PLASMA DONOR APPLICATION

Team ID:PNT2022TMID08712

SUBMITTED BY

TEAM LEADER	SATHISH KUMAR S	727619BEC045
TEAM MEMBER	AMRITHA M P	727619BEC089
TEAM MEMBER	DIVIYYA SHREE I	727619BEC097
TEAM MEMBER	SIVA KARTHINI G	727620BEC307

In partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

in

ELECTRONICS AND COMMUNICATION ENGINEERING

Dr . MAHALINGAM COLLEGE OF ENGINEERING AND TECHNOLOGY An Autonomous Institution Affiliated to ANNAUNIVERSITY CHENNAI – 600 025

ABSTRACT

The necessity of blood has become a significant concern in the present context all over the world. Due to a shortage of blood, people couldn’t save themselves or their friends and family members. A bag of blood can save a precious life. Statistics show that a tremendous amount of blood is needed yearly because of major operations, road accidents, blood disorders, including Anemia, Hemophilia, and acute viral infections like Dengue, etc. Approximately 85 million people require single or multiple blood transfusions for treatment. Voluntary blood donors per 1,000 population of some countries are quite promising, such as Switzerland (113/1,000), Japan (70/1,000), while others have an unsatisfying result like India has 4/1,000, and Bangladesh has 5/1000. Recently a life-threatening virus, COVID-19, spreading throughout the globe, which is more vulnerable for older people and those with pre-existing medical conditions. For them, plasma is needed to recover their illness. Our Purpose is to build a platform with clustering algorithms which will jointly help to provide the quickest solution to find blood or plasma donor. Closest blood or plasma donors of the same group in a particular area can be explored within less time and more efficiently.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Generally, plasma donors must be 18 years of age and weigh at least 110 pounds (50kg). All individuals must pass two separate medical examinations, a medical history screening and testing for transmissible viruses, before their donated plasma can be used to manufacture plasma protein therapies. Donor must have a pre-donation physical which includes answering medical history questions, tests for viruses such as HIV and Hepatitis and evaluating your protein and hemoglobin levels.

Blood is a liquid substance that circulates the necessary nutrients and oxygen to every cell of the body. Every year many people died because of the shortage of safe blood.

This problem becomes severe during pregnancy and major operation. Again blood requires for some other diseases like Cancer, Dengue and Leukemia, thalassemia, etc. Over

a million people died because of dengue in Bangladesh. There are time periods in which blood must be processed or preserved. Otherwise, collected or stored blood can’t be used. So, An efficient flow of blood is required in blood bank or blood donation camps to meet the regular demands of recipients. Recently concern grows about the plasma donation for COVID-19 during the pandemic situation. This convalescent plasma was used to recover

patients who are critically ill as it helps to grow antibodies on their body. People all over the world donate blood on these purpose. According to the World Health Organization (WHO),

around 118.5 million blood donations are collected globally, 40% of which are collected from high-income countries. Many organizations help blood donors donate blood and plasma via many applications and online social groups. But this application and online social groups
remain analog, and we need the quickest solution in this regard. In regular blood donation applications and social groups, people share their needs for blood and get some
information lately that can be less useful in an emergency condition.

The key objectives of our work are-

• To build a platform between blood donor and receiver.

•To implement a hybrid approach of K-Means and Agglomerative clustering algorithm.

• To find the nearest blood donor in a specific region in
the shortest possible time.

• To increase the number of voluntary unpaid blood
donations significantly

Plasma is used by pharmaceutical companies to make plasma-derived medicinal products (PDMPs). PDMPs are used to treat conditions such as immune deficiencies and bleeding disorders. Several PDMPS are included in the WHO Model Lists of Essential Medicines. According to the WHO, self-sufficiency driven by voluntary (non-remunerated) plasma donations is an important national goal to ensure an adequate supply is secured to meet the needs of the population. Australia, New Zealand, the UK, the Netherlands, and France only allow public or not-for-profit sectors to collect plasma for fractionation. Each of the 5 countries have toll or contract agreements with 1 private commercial plasma fractionator to manufacture PDMPs from the plasma collected within their respective countries. None of these countries pay plasma donors. Donors are only permitted to donate every 2 weeks (24 to 26 times per year) in these 5 countries. Austria, the Czech Republic, Germany, and the US allow both public and non-for-profit sectors, as well as commercial private plasma collection centres, to operate in the country. Private, not-for-profit, or public plasma collection centres in these 4 countries offer monetary compensation and other in-kind incentives to plasma donors. While the Czech Republic limits plasma donation to every 2 weeks, a much higher frequency of donation is allowed in other countries; up to 50 times per year in Austria, 60 times per year in Germany, and more than 100 times per year in the US. Austria, the Czech Republic, Germany, and the US (which allow commercial private plasma collectors to operate and pay donors) are 100% self-sufficient in immunoglobulins. These 4 countries collect the most plasma, which is from paid donors. In 2017, Austria, the Czech Republic, Germany, and the US collected 75 litres per 1,000 people, 45 litres per 1,000 people, 36 litres per 1,000 people, and 113 litres per 1,000 people of plasma for fractionation, respectively. Countries that do not pay donors including Australia, New Zealand, the UK, the Netherlands, and France are dependent to some extent on US and European Union donors who are paid for the supply of plasma or imported PDMPs.

1.1 PROJECT OVERVIEW

Plasma is the clear, straw-colored liquid portion of blood that remains after red blood cells, white blood cells, platelets, and other cellular components are removed. Plasma is composed of 90% water, plasma is a transporting medium for cells and a variety of substances vital to the human body. Plasma carries out a variety of functions in the body, including clotting blood, fighting diseases, and other critical functions. Plasma donation requires a commitment both in the amount of time for each donation and the frequency of donation. Typically it takes between one and three hours to donate source plasma, and plasma can be donated twice within a seven-day period. Whole plasma donation takes less time under 30 minutes and donors donate less frequently no more than once in eight weeks. The programs may fit into a donor's life differently at various times in the donor's life and are equally important in helping to fulfill a vital medical need. Source plasma is plasma that is collected from healthy, voluntary donors through a process called plasmapheresis and is used exclusively for further manufacturing into final therapies (fractionation). Source plasma donors may be compensated for their time and effort. Recovered plasma is collected through whole blood donation in which plasma is separated from its cellular components. Recovered plasma may be used for fractionation. The plasma protein therapeutics industry supports volunteerism donation in all of its forms. Source plasma donation and blood donation are critically important activities that contribute to saving lives. Source plasma and recovered plasma are used to produce therapies that treat people with rare, chronic diseases and disorders such as primary immunodeficiency, hemophilia, and genetic lung disease, as well as in the treatment of trauma, burns, and shock. Whole plasma donations most often are used locally in hospitals for transfusions required during surgery or other medical treatment.

1.2 PURPOSE

Individuals who experience a severe trauma, burn or shock often lose a significant amount of blood volume, and are depleted of many necessary electrolytes. Electrolytes are minerals that help to balance the amount of water, nutrients and pH level inside the body. In this situation, a plasma transfusion can provide the lifesaving blood volume needed to restore their blood pressure and volume status, as well as restore electrolyte levels.

In addition, people with liver disease or clotting factor deficiencies may not have the proper substances in their blood to allow their blood to clot normally. Whenever an individual has a cut or injury, these clotting factors ensure that they do not lose too much blood. Plasma donations ensure that these individuals can receive a plasma transfusion to supplement their body’s clotting ability and stop excessive bleeding from occurring. Finally, children and adults with cancer sometimes experience complications in which their body has used up all of their natural clotting factors. In cases of this disorder, called disseminated intravascular coagulation (DIC), transfusions of fresh frozen plasma may be critical.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING PROBLEM

The population of the world is multiplying with each coming year and so are the diseases and health issues. With an increase in the population there is an increase in the need of Plasma. The growing population of the world results in a lot of potential Plasma donors. With the growing population and the advancement in medical science the demand for Plasma has also increased.

The proposed method helps the users to check the availability of donors. A donor has to register to the website providing their details. The registered users can get the information about the donor count of each blood group. The database will have all the details such as name, email, phone number, infected status. Whenever a user requests for a particular blood group then the concerned blood group donors will receive the notification regarding the requirement. A Json code is written to store the information, to fetch the requested information in lambda.

2.2 REFERENCES

LITERATURE REVIEW

Arunkumar Chinnaswamy, Gurusankar Gopalakrishnan, Shabala Natarajan(2015). A study on Automation of Blood donor classification and Notification Techniques. This paper presents the increasing demand of blood donor in the field of healthcare related to automation processes. The present scenario tells us that blood donation services are manual and the demand for the blood is stably on the rise. Meanwhile, the number of voluntary donors is decreasing over the last few years. To improve this blood donor, automation and notification methods came to connect communication through all over the world.In this paper, we compare the various implementation and previous research doneon this techniques.

Sumazly Sulaiman, Abdul Aziz K.Abdul Hamid, Nurul Ain Najihah Yusri (2016). Development of a blood bank management system. This paper tells us about the development of blood bank system. There are 3 systems for blood bank management system. They are Blood Bank India, Lions Blood Bank & Research Foundation(LBBRF) and BBMS standalone version. The Blood Bank India is a website that provides the facility for the donor to register by him/himself as a blood donor. This website is only for Indian citizen can register to the system. It provides a feature where a person or hospital can request the blood stock from BBI. LBBRF is a private organisation that provides a place to donate blood. Theywill conduct an event and here the donor or public people can donate the blood. They will also inform when is their next event to the donor, public people or in their website. The standalone system uses the Microsoft as the database of the system. It contains user account management, view stock list, donor registration and customer registration.

Radha R. Mahalle, S. S. Thorat(2018). Smart Blood Bank Based on IOT. In this paper describes, blood is very important in the medical field. The main purpose of the blood bank is to provide the blood to the patients with minimal blood transfusion error. As the blood bank management system consists of number of manual steps, so it becomes difficult to the blood bank to provide a large level of accuracy, reliability and automation in blood storage as well as transfusion process. This IOT based system will improve the response time of the blood bank by connecting all the blood banks to cloud storage. The use of IOT system will provide benefits for blood bank.

Anish Hamlin M R, Albert Mayan J(2016). Blood Donation and Life Saver-Blood Donation App. This paper develops an application for finding the blood donation for making a request for the blood. If any blood seeker would login to the given application using GIS the patient will get detail about the nearby blooddonor. Also, any blood donor can add themselves for donating the blood then he/she will receive the notification related to the blood donation camp. In this appall the blood banks are connected to the cloud storage. The cloud storage providesthe real time information related to the available blood stock in every blood bank.If the blood is out of stock then the system will provide the contact details of theblood donors of different blood groups.

[1] “Blood safety and availability,” https://www.who.int/news-room/fact-sheets/detail/blood-safety-and-availability,(Accessed on 09/05/2020).

[2] T. Alanzi and B. Alsaeed, “Use of social media in the blooddonation process in saudi arabia,” Journal of Blood Medicine,vol. 10, p. 417, 2019.

[3] T. Wangchuk, K. Wangmo, U. Wangchuk, P. Gyem, P. R.Dhungyel et al., “Need of medium for nding blood donorin bhutan,” Asian Journal For Convergence In Technology(AJCT), 2018.

[4] V. K. Tatikonda and H. El-Ocla, “Bloodr: blood donor andrequester mobile application,” Mhealth, vol. 3, 2017.

[5] M. S. Hossain, N. Das, M. K. H. Patwary, and M. Al-Hasan, “Finding the nearest blood donors using dijkstra al-gorithm,” SISFORMA: Journal of Information Systems (e-Journal), vol. 5, no. 2, pp. 40–44, 2019.

[6] H. D. Das, R. Ahmed, N. Smrity, and L. Islam, “Bdonor:A geo-localised blood donor management system using mobilecrowdsourcing,” in 2020 IEEE 9th International Conference onCommunication Systems and Network Technologies (CSNT).IEEE, 2020, pp. 313–317.

[7] S.-Q. Wang and D.-M. Zhu, “Research on selecting initial pointsfor k-means clustering,” in 2008 International Conference onMachine Learning and Cybernetics, vol. 5. IEEE, 2008, pp.2673–2677.

[8] S. Na, L. Xumin, and G. Yong, “Research on k-means clusteringalgorithm: An improved k-means clustering algorithm,” in2010 Third International Symposium on intelligent informationtechnology and security informatics. IEEE, 2010, pp. 63–67.

2.3 PROBLEM STATEMENT DEFINITION

During the COVID 19 crisis, the requirement of plasma became a high priority and the donor count has become low. Saving the donor information and helping the Authorized user by notifying the current donors list, would be a helping hand. In regard to the problem faced, an application is to be built which would take the donor details, store them and inform them upon a request. People who need blood are increasing day by day. People who have diseases like anemia or people who have gotten into accidents and run out of blood need constant supply of blood to sustain their life and there is not enough blood available for them. It is not that people do not want to donate blood, but because they have no idea where they can donate. It is important for the people who are excited to donate, but yet are very busy, to be sure where and when they can donate,and therefore We are designing a system which contains all the information regarding blood donation camps ongoing in a particular area so that people who want to donate blood will get information regarding these camps. Our System is a mobile application which aims to serve as a communication tool between Blood Donation camp Organizers and blood donors. To become a member of the system, donors need to create their profile by providing the information like name, blood group, email address, password, and exact location from “Google Map”. In order to find out the exact location of a donor, Google Map is integrated with this application. The mobile application always keeps updating the location of a donor. As a result, the system can automatically keep showing the nearby Blood donation Camps to the registered donor wherever they go, and donors can easily get the idea of nearby blood donation camps. Also, users can get information regarding the type of blood which is available and information of past as well as future events.

