.jpg');
     background-repeat: no-repeat;
     background-attachment: fixed;
     background-size: cover;
   }
  </style>

  {% endif %}
{% endwith %}
<body>

          <div class="container-fluid"> <!-- Header content -->
                    <nav class="navbar navbar-inverse">
  <div class="container-fluid">
    <div class="navbar-header">
      <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#myNavbar">
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </button>
    <div>
    <ul class="nav navbar-nav navbar-right">
        <li><a href="{{ url_for('view') }}">Active Donors</a></li>
         <li><a href="{{ url_for('inactive') }}">Inactive Donors</a></li>
         <li><a href="{{url_for('logout')}}"><span class="glyphicon glyphicon-log-out"></span> Logout</a></li>
    </ul>
    </div>




</div>
    </nav>
</div>

</body>  <center><img src="..\static\images\inf.png" width="460" height="345"></center>
```

```
{% endblock %}
```

Layout.html

```
  <!DOCTYPE html>
<html>

<head>
  <title>Plasma Bank</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
  <link rel="stylesheet" type="text/css" href="https://cdn.datatables.net/1.10.21/css/jquery.dataTables.min.css">
  <link rel="stylesheet" type="text/css" href="https://cdn.datatables.net/buttons/1.6.2/css/buttons.dataTables.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script type="text/javascript" src="https://code.jquery.com/jquery-3.5.1.js"></script>
  <script type="text/javascript" src="https://cdn.datatables.net/1.10.21/js/jquery.dataTables.min.js"></script>
  <script type="text/javascript" src="https://cdn.datatables.net/buttons/1.6.2/js/dataTables.buttons.min.js"></script>
  <script type="text/javascript" src="https://cdn.datatables.net/buttons/1.6.2/js/buttons.flash.min.js"></script>
  <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/jszip/3.1.3/jszip.min.js"></script>
  <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/pdfmake/0.1.53/pdfmake.min.js"></script>
  <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/pdfmake/0.1.53/vfs_fonts.js"></script>
  <script type="text/javascript" src="https://cdn.datatables.net/buttons/1.6.2/js/buttons.html5.min.js"></script>
  <script type="text/javascript" src="https://cdn.datatables.net/buttons/1.6.2/js/buttons.print.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
</head>

<body>

  <div class="container-fluid">
    <!-- Header content -->
    <nav class="navbar navbar-inverse">
      <div class="container-fluid">
        <div class="navbar-header">
          <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#myNavbar">
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
```

```
        <span class="icon-bar"></span>
      </button>
      <a class="navbar-brand" href="{{ url_for('bloodbank') }}">Plasma Bank</a>
    </div>
    <div class="collapse navbar-collapse" id="myNavbar">
      <ul class="nav navbar-nav">
        <li><a href="{{ url_for('home') }}">Home</a></li>




      </ul>
      <ul class="nav navbar-nav navbar-right">

      </ul>
    </div>
  </div>
 </nav>

 <h1 style="color:green;"> Welcome Donor </h1>
</div>
<div class="container">
 <!-- Body content -->
 {% block content %}



 {% endblock%}
</div>


<div class="container_fluid bg-danger">
 <!-- Footer content -->
 <center>
  <h1 style="color:#ffd900;"> Donate Plasma_Save Life </h1>
 </center>
</div>
```

```
</body>

</html>
```

Login.html

```
<!DOCTYPE html>
<html>

 <head>
  <title>Login Form</title>
  <link rel="stylesheet" type="text/css" href="..\static\css\login.css">
  <link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap" rel="stylesheet">
  <script src="https://kit.fontawesome.com/a81368914c.js"></script>
  <meta name="viewport" content="width=device-width, initial-scale=1">
</head>

<body>
  <img class="wave" src="..\static\images\wave.png">
  <div class="container">

    <div class="img">
      <div id="png"><a href="/" title="HOME"><img style="width:75px; height:75px ; "
          src="..\static\images\home-page.png"></a></div>
      <img src="https://cdn.iconscout.com/icon/premium/png-256-thumb/blood-plasma-2500160-2092385.png">
    </div>



    <div class="login-content">

      <form action="{{url_for('login')}}" method="POST">
        <div class="msg">{{ msg }}</div>
        <img src="..\static\images\avatar.svg">
        <h2 class="title">Welcome</h2>
        <div class="input-div one">
          <div class="i">
            <i class="fas fa-user"></i>
          </div>
```

