

KONGUNADU COLLEGE OF ENGINEERING AND TECHNOLOGY

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION
ENGINEERING**

**HX 8001-PROFESSIONAL READINESS FOR INNOVATION,
EMPLOYABILITY AND ENTREPRENEURSHIP**

PLASMA DONOR APPLICATION

NALAIYA THIRAN PROJECT REPORT 2022

Submitted by

AKASH B	621319106004
BARATHKUMAR P	621319106008
DINESHKUMAR G	621319106017
HARIHARAN V M	621319106025

Team ID: PNT2022TMID13434

NOVEMBER 2022

TABLE OF CONTENT

CHAPTER NO	TITLE	PAGE NO
1	INTRODUCTION	4
	1.1 PROJECT OVERVIEW	5
	1.2 PURPOSE	5
2	LITERATURE SURVEY	6
	2.1 EXISTING SYSTEM	6
	2.2 REFERENCES	8
	2.3 PROBLEM STATEMENT DEFINITION	9
3	IDEATION & PROPOSED SOLUTION	10
	3.1 EMPATHY MAP CANVAS	10
	3.2 IDEATION & BRAINSTORMING	11
	3.3 PROPOSED SOLUTION	13
	3.4 PROBLEM SOLUTION FIT	14
4	REQUIREMENT ANALYSIS	15
	4.1 FUNCTIONAL REQUIREMENT	15
	4.2 NON – FUNCTIONAL REQUIREMENT	15
5	PROJECT DESIGN	16
	5.1 DATA FLOW DIAGRAMS	17
	5.2 ARCHITECTURES	17
	5.2.1 SOLUTION ARCHITECTURE	17
	5.2.2 TECHNICAL ARCHITECTURE	18
	5.3 USER STORIES	19

6	PROJECT PLANNING AND SCHEDULING	20
	6.1 SPRINT PLANNING & ESTIMATION	20
	6.2 SPRINT DELIVERY SCHEDULE	22
	6.3 REPORTS FROM JIRA	23
7	CODING & SOLUTION	
	7.1 FEATURE 1	24
	7.2 FEATURE 2	27
	7.3 DATABASE SCHEMA	30
8	TESTING	
	8.1 TEST CASES	33
	8.2 USER ACCEPTANCE TESTING	35
9	RESULTS	
	9.1 PERFORMANCE METRICS	36
10	ADVANTAGES & DISADVANTAGES	
	10.1 ADVANTAGES	40
	10.2 DISADVANTAGES	40
11	CONCLUSION	41
12	FUTURE SCOPE	42
13	APPENDIX	43
	13.1 SOURCE CODE	43
	13.2 GITHUB/PROJECT DEMO LINK	58

CHAPTER 1

INTRODUCTION

Blood is the most important component in the human body. Without it, a human can't live. Not only blood is important, also its various components are precious resources for the human circulatory system. With years passed on covid days on surviving and protecting ourselves proves that plasma's requirement is more important than we thought.

Plasma is a blood component that appears to be yellowish colored. Major portion of the blood consists of plasma components. It helps with immunity, controlling blood pressure & maintaining pH balance in our human body. It also plays a major role in transporting blood cells and other components such as nutrients, proteins, waste products throughout the body. How does it basically help in covid treatment? Major principle of this process is transfusing plasma retrieved from the blood of a person who has successfully recovered from covid-19 into a person who is suffering from the disease. As a person fights the covid-19 virus, they produce antibodies that attack the virus. These antibodies are secreted by immune cells known as B lymphocytes found in plasma or liquid part of the blood which helps the blood clot when needed and supports immunity.

Plasma is the liquid portion of blood that remains after red blood cells and major cells are removed. Those retrieved plasma transferred to the victim person as a donation. This neutralizes the virus entry into the cells, avoiding disease and also improving patients from covid-19. So, from the above process of treatment, it finalizes the point that a donation needs to be made. For this donation process, several sources are available to find a right donor such as from a medical organization, non-government organizations, blood bank donation centres etc.. The proposed model was meant to be deployed to ease the process of the donation of plasma which has been a major drawback during the covid

1.1 PROJECT OVERVIEW

The main aim of this proposed model is to deliver a communication bridge between the plasma requested person and plasma donor via web application. By this web application, a user is able to request for a needed plasma. Also on the other hand, plasma giving donors are able to apply for a donation on this portal. By collecting data from both plasma requested and plasma donor end's, this model achieves its aim of laying a bridge between them. This web application uses cloud storage services in order to store the details of both users. This model workflow is that initially the user interacts with a web application. Then register by giving details as a donor. Then the database will have all the details. If a user posts a request, then it will check that the requested blood group has a donor in the database and if it is available, then concerned blood group donors will get notified about it. Notification will be sent out via mail for that donor. By that the donor can reach out to the requested user with details and help them give the donation

1.2 PURPOSE

Smart apps are now considered as an important communication tool and could be best utilized in plasma donation if they are designed to fit the user's needs and preferences. The main purpose of this model is to give the details of the plasma requested person's details to the plasma donor. So, by using this web application could save a lot of time consumption on comparing with other mediums. Everyone in this digital era has smartphones and laptops to carry with them. By using those with proper implementation, could be a perfect medium for this donation field.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING SYSTEMS

BLOOD DONATION SMARTPHONE APP

The top-rated features were the ability to request for blood donors and the ability to locate the nearest blood centre on the map. The preferred method of contact was found to be SMS.