CHAPTER 3

IDEATION AND PROPSED SOLUTION

3.1 EMPATHY MAP CANVAS

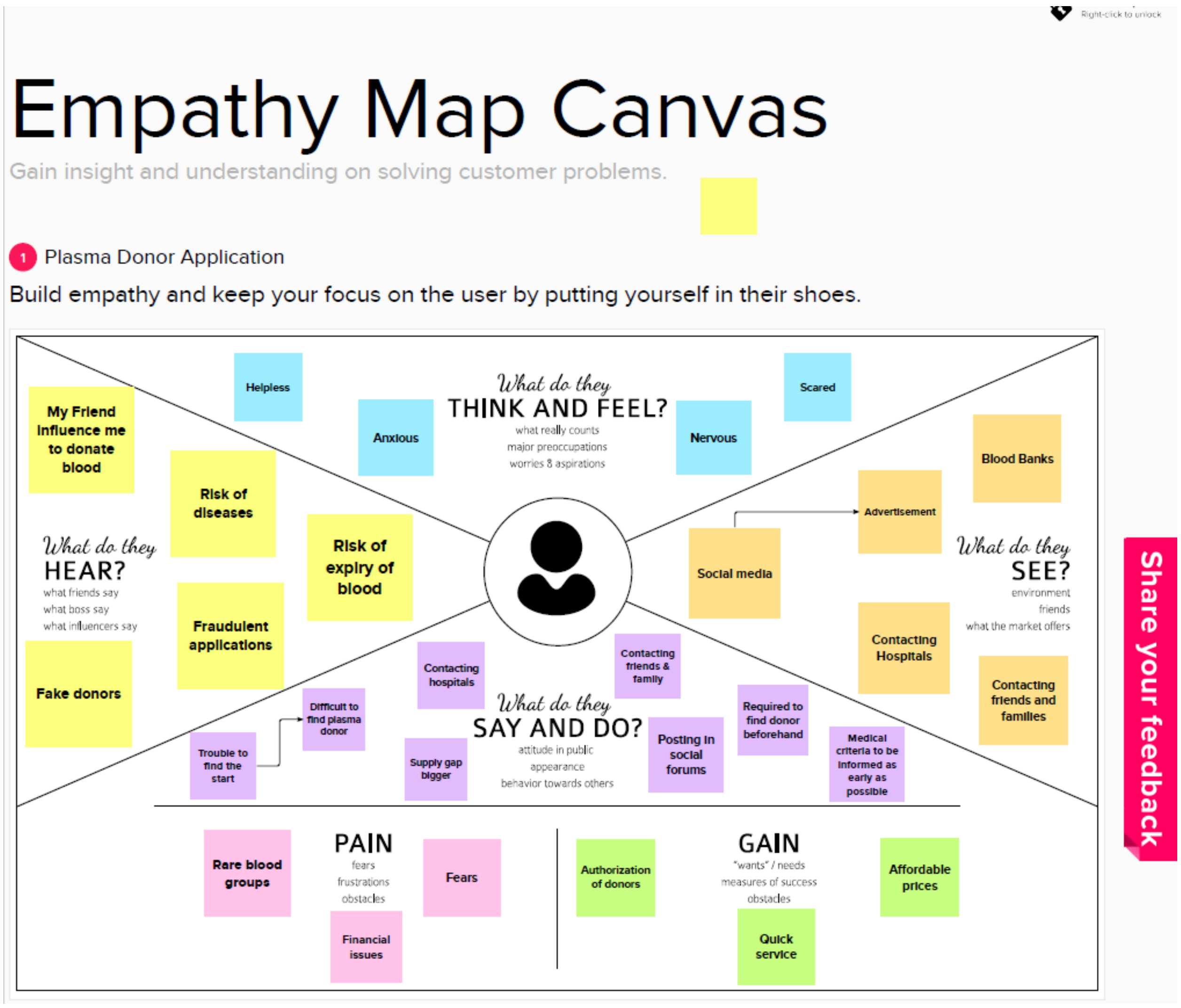
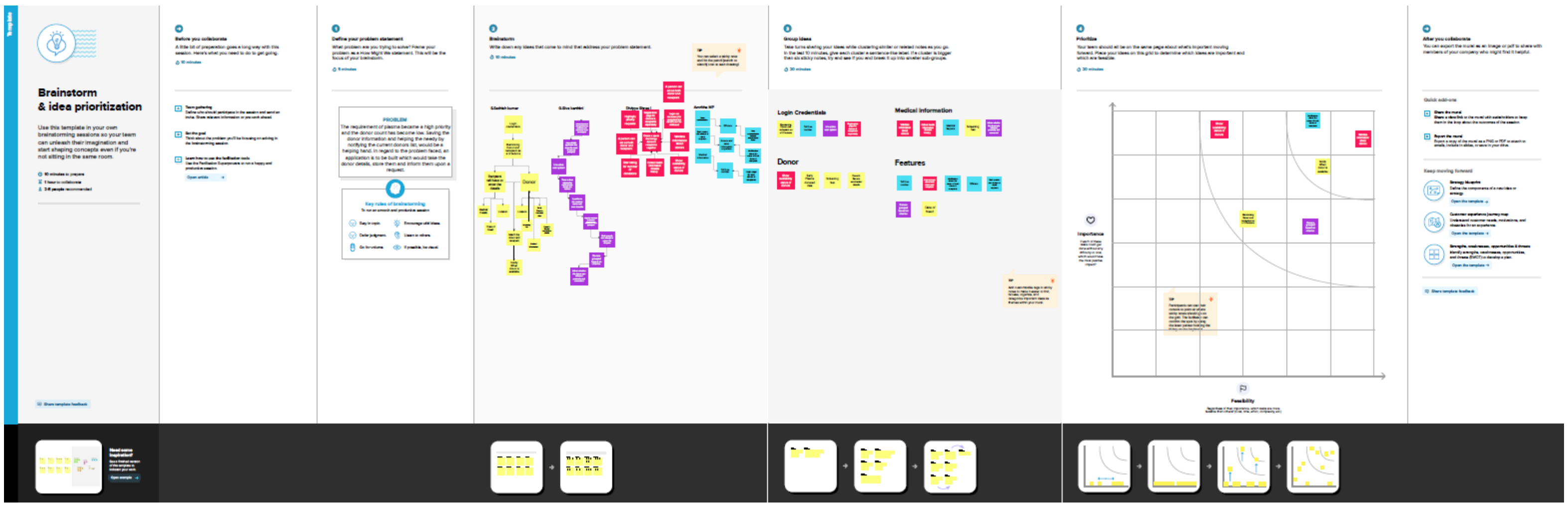


FIG : 1.1 Empathy map canvas

3.2 IDEATION AND PROCESSING



Brainstorm & idea prioritization

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

Before you collaborate

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

Define your problem statement

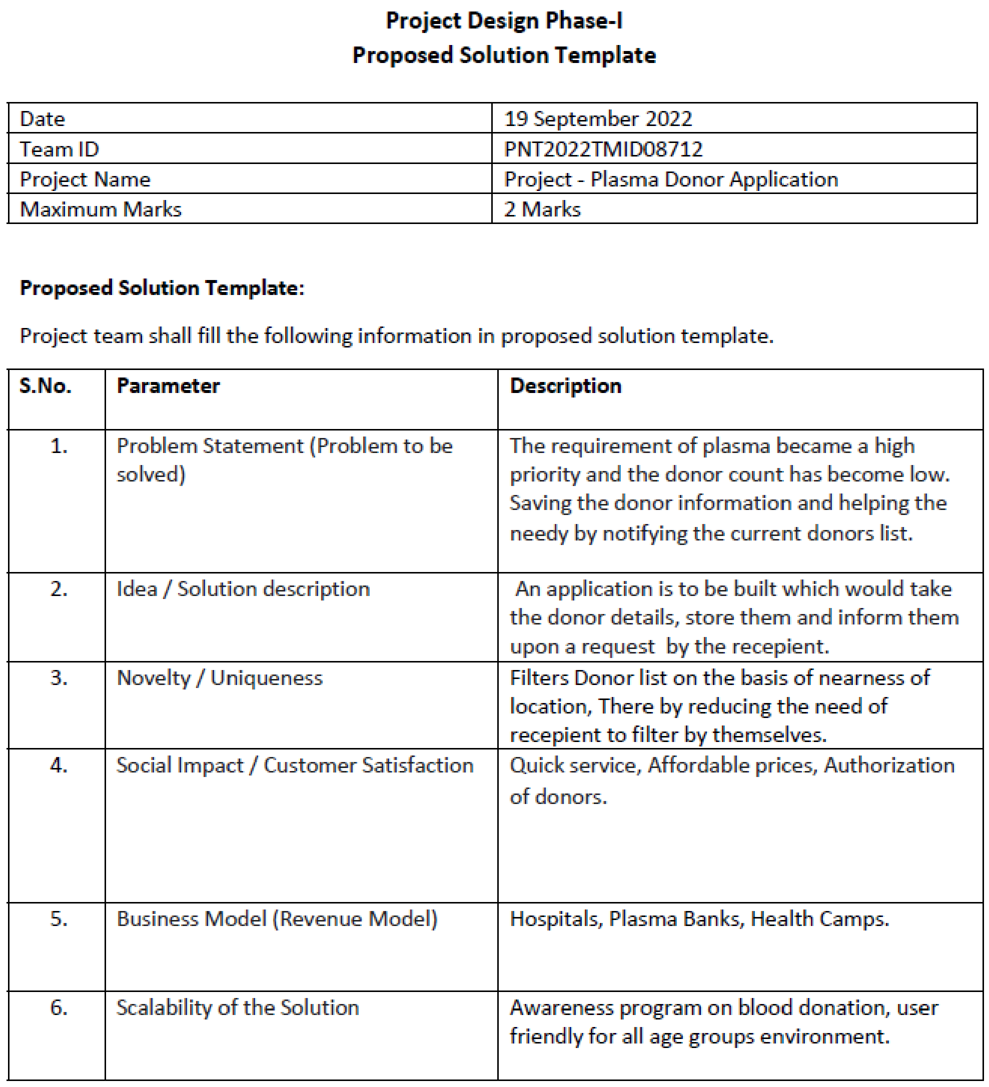
What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

PROBLEM

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

FIG : 1.2 Ideation and processing

3.3 PROPOSED SOLUTION



Project Design Phase-I
Proposed Solution Template

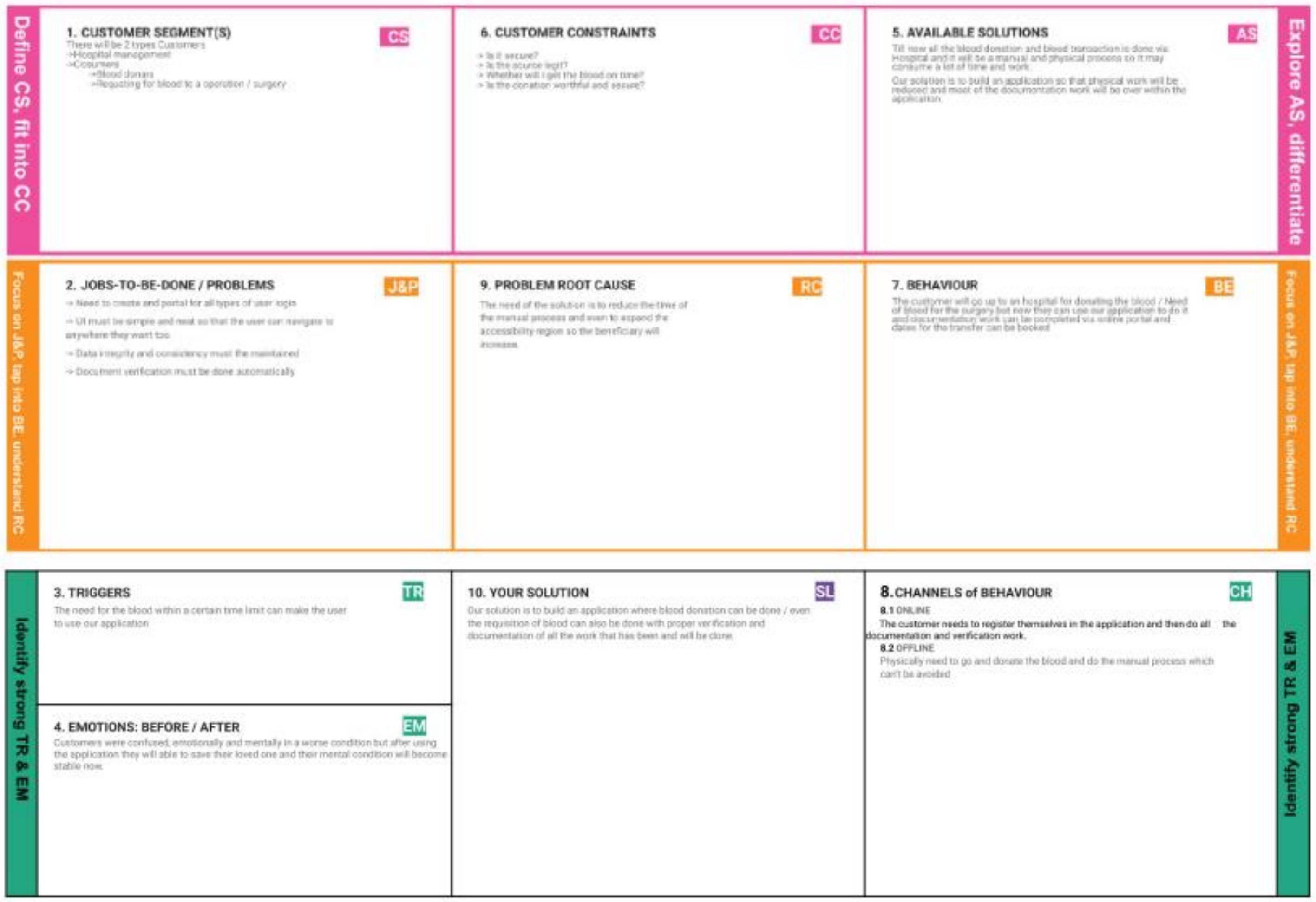
Date	19 September 2022
Team ID	PNT2022TMID08712
Project Name	Project - Plasma Donor Application
Maximum Marks	2 Marks

Proposed Solution Template:

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

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	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.
2.	Idea / Solution description	An application is to be built which would take the donor details, store them and inform them upon a request by the recepient.
3.	Novelty / Uniqueness	Filters Donor list on the basis of nearness of location, There by reducing the need of recepient to filter by themselves.
4.	Social Impact / Customer Satisfaction	Quick service, Affordable prices, Authorization of donors.
5.	Business Model (Revenue Model)	Hospitals, Plasma Banks, Health Camps.
6.	Scalability of the Solution	Awareness program on blood donation, user friendly for all age groups environment.

3.4 PROBLEM SOLUTION FIT



Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

There will be 2 types Customers
->Hospital management
->Cosumers
->Blood donars
->Requsting for blood to a operation / surgery

6. CUSTOMER CONSTRAINTS CC

-> Is it secure?
-> Is the source legit?
-> Whether will I get the blood on time?
-> Is the donation worthful and secure?

5. AVAILABLE SOLUTIONS AS

Till now all the blood donation and blood transaction is done via Hospital and it will be a manual and physical process so it may consume a lot of time and work.

Our solution is to build an application so that physical work will be reduced and most of the documentation work will be over within the application.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

-> Need to create and portal for all types of user login

-> UI must be simple and neat so that the user can navigate to anywhere they want too.

-> Data integrity and consistency must the maintained

-> Document verification must be done automatically

9. PROBLEM ROOT CAUSE RC

The need of the solution is to reduce the time of the manual process and even to expand the accessibility region so the beneficiary will increase.

7. BEHAVIOUR BE

The customer will go up to an hospital for donating the blood / Need of blood for the surgery but now they can use our application to do it and documentation work can be completed via online portal and dates for the transfer can be booked

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TR

The need for the blood within a certain time limit can make the user to use our application

4. EMOTIONS: BEFORE / AFTER EM

Customers were confused, emotionally and mentally in a worse condition but after using the application they will able to save their loved one and their mental condition will become stable now.

10. YOUR SOLUTION SL

Our solution is to build an application where blood donation can be done / even the requisition of blood can also be done with proper verification and documentation of all the work that has been and will be done.

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE

The customer needs to register themselves in the application and then do all the documentation and verification work.

8.2 OFFLINE

Physically need to go and donate the blood and do the manual process which can't be avoided

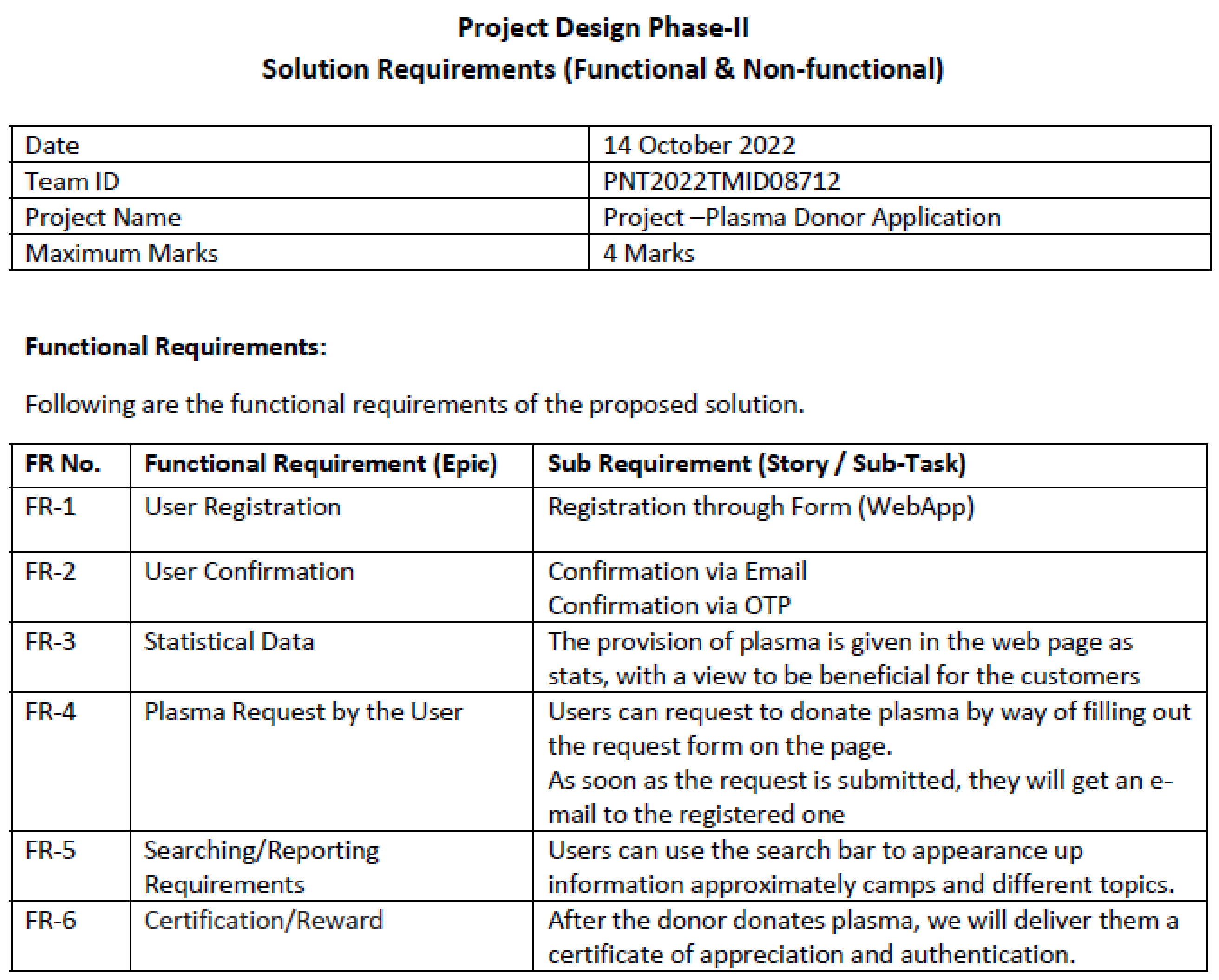
Identify strong TR & EM

FIG : 1.3 Problem solution fit

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT



Project Design Phase-II
Solution Requirements (Functional & Non-functional)

