

IBM NALAIYATHIRAN PROJECT REPORT

PLASMA DONOR APPLICATION

Team id	PNT2022TMID04796
Project Name	Plasma Donor Application
Team Members	Aravind S Arul Prasanna S ArulMurugan P Arun K

Table Of Contents

SI No	Title	Page No
1	INTRODUCTION 1.1 Project Overview 1.2 Purpose	4 4
2	LITERATURE SURVEY 2.1 Existing problem 2.2 References 2.3 Problem Statement Definition	5 5 5
3	IDEATION & PROPOSED SOLUTION 3.1 Empathy Map Canvas 3.2 Ideation & Brainstorming 3.3 Proposed Solution 3.4 Problem Solution fit	6 7 8 10
4	REQUIREMENT ANALYSIS 4.1 Functional requirement 4.2 Non-Functional requirements	11 11
5	PROJECT DESIGN 5.1 Data Flow Diagrams 5.2 Solution & Technical Architecture 5.3 User Stories	12 13 14

6	PROJECT PLANNING & SCHEDULING 6.1 Sprint Planning & Estimation 6.2 Sprint Delivery Schedule 6.3 Reports from JIRA	15 16 16
7	CODING & SOLUTIONING 7.1 Feature 1 7.2 Feature 2 7.3 Database Schema (if Applicable)	17 18 18
8	TESTING 8.1 Test Cases 8.2 User Acceptance Testing	20 21
9	RESULTS 9.1 Performance Metrics	22
10	ADVANTAGES & DISADVANTAGES	28
11	CONCLUSION	29
12	FUTURE SCOPE	29
13	APPENDIX 13.1 Source Code 13.2 Github & Demo link	30 42

1.INTRODUCTION

1.1 Project Overview

Plasma is a critical part of the treatment for many serious health problems. Therefore, there are blood drives asking people to donate blood plasma. The main goal of our project is to make it easier for the COVID-19 patients to get a plasma donor easily as well as donate plasma if they have recovered. The system targets two types of users: the people who want to donate plasma and the people who need plasma. The user can also view the total active cases, nearby vaccine centres, hospitals address.

The main objective of developing the website is to make it easier for the COVID-19 patients to get a plasma donor easily and as soon as possible. Yet, the need for plasma-derived products has been strongly increasing for some years, and blood collection agencies have to adapt if they want to meet this demand. This article aims to review the main motivations and deterrents to whole blood donation, and to compare them with those that we already know concerning plasma donation. Current evidence shows similarities between both behaviours, but also differences that indicate a need for further research regarding plasma donation.

1.2 Purpose

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. Regarding the problem faced, an 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

- Only web-based system is available no mobile based system is available
- Less Security
- No proper coordination between different applications and users
- Cannot upload and download the latest updates at right time
- Fewer users-friendly

2.2 REFERENCE

Several experiments have been carried out over the years by different groups of researchers. Here are some of the following groups:

[1] Denuis O'Neil (1999). "Blood component" Archived from the original on June 5, 2013.

[2] ways to keep your plasma healthy, Original Archived November 1, 2013, Accessed November 11, 2011.

[3] Ripathis S, Kumar V, Prabhakar A, Joshi S, Agarwal A (2015). "Microscale Passive Plasma Separation: A Review of Design Principles and Microdevices," J. Micromech Micro 25 (8): 083001;

[4] P. C. P. C. a. V. I. M. Yan, "Building a chatbot with server less computing," IBM watson research center, 2016.

[5] S. E. a. B. J. J. Short, "Cloud Event Programming Paradigms: Applications and Analysis," 9th IEEE International Conference on Cloud Computing (CLOUD), pp. pp. 4 00-406, 2017.

2.3 Problem Statement Definition

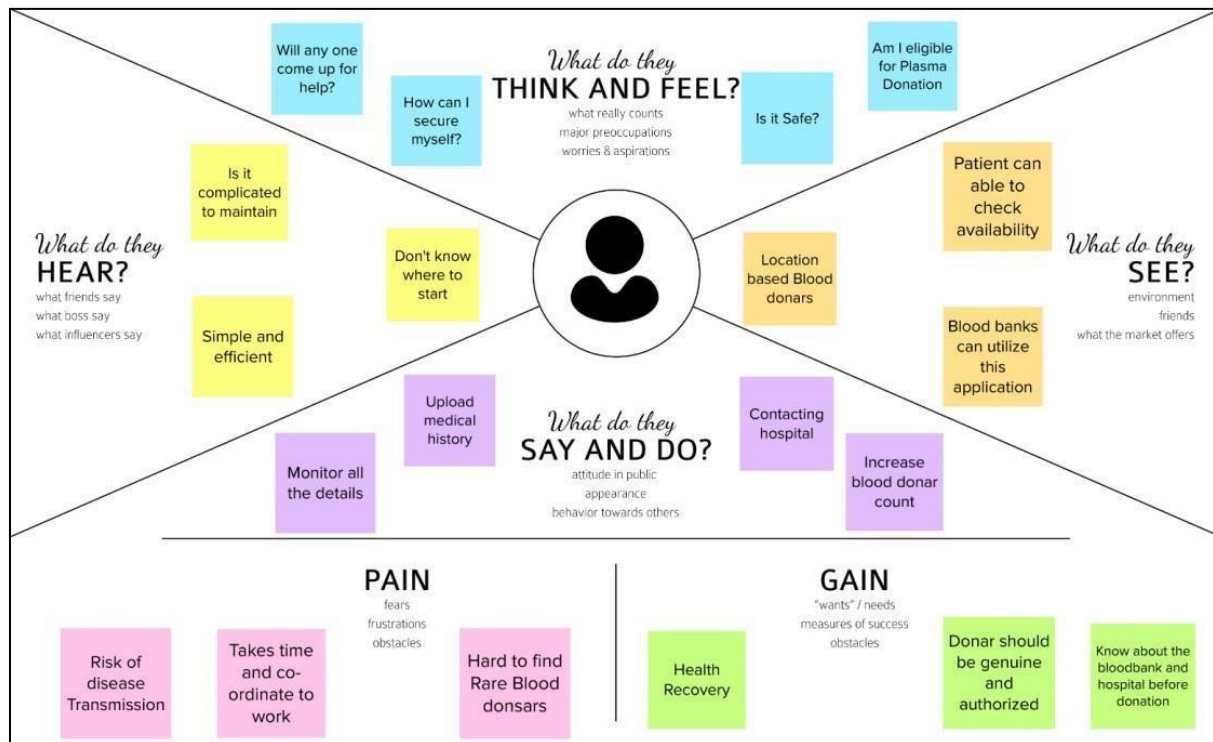
During COVID 19 crisis the requirement for plasma increased drastically as there were no vaccinations found in order to treat the infected patients.