INSTANT PLASMA DONOR RECIPIENT CONNECTOR WEB APPLICATION

In this proposed system, a donor who wants to donate plasma and can donate the plasma to a blood bank. The blood bank after checking the donor certificate can make a request to the donor.

BLOODR: BLOOD DONOR AND REQUESTER MOBILE APPLICATION

BLOODR application creates communication channel through authenticated clinics whenever a patient needs blood donation. It is a useful tool to find compatible blood donors who can receive blood request posts in their local area.

BLOOD DONATION AND LIFE SAVERBLOOD DONATION APP

At the emergency time of blood needed we can check for blood donor nearby by using GPS. Once the app user enters the blood group which he/she needed it will automatically show the donor nearby and send an alert message to the donor.

BLOOD DONATION MANAGEMENT SYSTEM

This system is very User-friendly and interactive between the donor and the recipient. Our system allows the donor to know the emergency by sending a web notification to the recipient.

FOOD DONATION APP FOR SMART LIVING

This app fits the general realm of AI for Smart Living in Smart Cities. In addition to entailing IoT and ubiquitous computing, this work makes positive impacts on both healthcare and environment by reducing hunger and food waste respectively.

APPLICATION FOR BOOK DONATION

Mobile application to connect people who are interested in donating their books to those who might be in need. It will connect the needy and the donors and enable them to get the books that they require from people that are done using them.

MOBILE APPLICATION FOR DONATION OF ITEMS

The proposed application shall reduce food wastage and also fulfil other requirements like clothes, etc. of needy organizations. The mobile donation apps collected information and analysed data in a specific way to determine the application's goals and potential issues after reviewing several online donation sites. The project therefore conducted quantitative research on data collection and analysis to achieve project goals. Donate Day mobile application offered a variety of features such as modern interface, databases and event page.

MOBILE PHONE BASED SYSTEM TO PROMOTE EYE DONATION

Eye Donor Form generates form using any smartphone. The donation form input is saved electronically and can be forwarded to your family members, healthcare.

2.2 REFERENCES

1. Kavita shirsat, Iqra shaikh, Pradnya Deshmukh, Mayuri lambhate⁴, Food donation application: food share, International research journal of engineering and technology (IRJET), volume: 08 issue: 05 | May 2021
2. Devanjan K. Srivastava, Utkarsh Tanwar & 2021, Blood Donation Management System, International Journal of Creative Research Thoughts, International research journal of engineering and technology, 2021
3. Shubham Singh Rana, Satvik Maheswari, Shiv kumar, Ms. Jyoti thakur, Charity-based application, Journal of xi'an university of architecture & technology, Volume XII, issue IV, 2020
4. A.Meiappane, K.Logavignesh, R.Prasanna, T.Sakthivel, Blood Donation App Using Android, IEEE International Conference on System, Computation, Automation and Networking (ICSCAN), 29-30 March 2019
5. Altahir Saad, Ahmed Saad, Lars Rune Christensen, Blood Donation & Blood Banks through Android Mobile App and Web Application System , International Journal of Computer Science Trends and Technology (IJCTST) ,Mar - Apr 2019 S Periyannayagi, A Manikandan, M Muthukrishnan ,BDoor AppBlood Donation Application, Journal of Physics: Conference Series, 2018. 13 References
6. Nikita M. Lunawat, Chetan D. Kshirsagar , Ashish A. Gawhande , Rohini M. Rathod, Blood and organ for patient using android application, International Journal of Research in Engineering and Technology, May-2016.
7. Sofhioubhi, Jose Luis Fernandez-Aleman, Ambrosio toval, ali idri, Free blood donation mobile applications, Journal of medical systems, May 2015.

2.3 PROBLEM STATEMENT DEFINITION

The statement of the problem is one of the first things that a colleague or potential client will read. It explains quickly to the reader, the problem at hand, the need for research, and how you intend to do it.

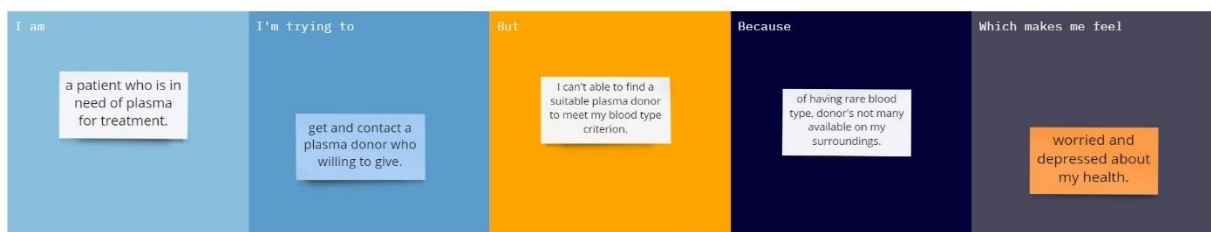
A problem statement is an explanation in research that describes the issue that is in need of study. What problem is the research attempting to address? Having a Problem Statement allows the reader to quickly understand the purpose and intent of the research. The purpose of the problem statement is to identify the issue that is a concern and focus it in a way that allows it to be studied in a systematic way. It defines the problem and proposes a way to research a solution, or demonstrates why further information is needed in order for a solution to become possible.