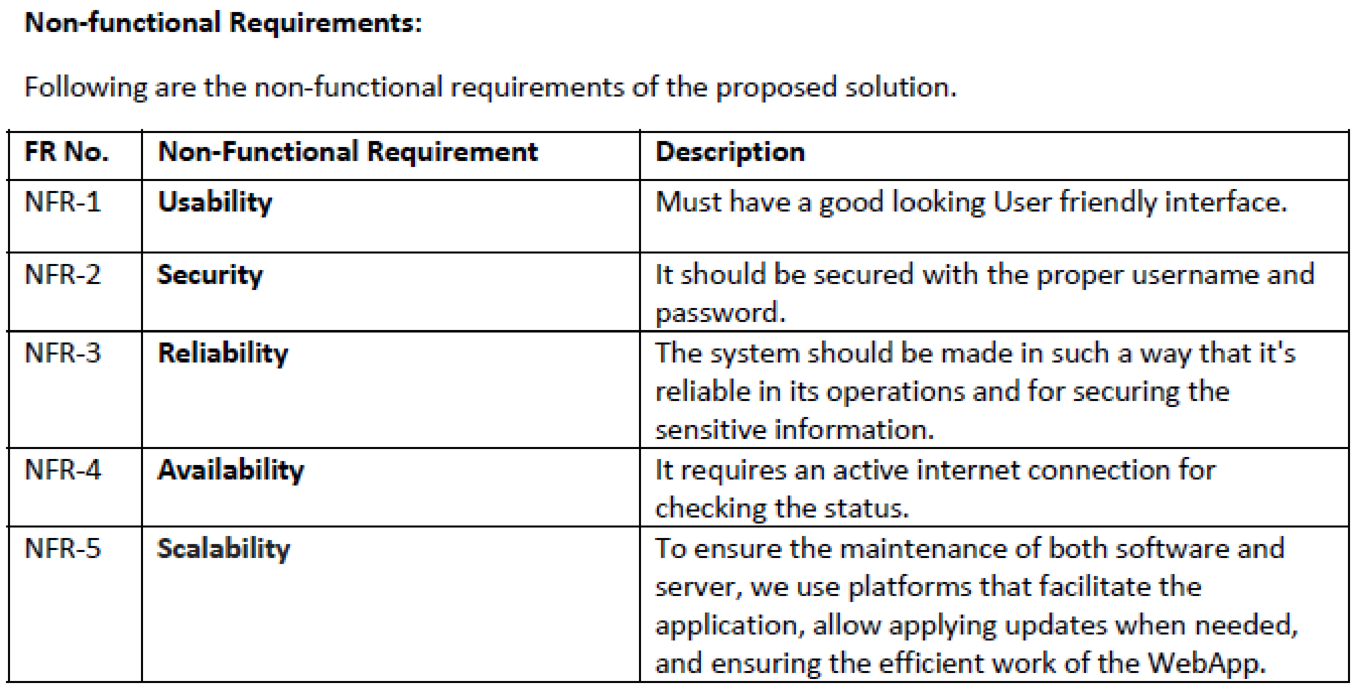
Date	14 October 2022
Team ID	PNT2022TMID08712
Project Name	Project –Plasma Donor Application
Maximum Marks	4 Marks

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form (WebApp)
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Statistical Data	The provision of plasma is given in the web page as stats, with a view to be beneficial for the customers
FR-4	Plasma Request by the User	Users can request to donate plasma by way of filling out the request form on the page. As soon as the request is submitted, they will get an e-mail to the registered one
FR-5	Searching/Reporting Requirements	Users can use the search bar to appearance up information approximately camps and different topics.
FR-6	Certification/Reward	After the donor donates plasma, we will deliver them a certificate of appreciation and authentication.

4.2 NON-FUNCTIONAL REQUIREMENTS



Non-functional Requirements:

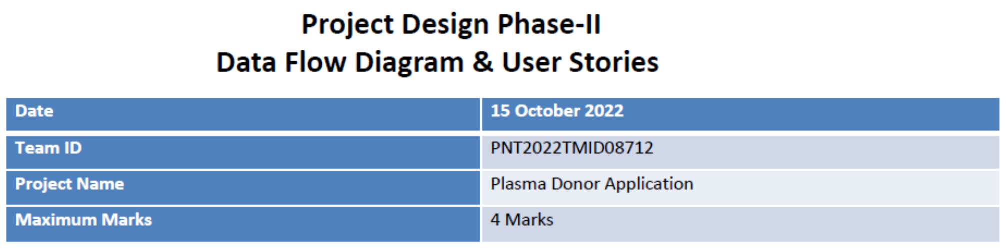
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Must have a good looking User friendly interface.
NFR-2	**Security**	It should be secured with the proper username and password.
NFR-3	**Reliability**	The system should be made in such a way that it's reliable in its operations and for securing the sensitive information.
NFR-4	**Availability**	It requires an active internet connection for checking the status.
NFR-5	**Scalability**	To ensure the maintenance of both software and server, we use platforms that facilitate the application, allow applying updates when needed, and ensuring the efficient work of the WebApp.

CHAPTER 5

PROJECT DESIGN

5.1 PROJECT DESIGN PHASE II DATA FLOW DIAGRAMS AND USER STORIES



Project Design Phase-II
Data Flow Diagram & User Stories

Date	**15 October 2022**
Team ID	PNT2022TMID08712
Project Name	Plasma Donor Application
Maximum Marks	4 Marks

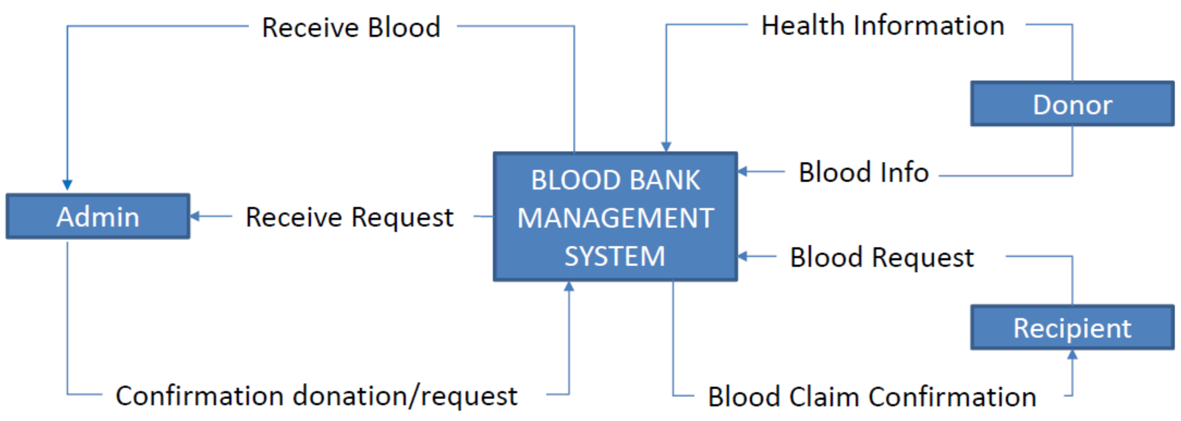


FIG : 1.4 Data flow diagram

5.2 SOLUTION ARCHITECTURE

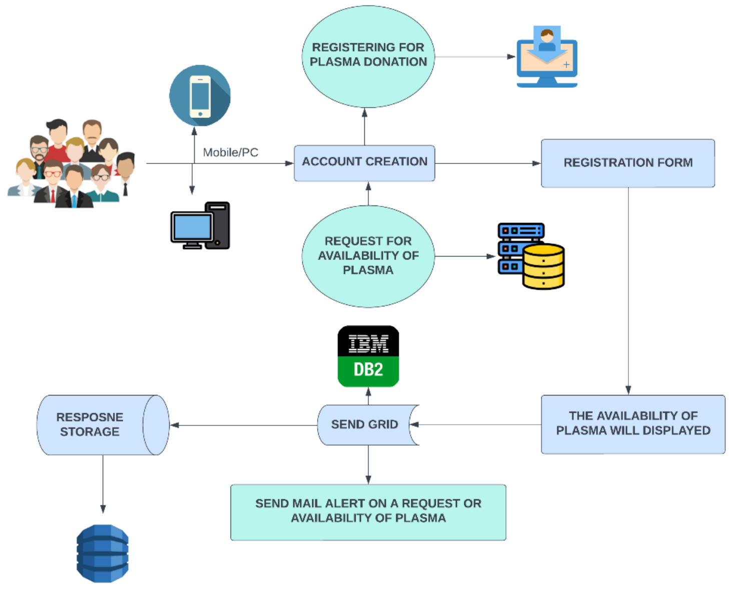

REGISTERING FOR PLASMA DONATION
Mobile/PC
ACCOUNT CREATION
REGISTRATION FORM
REQUEST FOR AVAILABILITY OF PLASMA
IBM DB2
RESPOSNE STORAGE
SEND GRID
THE AVAILABILITY OF PLASMA WILL DISPLAYED
SEND MAIL ALERT ON A REQUEST OR AVAILABILITY OF PLASMA

FIG : 1.5 Solution architechture

TECHNICAL ARCHITECTURE

Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	03 October 2022
Team ID	PNT2022TMID08712
Project Name	Plasma Donor Application.
Maximum Marks	4 Marks

Technical Architecture:

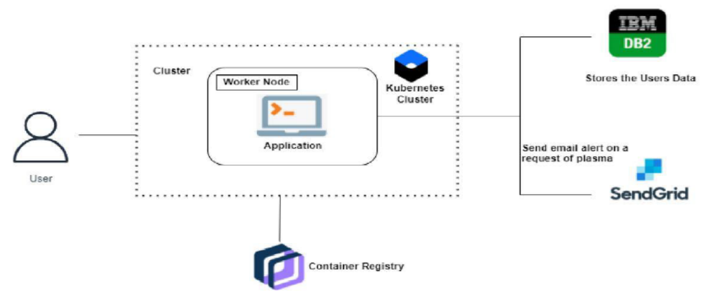


FIG : 1.6 Technical architechture

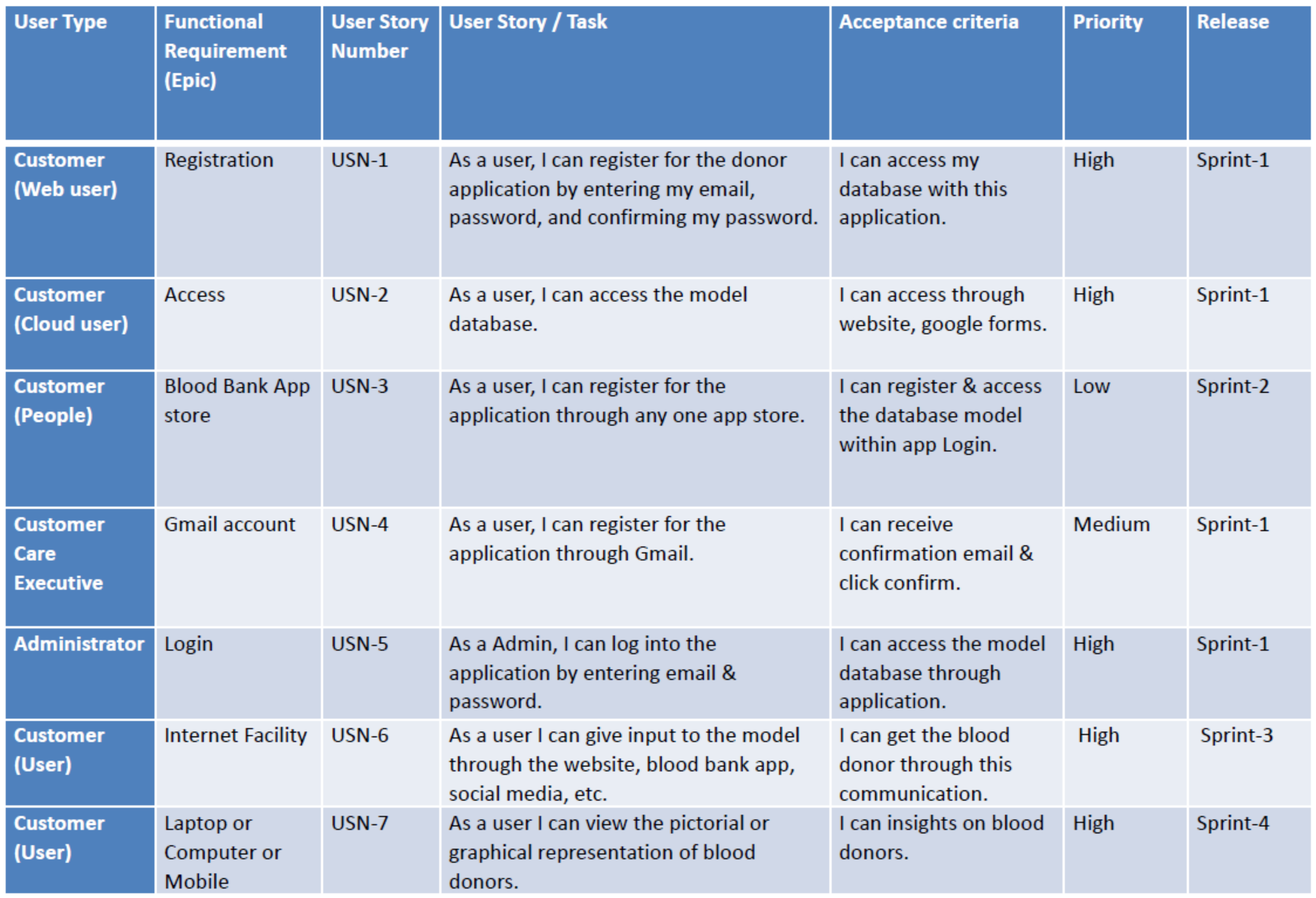
Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	User interacts with application Web UI	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Application Logic-1	Framework used for designing the application	Python - Flask
3.	Application Logic-2	Communication between users and the application via mails.	SendGrid
4.	Application Logic-3	Storing the details of the users both donors and patients	IBM DB2
5.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
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-2	They make it easier for your developers to store, manage and deploy container images.	Container Registry

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Python – flask is an open-source framework used to develop the application.	Python -flask
2.	Security Implementations	Container registry and Kubernetes Cluster are used for encryption of data.	Container registry and Kubernetes Cluster
3.	Scalable Architecture	Kubernetes Cluster allow containers to run across multiple machines and environments	Kubernetes Cluster
4.	Availability	Kubernetes Cluster provides all time availability.	Kubernetes Cluster
5.	Performance	Docker improves the application performance.	Docker