In such situation it was very difficult to find the plasma donor, check whether the donor was infected previously and was recovered, and which donor is eligible to donate plasma was a challenging task.

As the plasma therapy was one of the ways to treat the infected patients getting the donor details played a major role.

3. IDEATION AND PROPOSED SYSTEM

3.1 Empathy Map Canvas



3.2 Ideation and Brainstorming

Conducting a brainstorm

Executing a brainstorm isn't unique; holding a productive brainstorm is. Great brainstorming is one that sets the stage for fresh and generative thinking through simple guidelines and an open and collaborative environment. Use this when you're just kicking-off a new project and want to hit the ground running with big ideas that will move your team forward.

- 15 minutes to complete
- 10-15 minutes to facilitate
- 2-4 people recommended

Meta Meta

Show template feedback

Set the goal

To develop an plasma donor application that enables user to join as a donor anywhere and in anyplace and the information can be used whenever necessary situation occur

Tools used for development

Flink
IBM cloud
Docker
send grid

problem statement

To develop an plasma donor application that enables user to join as a donor anywhere and in anyplace and the information can be used whenever necessary situation occur

15 minutes

Problem statement

To propose an application where the blood banks can timely update the Blood stock availability and helps to register themselves to donor and user can find blood availability near by their location

Brainstorm solo

Define the ideas that come to mind that address your problem statement

15 minutes

shriya

Find Plasma availability nearby

sujitha

Donor shall be aged between 18 to 60

Female Donor must conceived in past

Donors should not have thyroid

satish kumar

urgent time of a blood requirement, user can quickly check for blood banks

shriram

Donor should not have covid

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

Donor details are stored in cloud

it is available in both android device and web portal

LOGIN CREDENTIAL

This application provides a reliable platform to connect local donors with

it is active user friendly.

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.

15 minutes

Importance

Feasibility

After you collaborate

You can export the mind as an image or pdf to share with members of your company who might find it helpful.

Quick add-ons

- Show the mind: Share a link to the mind with collaborators to keep them in the loop about the activity of the session.
- Export the mind: Export a copy of the mind as a PDF or PNG to share with others, including a link to the mind.

Keep moving forward!

Sharing templates

Share the template of a template as a template.

Customer experience journey map

Understand customer needs, motivations, and behaviors for an experience.

Strengths, weaknesses, opportunities & threats

Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.

Show template feedback

3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The main aim of this project is to help the people who needs blood in emergency and to associate some donors who are willing to donate their blood to needy people and save their lives.
2.	Idea / Solution description	The user will be able to : <ul style="list-style-type: none"> ➤ Search donors of suitable blood groups and contact them if needed. ➤ Donate blood by registering themselves with our system and can also become donors. ➤ Will be able to see the stock of various blood groups. ➤ Send request for blood via “contact us”. ➤ Get information about all the blood campaigns..
3.	Novelty / Uniqueness	All of them have different ideas and different queries. Based on the user request and experience we will update our project based on user convenience .
4.	Social Impact / Customer Satisfaction	With the right implementation of the software you can benefit in many ways and also it makes the management very easy and error free. The software helps in tracking donors, getting Prompt and Correct Reports when required, and Centralized data storage with security. And last but not the least; the software will help in Customer Satisfaction.
5.	Business Model (Revenue Model)	Hospitals, NGOs, and private groups will profit from this donation application. Anyone with a basic understanding can use this software. This can be utilised at any time, anywhere. working With the assistance of the government, we can a programme to assist persons in need of plasma.

		<pre> graph TD User1((User)) -- "BLOOD REQUEST" --> WA[WEB APPLICATION] WA -- "SEARCH" --> WA WA -- "DISPLAY" --> WA User2((User)) -- "POST DETAILS" --> WA WA -- "FINDS DONOR" --> User1 </pre>
6.	Scalability of the Solution	<p>Instead of scouring the entire world for plasma donors, this programme enables users to find donors while sitting at home. once there is an emergency, send a plasma request to all people. the donor is prepared to Donor recipient is informed of the donation. Receiver may get in touch with the donor. Due to this Donors can check their eligibility on an app as well as making it simpler to find a suitable donor.</p>

3.4 Problem Statement Fit

Identify strong TR & EM	3. TRIGGERS TR What triggers customers to act? Volunteering interest and social responsibility towards society triggers the people to use this application	10. YOUR SOLUTION SL <ul style="list-style-type: none"> Donors will be searched with blood groups in our database, if needed. The volunteers can donate the blood with their interest and become donors by registering themselves. Stock monitoring will be done and updation happens at the same time. An application which will act as the intermediate between the hospital and donors and bridge the gap between them. 	8. CHANNELS of BEHAVIOUR CU 8.1 ONLINE What kind of actions do customers take online? Registering for plasma donation and requesting for plasma will be carried out through online 8.1 OFFLINE What kind of actions do customers take offline? <ul style="list-style-type: none"> Arrangements for plasma donation Awareness for more plasma donation 	Identify strong TR & EM
	4. EMOTIONS: BEFORE / AFTER EM How do customers feel when they face a problem or a job and afterwards? <u>Before :</u> <ul style="list-style-type: none"> Hard to find the donors for plasma donation at the right time. <u>After :</u> <ul style="list-style-type: none"> Satisfactory feel and relaxed feel after getting the right donor. 			