Problem Statement 1



mina

Problem statement 2

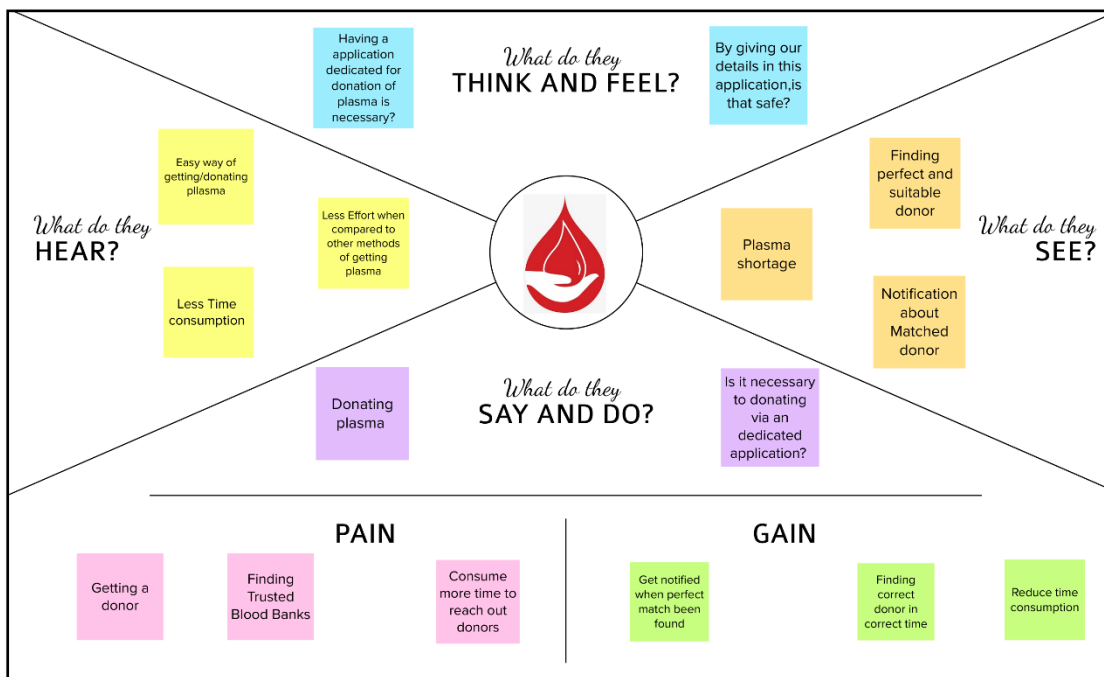


CHAPTER 3

IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

It is a visual scheme with questions about the ideal target customer of a specific product or service of our company: What does he think and feel, what does he see, what does he hear, what does he say and do, what efforts does he make and what results do he expect? It serves to better understand this prototype of person and therefore to be able to better satisfy her needs and communicate by empathizing with her as much as possible. This canvas helps to reflect on how the inputs from the environment affect her and the internal emotions she experiences. This approach is intended to be deeper than other more superficial canvases, so it is important to take it as one of the starting points to identify what this user persona is like. The **user persona** (also called the buyer persona) is a concept that is closely linked to the empathy map and that helps to put this person in context:



3.2 IDEATION & BRAINSTORMING

Brainstorming is a group problem-solving method that involves the spontaneous contribution of creative ideas and solutions. This technique requires intensive, freewheeling discussion in which every member of the group is encouraged to think aloud and suggest as many ideas as possible based on their diverse knowledge. Brainstorming combines an informal approach to problem-solving with lateral thinking, which is a method for developing new concepts to solve problems by looking at them in innovative ways

<div>1</div> <div>PROBLEM STATEMENT</div> <div>PROBLEM</div> <div>How might we handle to create an application fully dedicated for Plasma Donation?</div>	<div>2</div> <div>BRAINSTROM</div> <div>AKASH B</div> <div>Using hospital/clinic organization to improve the service.</div> <div>Chatbot service can be included to clear doubts about plasma donation.</div> <div>If the blood group is suitable with the requested blood type ,then only the notification are send.</div> <div>Organizing various activities to promote the application's interest among the people.</div> <div>BARATHKUMAR P</div> <div>Features like history of the donations made,finding donor's location using GPS</div> <div>Only donors from the age of 18 having a weight of 50kg can register in the application.</div> <div>Rapid contacting features can be added when there is a immediate need of plasma.</div> <div>By using location detecting features,one can able to find accurate location of the donor.</div> <div>DINESH KUMAR G</div> <div>Details such as gender, D.O.B, age,contact details are collected and stored in database.</div> <div>Plasma of the body are to be examined by a medical expert before the donation.</div> <div>Verifications are to be made at registration stage inorder to make the donation.</div> <div>To ensure and verify whether the donor is free from any other cautionary diseases.</div> <div>HARIHARAN V M</div> <div>Blood group description from both the parties(Donor & collector) is collected.</div> <div>If there is any misinformation and proved that he/she had any other diseases,immediate action of removing his/her profile are to made.</div> <div>The exact date of the plasma extraction must be mentioned on his profile.</div> <div>The user during the initial stage of registration should given whether it is his/her first time at donating or already donated person.</div>
---	---

3

GROUP IDEAS



4

PRIORITIZE



3.3 PROPOSED SOLUTION

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, would be a helping hand.
2.	Idea / Solution description	In regard to the problem, an application is to be built which would take the donor details, store them and inform them upon a request.
3.	Novelty / Uniqueness	The fresh & responsive interface along with the quick response on giving notification to recipient & donor has been the novelty of the model. A chatbot, which is fully dedicated to that application environment will interact with the users.
4.	Social Impact / Customer Satisfaction	In society, it will create an awareness among the people about donation of plasma which will be done in an easy way of connecting the donor and the recipient. For sure, this application will satisfy the customers with its services.
5.	Business Model (Revenue Model)	The proposed model can be deployed in collaboration with the NGO's and in turn can yield revenue for each and every request from the user.
6.	Scalability of the Solution	The model can be deployed in a large scale based on the requirement and usage by the users.