5.3 USER STORIES

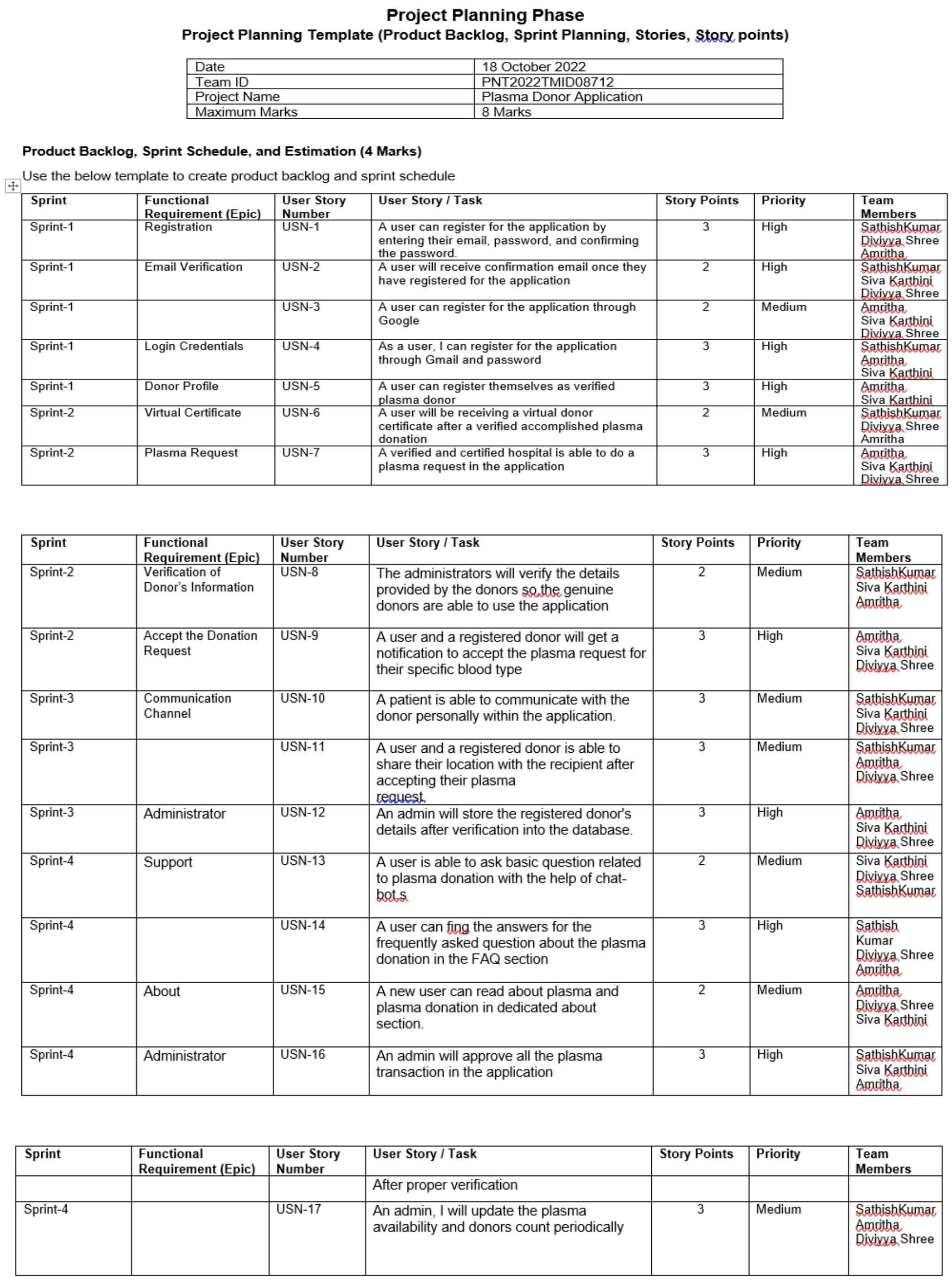


User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Web user)	Registration	USN-1	As a user, I can register for the donor application by entering my email, password, and confirming my password.	I can access my database with this application.	High	Sprint-1
Customer (Cloud user)	Access	USN-2	As a user, I can access the model database.	I can access through website, google forms.	High	Sprint-1
Customer (People)	Blood Bank App store	USN-3	As a user, I can register for the application through any one app store.	I can register & access the database model within app Login.	Low	Sprint-2
Customer Care Executive	Gmail account	USN-4	As a user, I can register for the application through Gmail.	I can receive confirmation email & click confirm.	Medium	Sprint-1
Administrator	Login	USN-5	As a Admin, I can log into the application by entering email & password.	I can access the model database through application.	High	Sprint-1
Customer (User)	Internet Facility	USN-6	As a user I can give input to the model through the website, blood bank app, social media, etc.	I can get the blood donor through this communication.	High	Sprint-3
Customer (User)	Laptop or Computer or Mobile	USN-7	As a user I can view the pictorial or graphical representation of blood donors.	I can insights on blood donors.	High	Sprint-4

CHAPTER 6

PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION



Project Planning Phase

Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)

Date	18 October 2022
Team ID	PNT2022TMID08712
Project Name	Plasma Donor Application
Maximum Marks	8 Marks

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

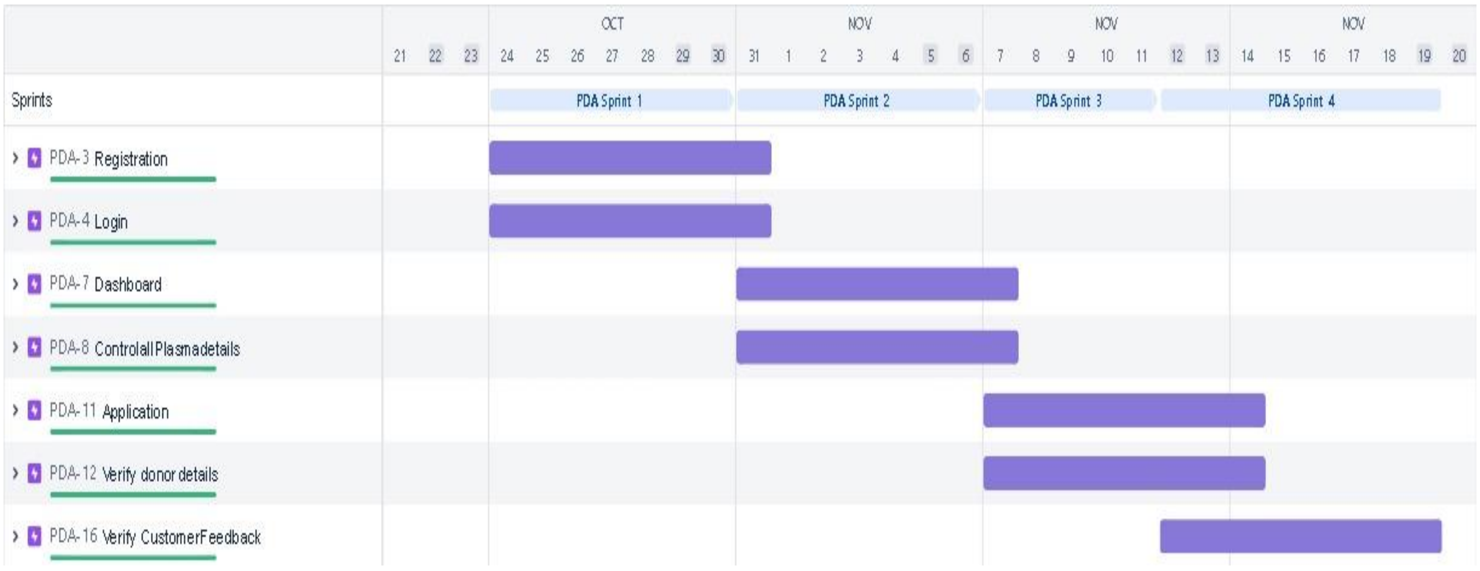
Use the below template to create product backlog and sprint schedule

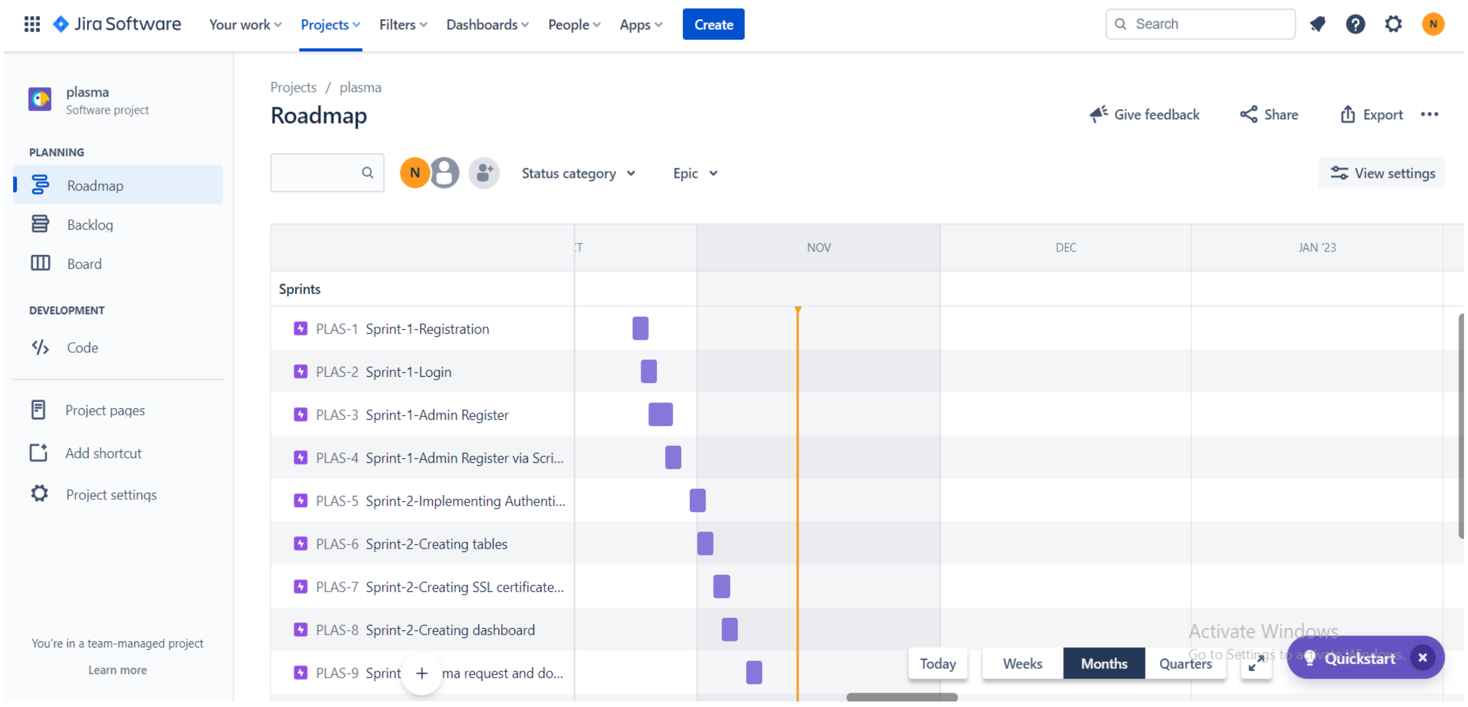
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	A user can register for the application by entering their email, password, and confirming the password.	3	High	SathishKumar Diviyya Shree Amritha
Sprint-1	Email Verification	USN-2	A user will receive confirmation email once they have registered for the application	2	High	SathishKumar Siva Karthini Diviyya Shree
Sprint-1		USN-3	A user can register for the application through Google	2	Medium	Amritha Siva Karthini Diviyya Shree
Sprint-1	Login Credentials	USN-4	As a user, I can register for the application through Gmail and password	3	High	SathishKumar Amritha Siva Karthini
Sprint-1	Donor Profile	USN-5	A user can register themselves as verified plasma donor	3	High	Amritha Siva Karthini
Sprint-2	Virtual Certificate	USN-6	A user will be receiving a virtual donor certificate after a verified accomplished plasma donation	2	Medium	SathishKumar Diviyya Shree Amritha
Sprint-2	Plasma Request	USN-7	A verified and certified hospital is able to do a plasma request in the application	3	High	Amritha Siva Karthini Diviyya Shree
Sprint-2	Verification of Donor's Information	USN-8	The administrators will verify the details provided by the donors so the genuine donors are able to use the application	2	Medium	SathishKumar Siva Karthini Amritha
Sprint-2	Accept the Donation Request	USN-9	A user and a registered donor will get a notification to accept the plasma request for their specific blood type	3	High	Amritha Siva Karthini Diviyya Shree
Sprint-3	Communication Channel	USN-10	A patient is able to communicate with the donor personally within the application.	3	Medium	SathishKumar Siva Karthini Diviyya Shree
Sprint-3		USN-11	A user and a registered donor is able to share their location with the recipient after accepting their plasma request.	3	Medium	SathishKumar Amritha Diviyya Shree
Sprint-3	Administrator	USN-12	An admin will store the registered donor's details after verification into the database.	3	High	Amritha Siva Karthini Diviyya Shree
Sprint-4	Support	USN-13	A user is able to ask basic question related to plasma donation with the help of chat-bot.s	2	Medium	Siva Karthini Diviyya Shree SathishKumar
Sprint-4		USN-14	A user can fing the answers for the frequently asked question about the plasma donation in the FAQ section	3	High	Sathish Kumar Diviyya Shree Amritha
Sprint-4	About	USN-15	A new user can read about plasma and plasma donation in dedicated about section.	2	Medium	Amritha Diviyya Shree Siva Karthini
Sprint-4	Administrator	USN-16	An admin will approve all the plasma transaction in the application	3	High	SathishKumar Siva Karthini Amritha
			After proper verification			
Sprint-4		USN-17	An admin, I will update the plasma availability and donors count periodically	3	Medium	SathishKumar Amritha Diviyya Shree

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	18	6 Days	24 Oct 2022	29 Oct 2022	18	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	18	6 Days	07 Nov 2022	12 Nov 2022	18	12 Nov 2022
Sprint-4	18	6 Days	14 Nov 2022	19 Nov 2022	18	19 Nov 2022

6.3 REPORTS FROM JIRA

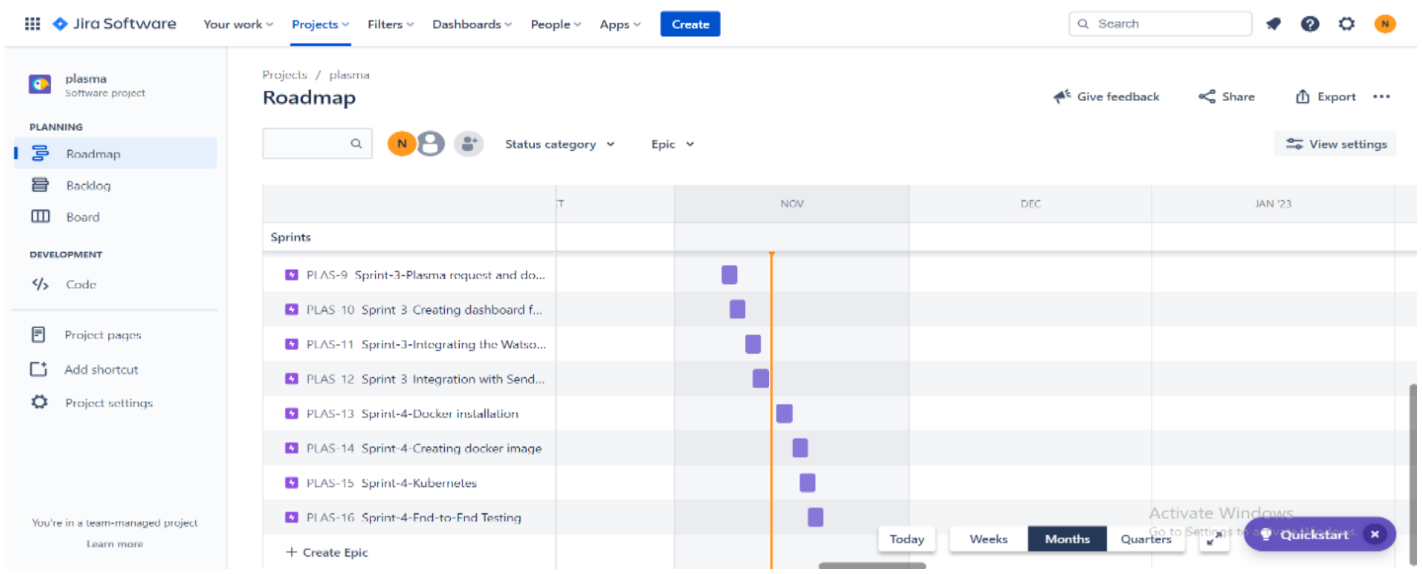




Roadmap

Sprints

PLAS-1 Sprint-1-Registration
PLAS-2 Sprint-1-Login
PLAS-3 Sprint-1-Admin Register
PLAS-4 Sprint-1-Admin Register via Scri...
PLAS-5 Sprint-2-Implementing Authenti...
PLAS-6 Sprint-2-Creating tables
PLAS-7 Sprint-2-Creating SSL certificate...
PLAS-8 Sprint-2-Creating dashboard
PLAS-9 Sprint ma request and do...