```
        <div class="div">

          <input type="text" name="Username" placeholder="username" class="input" required>
        </div>
      </div>
      <div class="input-div pass">
        <div class="i">
          <i class="fas fa-lock"></i>
        </div>
        <div class="div">

          <input type="password" name="Password" placeholder="password" class="input" required>
        </div>
      </div>
      <a href="#">Forgot Password?</a>
      <div class="btn">
        <button type="login" class="btn btn-default">Login</button>
      </div><br><br><br><br>


      <div class="app"><b>Don't have an account?</b><a id="app1" href="\signup">Register Here </a></div>
    </form>

   </div>

  </div>

  <script type="text/javascript" src="..\static\js\login.js"></script>
</body>

</html>
```

Select.html

```
{% extends "layout.html" %}
```

```
{% block content %}



{% with messages = get_flashed_messages() %}
        {% if messages %}

        <script type="text/javascript">
                var m={{ messages|safe }};
                for(var i=0;i<m.length;i++)
                {
                        alert(m[i]);
                }
        </script>

        {% endif %}
{% endwith %}

<form class="form-horizontal" action="{{url_for('viewselected')}}" method="POST">
        <select id="blood_group" name="blood_group">
                <option value="">select</option>
                        <option value="A+">A+</option>
                        <option value="A-">A-</option>
                        <option value="B+">B+</option>
                        <option value="B-">B-</option>
                        <option value="AB+">AB+</option>
                        <option value="AB-">AB-</option>
                        <option value="O+">O+</option>
                        <option value="O-">O-</option>
        </select>




        <input type="submit" name="submit">
        <p></p>
</form>
<script type="text/javascript">
        $('#blood_group').change(function()){
```

```
                $.ajax({
                        data:{
                                blood_group:$('#blood_group').val()
                        },
                        type:'POST'
                        url:'/viewselected'
                })
                .done(function(data){
                        alert(data.blood_group)
                });

</script>

   <td>
      <form action="{{url_for('viewall')}}" method="POST">
      <button value="" name="viewall">View all</button>
      </form>
   </td>

<center><img src="http://www.gkh.co.in/files/abstract-images/blood_bank_581151803.jpg" width="400" height="400"></center>

{% endblock %}
```

Signup.html

```
<html>
<head>
<meta charset="utf-8">
<title>Sign-up</title>
<link href="..\static\css\signup.css" rel="stylesheet">
<script src="https://kit.fontawesome.com/a81368914c.js"></script>
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">
</head>
<body>
 <!--container---------------------->
```

```
<div class="container" >
 <!--sign-up-box-container--->
<div class="sign-up">

   <div id="png"><a href="/" title="HOME"><img style="width:55px; height:55px ; " src="..\static\images\home-page.png"></a></div>
 <!--heading-->
 <form action="/register1" method="post">

 <h1 class="heading">Register Here</h1><br>
<div
class="para">               <b>Welcome</b>,here you have to fill the blanks for registration</div><br>
 <!--name-box-->
 <div class="text">
 <img height="20px" src="..\static\images\user.png" />
 <input placeholder="Name" type="text" name="username"/>
 </div>
 <!--Email-box-->
 <div class="text">
 <img height="12px" src="..\static\images/email.png" />
 <input placeholder=" Example@gmail.com" type="email" name="email" />
 </div>
 <!--Password-box-->
 <div class="text">
 <img height="20px" src="..\static\images\password.png" />
 <input placeholder=" Password" type="password" name="password"/>
 </div>

<!--trems-->

<div class="trems">
   <input class="check" type="checkbox" required/>
 <p class="conditions">I read and agree to <a href="#">Terms &amp;amp; Conditions</a></p>
 </div>
<!--button-->
<div class="toop">
```

```
<button type="submit" class="btn btn-primary" >CREATE ACCOUNT</button> </div><br>

</form>
<!--sign-in-->
<div class="t">             <p
class="conditions" id="p3">Already have an account ?  <a href="/login">Login</a></p> </div></div>
 </div>
<!--text-container-->
<div class="text-container">

 <h1 style="color: #2d2c2c;font-family:cursive;">Glad to see you</h1>

 <div class="diag"><img class="fig1" width="70%" height="100%" src="..\static\images\signup2.png"></div>