3.4 PROBLEM SOLUTION FIT

Define CS, fit	1.CUSTOMER SEGMENT CS <ul style="list-style-type: none"> -The recipient who are in need of plasma. -The NGO's & hospital managements. 	6.CUSTOMER CONSTRAINTS CC <ul style="list-style-type: none"> -There is no connection details between the customers. -Unavailability of plasma at the needed time. 	5.AVAILABLE SOLUTIONS AS <ul style="list-style-type: none"> -Seeking help through social media. - Existing system involves, only the collection of donor data and will not notify the about the recipient. 	Explore differentiate AS,
Focus on J&P, tap into BE, understand RC	2.JOBS TO BE DONE/PROBLEMS J&P <ul style="list-style-type: none"> - Establish a connection between the donor and the recipient. - Notify donors at the correct time. - Demand has increased. 	9.PROBLEM ROOT CAUSE RC <ul style="list-style-type: none"> - 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. 	7.BEHAVIOUR BE <ul style="list-style-type: none"> - The recipient will get the plasma at the right time. - The donors whose details, stored in database during registration will be notified. 	Focus on J&P, tap into BE, understand RC
Identify strong TR & EM	3.TRIGGERS TR <ul style="list-style-type: none"> - We can advertise the web app through the NGO's and through the pharmaceutical companies. <hr/> 4.EMOTIONS: BEFORE/AFTER EM <ul style="list-style-type: none"> - Before : Anxiety, Stress, Scared - After : Relaxed, Happy 	10.YOUR SOLUTION SL <ul style="list-style-type: none"> - Finding the respective donor and notify them through email for the requests. 	8.CHANNELS OF BEHAVIOUR CH <ul style="list-style-type: none"> - The donor will register and they will be notified through the mail. - It will acts as a communication channel. 	Identify strong TR & EM

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through website
FR-2	User Confirmation	Confirmation via Email
FR-3	User Login	Login through registered email id
FR-4	Send Request	If plasma required then donor get thenotification
FR-5	Contact Donor	Contact donor directly if emergency

4.1 NON - FUNCTIONAL REQUIREMENT

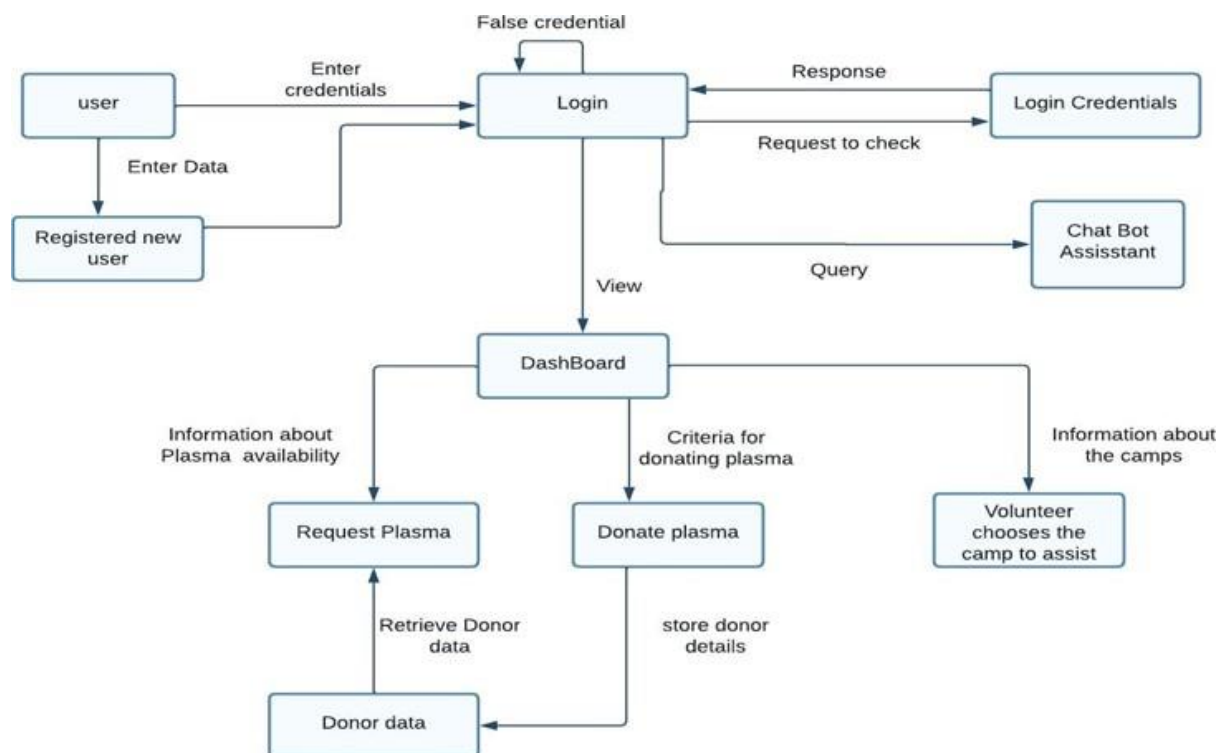
NFR No.	Non-Functional Requirement	Description
NFR-1	Usability	The plasma Donor application is user friendlyand easy to access
NFR-2	Security	The users/donor details are stored in thecloud and it is secured with the user email id andpassword
NFR-3	Reliability	The system have the ability to work all thetimes without failure apart from network failure. The contact list of the donor are provided
NFR-4	Performance	The plasma donor application works well inevery emergency situation. The easy interactive with the user and less interrupts
NFR-5	Availability	The plasma Application is an online webapplication and it monitor 24/7
NFR-6	Scalability	The application offers multiple users and it is designed to protect the user's information.