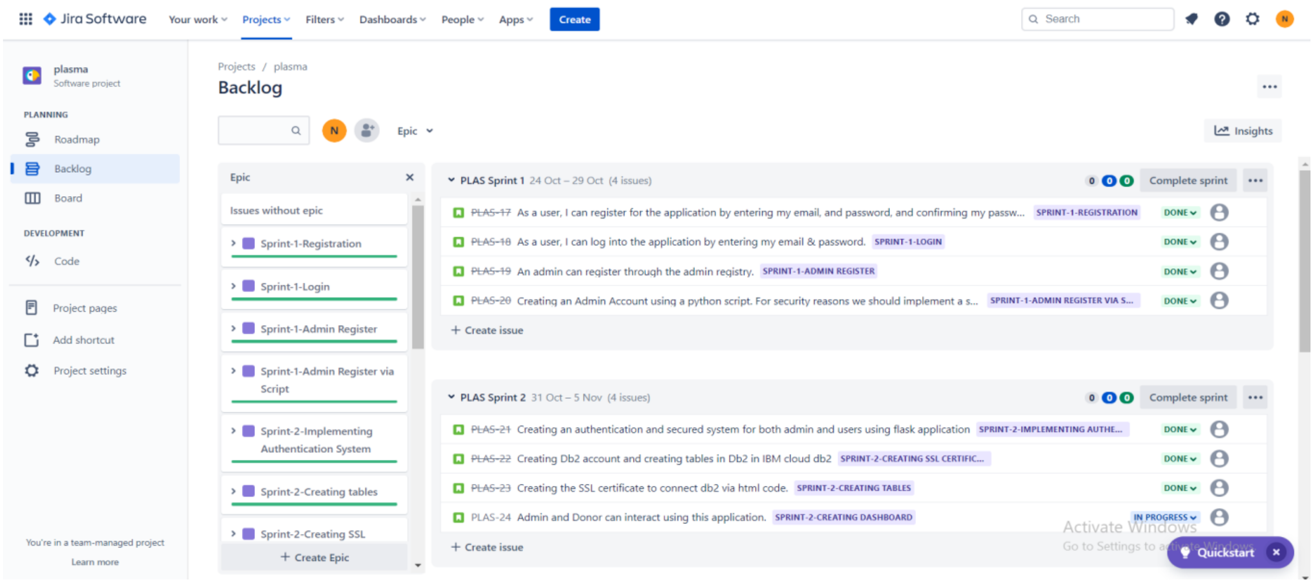


Roadmap

Sprints

PLAS-9 Sprint-3-Plasma request and do...
PLAS-10 Sprint-3-Creating dashboard f...
PLAS-11 Sprint-3-Integrating the Watso...
PLAS-12 Sprint-3-Integration with Send...
PLAS-13 Sprint-4-Docker installation
PLAS-14 Sprint-4-Creating docker image
PLAS-15 Sprint-4-Kubernetes
PLAS-16 Sprint-4-End-to-End Testing
+ Create Epic

BACKLOG



Projects / plasma

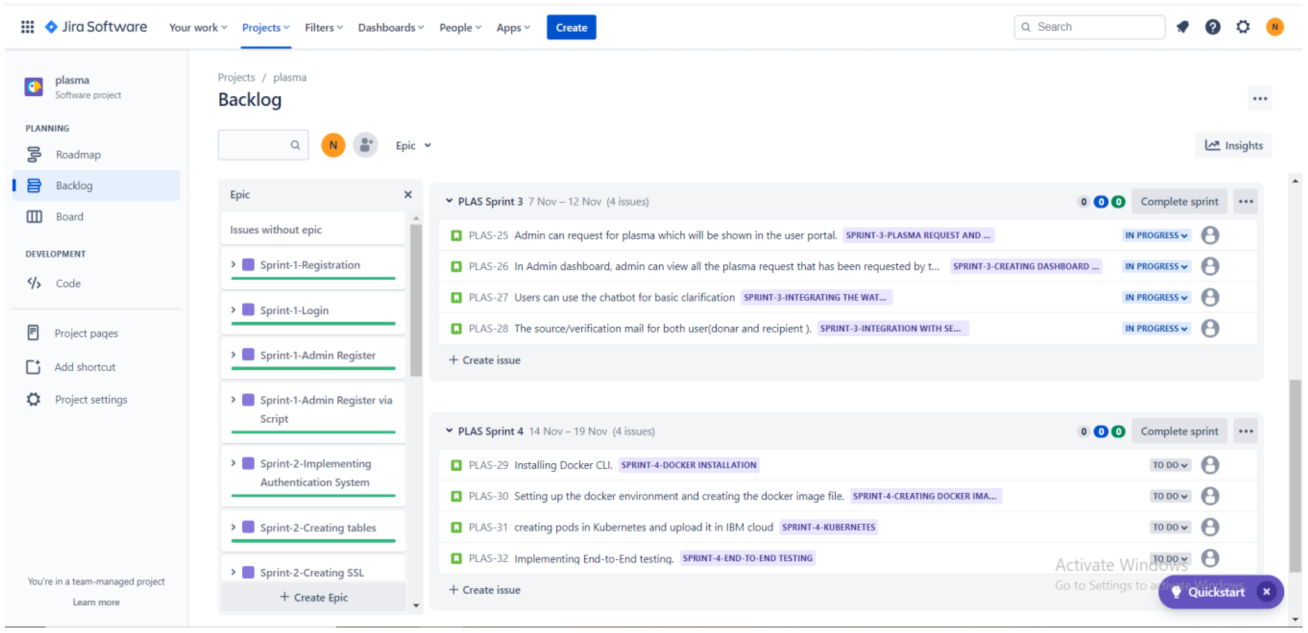
Backlog

PLAS Sprint 1 24 Oct – 29 Oct (4 issues)

- ~~PLAS-17~~ As a user, I can register for the application by entering my email, and password, and confirming my passw... SPRINT-1-REGISTRATION DONE
- ~~PLAS-18~~ As a user, I can log into the application by entering my email & password. SPRINT-1-LOGIN DONE
- ~~PLAS-19~~ An admin can register through the admin registry. SPRINT-1-ADMIN REGISTER DONE
- ~~PLAS-20~~ Creating an Admin Account using a python script. For security reasons we should implement a s... SPRINT-1-ADMIN REGISTER VIA S... DONE

PLAS Sprint 2 31 Oct – 5 Nov (4 issues)

- ~~PLAS-21~~ Creating an authentication and secured system for both admin and users using flask application SPRINT-2-IMPLEMENTING AUTHE... DONE
- ~~PLAS-22~~ Creating Db2 account and creating tables in Db2 in IBM cloud db2 SPRINT-2-CREATING SSL CERTIFIC... DONE
- ~~PLAS-23~~ Creating the SSL certificate to connect db2 via html code. SPRINT-2-CREATING TABLES DONE
- PLAS-24 Admin and Donor can interact using this application. SPRINT-2-CREATING DASHBOARD IN PROGRESS



Projects / plasma

Backlog

PLAS Sprint 3 7 Nov – 12 Nov (4 issues)

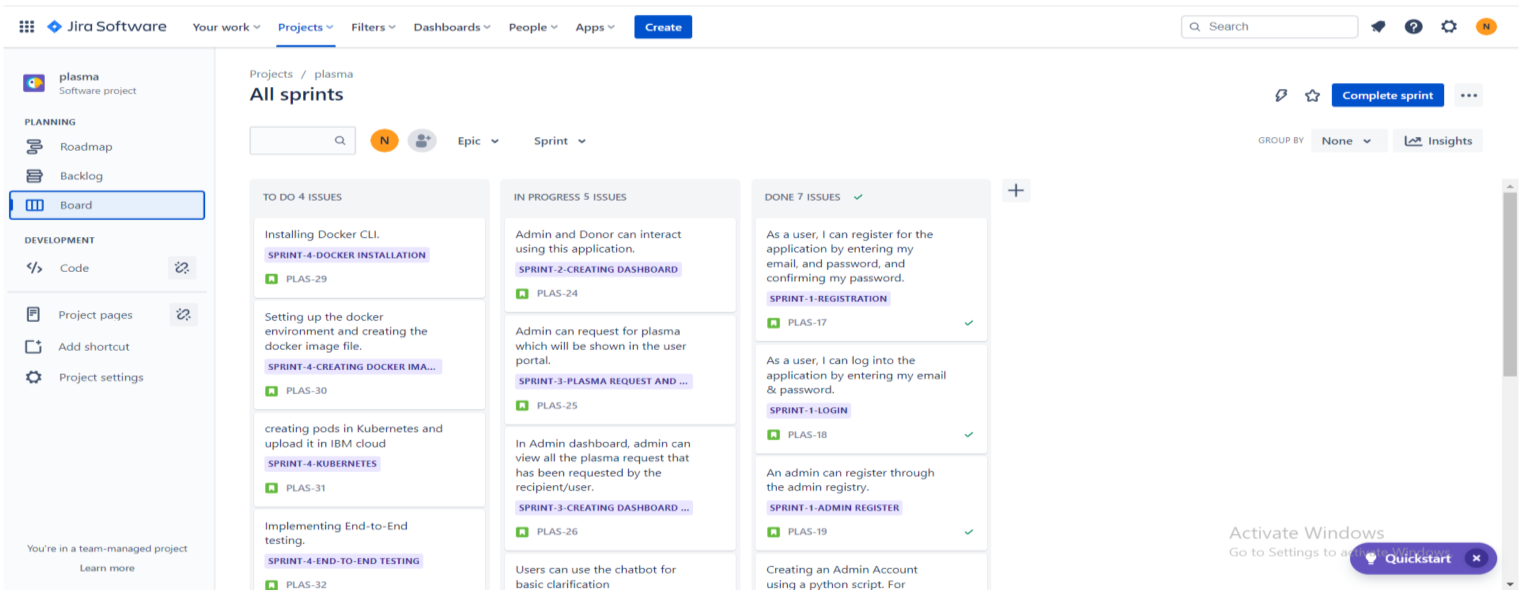
- PLAS-25 Admin can request for plasma which will be shown in the user portal. SPRINT-3-PLASMA REQUEST AND ... IN PROGRESS
- PLAS-26 In Admin dashboard, admin can view all the plasma request that has been requested by t... SPRINT-3-CREATING DASHBOARD ... IN PROGRESS
- PLAS-27 Users can use the chatbot for basic clarification SPRINT-3-INTEGRATING THE WAT... IN PROGRESS
- PLAS-28 The source/verification mail for both user(donar and recipient). SPRINT-3-INTEGRATION WITH SE... IN PROGRESS

PLAS Sprint 4 14 Nov – 19 Nov (4 issues)

- PLAS-29 Installing Docker CLI. SPRINT-4-DOCKER INSTALLATION TO DO
- PLAS-30 Setting up the docker environment and creating the docker image file. SPRINT-4-CREATING DOCKER IMA... TO DO
- PLAS-31 creating pods in Kubernetes and upload it in IBM cloud SPRINT-4-KUBERNETES TO DO
- PLAS-32 Implementing End-to-End testing. SPRINT-4-END-TO-END TESTING TO DO

Epics: Sprint-1-Registration, Sprint-1-Login, Sprint-1-Admin Register, Sprint-1-Admin Register via Script, Sprint-2-Implementing Authentication System, Sprint-2-Creating tables, Sprint-2-Creating SSL

BOARD



Jira Software | Your work | Projects | Filters | Dashboards | People | Apps | Create | Search

plasma
Software project

PLANNING
Roadmap
Backlog
Board

DEVELOPMENT
Code

Project pages
Add shortcut
Project settings

You're in a team-managed project
Learn more

Projects / plasma

All sprints

Complete sprint

Epic | Sprint | GROUP BY None | Insights

TO DO 4 ISSUES

- Installing Docker CLI.
  SPRINT-4-DOCKER INSTALLATION
  PLAS-29
- Setting up the docker environment and creating the docker image file.
  SPRINT-4-CREATING DOCKER IMA...
  PLAS-30
- creating pods in Kubernetes and upload it in IBM cloud
  SPRINT-4-KUBERNETES
  PLAS-31
- Implementing End-to-End testing.
  SPRINT-4-END-TO-END TESTING
  PLAS-32

IN PROGRESS 5 ISSUES

- Admin and Donor can interact using this application.
  SPRINT-2-CREATING DASHBOARD
  PLAS-24
- Admin can request for plasma which will be shown in the user portal.
  SPRINT-3-PLASMA REQUEST AND ...
  PLAS-25
- In Admin dashboard, admin can view all the plasma request that has been requested by the recipient/user.
  SPRINT-3-CREATING DASHBOARD ...
  PLAS-26
- Users can use the chatbot for basic clarification

DONE 7 ISSUES

- As a user, I can register for the application by entering my email, and password, and confirming my password.
  SPRINT-1-REGISTRATION
  PLAS-17 ✓
- As a user, I can log into the application by entering my email & password.
  SPRINT-1-LOGIN
  PLAS-18 ✓
- An admin can register through the admin registry.
  SPRINT-1-ADMIN REGISTER
  PLAS-19 ✓
- Creating an Admin Account using a python script. For

Activate Windows
Go to Settings to activate Windows.

Quickstart

CHAPTER 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.

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

KUBERNATES

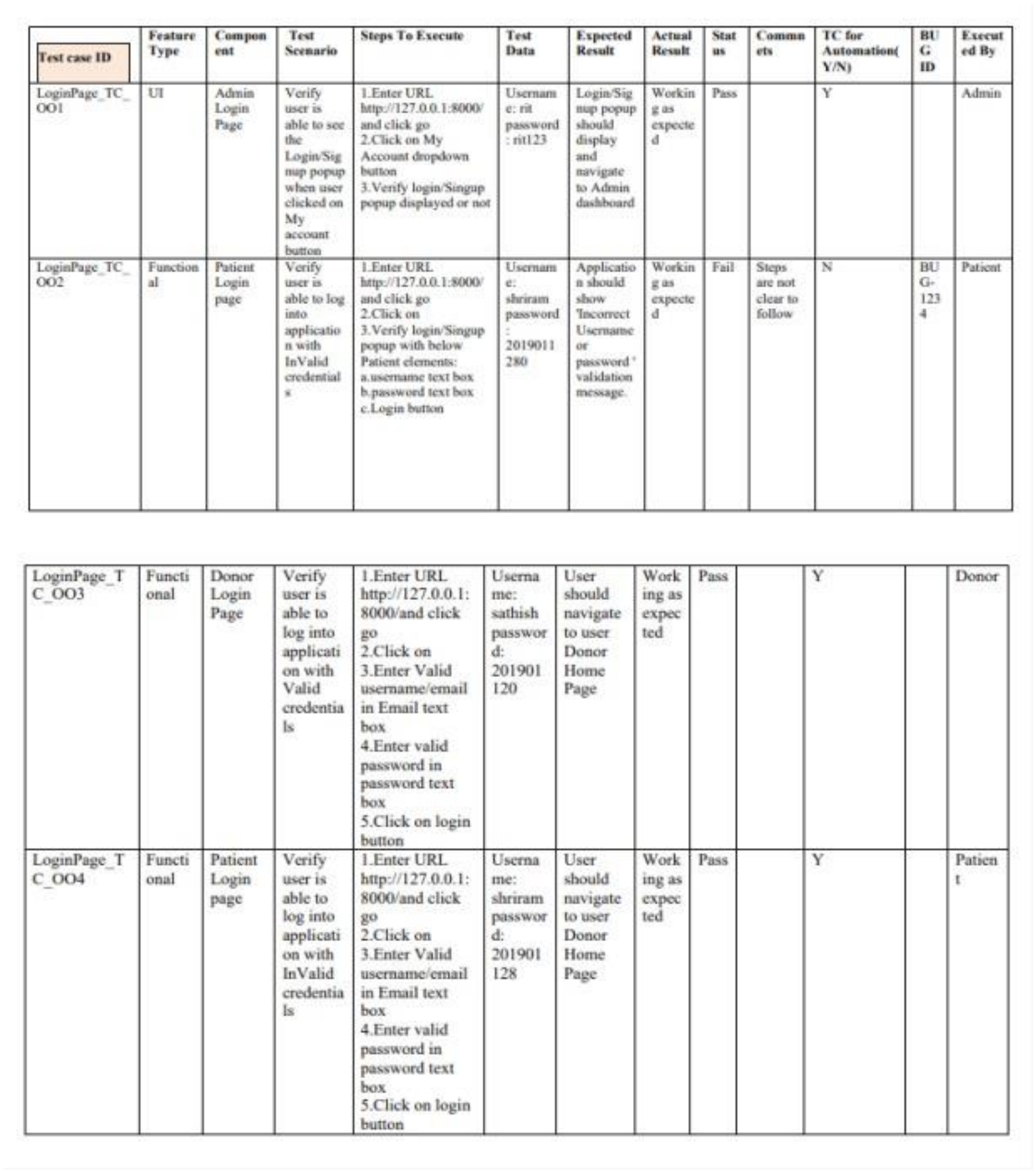
- **Kubernetes** is also known as **'k8s'.**
- **Kubernetes** is an extensible, portable, and open-source platform designed by **Google** in **2014**.
- It is mainly used to automate the deployment, scaling, and operations of the container-based applications across the cluster of nodes.
- Kubernetes 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.