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

View.html

```
{% extends "layout.html" %}

{% block content %}
{% with messages = get_flashed_messages() %}
  {% if messages %}

  <script type="text/javascript">
    var m={{ messages|safe }};
    for(var i=0;i<m.length;i++)
    {
      alert(m[i]);
    }
  </script>

  {% endif %}
```

```
{% endwith %}

<table id="example" class="display" style="width:100%">
    <thead>
        <tr>
            <th>Sno</th>
            <th>Name</th>
            <th>Email</th>
            <th>Phno</th>
            <th>Blood_group</th>
            <th>Weight</th>
            <th>Gender</th>
            <th>Dob</th>
            <th>Address</th>
            <th>Adharno</th>
            <th>Action</th>
            <th></th>
            <th></th>
        </tr>
    </thead>
    <tbody>
      {% for d in data %}
        <tr>
            <td>{{d[0]}}
            <td>{{d[1]}}</td>
            <td>{{d[2]}}</td>
            <td>{{d[3]}}</td>
            <td>{{d[4]}}</td>
            <td>{{d[5]}}</td>
            <td>{{d[6]}}</td>
            <td>{{d[7]}}</td>
            <td>{{d[8]}}</td>
            <td>{{d[9]}}</td>



            <td>
              <form action="{{url_for('edit')}}" method="POST">
```

```
            <button value={{d[0]}} name="edit">Edit</button>
            </form>
           </td>

           <td>
            <form action="{{url_for('hold')}}" method="POST">
            <button value={{d[0]}} name="hold">Hold</button>
            </form>
           </td>



           <td>
            <form action="{{url_for('delete')}}" method="POST">
            <button value={{d[0]}} name="delete">Delete</button>
            </form>
           </td>

         </tr>
        {% endfor %}
      </tbody>
</table>

<script type="text/javascript">

$(document).ready(function() {
   $('#example').DataTable( {
      dom: 'Bfrtip',
      buttons: [
         'copy', 'csv', 'excel', 'pdf', 'print'
      ]
   } );
} );

</script>

<div class="container-fluid"> <!-- Header content -->
   <nav class="navbar navbar-inverse">
```

```
 <div class="container-fluid">
  <div class="navbar-header">
    <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#myNavbar">
      <span class="icon-bar"></span>
      <span class="icon-bar"></span>
      <span class="icon-bar"></span>
    </button>

    <div>
  <ul class="nav navbar-nav navbar-right">
      <li><a href="{{url_for('inactive')}}"><span class="glyphicon glyphicon-log-out"></span> Inactive</a></li>
 </ul>
 </div>

  <div>
  <ul class="nav navbar-nav navbar-right">
      <li><a href="{{url_for('logout')}}"><span class="glyphicon glyphicon-log-out"></span> Logout</a></li>
  </ul>
  </div>

 </div>
 </div>
 </nav>
</div>
{% endblock %}
```

View2.html

```
{% extends "layout.html" %}

{% block content %}

{% with messages = get_flashed_messages() %}
  {% if messages %}

 <script type="text/javascript">
   var m={{ messages|safe }};
```

```
   for(var i=0;i<m.length;i++)
   {
     alert(m[i]);
   }
  </script>

  {% endif %}
{% endwith %}

<table id="example" class="display" style="width:100%">
        <thead>
            <tr>
                <th>Sno</th>
                <th>Name</th>
                <th>Email</th>
                <th>Blood_group</th>
                <th>Weight</th>
                <th>Gender</th>
                <th>Dob</th>
                <th>Address</th>
                <th>Adharno</th>
                <th>Action</th>
            </tr>
        </thead>
        <tbody>
            {% for d in data %}
            <tr>
                <td>{{d[0]}}
                <td>{{d[1]}}</td>
                <td>{{d[2]}}</td>
                <td>{{d[4]}}</td>
                <td>{{d[5]}}</td>
                <td>{{d[6]}}</td>
                <td>{{d[7]}}</td>
                <td>{{d[8]}}</td>
                <td>{{d[9]}}</td>
```

```
            <td>
              <form action="{{url_for('send')}}" method="POST">
              <button value={{d[0]}} name="send">Send sms</button>
              </form>
            </td>


          </tr>
          {% endfor %}
      </tbody>
</table>


<script type="text/javascript">

$(document).ready(function() {
   $('#example').DataTable( {
      dom: 'Bfrtip',
      buttons: [
         'copy', 'csv', 'excel', 'pdf', 'print'
      ]
   } );
} );

</script>
{% endblock %}
```