Define CS, fit into CC	1. CUSTOMER SEGMENT(S) CS Who is your customer? People who wish to donate plasma and Hospitals & Blood banks which needs plasma donors.	6. CUSTOMER CONSTRAINTS CC What constraints prevent your customers from taking action or limit their choices of solutions? <ul style="list-style-type: none"> Network Bandwidth Donor Health condition Lack of knowledge about app Unavailability of plasma 	5. AVAILABLE SOLUTIONS AS Which solutions are available to the customers when they face the problem <ul style="list-style-type: none"> Available solutions notify about the donors, patients and the availability of plasma & need for the plasma. The notification regarding the need for plasma was not send to the donors. 	Explore AS, differentiate
	2. JOBS-TO-BE-DONE / PROBLEMS J&P Which jobs-to-be-done (or problems) do you address for your customers? <ul style="list-style-type: none"> Data collection should be monitored properly with donor's data security. Unawareness about the need of plasma donation. Demand for donors. 			
Focus on J&P, tap into BE, understand RC	9. PROBLEM ROOT CAUSE RC What is the real reason that this problem exists? What is the back story behind the need to do this job? <ul style="list-style-type: none"> Lack of unawareness about the importance of plasma donation. Inability to find the donors at the time of emergency. Decrease in donors count 	7. BEHAVIOUR BE What does your customer do to address the problem and get the job done? <ul style="list-style-type: none"> An unique ID will be provided for the donor's, in order to maintain their personal privacy. At the same time, an unique ID will be issued to the patient and the records will be monitored. Both donor and patient can access the application at ease. 	Focus on BE, tap into CS, understand RC	

4. REQUIREMENT ANALYSIS

4.1 Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Access Website	Software operator should be capable to access web- application through either an application browser or similar on the pc.
FR-2	Software operator Registration	The software operator should be able to register through the web-application. The donor software operator must provide user name,gender,blood/plasma group,location,contact.
FR-3	Login/logout/update details	The login information will be stored on the database for future use.
FR-4	Search for donor	Search result can be viewed in a list.Each element in the list represents a specific donor with the donor details.
FR-5	User plasma request	Users can request to donate plasma by filling out the request form on the page. Once the request is submitted, they will get an email.
FR-6	View distribution details	The plasma bank should be able to view the status of the distribution details.

4.2 Non-functional Requirements:

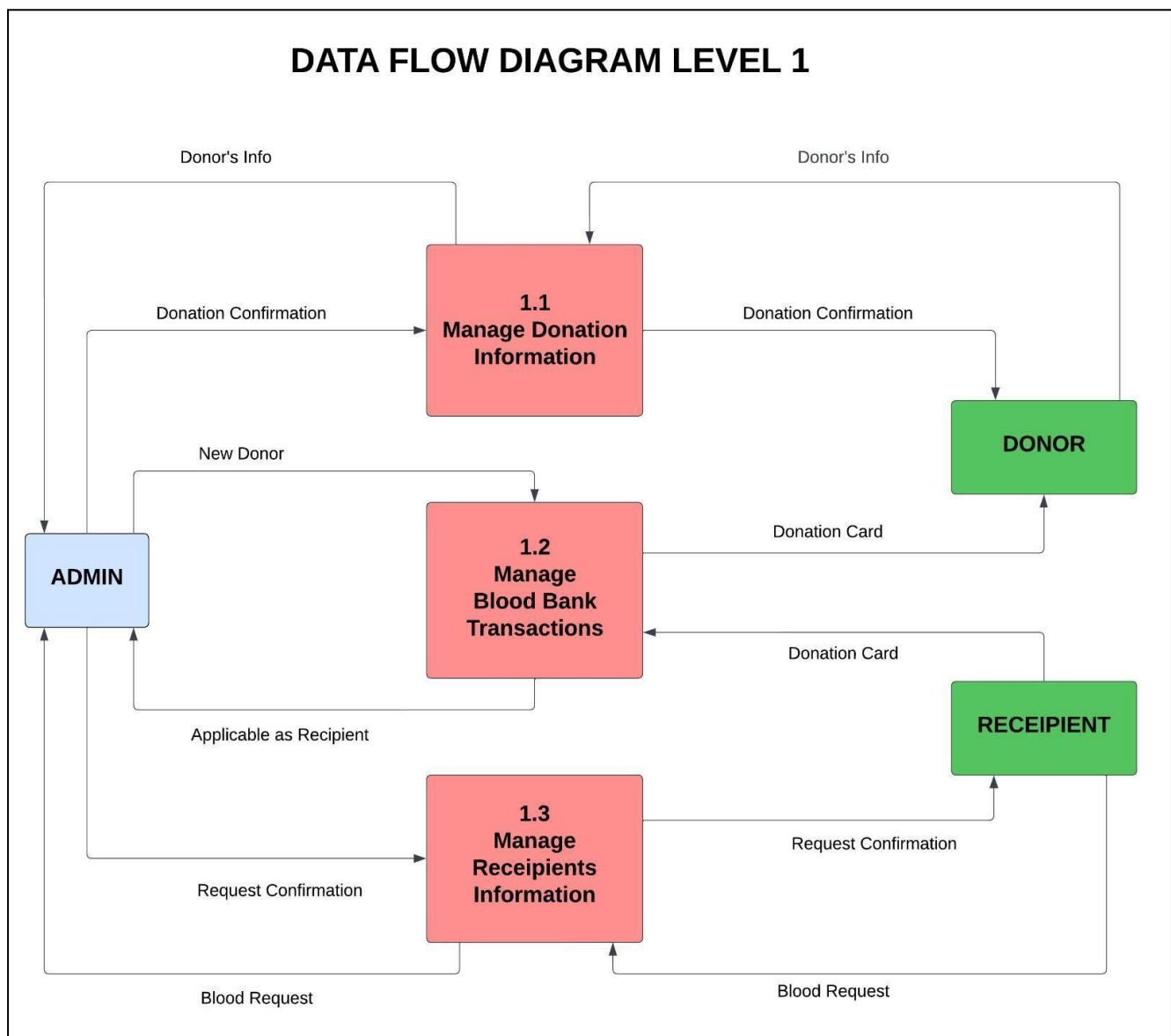
Following are the non-functional requirements of the proposed solution.