CHAPETR 8

TESTING

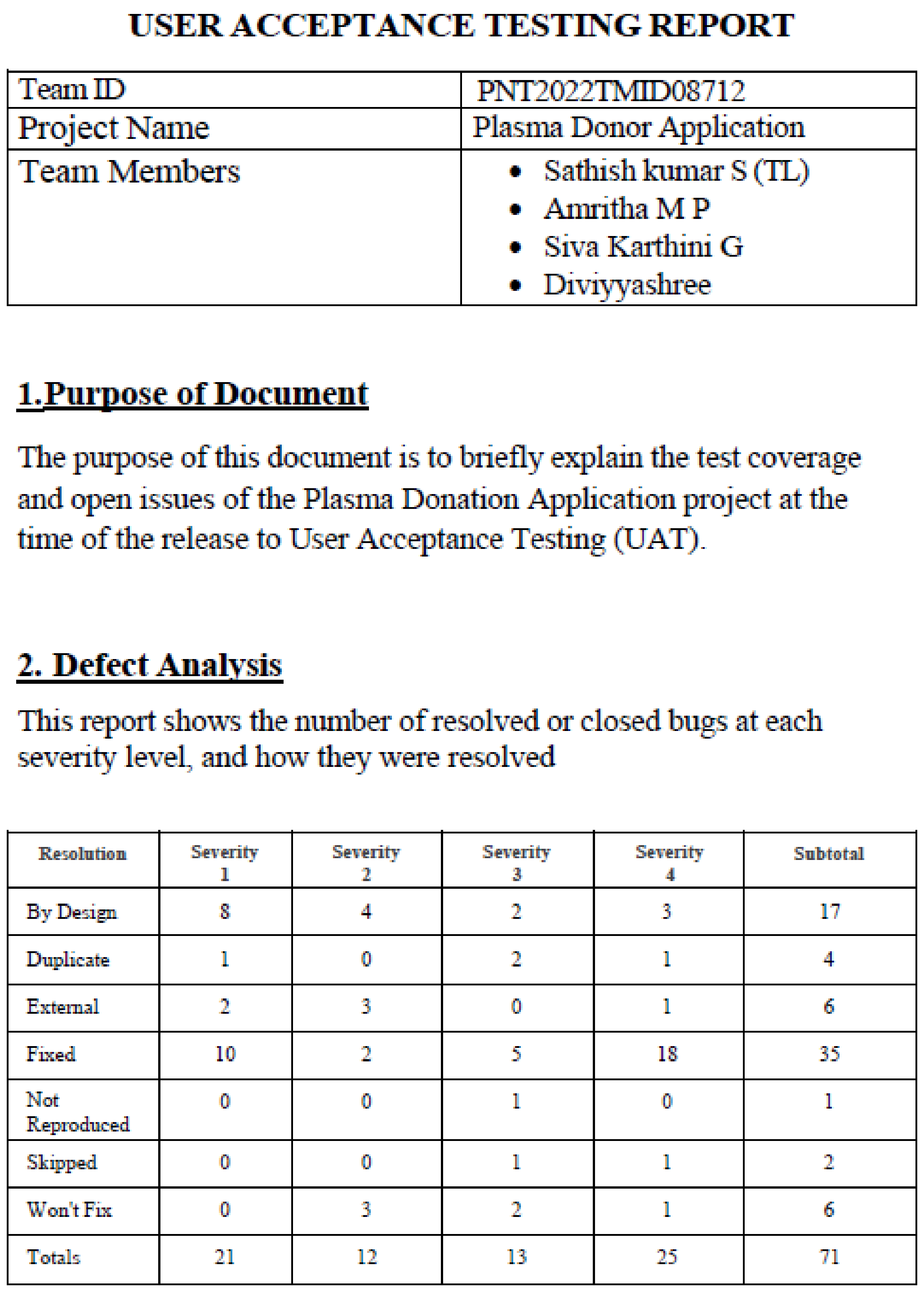
8.1 TEST CASE

- 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/Signup 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



USER ACCEPTANCE TESTING REPORT

Team ID	PNT2022TMID08712
Project Name	Plasma Donor Application
Team Members	• Sathish kumar S (TL) • Amritha M P • Siva Karthini G • Diviyyashree

<u>1.Purpose of Document</u>

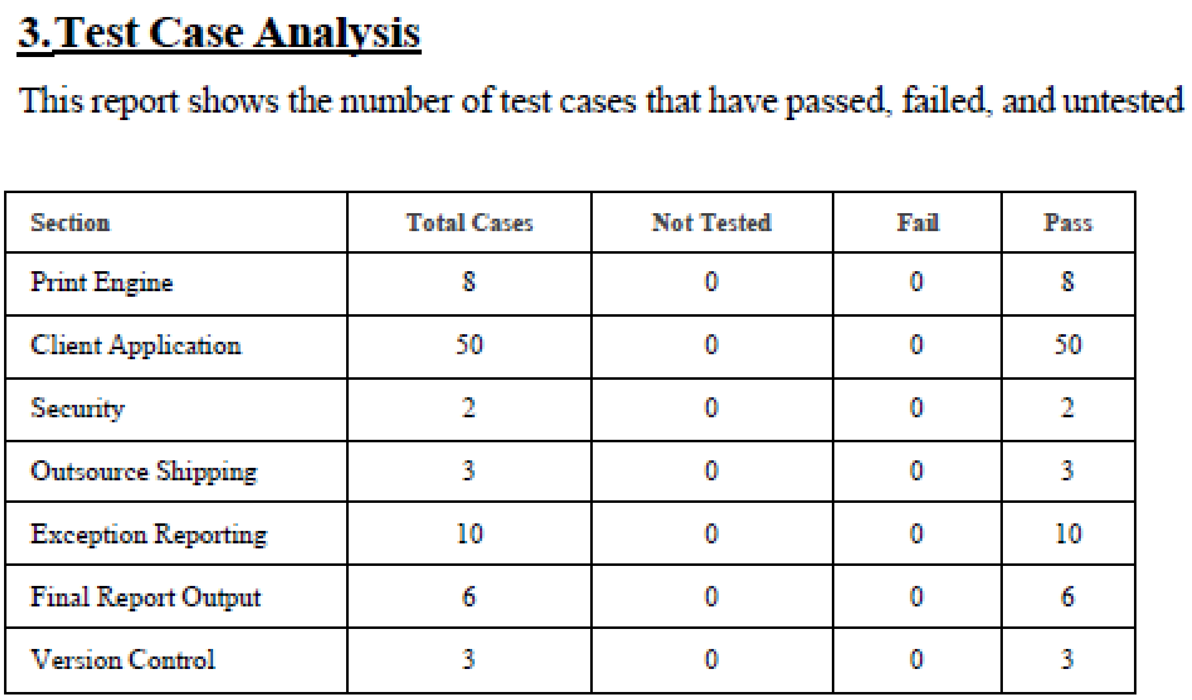
The purpose of this document is to briefly explain the test coverage and open issues of the Plasma Donation Application project at the time of the release to User Acceptance Testing (UAT).

<u>2. Defect Analysis</u>

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

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	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	71

3.TEST CASE ANALYSIS



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

CHAPTER 9

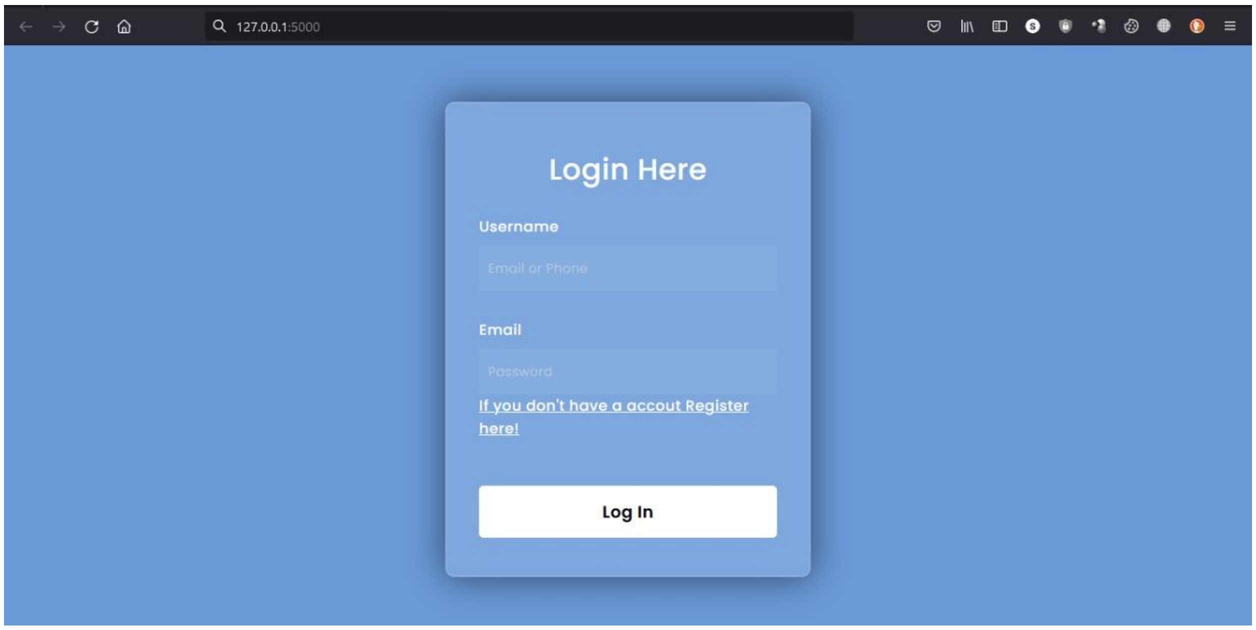
RESULTS

9.1 PERFORMANCE METRICS

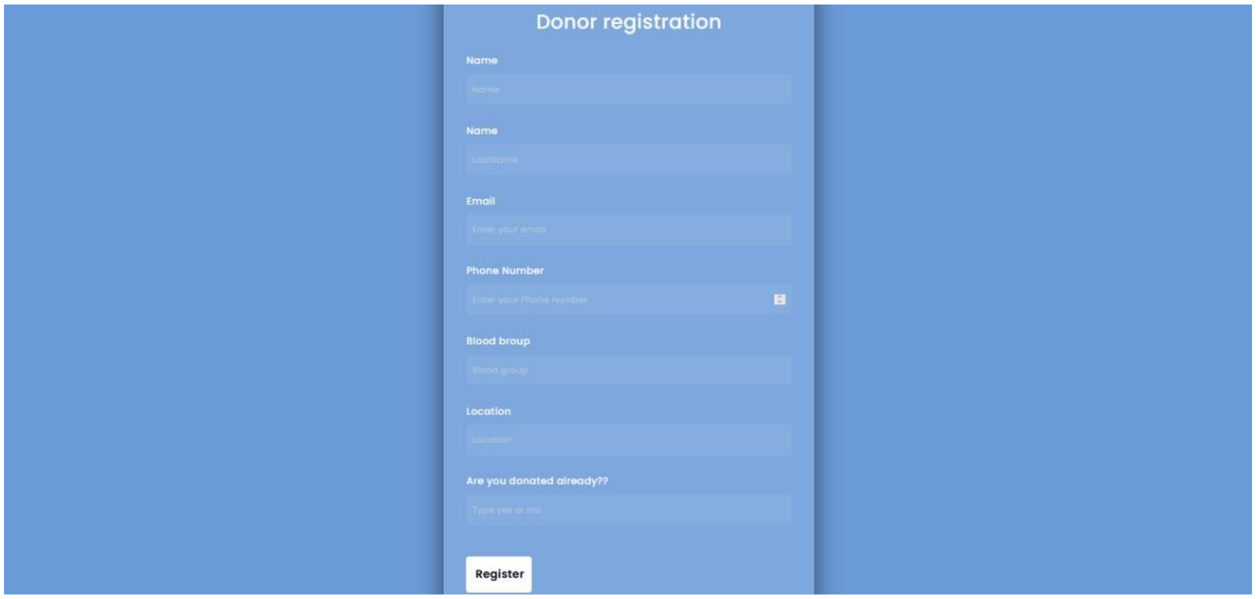
- Project metrics are used to track the progress and performance of a project.
- Monitoring parts of a project like productivity, scheduling, and scope make it easier for team leaders to see what's on track.
- As a project evolves, managers need access to changing
- deadlines or budgets to meet their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

OUTPUT SCREENS

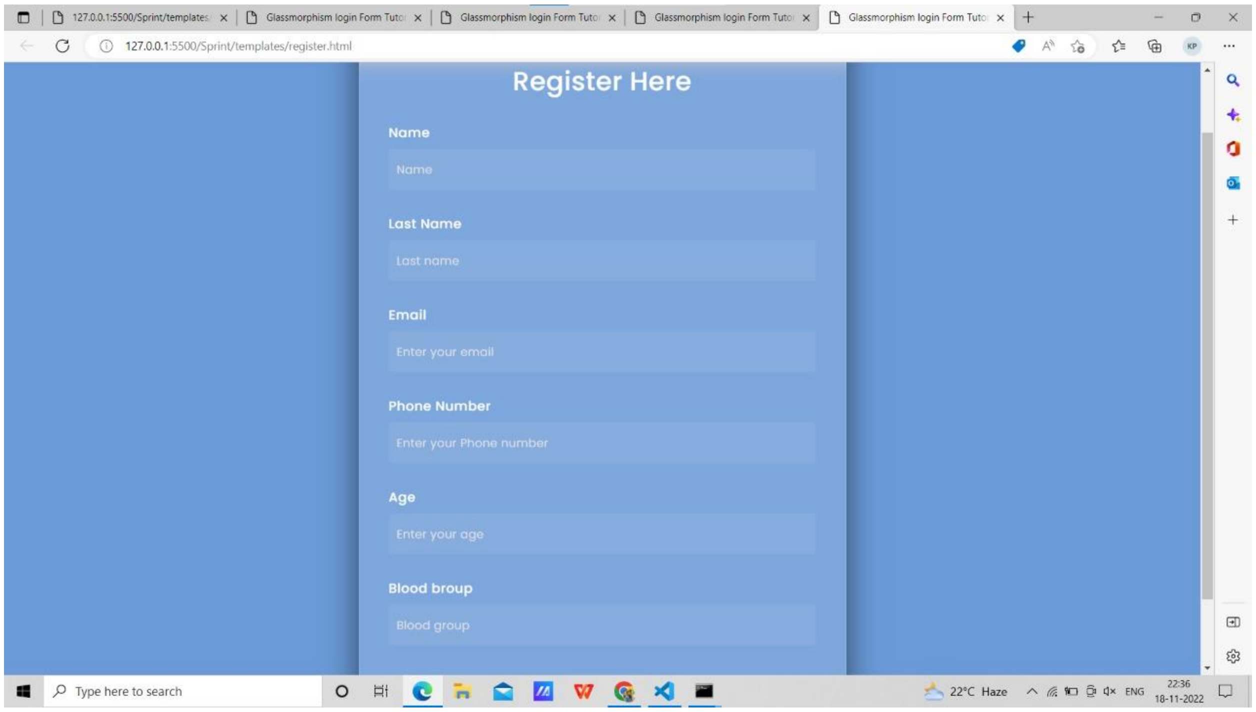
LOGIN PAGE



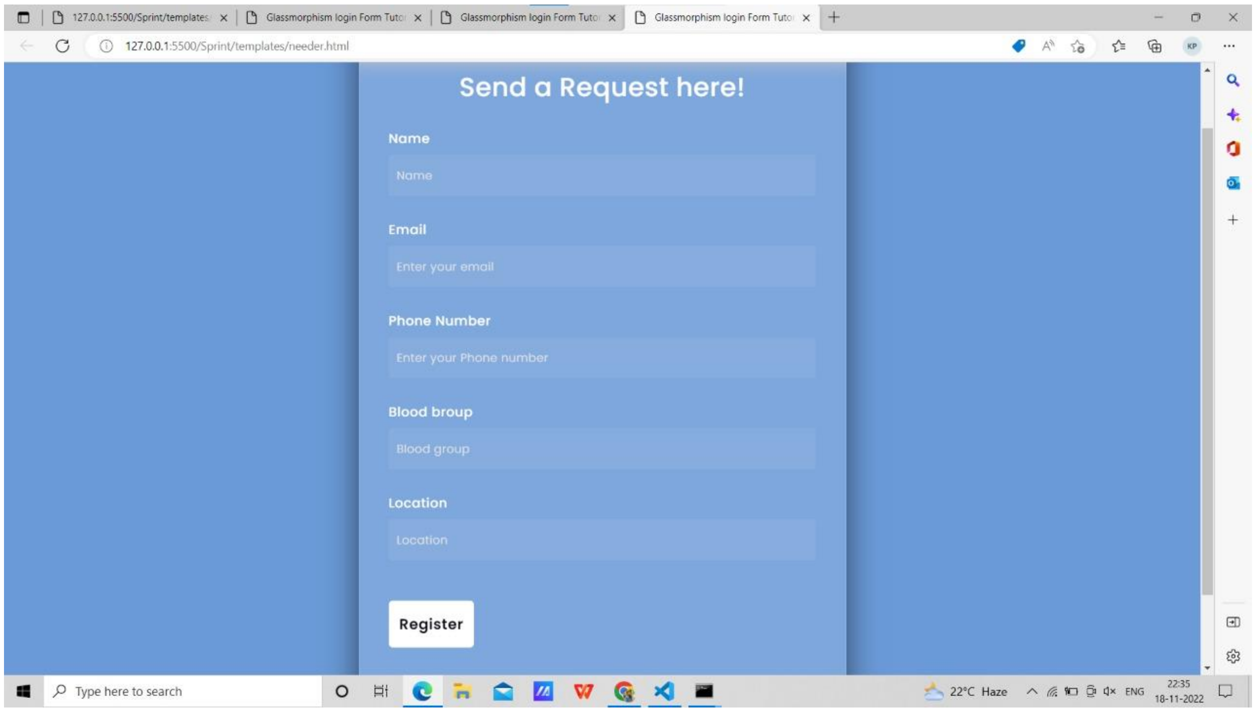
DONOR REGISTRATION



REGISTER PAGE



REQUEST PAGE


127.0.0.1:5500/Sprint/templates/needer.html

Send a Request here!