Login.css

```
*{
          padding: 0;
          margin: 0;
          box-sizing: border-box;
}
```

```
body{
    font-family: 'Poppins', sans-serif;
    overflow: hidden;
}

.wave{
        position: fixed;
        bottom: 0;
        left: 0;
        height: 100%;
        z-index: -1;
}

.container{
    width: 100vw;
    height: 100vh;
    display: grid;
    grid-template-columns: repeat(2, 1fr);
    grid-gap :7rem;
    padding: 0 2rem;
}

.img{
        display: flex;
        justify-content: flex-end;
        align-items: center;
}

.login-content{
        display: flex;
        justify-content: flex-start;
        align-items: center;
        text-align: center;
}

.img img{
        width: 500px;
```

```
}

form{
        width: 360px;
}

.login-content img{
   height: 100px;
}

.login-content h2{
        margin: 15px 0;
        color: #333;
        text-transform: uppercase;
        font-size: 2.9rem;
}

.login-content .input-div{
        position: relative;
   display: grid;
   grid-template-columns: 7% 93%;
   margin: 25px 0;
   padding: 5px 0;
   border-bottom: 2px solid #d9d9d9;
}

.login-content .input-div.one{
        margin-top: 0;
}

.i{
        color: #d9d9d9;
        display: flex;
        justify-content: center;
        align-items: center;
}
```

```
.i i{
	transition: .3s;
}

.input-div > div{
   position: relative;
	height: 45px;
}

.input-div > div > h5{
	position: absolute;
	left: 10px;
	top: 50%;
	transform: translateY(-50%);
	color: #999;
	font-size: 18px;
	transition: .3s;
}

.input-div:before, .input-div:after{
	content: '';
	position: absolute;
	bottom: -2px;
	width: 0%;
	height: 2px;
	background-color: #38d39f;
	transition: .4s;
}

.input-div:before{
	right: 50%;
}

.input-div:after{
	left: 50%;
}
```

```
.input-div.focus:before, .input-div.focus:after{
	width: 50%;
}

.input-div.focus > div > h5{
	top: -5px;
	font-size: 15px;
}

.input-div.focus > .i > i{
	color: #38d39f;
}

.input-div > div > input{
	position: absolute;
	left: 0;
	top: 0;
	width: 100%;
	height: 100%;
	border: none;
	outline: none;
	background: none;
	padding: 0.5rem 0.7rem;
	font-size: 1.2rem;
	color: #555;
	font-family: 'poppins', sans-serif;
}

.input-div.pass{
	margin-bottom: 4px;
}

a{
	display: block;
	text-align: right;
	text-decoration: none;
	color: #999;
```

```
        font-size: 0.9rem;
        transition: .3s;
}

a:hover{
        color: #38d39f;
}

.btn{
        display: block;
        width: 100%;
        height: 50px;
        border-radius: 25px;
        outline: none;
        border: none;
        background-image: linear-gradient(to right, #32be8f, #38d39f, #32be8f);
        background-size: 200%;
        font-size: 1.2rem;
        color: #fff;
        font-family: 'Poppins', sans-serif;
        text-transform: uppercase;
        margin: 1rem 0;
        cursor: pointer;
        transition: .5s;
}
.btn:hover{
        background-position: right;
}


@media screen and (max-width: 1050px){
        .container{
                grid-gap: 5rem;
        }
}

@media screen and (max-width: 1000px){
```

```
        form{
                width: 290px;
        }

        .login-content h2{
    font-size: 2.4rem;
    margin: 8px 0;
        }

        .img img{
                width: 400px;
        }
}

@media screen and (max-width: 900px){
        .container{
                grid-template-columns: 1fr;
        }

        .img{
                display: none;
        }

        .wave{
                display: none;
        }

        .login-content{
                justify-content: center;
        }
}
.container{
        overflow:scroll
}
.container::-webkit-scrollbar {
   display: none;
}
```

```
ul {
        position:relative;
        top:-20px;
        left:0%;
        right:10%;
        transform: translate(-50%, -50%);
        margin: 75px;
        padding:0;
        display:flex;
        flex:auto;
 }

 ul li {
        list-style: none;
 }

 ul li a {
        position: relative;
        width:60px;
        height:60px;
        display:block;
        text-align:center;
        margin:0 10px;
        border-radius: 50%;
        padding: 6px;
        box-sizing: border-box;
        text-decoration:none;
        box-shadow: 0 10px 15px rgba(0,0,0,0.3);
        background: linear-gradient(0deg, #ddd, #fff);
        transition: .5s;
 }

 ul li a:hover {
        box-shadow: 0 2px 5px rgba(0,0,0,0.3);
        text-decoration:none;
 }
```

```
ul li a .fab {
        width: 100%;
        height:100%;
        display:block;
        background: linear-gradient(0deg, #fff, #ddd);
        border-radius: 50%;
        line-height: calc(60px - 12px);
        font-size:24px;
        color: #262626;
        transition: .5s;
}

ul li:nth-child(1) a:hover .fab {
        color: #3b5998;
}

ul li:nth-child(2) a:hover .fab {
        color: #00aced;
}

ul li:nth-child(3) a:hover .fab {
        color: #dd4b39;
}

ul li:nth-child(4) a:hover .fab {
        color: #007bb6;
}

ul li:nth-child(5) a:hover .fab {
        color: #e4405f;
}
.app{
          position: relative;
          top: -70px;
          height: 5%;
```

```
 }
 #app1{
        font-style: oblique;
        color:blue ;
 }
 #png{
        position: relative;
        top: -300px;
        right: 50px;

 }
```