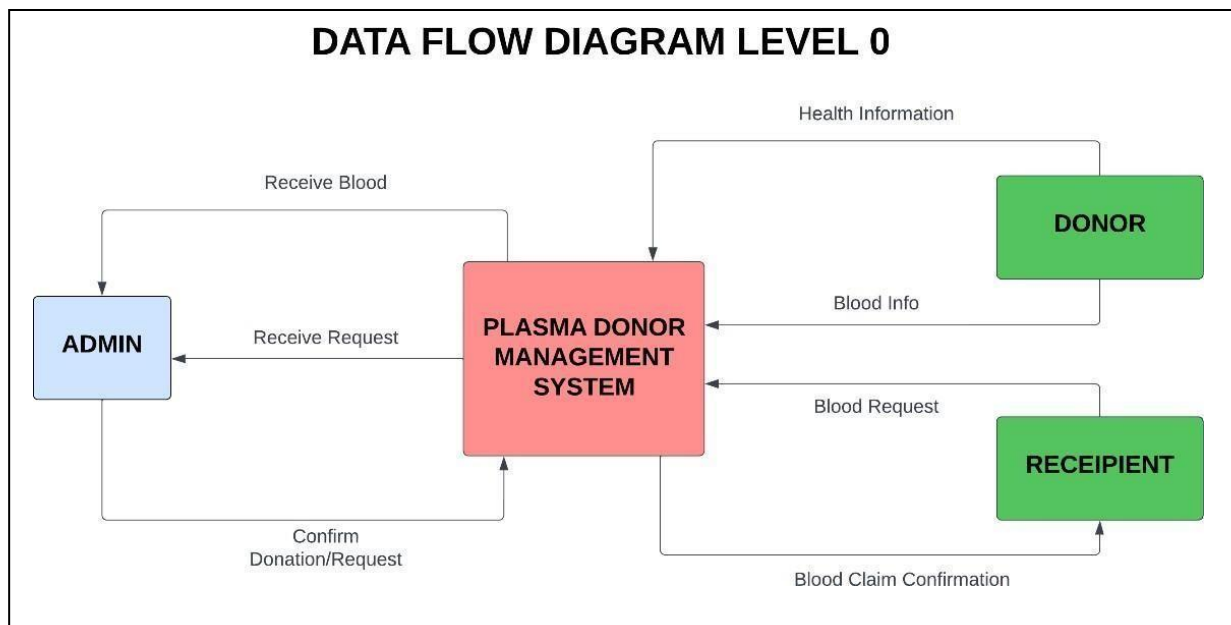
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	The plasma donor application must have a good looking user friendly interface.
NFR-2	Security	The plasma donor application must be secured with proper user name and passwords.
NFR-3	Reliability	The plasma donor application should work properly,even when faults occur.
NFR-4	Performance	The plasma donor application must perform well

		in different scenarios.
NFR-5	Availability	The plasma donor application must available 24 hours a day with no bandwidth issues.
NFR-6	Scalability	The plasma donor application should able to increase or decrease in performance and cost in response to changes in application and system processing demands.

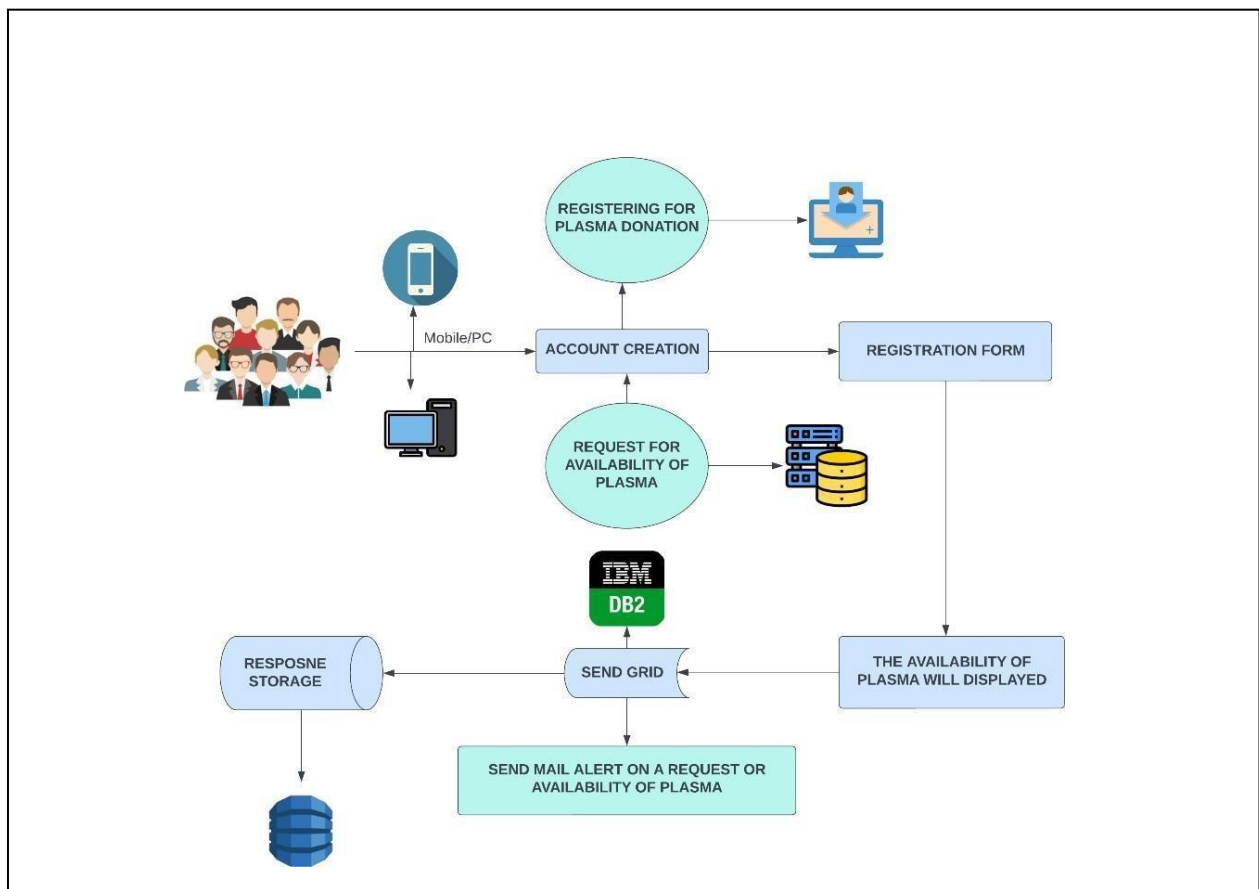
5. PROJECT DESIGN

5.1 Data Flow Diagram:





5.2 Solution and Technical Architecture :



5.3 User Stories:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail	I can receive confirmation email & click confirm	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can enter into my account	High	Sprint-1
	Dashboard	USN-6	As a user, Display all details about plasma application	I can donate/get details about the plasma	High	Sprint-2
Customer (Web user)	Application	USN-7	As a user, I can register, login and see details about plasma	I can access the donor details and availability of plasma	High	Sprint-3
Customer Care Executive	Update Plasma storage	USN-8	Keep track the availability of the Plasma	I can provide application for customer needs	High	Sprint-4
Administrator	Verify donor details	USN-9	To add the donor plasma details in application	I can Control the all details in this application	Medium	Sprint-3
Customer Care Executive	Verify Customer Feedback	USN-10	To design the application that meets user's desires	I can satisfy the customer expectations	Medium	Sprint-4
Customer Care Executive	Control all Plasma details	USN-11	Make sure to check the availability of plasma in application	I can alert notification through email and SMS	High	Sprint-2
Administrator	Performance of application	USN-12	To make the process more efficient	I can save time, cost by improving the Plasma management application	High	Sprint-4

6.