CHAPTER 5

PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS

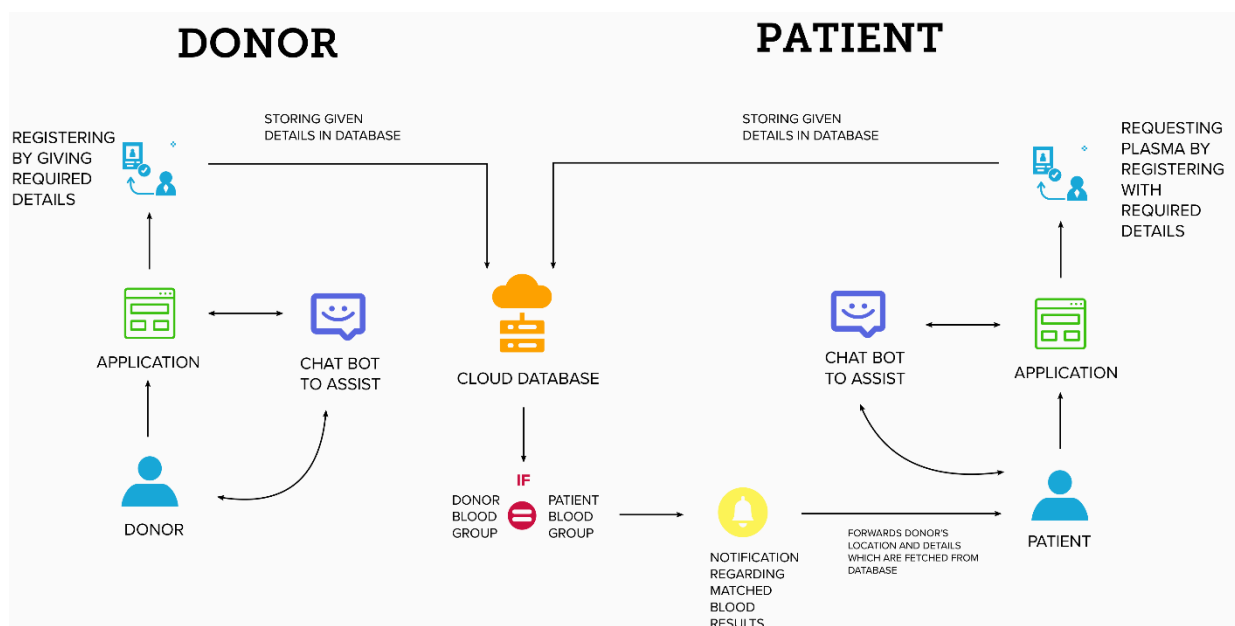
A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled.



5.2 ARCHITECTURES

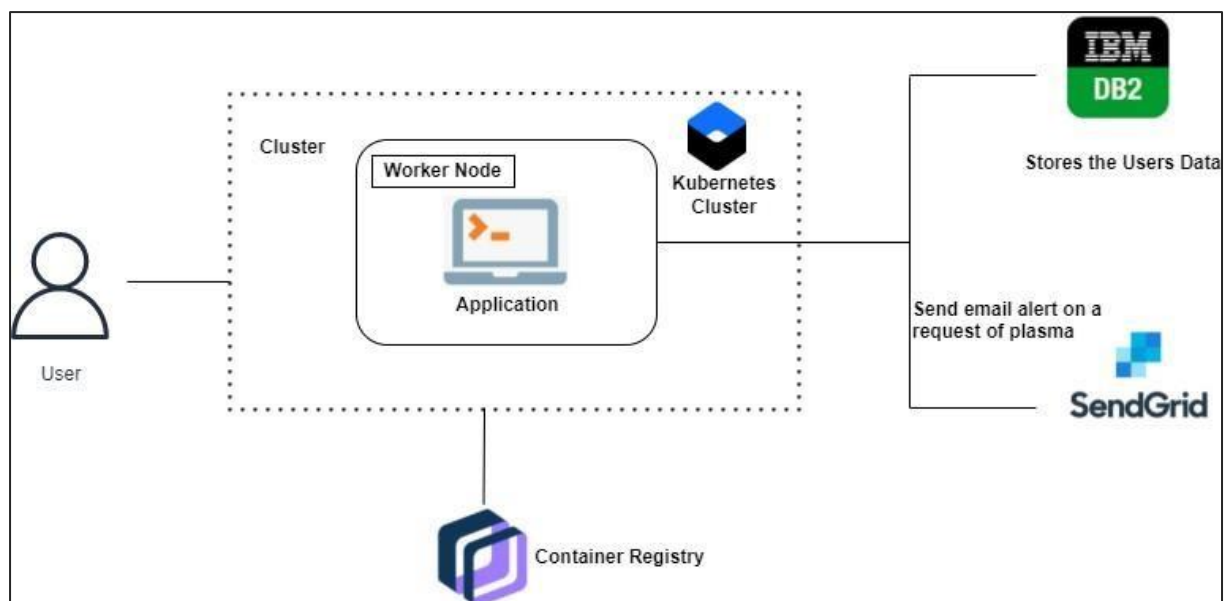
5.2.1 SOLUTION ARCHITECTURE

Solution architecture is the process of developing solutions based on predefined processes, guidelines and best practices with the objective that the developed solution fits within the enterprise architecture in terms of information architecture, system portfolios, integration requirements and many more. It can then be viewed as a combination of roles, processes and documentation that are intended to address specific business needs, requirements or problems through the design and development of applications and information systems.