Signup.css

```
@charset "utf-8";
/* CSS Document */
body{
 background-color:#eef1f8;
 margin:0px;
 padding:0px;
}
a{
 text-decoration:none;
}
.container{
 width:100%;
 height:100%;
 background-color:#FFFFFF;
 position: absolute;
 left:50%;
 top:49%;
 transform:translate(-50%,-50%);
 box-shadow:2px 2px 30px rgba(66,57,238,0.2);
 border-radius:20px;
 display:flex;
 justify-content: center;
 align-items:center;
 background-image: url('../images/bg.gif');
```

```
 background-repeat:no-repeat;
 background-size:cover;
}
.sign-up{
        position: relative;
        left: -250px;
 width:50%;
 display:flex;
 flex-direction:column;
 align-items: center;

}
.text-container{
   position: relative;
   top:-100px; padding-bottom:-50px;
 width:50%;
 height:100%;
 display:flex;
 flex-direction:column;
 justify-content: center;
 align-items: center;
 margin-left:700px;
 margin-bottom:-10px;
 top:-40px

}
.heading{
 font-family:calibri;
 color:rgba(30,30,30,1);
position: relative;
left:80px ;
}
.text{
 width:350px;
 height:50px;
 box-shadow:2px 6px 18px rgba(66,57,238,0.3);
 border-radius: 30px;
```

```
 display:flex;
 align-items:center;
 margin:10px;
}
.text input{
 height:40px;
 width:80%;
 outline:none;
 border:none;
 font-size:14px;
 margin:5px;
}
.text img{
 margin-left:20px;
}
.conditions{
 font-family:myriad pro;
 color:#bbc1cb;
 font-size:14px;
}
.trems{
    top:20px ;
    position: relative;
            left:37px ;

 display: flex;
 align-items:center;
}
.conditions a{
 color:#7d22e3;
 font-weight:500;
}
button{
 width:200px;
 height:40px;
 outline:none;
 border:none;
```

```
 border-radius:20px;
 background:linear-gradient(-30deg,#3b02ed,#8e2ae0 55%);
 box-shadow:2px 6px 16px rgba(66,57,238,0.3);
 color:#FFFFFF;
 font-weight:600;
 letter-spacing:1px;
 font-weight: 13px;
}
button:active{
 transform:scale(1.1);
}
.text-container p{
 width:70%;
 text-align: center;
 font-family:arial;
 font-size: 15px;
 font-weight: 400;
 line-height:0px;
}
ul {
        position:relative;
        top:40px;
        left:25%;
        right:100px;
        transform: translate(-50%, -50%);
        margin: 75px;
        padding:0;
        display:flex;
        flex:auto;
  }

  ul li {
        list-style: none;
  }

  ul li a {
        position: relative;
```

```
        width:60px;
        height:60px;
        display:block;
        text-align:center;
        margin:0 10px;
        border-radius: 50%;
        padding: 6px;
        box-sizing: border-box;
        text-decoration:none;
        box-shadow: 0 10px 15px rgba(0,0,0,0.3);
        background: linear-gradient(0deg, #ddd, #fff);
        transition: .5s;
}

ul li a:hover {
        box-shadow: 0 2px 5px rgba(0,0,0,0.3);
        text-decoration:none;
}

ul li a .fab {
        width: 100%;
        height:100%;
        display:block;
        background: linear-gradient(0deg, #fff, #ddd);
        border-radius: 50%;
        line-height: calc(60px - 12px);
        font-size:24px;
        color: #262626;
        transition: .5s;
}

ul li:nth-child(1) a:hover .fab {
        color: #3b5998;
}

ul li:nth-child(2) a:hover .fab {
        color: #00aced;
```

```
}

ul li:nth-child(3) a:hover .fab {
        color: #dd4b39;
}

ul li:nth-child(4) a:hover .fab {
        color: #007bb6;
}

ul li:nth-child(5) a:hover .fab {
        color: #e4405f;
}
.toop{
   position: relative;
   top: 20px;
         left: 75px;
}
.check{
   position: relative;
   top: -8px;
   left: -5px;
}
.para{
   position: relative;
   top: 0px;
         left: -45px;

}
.diag{
         position: relative;

         top:0px;
         margin: 0px;
         padding: 0px;
         left:-10px;
}
```

```
.fig1{
        position: relative;
        size:200%;
}
#png{
        position: relative;
        left: -270px;
        top: -45px;

}
.or{
        position: relative;
        left:180px;
}
.s1{
        position: relative;
        left: 140px;
}
.t{
        position: relative;
        left:-100px ;
        top:10px;
}
```