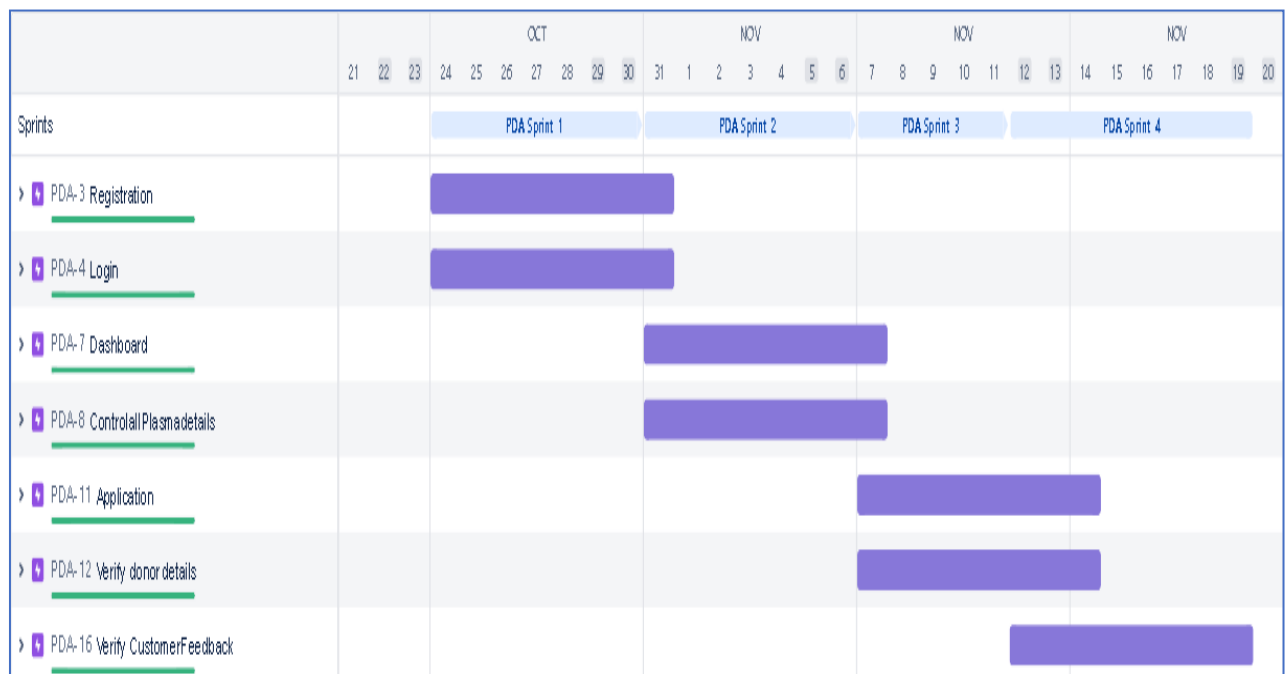
PROJECT PLANNING AND SCHEDULING**6.1 Sprint Planning & Estimation**

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	1	High	Arul Prasanna Arun
Sprint-1	Login	USN-2	As a user, I can log into the application by entering email & password	1	High	Arun Arul Murugan
Sprint-2	Dashboard	USN-3	As a user ,Display all details about plasma application	1	High	Arul Prasanna Aravind
Sprint-3	Application	USN-4	As a user ,I can register, login and see details about plasma	1	High	ArulMurugan Arun
Sprint-3	Verify donor details	USN-5	To add the donor plasma details in application	1	Medium	Arun Aravind
Sprint-2	Control all Plasma details	USN-6	Make sure to check the availability of plasma in application	1	High	Arul Murugan Arul Prasanna
Sprint-4	Verify Customer Feedback	USN-7	To design the application that meets user's desires	2	Medium	Arul Prasanna Aravind

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	30	6 Days	25 Oct 2022	30 Oct 2022	30	30 Oct 2022
Sprint-2	30	6 Days	1 Nov 2022	6 Nov 2022	30	6 Nov 2022
Sprint-3	30	6 Days	8 Nov 2022	13 Nov 2022	30	13 Nov 2022
Sprint-4	30	6 Days	15 Nov 2022	20 Nov 2022	30	20 Nov 2022

6.3 Reports from JIRA



7. CODING & SOLUTIONING

7.1 Feature 1:

Python

- Python is a widely-used, interpreted, object-oriented, and high-level programming language with dynamic semantics, used for general-purpose programming. It's everywhere, and people use numerous Python-powered devices on a daily basis, whether they realize it or not.
- Python was created by Guido van Rossum, and first released on February 20, 1991.
- Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, Smalltalk, and Unix shell and other scripting languages.
- Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL)
- It is easy to learn – the time needed to learn Python is shorter than for many other languages; this means that it's possible to start the actual programming fast
- It is easy to use for writing new software – it's often possible to write code faster when using Python.
- It is easy to obtain, install and deploy – Python is free, open and multiplatform; not all languages can boast that.
- Programming skills prepare you for careers in almost any industry and are required if you want to continue to more advanced and higher-paying software development and engineering roles.
- Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

7.2 Feature 2:

Flask

- **Flask** is a micro web framework written in Python. It is classified as a microframework 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.
- Applications that use the Flask framework include Pinterest and LinkedIn.

7.3 Database Scheme

IBM Db2

- DB2 is a database product from IBM.
- It is a Relational Database Management System (RDBMS). DB2 is designed to store, analyze and retrieve the data efficiently.
- DB2 product is extended with the support of Object-Oriented features and non-relational structures with XML.
- Provide a massively parallel processing (MPP) architecture Exploits Hive, HBase and Apache Spark concurrently for best-in-class analytic capabilities.
- 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

Kubernetes

- **Kubernetes** is also known as '**k8s**'.
- **Kubernetes** is an extensible, portable, and open-source platform designed by **Google** in **2014**.
- It is mainly used to automate the deployment, scaling, and operations of the container-based applications across the cluster of nodes.
- Kubernetes helps to manage containerised applications in various types of physical, virtual, and cloud environments.
- Google Kubernetes is a highly flexible container tool to consistently deliver complex applications running on clusters of hundreds to thousands of individual servers
- Kubernetes is the Linux kernel which is used for distributed systems.
- It helps you to be abstract the underlying hardware of the nodes(servers) and offers a consistent interface for applications that consume the shared pool of resources.