5.2.2 TECHNICAL ARCHITECTURE

Technical Architecture (TA) is a form of IT architecture that is used to design computer systems. It involves the development of a technical blueprint with regard to the arrangement, interaction, and interdependence of all elements so that system-relevant requirements are met. Throughout the past decade, architecture has become a broadly used term in the context of information technology. This doesn't come as a surprise considering how most companies had to redesign their IT landscape to adopt digital trends like cloud computing and software as service (SaaS).



5.3 USER STORIES

Functional Requirement (Epic)	User Story Number	User Story / Task	Priority	Release
Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	High	Sprint-1
	USN-2	As a user, I will receive confirmation email once I have registered for the application	High	Sprint-1
	USN-3	As a user, I can register for the application through Gmail	Medium	Sprint-1
Login	USN-4	As a user, I can log into the application by entering email & password	High	Sprint-1
Dashboard	USN-5	As a user, I can send the proper requests to donate and obtain plasma.	High	Sprint-1
Login	USN-6	As a user, I can register and log into the application by entering email & password to view the profile	High	Sprint-1
Dashboard	USN-7	As a user, I can send the proper requests to donate and obtain plasma.	High	Sprint-1
Application	USN-8	As a customer care executive, I can try to address user's concerns and questions	Medium	Sprint-2
Application	USN-9	As an administrator I can help with user-facing aspects of a website, like its appearance, navigation and use of media.	Medium	Sprint-3
	USN-10	As an administrator, I can involve working with the technical side of websites.	Medium	Sprint-1
Functional Requirement (Epic)	User Story Number	User Story / Task	Priority	Release
Dashboard	USN-11	In addition the Customer care executive, chatbot can try to address user's concerns and questions	Medium	Sprint-3

CHAPTER 6

PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint - 1	Registration	USN -1	As a user, I can register for the application by entering my email, password, and confirming my password.	8	High
		USN -2	As a user, I will receive confirmation email once I have registered for the application	4	Low
	Login	USN -3	As a user, I can log into the application by entering email & password	8	High
Sprint - 2	Register for plasma donation	USN -4	As a donor, I have to register to intimate the users that I am interested in donating the plasma.	10	High
	Request for Plasma	USN -5	As a recipient, I have to request the plasma from the donors.	10	High

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint - 3	Database	USN-7	The user data has to be saved and needs to be maintained in the database.	13	High
	Chatbot	USN-8	As a user, I can get clarify my doubts with the help of the Watson assistant.	7	Moderate
Sprint - 4	Send Notification	USN-9	As a donor, myself have to be notified for donation of the plasma when it was in need.	12	High
	Deployment	USN-10	As a user, I want to access the application in a public IP domain.	8	High