Docker file

```
FROM python:3.6.5-alpine
WORKDIR /app
ADD . /app

RUN set -e; \
    apk add --no-cache --virtual .build-deps \
        gcc \
        libc-dev \
        linux-headers \
        mariadb-dev \
        python3-dev \
```

```
            postgresql-dev \
      ;
COPY requirements.txt /app
RUN pip install -r requirements.txt
CMD ["python","app.py"]
```

Index.py

```
from flask import *
from flask import Flask, render_template, request, redirect, url_for, session
from twilio.rest import Client
from werkzeug.utils import secure_filename
import ibm_db
import re
import os
from sendgrid import SendGridAPIClient
from sendgrid.helpers.mail import Mail
import csv
app=Flask(__name__)
app.secret_key="don't share"
myconn=ibm_db.connect('DATABASE=bludb;HOSTNAME=3883e7e4-18f5-4afe-be8c-
fa31c41761d2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=31498;SECURITY=SSL;SSLServerCertificate
=DigiCertGlobalRootCA.crt;UID=wqr23763;PWD=dyDwgsrX1kt4wgJn', '', ''
            )
@app.route("/signup")
def signup():
   return render_template("signup.html")



@app.route('/register1', methods =['GET', 'POST'])
def register1():
   msg = ''
   if request.method == 'POST' :
      username = request.form['username']
      email = request.form['email']
      password = request.form['password']
```

```
    query = 'SELECT * FROM admin WHERE username =?;'
    stmt=ibm_db.prepare(myconn,query)
    ibm_db.bind_param(stmt,1,username)
    ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)
    print(account)
    if account:
       msg = 'Account already exists !'
    elif not re.match(r'[^@]+@[^@]+\.[^@]+', email):
       msg = 'Invalid email address !'
    elif not re.match(r'[A-Za-z0-9]+', username):
       msg = 'name must contain only characters and numbers !'
    else:
       query = "INSERT INTO ADMIN VALUES (?,?,?)"
       stmt=ibm_db.prepare(myconn,query)
       ibm_db.bind_param(stmt,1,username)
       ibm_db.bind_param(stmt,2,email)
       ibm_db.bind_param(stmt,3,password)
       ibm_db.execute(stmt)
       msg = 'You have successfully registered !'
       return render_template('login.html', msg = msg)


@app.route("/login",methods=['GET','POST'])
def login():
        if request.method=="POST":
                Username=request.form['Username']
                Password=request.form['Password']
                query="select * from admin where Username=? and password=?;"
                stmt=ibm_db.prepare(myconn, query)
                ibm_db.bind_param(stmt, 1, Username)
                ibm_db.bind_param(stmt, 2, Password)
                ibm_db.execute(stmt)
                data=ibm_db.fetch_assoc(stmt)
                if data:
                        session['loggedin']=True
```

```
                        flash("Login Successfully")
                        return  render_template('info.html')

                else:
                        flash("Incorrect Username or Password")
        return render_template("login.html")



@app.route("/")
@app.route("/bloodbank")
def bloodbank():
                return render_template("bloodbank.html")




@app.route("/home")
def home():

        query="select count(*) from donor where status=1"
        stmt = ibm_db.prepare(myconn, query)
        ibm_db.execute(stmt)
        data = ibm_db.fetch_tuple(stmt)
        return render_template("index.html",data=[data])



@app.route("/register",methods=['GET','POST'])
def register():
        if request.method=="POST":
                name=request.form['name']
                email=request.form['email']
                phno=request.form['phno']
                blood_group=request.form['blood_group']
                weight=request.form['weight']
                gender=request.form['gender']
                dob=request.form['dob']
                address=request.form['address']
```

```
                adharno=request.form['adharno']
                status=1

                query="select * from donor where adharno=(?);"
                stmt = ibm_db.prepare(myconn, query)
                ibm_db.bind_param(stmt, 1, adharno)
                ibm_db.execute(stmt)
                data = ibm_db.fetch_assoc(stmt)
                if (data)==0:
                        query = "INSERT INTO donor (NAME,EMAIL,PHNO,BLOOD_GROUP,WEIGHT,GENDER,DOB,ADDRESS,ADHARNO,STATUS) values(?,?,?,?,?,?,?,?,?,?)"
                        stmt = ibm_db.prepare(myconn, query)
                        ibm_db.bind_param(stmt, 1, name)
                        ibm_db.bind_param(stmt, 2, email)
                        ibm_db.bind_param(stmt, 3, phno)
                        ibm_db.bind_param(stmt, 4, blood_group)
                        ibm_db.bind_param(stmt, 5, weight)
                        ibm_db.bind_param(stmt, 6, gender)
                        ibm_db.bind_param(stmt, 7, dob)
                        ibm_db.bind_param(stmt, 8, address)
                        ibm_db.bind_param(stmt, 9, adharno)
                        ibm_db.bind_param(stmt, 10, status)
                        ibm_db.execute(stmt)
                        msg = 'You have successfully Logged In!!'
                        return redirect(url_for('viewall'))

                else:
                        flash("Already Registered")
                return redirect(url_for('register'))
        else:
                return render_template("about.html")



@app.route("/view",methods=['GET','POST'])
def view():
```

```
    if not session.get('loggedin'):
        return  render_template("login.html")
    query="select * from donor where status=1"
    stmt = ibm_db.prepare(myconn, query)
    ibm_db.execute(stmt)
    data=[]
    tuple = ibm_db.fetch_tuple(stmt)
    while tuple!=False:
        data.append(tuple)
        tuple=ibm_db.fetch_tuple(stmt)
    return render_template("view.html",data=data)



@app.route("/delete",methods=['GET','POST'])
def delete():
    if not session.get('loggedin'):
        return  render_template("login.html")
    if request.method=="POST":
        id=request.form['delete']

        query="delete from donor where sno=?"