8. TESTING

8.1 Test case

- 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 test. Each test type addresses a specific testing requirement

Test case ID	Feature Type	Component	Test Scenario	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
LoginPage_TC_OO1	UI	Admin Login Page	Verify user is able to see the Login/Signup popup when user clicked on My account button	1.Enter URL http://127.0.0.1:8000/ and click go 2.Click on My Account dropdown button 3.Verify login/Singup popup displayed or not	Username: rit password: rit123	Login/Singup popup should display and navigate to Admin dashboard	Working as expected	Pass		Y		Admin
LoginPage_TC_OO2	Functional	Patient Login page	Verify user is able to log into application with Invalid credentials	1.Enter URL http://127.0.0.1:8000/ and click go 2.Click on 3.Verify login/Singup popup with below Patient elements: a.username text box b.password text box c.Login button	Username: shriram password: 2019011280	Application should show 'Incorrect Username or password' validation message.	Working as expected	Fail	Steps are not clear to follow	N	BUG-1234	Patient

LoginPage_TC_OO3	Functional	Donor Login Page	Verify user is able to log into application with Valid credentials	1.Enter URL http://127.0.0.1:8000/ and click go 2.Click on 3.Enter Valid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username: sathish password: 201901120	User should navigate to user Donor Home Page	Working as expected	Pass		Y		Donor
LoginPage_TC_OO4	Functional	Patient Login page	Verify user is able to log into application with Invalid credentials	1.Enter URL http://127.0.0.1:8000/ and click go 2.Click on 3.Enter Valid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username: shriram password: 201901128	User should navigate to user Donor Home Page	Working as expected	Pass		Y		Patient

8.2 User Acceptance Testing

1. Purpose of Document

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

2. Defect Analysis

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

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Sub total
By Design	8	4	2	3	17
Duplicate	1	0	2	1	4
External	2	3	0	1	6
Fixed	10	2	5	18	35
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	3	2	1	6
Totals	21	12	13	25	7 1

3. Test Case Analysis

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

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	8	0	0	8
Client Application	50	0	0	50
Security	2	0	0	2
Outsource Shipping	3	0	0	3
Exception Reporting	10	0	0	10
Final Report Output	6	0	0	6
Version Control	3	0	0	3

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 SCREENS

Login page

The screenshot shows a web browser window with the URL `127.0.0.1:8000/adminlogin`. The page has a dark red header with the text "Plasma Donor Application" and navigation links for Home, Patient, Donor, and Admin. The main content area has a blue and purple gradient background. In the center, there is a white box titled "ADMIN LOGIN". Inside this box, there are two input fields: "Username" with the value "rit" and "Password" with masked characters "*****". Below the password field is a red "LOGIN" button. The browser's taskbar at the bottom shows the system clock as 22:28 on 16-11-2022.

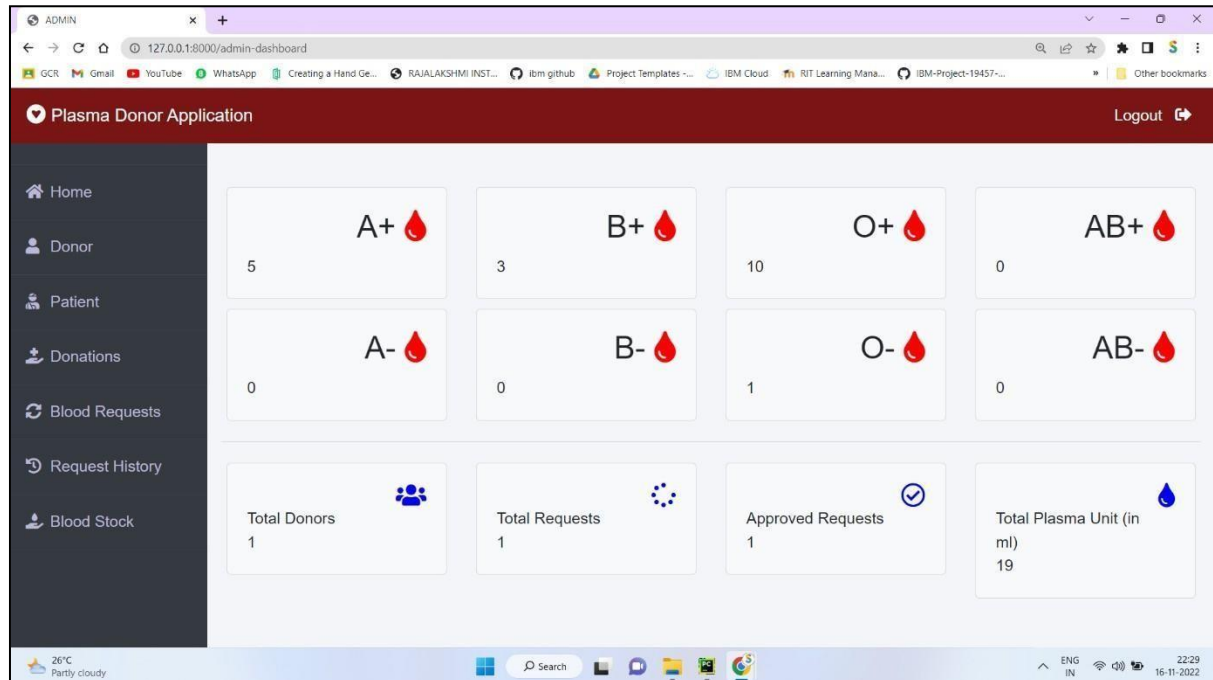
Donor Login

The screenshot shows a web browser window with the URL `127.0.0.1:8000/donor/donorlogin`. The page has a dark red header with the text "Plasma Donor Application" and navigation links for Home, Patient, Donor, and Admin. The main content area has a blue and purple gradient background. In the center, there is a white box titled "DONOR LOGIN". Inside this box, there are two input fields: "Username" with the value "sathish" and "Password" with masked characters "*****". Below the password field is a red "LOGIN" button. At the bottom of the white box, there is a link that says "Does not have an account ? [Click here to register](#)". The browser's taskbar at the bottom shows the system clock as 22:31 on 16-11-2022.