6.2 SPRINT DELIVERY SCHEDULE

Sprint	Total Story Point	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

SPRINT VELOCITY

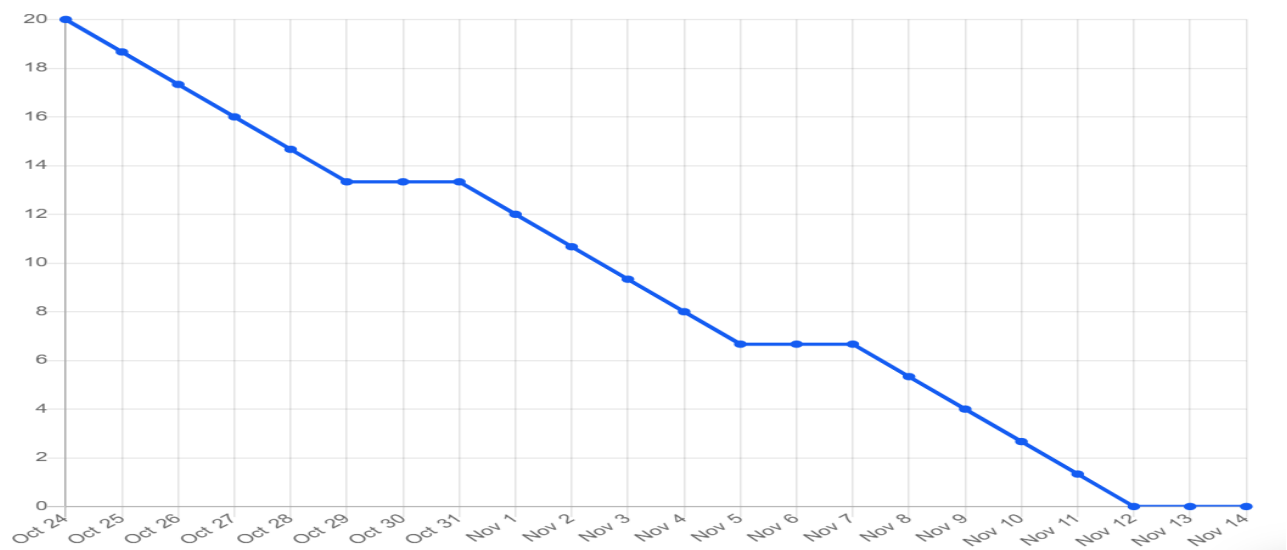
Sprint 1 = $20/6 = 3.66$

Sprint 2 = $20/6 = 3.66$

Sprint 3 = $20/6 = 3.66$

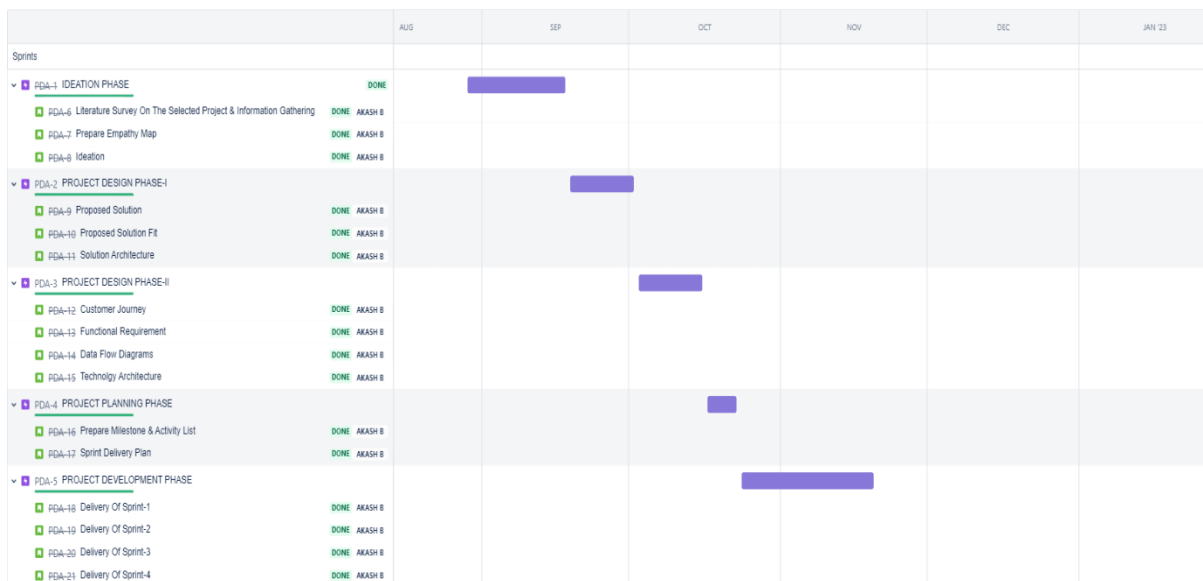
Sprint 4 = $20/6 = 3.66$

BURNDOWN CHART



6.3 REPORTS FROM JIRA

Jira Software is part of a family of products designed to help teams of all types manage work. For teams who practice agile methodologies, Jira Software provides scrum and Kanban boards out-of-the-boxboards task management hubs, where tasks are mapped to customizable workflows. Boards provide transparency across teamwork and visibility into the status of every work item. Time tracking capabilities and real-time performance reports (burn-up/down charts, sprint reports, velocity charts) enable teams to closely monitor their productivity over time.



CHAPTER 7

CODING & SOLUTIONING

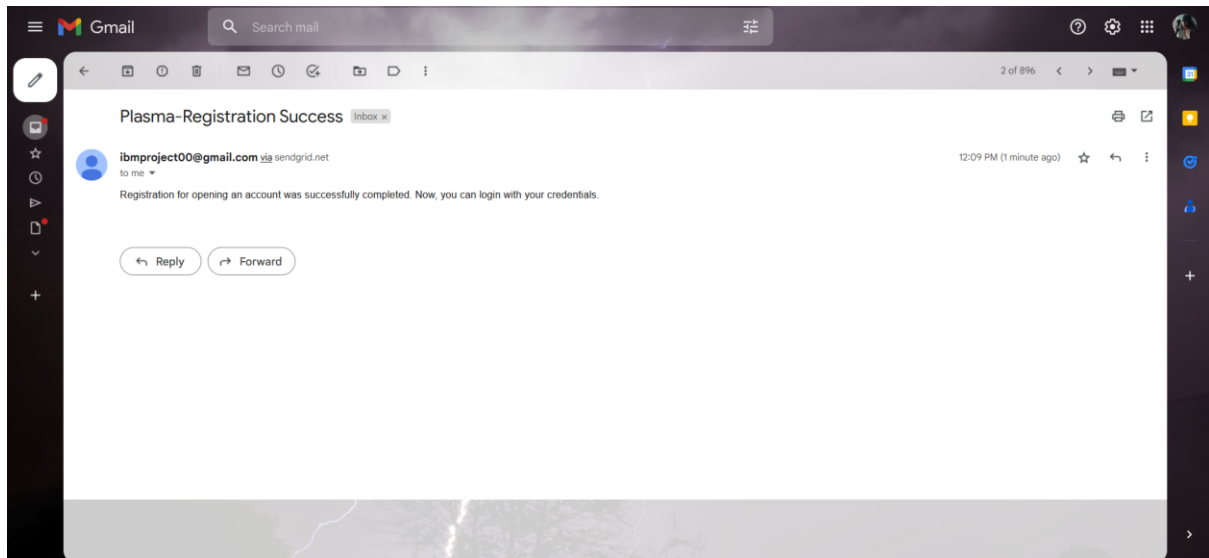
7.1 FEATURE 1 – MAIL NOTIFICATION

One of the features that added to this model is sending Notification via Mail. Notification plays a major role on every communication medium process. On implementing this feature on application which is fully dedicated for providing a communication bridge between a plasma donor and plasma requester is a meaningful way. In this process, Mail system been used on several stages. Those stages such as after registration, after successfully applied for a plasma donation, after successfully request for a plasma on that portal. We using SendGrid Mail service for integration and other purposes. SendGrid API key is use for this system in order to connect the SendGrid account service and our web application portal.