        stmt = ibm_db.prepare(myconn, query)
        ibm_db.bind_param(stmt, 1, id)
        ibm_db.commit(stmt)
        flash("Deleted Successfully")
        return redirect(url_for('view'))



@app.route("/edit",methods=['GET','POST'])
def edit():
    if not session.get('loggedin'):
        return  render_template("login.html")
    if request.method=="POST":
        id=request.form['edit']
```

```
                query="select * from donor where sno=?"

                stmt = ibm_db.prepare(myconn, query)
                ibm_db.bind_param(stmt, 1, id)
                ibm_db.execute(stmt)
                data = ibm_db.fetch_tuple(stmt)
                return render_template("edit.html",data=data)



@app.route("/update",methods=['GET','POST'])
def update():
        if not session.get('loggedin'):
                return  render_template("login.html")
        if request.method=="POST":
                id=request.form['id']
                name=request.form['name']
                email=request.form['email']
                phno=request.form['phno']
                blood_group=request.form['blood_group']
                weight=request.form['weight']
                gender=request.form['gender']
                dob=request.form['dob']
                address=request.form['address']
                adharno=request.form['adharno']


                query = "INSERT INTO USER1 values(?,?,?,?,?,?,?,?,?,?)"
                stmt = ibm_db.prepare(myconn, query)
                ibm_db.bind_param(stmt, 1, id)
                ibm_db.bind_param(stmt, 2, name)
                ibm_db.bind_param(stmt, 3, email)
                ibm_db.bind_param(stmt, 4, phno)
                ibm_db.bind_param(stmt, 5, blood_group)
                ibm_db.bind_param(stmt, 6, weight)
                ibm_db.bind_param(stmt, 7, gender)
                ibm_db.bind_param(stmt, 8, dob)
```

```
                ibm_db.bind_param(stmt, 9, address)
                ibm_db.bind_param(stmt, 10, adharno)
                ibm_db.commit(stmt)
                return redirect(url_for('view'))



@app.route("/view2",methods=['GET','POST'])
def view2():

        query="select distinct blood_group from donor where status=1"
        stmt = ibm_db.prepare(myconn, query)
        ibm_db.execute(stmt)
        data=[]
        tuple = ibm_db.fetch_tuple(stmt)
        while tuple!=False:
                data.append(tuple)
                tuple=ibm_db.fetch_tuple(stmt)
        return render_template("select.html",data=data)



@app.route("/viewselected",methods=['GET','POST'])
def viewselected():
        blood_group=request.form['blood_group']
        query="select * from donor where blood_group= ? and status=1"
        stmt = ibm_db.prepare(myconn, query)
        ibm_db.bind_param(stmt, 1, blood_group)
        ibm_db.execute(stmt)
        data=[]
        tuple = ibm_db.fetch_tuple(stmt)
        while tuple!=False:
                data.append(tuple)
                tuple=ibm_db.fetch_tuple(stmt)
        return render_template("view2.html",data=data)



@app.route("/viewall",methods=['GET','POST'])
```

```
def viewall():

        query="select * from donor where status=1"
        stmt = ibm_db.prepare(myconn, query)
        ibm_db.execute(stmt)
        data=[]
        tuple = ibm_db.fetch_tuple(stmt)
        while tuple!=False:
                data.append(tuple)
                tuple=ibm_db.fetch_tuple(stmt)
        return render_template("view2.html",data=data)



@app.route("/")
@app.route("/send",methods=['GET','POST'])
def send():
        if request.method=="POST":
                id=request.form['send']


                query="select email from donor where sno=?"


                stmt = ibm_db.prepare(myconn, query)
                ibm_db.bind_param(stmt, 1, id)
                ibm_db.execute(stmt)
                data = ibm_db.fetch_assoc(stmt)
                print(data)
                message = Mail(from_email='deepasomu28@gmail.com',to_emails=data['EMAIL'],subject='Sending
with Twilio SendGrid is Fun',html_content='<strong>and easy to do anywhere, even with Python</strong>')
                try:
                        sg = SendGridAPIClient('key')
                        response = sg.send(message)
                        print(response.status_code)
                        print(response.body)
                        print(response.headers)
                except Exception as e:
                        print(e)
        return redirect('/viewall')
```

```
@app.route("/logout")
def logout():
        session['loggedin']=False
        return render_template("index.html")



@app.route("/hold",methods=['GET','POST'])
def hold():
        if not session.get('loggedin'):
                return  render_template("login.html")
        if request.method=="POST":
                id=request.form['hold']
                query="update donor set  status=0 where sno=?"
                stmt = ibm_db.prepare(myconn, query)
                ibm_db.bind_param(stmt, 1, id)
                ibm_db.execute(stmt)

                return redirect(url_for('view'))



@app.route("/activate",methods=['GET','POST'])
def activate():
        if not session.get('loggedin'):
                return  render_template("login.html")
        if request.method=="POST":
                id=request.form['hold']

                query="update donor set  status=1 where sno=?"
                stmt = ibm_db.prepare(myconn, query)
                ibm_db.bind_param(stmt, 1, id)
                ibm_db.execute3(stmt)

                return redirect(url_for('inactive'))
```

```
@app.route("/inactive",methods=['GET','POST'])
def inactive():
        if not session.get('loggedin'):
                return  render_template("login.html")

        query="select * from donor where status=0"
        stmt = ibm_db.prepare(myconn, query)
        ibm_db.execute(stmt)
        data = ibm_db.fetch_tuple(stmt)
        print(data)
        return render_template('inactive.html',data=[data])



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

Mail.py