Name: Name

Email: Enter your email

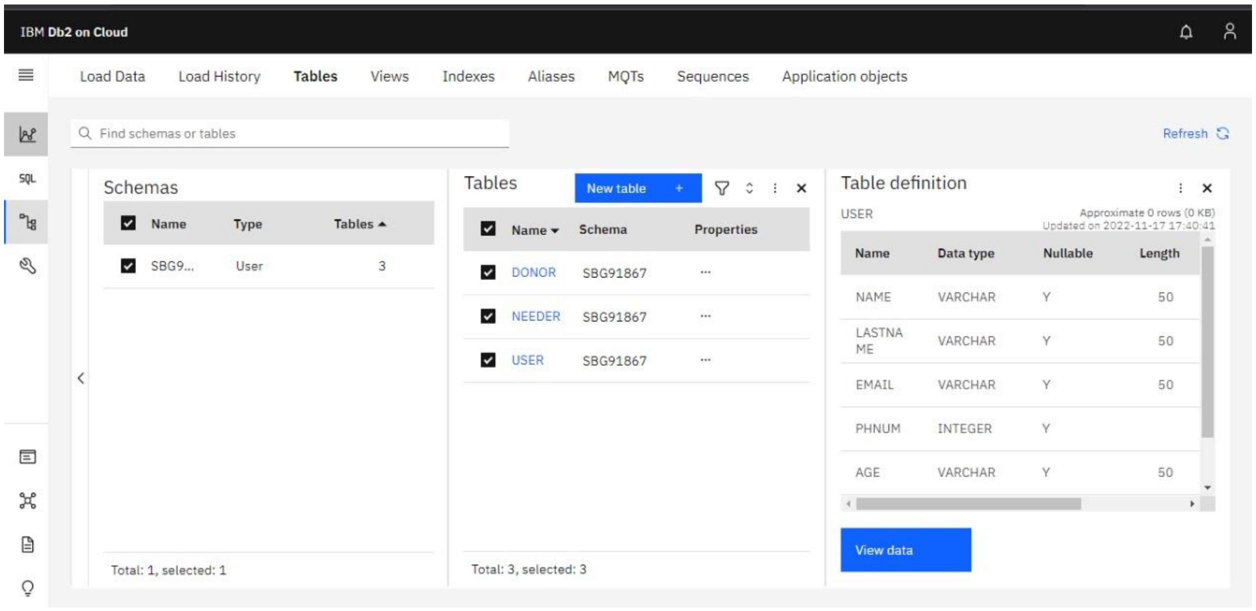
Phone Number: Enter your Phone number

Blood broup: Blood group

Location: Location

Register

IBM DB2


IBM Db2 on Cloud

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

Schemas

Name	Type	Tables
SBG9...	User	3

Total: 1, selected: 1

Tables

Name	Schema	Properties
DONOR	SBG91867	...
NEEDER	SBG91867	...
USER	SBG91867	...

Total: 3, selected: 3

Table definition

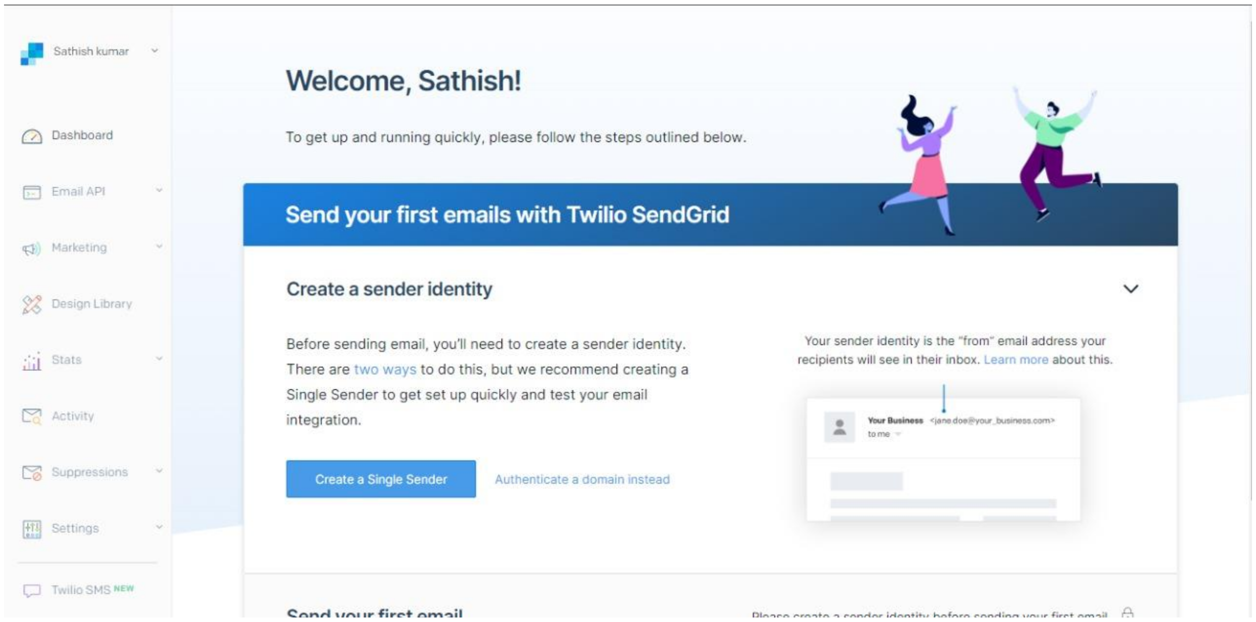
USER

Approximate 0 rows (0 KB)
Updated on 2022-11-17 17:40:41

Name	Data type	Nullable	Length
NAME	VARCHAR	Y	50
LASTNAME	VARCHAR	Y	50
EMAIL	VARCHAR	Y	50
PHNUM	INTEGER	Y	
AGE	VARCHAR	Y	50

View data

SENDGRID INTEGRATION



Welcome, Sathish!

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

Send your first emails with Twilio SendGrid

Create a sender identity

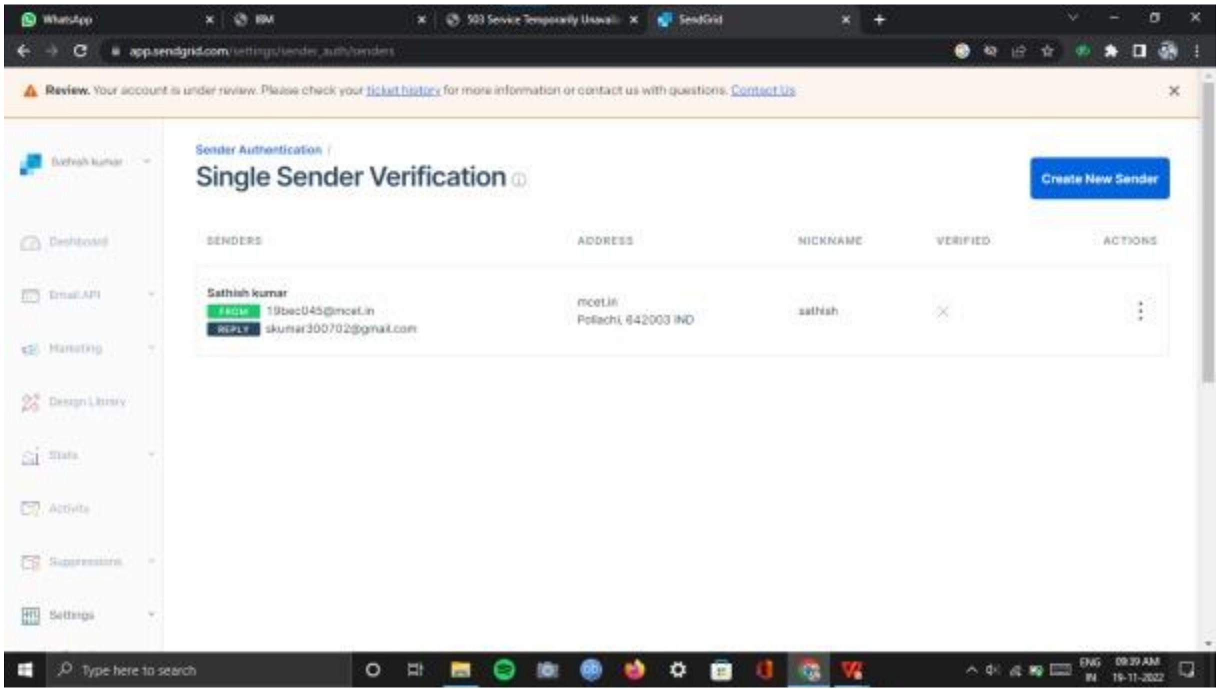
Before sending email, you'll need to create a sender identity. There are two ways to do this, but we recommend creating a Single Sender to get set up quickly and test your email integration.

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

Create a Single Sender
Authenticate a domain instead
Your Business <jane.doe@your_business.com>
to me

Send your first email

Please create a sender identity before sending your first email


WhatsApp
IBM
503 Service Temporarily Unavail
SendGrid
app.sendgrid.com/settings/sender_auth/senders

Review. Your account is under review. Please check your ticket history for more information or contact us with questions. Contact Us

Sender Authentication /

Single Sender Verification

SENDERS	ADDRESS	NICKNAME	VERIFIED	ACTIONS
Sathish kumar FROM 19bec045@mcet.in REPLY skumar300702@gmail.com	mcet.in Pollachi, 642003 IND	sathish	✗	⋮

CHAPTER 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.

CHAPTER 11
CONCLUSION

- The efficient way of finding plasma door 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 & Kubernates Cluster to make sure the operations are running successfully Cloud lambda function is used and to deploy the application IBM Db2 service is used.

CHAPTER 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

CHAPTER 13

APPENDIXES

13.1 SAMPLE SOURCE CODE:

DONOR.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <!-- Design by foolishdeveloper.com -->
    <title>Glassmorphism login Form Tutorial in html css</title>
    <link rel="preconnect" href="https://fonts.gstatic.com">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.15.4/css/all.min.css">
    <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;500;600&dis
play=swap" rel="stylesheet">
    <!--Stylesheet-->
    <style media="screen">
      *,
*:before,
*:after{
    padding: 0;
    margin: 0;
    box-sizing: border-box;
}
body{
    background-color: #080710;
}
.background{
    width: 230px;
    height: 520px;
    position: absolute;
    transform: translate(-50%,-50%);
    left: 50%;
    top: 50%;
}
.background .shape{
    height: 200px;
    width: 200px;
    position: absolute;
    border-radius: 50%;
}
.vertical-center {
  margin: 0;
  position: absolute;
  top: 50%;
  -ms-transform: translateY(-50%);
  transform: translateY(-50%);
}
form{
    margin-top:8%;
    height: 120%;
    width: 600px;
    background-color: rgba(255,255,255,0.13);
    position: absolute;
    transform: translate(-50%,-50%);
    top: 50%;
    left: 50%;
    border-radius: 10px;
    backdrop-filter: blur(10px);
    border: 2px solid rgba(255,255,255,0.1);
    box-shadow: 0 0 40px rgba(8,7,16,0.6);
    padding: 50px 35px;
}
form *{
    font-family: 'Poppins',sans-serif;
    color: #ffffff;
    letter-spacing: 0.5px;
    outline: none;
    border: none;
}
form h3{
    font-size: 32px;
    font-weight: 500;
    line-height: 42px;
    text-align: center;
}
label{
    display: block;
    margin-top: 30px;
    font-size: 16px;
    font-weight: 500;
}
input{
    display: block;
    height: 50px;
    width: 100%;
    background-color: rgba(255,255,255,0.07);
    border-radius: 3px;
    padding: 0 10px;
    margin-top: 8px;
    font-size: 14px;
    font-weight: 300;
}
::placeholder{
    color: #e5e5e5;
}
button{
    margin-top: 50px;
    width: 100%;
    background-color: #ffffff;
    color: #080710;
    padding: 15px 0;
    font-size: 18px;
    font-weight: 600;
    border-radius: 5px;
    cursor: pointer;
}
.social{
  margin-top: 30px;
  display: flex;
}
.social div{
  background: red;
  width: 150px;
  border-radius: 3px;
  padding: 5px 10px 10px 5px;
  background-color: rgba(255,255,255,0.27);
  color: #eaf0fb;
  text-align: center;
}
.social div:hover{
  background-color: rgba(255,255,255,0.47);
}
.social .fb{
  margin-left: 25px;
}
.social i{
  margin-right: 4px;
}
    </style>
</head>
{% include 'navbar.html' %}
<body>
    <div class="background">
        <div class="shape"></div>
        <div class="shape"></div>
    </div>
    <form action="{{url_for('donorpage')}}" method="POST">

        <h3>Donor registration</h3>

        <label for="uname">Name</label>

        <input type="text" placeholder="Name" name="name">

        <label for="uname">Name</label>

        <input type="text" placeholder="Name" name="lname">

        <label for="Email">Email</label>

        <input type="text" placeholder="Enter your email" name="email">

        <label for="phnum">Phone Number</label>

        <input type="number" placeholder="Enter your Phone number"
name="phnum">

        <label for="blood group">Blood broup</label>

        <input type="text" placeholder="Blood group" name="bloodgrp">

        <label for="location">Location</label>

        <input type="text" placeholder="Location" name="location">

        <label>Are you donated already??</label>

        <input type="text" placeholder="Type yes or no!" name="donated">

        <button style="height: 5%; width: 20%; align-items:
center;">Register</button>

    </form>

</body>

</html>
```

INDEX.HTML

```
{%include 'navbar.html' %}
```

```
<!DOCTYPE html>

<html>

<head>

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <style>

    * {box-sizing: border-box}

    body {font-family: Verdana, sans-serif; margin:0}

    .mySlides {display: none}

    img {vertical-align: middle;}

    /* Slideshow container */

    .slideshow-container {

      max-width: 1000px;

      position: relative;

      margin: auto;

    }

    /* Next & previous buttons */

    .prev, .next {

      cursor: pointer;

      position: absolute;

      top: 50%;

      width: auto;

      padding: 16px;

      margin-top: -22px;
      color: white;
      font-weight: bold;
      font-size: 18px;
      transition: 0.6s ease;
      border-radius: 0 3px 3px 0;
      user-select: none;
    }
    /* Position the "next button" to the right */
    .next {
      right: 0;
      border-radius: 3px 0 0 3px;
    }
    /* On hover, add a black background color with a little bit seethrough */
    .prev:hover, .next:hover {
      background-color: rgba(0,0,0,0.8);
    }
    /* Caption text */
    .text {
      color: black;
      font-size: 15px;
      padding: 8px 12px;
      position: absolute;
      bottom: 8px;
      width: 100%;
      text-align: center;
    }
    /* Number text (1/3 etc) */
    .numbertext {
      color: #f2f2f2;
      font-size: 12px;
      padding: 8px 12px;
      position: absolute;
      top: 0;
    }
    /* The dots/bullets/indicators */
    .dot {
      cursor: pointer;
      height: 15px;
      width: 15px;
      margin: 0 2px;
      background-color: #bbb;
      border-radius: 50%;
      display: inline-block;
      transition: background-color 0.6s ease;
    }
    .active, .dot:hover {
        background-color: #717171;
      }
  </style>
</head>
<body>
  <div class="slideshow-container">
      <div class="mySlides fade">
        <div class="numbertext">1 / 3</div>
        <img src="https://cdn.pixabay.com/photo/2019/04/29/16/56/blood-donation-
4166552_960_720.jpg" style="width:100%">
        <div class="text">Donate blood</div>
        </div>
        <div class="mySlides fade">
          <div class="numbertext">2 / 3</div>
          <img src="https://cdn.pixabay.com/photo/2017/10/11/21/07/blood-
2842450_960_720.jpg  " style="width:100%;">
          <div class="text"></div>
        </div>
        <div class="mySlides fade">
          <div class="numbertext">3 / 3</div>
          <img src="https://cdn.pixabay.com/photo/2020/02/19/06/23/earth-
4861456_960_720.jpg" style="width:100%">
          <div class="text">Save World!!!</div>
        </div>
```