Register Page



Admin Dashboard Page



Plasma Donor Page

The screenshot shows a web browser window with the URL `127.0.0.1:8000/admin-donation`. The page title is "Plasma Donor Application" and it includes a "Logout" link. A sidebar on the left contains navigation links: Home, Donor, Patient, Donations (highlighted), Blood Requests, Request History, and Blood Stock. The main content area is titled "PLASMA DONATION DETAILS" and features a table with the following data:

Donor Name	Disease	Age	Blood Group	Unit	Request Date	Status	Action
Sathish Kumar	Nothing	20	B+	1	Nov. 16, 2022	Approved	1 Unit Added To Stock

The browser's taskbar at the bottom shows the system clock as 22:29 on 16-11-2022 and the weather as 26°C Partly cloudy.

SEND GRID

The screenshot displays the SendGrid Sender Authentication Center interface. A notification at the top states: "Review. Your account is under review and we'd like to know a little more about you and how you intend to use Twilio SendGrid. [Contact Us](#)". The page title is "Single Sender Verification". A sidebar on the left lists navigation options: Dashboard, Email API, Marketing, Design Library, Stats, Activity, Suppressions, and Settings. The main content area shows a table of senders:

SENDERS	ADDRESS	NICKNAME	VERIFIED	ACTIONS
Sathish FROM: sathishkumar.r.2019.cse@ritchennai.edu.in REPLY: shriram.sk.2019.cse@ritchennai.edu.in	Chennai Chennai, 600124 IND	IBM Plasma		

A "Create New Sender" button is located in the top right corner of the main content area.

IBM DB2

IBM Db2 on Cloud

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

Find schemas or tables Refresh

Tables New table

Name	Schema	Properties
REGTBL	ZTG96113	...
USERTBL	ZTG96113	...

Total: 2, selected: 0

Table definition REGTBL No statistics available

Name	Data type	Nullable	Length	Scale
USERNAME	CHAR	Y	5	0
EMAIL	CHAR	Y	5	0
PASSWORD	CHAR	Y	5	0
re-password	CHAR	Y	5	0

View data

IBM Db2 on Cloud

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

ZTG96113.REGTBL Back

Export to CSV

USERNAME	EMAIL	PASSWORD	re-password	PHONNUMBER
Kis	ksho	pass	pass	999

10. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- **Speed:** This website is fast and offers great accuracy as compared to manual registered keeping.
- **Maintenance:** Less maintenance is required
- **User Friendly:** It is very easy to use and understand. It is easily workable and accessible for everyone.
- **Fast Results:** It would help you to provide plasma donors easily depending upon the availability of it.

DISADVANTAGES:

- **Internet:** It would require an internet connection for the working of the website.
- **Auto- Verification:** It cannot automatically verify the genuine users.

11. CONCLUSION

- The efficient way of finding plasma donor for the infected people is implemented using the plasma donor website that is hosted on IBM Cloud platform.
- To ensure the smooth functioning of the web site operation. I have hosted the website in IBM Db2 & Kubernetes Cluster to make sure the operations are running successfully Cloud lambda function is used and to deploy the application IBM Db2 service is used.

12. FUTURE SCOPE

- Upgrading the UI that is more user-friendly which will help many users to access the website and also ensures that many plasma donors can be added into the community.
- Using elastic load balancer, it helps to handle multiple requests at the same time which will maintain the uptime of the website with negligible downtime

13.

APPENDIXES

13.1 SAMPLE SOURCE CODE:

DONOR

form.py

```
class DonorUserForm(forms.ModelForm):

    class Meta:
        model=User
        fields=['first_name','last_name','username','password']
        widgets = {
            'password': forms.PasswordInput()
        }
```

```
class DonorForm(forms.ModelForm):
    class Meta:
        model=models.Donor
        fields=['bloodgroup','address','mobile','profile_pic']
```

```
class DonationForm(forms.ModelForm):
    class Meta:
        model=models.BloodDonate
        fields=['age','bloodgroup','disease','unit']
```

model.py

```
class Donor(models.Model):
    user=models.OneToOneField(User,on_delete=models.CASCADE)
    profile_pic=
models.ImageField(upload_to='profile_pic/Donor/',null=True,blank=True)
```

```
bloodgroup=models.CharField(max_length=10)
```

```
address = models.CharField(max_length=40)
mobile = models.CharField(max_length=20,null=False)
```

```
@property
```

```

def get_name(self):
    return self.user.first_name+" "+self.user.last_name
@property
def get_instance(self):
    return self
def __str__(self):
    return self.user.first_name

```

```

class BloodDonate(models.Model):
    donor=models.ForeignKey(Donor,on_delete=models.CASCADE)
    disease=models.CharField(max_length=100,default="Nothing")
    age=models.PositiveIntegerField()
    bloodgroup=models.CharField(max_length=10)
    unit=models.PositiveIntegerField(default=0)
    status=models.CharField(max_length=20,default="Pending")
    date=models.DateField(auto_now=True)
    def __str__(self):
        return self.donor

```

view.py

```

def donor_signup_view(request):
    userForm=forms.DonorUserForm()
    donorForm=forms.DonorForm()
    mydict={'userForm':userForm,'donorForm':donorForm}
    if request.method=='POST':
        userForm=forms.DonorUserForm(request.POST)
        donorForm=forms.DonorForm(request.POST,request.FILES)
        if userForm.is_valid() and donorForm.is_valid():
            user=userForm.save()
            user.set_password(user.password)
            user.save()
            donor=donorForm.save(commit=False)

```

```

        donor.user=user
        donor.bloodgroup=donorForm.cleaned_data['bloodgroup']
        donor.save()
        my_donor_group = Group.objects.get_or_create(name='DONOR')
        my_donor_group[0].user_set.add(user)
        return HttpResponseRedirect('donorlogin')
    return render(request,'donor/donorsignup.html',context=mydict)

```

```

def donor_dashboard_view(request):
    donor= models.Donor.objects.get(user_id=request.user.id)
    dict={

                                                    'requestpending':
bmodels.BloodRequest.objects.all().filter(request_by_donor=donor).filter(status
='Pending').count(),

                                                    'requestapproved':
bmodels.BloodRequest.objects.all().filter(request_by_donor=donor).filter(status
='Approved').count(),

                                                    'requestmade':
bmodels.BloodRequest.objects.all().filter(request_by_donor=donor).count(),

                                                    'requestrejected':
bmodels.BloodRequest.objects.all().filter(request_by_donor=donor).filter(status
='Rejected').count(),
    }
    return render(request,'donor/donor_dashboard.html',context=dict)