```
import os
from sendgrid import SendGridAPIClient
from sendgrid.helpers.mail import Mail

message = Mail(
    from_email='deepasomu28@gmail.com',
    to_emails='sowmiyasivakalpana@gmail.com',
    subject='Sending with Twilio SendGrid is Fun',
    html_content='<strong>and easy to do anywhere, even with Python</strong>')
try:
    sg = SendGridAPIClient('key')
    response = sg.send(message)
    print(response.status_code)
    print(response.body)
    print(response.headers)
except Exception as e:
    print€
```

Sendmail.py

```
import smtplib
from flask import Flask
import sendgrid
import os
from sendgrid.helpers.mail import Mail, Email, To, Content
app=Flask(__name__)
SUBJECT = "expense tracker"
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.deepasomu28@gmail.com", "oms@1Ram")
    message  = 'Subject: {}\n\n{}'.format(SUBJECT, TEXT)
    s.sendmail("il.deepasomu28@gmail.com", email, message)
    s.quit()
def sendgridmail(user,TEXT):

    from_email = Email("deepasomu28@gmail.com")
    to_email = To(user)
    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 = s.client.mail.send.post(request_body=mail_json)
    print(response.status_code)
    print(response.headers)

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

12.2.GitHub & Project Demo Link

https://youtu.be/gr7zCIsOBh4