```
    <a class="prev" onclick="plusSlides(-1)">❮</a>
    <a class="next" onclick="plusSlides(1)">❯</a>
</div>
<br>
<div style="text-align:center">
  <span class="dot" onclick="currentSlide(1)"></span>
  <span class="dot" onclick="currentSlide(2)"></span>
  <span class="dot" onclick="currentSlide(3)"></span>
</div>
<script>
  var slideIndex = 1;
  showSlides(slideIndex);
  function plusSlides(n) {
    showSlides(slideIndex += n);
  }
  function currentSlide(n) {
    showSlides(slideIndex = n);
  }
  function showSlides(n) {
    var i;
    var slides = document.getElementsByClassName("mySlides");
    var dots = document.getElementsByClassName("dot");
    if (n > slides.length) {slideIndex = 1}
```

```
        if (n < 1) {slideIndex = slides.length}
        for (i = 0; i < slides.length; i++) {
          slides[i].style.display = "none";
        }
        for (i = 0; i < dots.length; i++) {
          dots[i].className = dots[i].className.replace(" active", "");
        }
        slides[slideIndex-1].style.display = "block";
        dots[slideIndex-1].className += " active";
      }
  </script>
</body>
</html>
```

LOGIN.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <!-- Design by foolishdeveloper.com -->
    <title>Glassmorphism login Form Tutorial in html css</title>
    <link rel="preconnect" href="https://fonts.gstatic.com">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.15.4/css/all.min.css">
```

```
    <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;500;600&dis
play=swap" rel="stylesheet">

    <!--Stylesheet-->

    <style media="screen">

      *,

*:before,

*:after{

    padding: 0;

    margin: 0;

    box-sizing: border-box;

}

body{

    background-color: #080710;

}

.background{

    width: 430px;

    height: 520px;

    position: absolute;

    transform: translate(-50%,-50%);

    left: 50%;

    top: 50%;

}

.background .shape{
    height: 200px;
    width: 200px;
    position: absolute;
    border-radius: 50%;
}
form{
    height: 520px;
    width: 400px;
    background-color: rgba(255,255,255,0.13);
    position: absolute;
    transform: translate(-50%,-50%);
    top: 50%;
    left: 50%;
    border-radius: 10px;
    backdrop-filter: blur(10px);
    border: 2px solid rgba(255,255,255,0.1);
    box-shadow: 0 0 40px rgba(8,7,16,0.6);
    padding: 50px 35px;
}
form *{
    font-family: 'Poppins',sans-serif;
    color: #ffffff;
    letter-spacing: 0.5px;
    outline: none;
    border: none;
}
form h3{
    font-size: 32px;
    font-weight: 500;
    line-height: 42px;
    text-align: center;
}
label{
    display: block;
    margin-top: 30px;
    font-size: 16px;
    font-weight: 500;
}
input{
    display: block;
    height: 50px;
    width: 100%;
    background-color: rgba(255,255,255,0.07);
    border-radius: 3px;
    padding: 0 10px;
    margin-top: 8px;
    font-size: 14px;
    font-weight: 300;
}
::placeholder{
    color: #e5e5e5;
}
button{
    margin-top: 50px;
    width: 100%;
    background-color: #ffffff;
    color: #080710;
    padding: 15px 0;
    font-size: 18px;
    font-weight: 600;
    border-radius: 5px;
    cursor: pointer;
}
.social{
  margin-top: 30px;
  display: flex;
}
.social div{
  background: red;
  width: 150px;
  border-radius: 3px;
  padding: 5px 10px 10px 5px;
  background-color: rgba(255,255,255,0.27);
  color: #eaf0fb;
  text-align: center;
}
.social div:hover{
  background-color: rgba(255,255,255,0.47);
}
.social .fb{
  margin-left: 25px;
}
.social i{
  margin-right: 4px;
}
    </style>
</head>
{%include 'navbar.html' %}
<body>
    <div class="background">
        <div class="shape"></div>
        <div class="shape"></div>
    </div>

    <form action="{{url_for('loginpage')}}" method="POST">

        <h3>Login Here</h3>

        <label for="username">Username</label>

        <input type="text" placeholder="Email or Phone" name="user">

        <label for="password">Email</label>

        <input type="password" placeholder="Password" name="passw">

        <a href="{{url_for('register')}}"><p>If you don't have a accout Register
here!</p></a>

        <button>Log In</button>

    </form>

</body>

</html>
```

NAVABAR.HTML

```
<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width, initial-scale=1">

<style>

body {

  margin: 0;

  font-family: Arial, Helvetica, sans-serif;

}
```

```
.topnav {
  overflow: hidden;
  background-color: rgb(248, 35, 35);
}
.topnav a {
  float: left;
  color: #f2f2f2;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  font-size: 17px;
}
.topnav a:hover {
  background-color: #ddd;
  color: black;
}
</style>
</head>
<body>
<div class="topnav">
  <a class="active" href="{{url_for('home')}}">Home</a>
  <a href="{{url_for('donor')}}">Donor</a>
  <a href="{{url_for('needer')}}">Needer</a>
  <a href="{{url_for('login')}}">Login</a>

</div>

</body>

</html>
```

NEEDER.HTML

```
<!DOCTYPE html>

<html lang="en">

<head>

  <!-- Design by foolishdeveloper.com -->

    <title>Glassmorphism login Form Tutorial in html css</title>

    <link rel="preconnect" href="https://fonts.gstatic.com">

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.15.4/css/all.min.css">

    <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;500;600&dis
play=swap" rel="stylesheet">

    <!--Stylesheet-->

    <style media="screen">

      *,

*:before,

*:after{

    padding: 0;

    margin: 0;

    box-sizing: border-box;
```

```
}
body{
    background-color: #080710;
}
.background{
    width: 230px;
    height: 520px;
    position: absolute;
    transform: translate(-50%,-50%);
    left: 50%;
    top: 50%;
}
.background .shape{
    height: 200px;
    width: 200px;
    position: absolute;
    border-radius: 50%;
}
.vertical-center {
  margin: 0;
  position: absolute;
  top: 50%;
  -ms-transform: translateY(-50%);
  transform: translateY(-50%);
}
form{
    margin-top:8%;
    height: 120%;
    width: 600px;
    background-color: rgba(255,255,255,0.13);
    position: absolute;
    transform: translate(-50%,-50%);
    top: 50%;
    left: 50%;
    border-radius: 10px;
    backdrop-filter: blur(10px);
    border: 2px solid rgba(255,255,255,0.1);
    box-shadow: 0 0 40px rgba(8,7,16,0.6);
    padding: 50px 35px;
}
form *{
    font-family: 'Poppins',sans-serif;
    color: #ffffff;
    letter-spacing: 0.5px;
    outline: none;
    border: none;
```

```
}
form h3{
    font-size: 32px;
    font-weight: 500;
    line-height: 42px;
    text-align: center;
}
label{
    display: block;
    margin-top: 30px;
    font-size: 16px;
    font-weight: 500;
}
input{
    display: block;
    height: 50px;
    width: 100%;
    background-color: rgba(255,255,255,0.07);
    border-radius: 3px;
    padding: 0 10px;
    margin-top: 8px;
    font-size: 14px;
    font-weight: 300;
```

```
}
::placeholder{
    color: #e5e5e5;
}
button{
    margin-top: 50px;
    width: 100%;
    background-color: #ffffff;
    color: #080710;
    padding: 15px 0;
    font-size: 18px;
    font-weight: 600;
    border-radius: 5px;
    cursor: pointer;
}
.social{
  margin-top: 30px;
  display: flex;
}
.social div{
  background: red;
  width: 150px;
  border-radius: 3px;
  padding: 5px 10px 10px 5px;
  background-color: rgba(255,255,255,0.27);
  color: #eaf0fb;
  text-align: center;
}
.social div:hover{
  background-color: rgba(255,255,255,0.47);
}
.social .fb{
  margin-left: 25px;
}
.social i{
  margin-right: 4px;
}
    </style>
</head>
{% include 'navbar.html' %}
<body>
    <div class="background">
        <div class="shape"></div>
        <div class="shape"></div>
    </div>
    <form action="{{url_for('neederpage')}}" method="POST">
```

```
        <h3>Send a Request here!</h3>

        <label for="uname">Name</label>

        <input type="text" placeholder="Name" name="name">

        <label for="Email">Email</label>

        <input type="text" placeholder="Enter your email" name="email">

        <label for="phnum">Phone Number</label>

        <input type="number" placeholder="Enter your Phone number"
name="phnum">

        <label for="blood group">Blood broup</label>

        <input type="text" placeholder="Blood group" name="bloodgrp">

        <label for="location">Location</label>

        <input type="text" placeholder="Location" name="location">

        <button style="height: 5%; width: 20%; align-items:
center;">Register</button>

    </form>

</body>

</html>
```

REGISTER.HTML

```
<!DOCTYPE html>

<html lang="en">

<head>

  <!-- Design by foolishdeveloper.com -->

    <title>Glassmorphism login Form Tutorial in html css</title>

    <link rel="preconnect" href="https://fonts.gstatic.com">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.15.4/css/all.min.css">

    <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;500;600&dis
play=swap" rel="stylesheet">

    <!--Stylesheet-->

    <style media="screen">

      *,

*:before,

*:after{

    padding: 0;

    margin: 0;

    box-sizing: border-box;

}

body{

    background-color: #080710;

}

.background{

    width: 230px;

    height: 520px;

    position: absolute;

    transform: translate(-50%,-50%);

    left: 50%;

    top: 50%;
```

```
}
.background .shape{
    height: 200px;
    width: 200px;
    position: absolute;
    border-radius: 50%;
}
.vertical-center {
  margin: 0;
  position: absolute;
  top: 50%;
  -ms-transform: translateY(-50%);
  transform: translateY(-50%);
}
form{
    margin-top:8%;
    height: 120%;
    width: 600px;
    background-color: rgba(255,255,255,0.13);
    position: absolute;
    transform: translate(-50%,-50%);
    top: 50%;
    left: 50%;
    border-radius: 10px;
    backdrop-filter: blur(10px);
    border: 2px solid rgba(255,255,255,0.1);
    box-shadow: 0 0 40px rgba(8,7,16,0.6);
    padding: 50px 35px;
}
form *{
    font-family: 'Poppins',sans-serif;
    color: #ffffff;
    letter-spacing: 0.5px;
    outline: none;
    border: none;
}
form h3{
    font-size: 32px;
    font-weight: 500;
    line-height: 42px;
    text-align: center;
}

label{
    display: block;
    margin-top: 30px;
    font-size: 16px;
    font-weight: 500;
}
input{
    display: block;
    height: 50px;
    width: 100%;
    background-color: rgba(255,255,255,0.07);
    border-radius: 3px;
    padding: 0 10px;
    margin-top: 8px;
    font-size: 14px;
    font-weight: 300;
}
::placeholder{
    color: #e5e5e5;
}
button{
    margin-top: 50px;
    width: 100%;
    background-color: #ffffff;
    color: #080710;
    padding: 15px 0;
    font-size: 18px;
    font-weight: 600;
    border-radius: 5px;
    cursor: pointer;
}
.social{
  margin-top: 30px;
  display: flex;
}
.social div{
  background: red;
  width: 150px;
  border-radius: 3px;
  padding: 5px 10px 10px 5px;
  background-color: rgba(255,255,255,0.27);
  color: #eaf0fb;
  text-align: center;
}
.social div:hover{
  background-color: rgba(255,255,255,0.47);
}
.social .fb{
  margin-left: 25px;
```

```
}
.social i{
  margin-right: 4px;
}
    </style>
</head>
{% include 'navbar.html' %}
<body>
    <div class="background">
        <div class="shape"></div>
        <div class="shape"></div>
    </div>
    <form action="{{url_for('addrec')}}" method="POST">
        <h3>Register Here</h3>
        <label for="uname">Name</label>
        <input type="text" placeholder="Name" name="name">
        <label for="username">Last Name</label>
        <input type="text" placeholder="Last name" name="lname">
        <label for="Email">Email</label>
        <input type="text" placeholder="Enter your email" name="email">
        <label for="phnum">Phone Number</label>
        <input type="number" placeholder="Enter your Phone number"
name="phnum">
```

```
        <label for="age">Age</label>

        <input type="number" placeholder="Enter your age" name="age">

        <label for="blood group">Blood broup</label>

        <input type="text" placeholder="Blood group" name="bloodgrp">

        <button style="height: 5%; width: 20%; align-items:
center;">Register</button>

    </form>

</body>

</html>
```

APP.PY

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

import ibm_db

app = Flask(__name__)

conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=21fecfd8-47b7-4937-
840d-
d791d0218660.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=31864;S
ECURITY=SSL;SSLServiceCertificate=DigiCertGlobalRootCA.crt;UID=jhs6873
1;PWD=IA0K22DLUUq1ncnj","","")

@app.route('/')

def home():

  return render_template('index.html')

@app.route('/register')

def register():

  return render_template('register.html')

@app.route('/login')
```

```
def login():
  return render_template('login.html')
@app.route('/donor')
def donor():
  return render_template('donor.html')
@app.route('/needer')
def needer():
  return render_template('needer.html')
@app.route('/addrec',methods = ['POST', 'GET'])
def addrec():
  if request.method == 'POST':
    name = request.form['name']
    lname = request.form['lname']
    email = request.form['email']
    phnum = request.form['phnum']
    age=request.form['age']
    bloodgrp = request.form['bloodgrp']
    sql = "SELECT * FROM user WHERE name =?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,name)
    ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)
    if account:
      return render_template('index.html', msg="You are already a member, please
login using your details")
    else:
      insert_sql = "INSERT INTO user VALUES (?,?,?,?,?,?)"
      prep_stmt = ibm_db.prepare(conn, insert_sql)
      ibm_db.bind_param(prep_stmt, 1, name)
      ibm_db.bind_param(prep_stmt, 2, lname)
      ibm_db.bind_param(prep_stmt, 3, email)
      ibm_db.bind_param(prep_stmt, 4, phnum)
      ibm_db.bind_param(prep_stmt, 5, age)
      ibm_db.bind_param(prep_stmt, 6, bloodgrp)
      ibm_db.execute(prep_stmt)
    return render_template('index.html', msg="Student Data saved successfuly.")
@app.route('/loginpage',methods=['POST'])
def loginpage():
    user = request.form['user']
    passw = request.form['passw']
    sql = "SELECT * FROM user WHERE name =? AND email=?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,user)
    ibm_db.bind_param(stmt,2,passw)
    ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)
    if account:
            return render_template('index.html')
    else:
        return render_template('login.html', pred="Login unsuccessful. Incorrect
username / password !")
@app.route('/donorpage',methods = ['POST', 'GET'])
def donorpage():
  if request.method == 'POST':
    name = request.form['name']
    lname = request.form['lname']
    email = request.form['email']
    phnum = request.form['phnum']
    bloodgrp = request.form['bloodgrp']
    location=request.form['location']
    donated=request.form['donated']
    sql = "SELECT * FROM donor WHERE name =?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,name)
    ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)
    if account:
      return render_template('index.html', msg="You are already a member, please
login using your details")
    else:
      insert_sql = "INSERT INTO donor VALUES (?,?,?,?,?,?,?)"
      prep_stmt = ibm_db.prepare(conn, insert_sql)
      ibm_db.bind_param(prep_stmt, 1, name)
      ibm_db.bind_param(prep_stmt, 2, lname)
      ibm_db.bind_param(prep_stmt, 3, email)
      ibm_db.bind_param(prep_stmt, 4, phnum)
      ibm_db.bind_param(prep_stmt, 5, bloodgrp)
      ibm_db.bind_param(prep_stmt,6, location)
      ibm_db.bind_param(prep_stmt, 7, donated)
      ibm_db.execute(prep_stmt)
    return render_template('index.html', msg="Student Data saved successfuly.")
@app.route('/neederpage',methods = ['POST', 'GET'])
def neederpage():
  if request.method == 'POST':
    name = request.form['name']
    email = request.form['email']
    phnum = request.form['phnum']
    bloodgrp = request.form['bloodgrp']
    location=request.form['location']
    insert_sql = "INSERT INTO needer VALUES (?,?,?,?,?)"
    prep_stmt = ibm_db.prepare(conn, insert_sql)
    ibm_db.bind_param(prep_stmt, 1, name)
    ibm_db.bind_param(prep_stmt, 2, email)
    ibm_db.bind_param(prep_stmt, 3, phnum)
    ibm_db.bind_param(prep_stmt, 4, bloodgrp)
    ibm_db.bind_param(prep_stmt,5, location)
    ibm_db.execute(prep_stmt)
    return render_template('index.html', msg="Student Data saved successfuly.")
```

13.2 GITHUB

https://github.com/IBM-EPBL/IBM-Project-6818-1658838912

PROJECT DEMO LINK

https://youtu.be/aqAXV-kZQr4