```

```

def donate_blood_view(request):
    donation_form=forms.DonationForm()
    if request.method=='POST':

```



```

donation_form=forms.DonationForm(request.POST)
if donation_form.is_valid():
    blood_donate=donation_form.save(commit=False)
    blood_donate.bloodgroup=donation_form.cleaned_data['bloodgroup']
    donor= models.Donor.objects.get(user_id=request.user.id)
    blood_donate.donor=donor
    blood_donate.save()
    return HttpResponseRedirect('donation-history')
return
render(request,'donor/donate_blood.html',{'donation_form':donation_form})

def donation_history_view(request):
    donor= models.Donor.objects.get(user_id=request.user.id)
    donations=models.BloodDonate.objects.all().filter(donor=donor)
    return render(request,'donor/donation_history.html',{'donations':donations})

def make_request_view(request):
    request_form=bforms.RequestForm()
    if request.method=='POST':
        request_form=bforms.RequestForm(request.POST)
        if request_form.is_valid():
            blood_request=request_form.save(commit=False)
            blood_request.bloodgroup=request_form.cleaned_data['bloodgroup']
            donor= models.Donor.objects.get(user_id=request.user.id)
            blood_request.request_by_donor=donor
            blood_request.save()
            return HttpResponseRedirect('request-history')

```

```

return
render(request,'donor/makerequest.html',{'request_form':request_form})

def request_history_view(request):
    donor= models.Donor.objects.get(user_id=request.user.id)
    blood_request=bmodels.BloodRequest.objects.all().filter(request_by_donor=
donor)
    return
render(request,'donor/request_history.html',{'blood_request':blood_request})

```

BLOOD

admin.html

```

{% extends 'blood/adminbase.html' %}
{% block content %}
{% load widget_tweaks %}
<style>
    .xyz{
        display: table;
        margin-right: auto;
        margin-left: auto;
    }
</style>
<br><br>
<div class="container">
<div class="row">
    <div class="col-sm-3">

```

```

<div class="card bg-light">
  <div class="card-body">
    <div class="blood">
      <h2>A+ <i class="fas fa-tint"></i></h2>
    </div><br><br>
    <div>
      { { A1.unit } }
    </div>
  </div>
</div>
<div class="col-sm-3">
  <div class="card bg-light">
    <div class="card-body">
      <div class="blood">
        <h2>B+ <i class="fas fa-tint"></i></h2>
      </div><br><br>
      <div>
        { { B1.unit } }
      </div>
    </div>
  </div>
</div>
<div class="col-sm-3">

```

```

<div class="card bg-light">
  <div class="card-body">
    <div class="blood">
      <h2>O+ <i class="fas fa-tint"></i></h2>
    </div><br><br>
    <div>
      { {O1.unit} }
    </div>
  </div>
</div>
<div class="col-sm-3">
  <div class="card bg-light">
    <div class="card-body">
      <div class="blood">
        <h2>AB+ <i class="fas fa-tint"></i></h2>
      </div><br><br>
      <div>
        { {AB1.unit} }
      </div>
    </div>
  </div>
</div>
</div>

```

```

<div class="row">

  <div class="col-sm-3">

    <div class="card bg-light">

      <div class="card-body">

        <div class="blood">

          <h2>A- <i class="fas fa-tint"></i></h2>

          </div><br><br>

          <div>

            {{ A2.unit }}

          </div>

        </div>

      </div>

    </div>

  </div>

  <div class="col-sm-3">

    <div class="card bg-light">

      <div class="card-body">

        <div class="blood">

          <h2>B- <i class="fas fa-tint"></i></h2>

          </div><br><br>

          <div>

            {{ B2.unit }}

          </div>

        </div>

      </div>

    </div>

  </div>

```

</div>

<div class="col-sm-3">

<div class="card bg-light">

<div class="card-body">

<div class="blood">

<h2>O- <i class="fas fa-tint"></i></h2>

</div>

<div>

{ { O2.unit } }

</div>

</div>

</div>

</div>

<div class="col-sm-3">

<div class="card bg-light">

<div class="card-body">

<div class="blood">

<h2>AB- <i class="fas fa-tint"></i></h2>

</div>

<div>

{ { AB2.unit } }

</div>

</div>

</div>

```

        </div>

    </div>

<hr>

<br>

<h3 class="text-center">Update Blood Unit</h3><br>

<div class="xyz">

    <form class="form-inline" method="POST">

        { % csrf_token % }

        <div class="form-group mx-sm-3 mb-6">

            <select name="bloodgroup" class="form-control">

                <option disabled="disabled" selected="selected">Choose Blood
Group</option>

                <option>O+</option>

                <option>O-</option>

                <option>A+</option>

                <option>A-</option>

                <option>B+</option>

                <option>B-</option>

                <option>AB+</option>

                <option>AB-</option>

            </select>

        </div>

        <div class="form-group mx-sm-3 mb-6">

            <input type="number" class="form-control" name="unit" placeholder="Unit">

        </div>

```

```
        <button type="submit" class="btn btn-primary mb-2">Update</button>
    </form>
</div>
</div>
{% endblock content %}
```

Index.html

```
{% load static %}

<!DOCTYPE html>

<head>

    <style>

        .xyz{

            margin-bottom: 0px;

            background-image: url('{% static "image/homepage3.png" %}');

            background-size: cover;

            background-repeat: no-repeat;

        }

    </style>

</head>

<body>

    {% include "blood/navbar.html" %}

    <br>
```



```

<section      id="section-jumbotron"      style="margin-bottom:      0px;"
class="jumbotron jumbotron-fluid d-flex justify-content-center align-items-
center xyz">

    <div class="container text-center">

        <br>

<br>

</div>

</section>

<div class="jumbotron" style="margin-top: 0px;margin-bottom: 0px;">

    <p      class="lead      text-center"><h3      align      =
"center">“PNT2022TMID26437”</h3></p>

    <p align="center">SHRIYA R </p>

    <p align="center">SATHISH KUMAR R </p>

    <p align="center">SHRIRAM S K</p>

    <p align="center">SUJITHA A</p><br>

<p class="lead text-center" ><h5 align = "center"><b>“Saving a life won’t cost
you anything. Go ahead and donate Plasma”</b></h5>

</p>

<p class="text-center">- Anonymous</p>

</div>

{ % include "blood/footer.html" % }

</body>

```

13.2

GITHUB

<https://github.com/IBM-EPBL/IBM-Project-29134-1660121370>