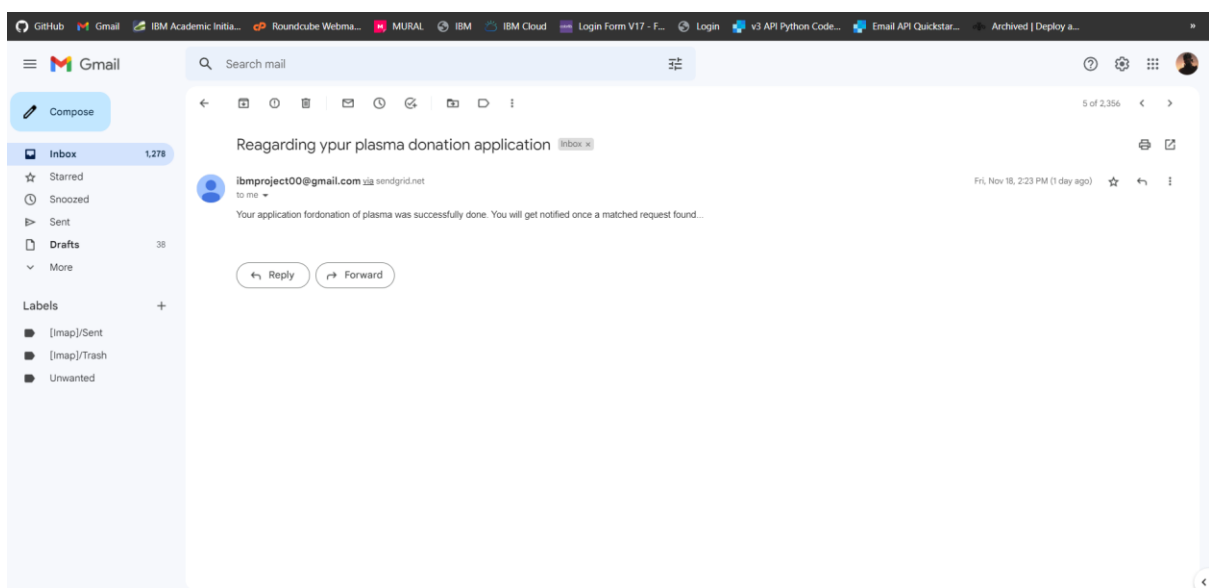
Our web application majorly based on python, we connecting the SendGrid services via python. The mail been sent in a short time which reflects the responsive characteristics of this mailing system. Without integrating notification system, the donor and plasma requester can't able to communicating between them.

Notification been sent via many mediums. But here we choose the mail system because of the transmission speed between the users. Some of the stages are

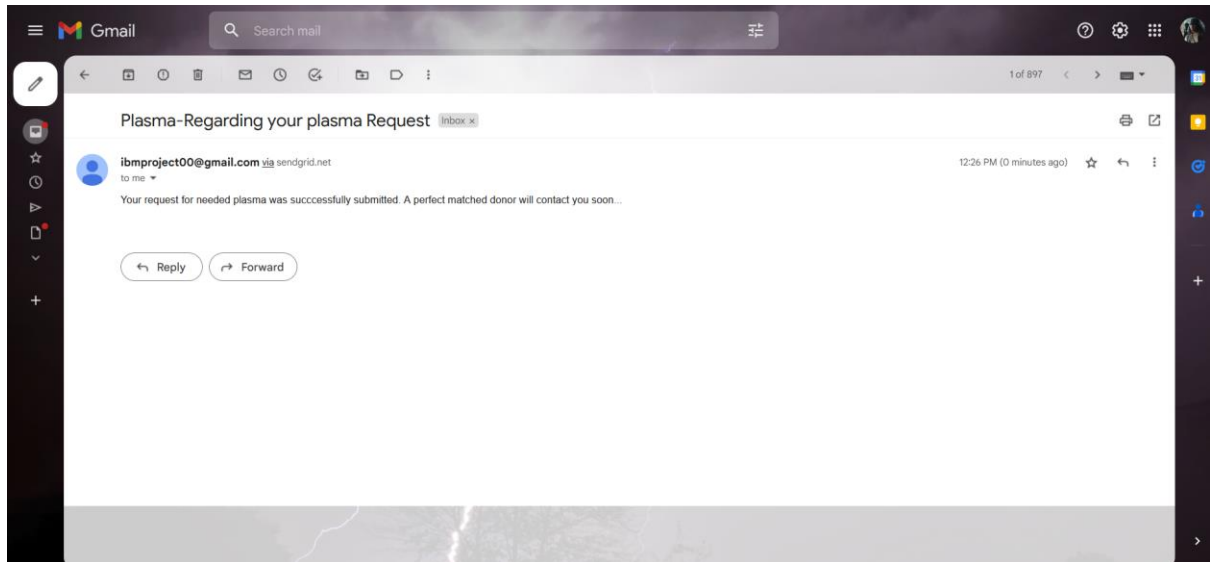
AFTER REGISTRATION, THE USER GET NOTIFICATION ABOUT THE SUCCESSFUL REGISTRATION ON THIS PORTAL.



AFTER APPLIED FOR A PLASMA DONATION ON PORTAL



AFTER SUCCESSFULLY REQUESTED FOR A PLASMA ON PORTAL

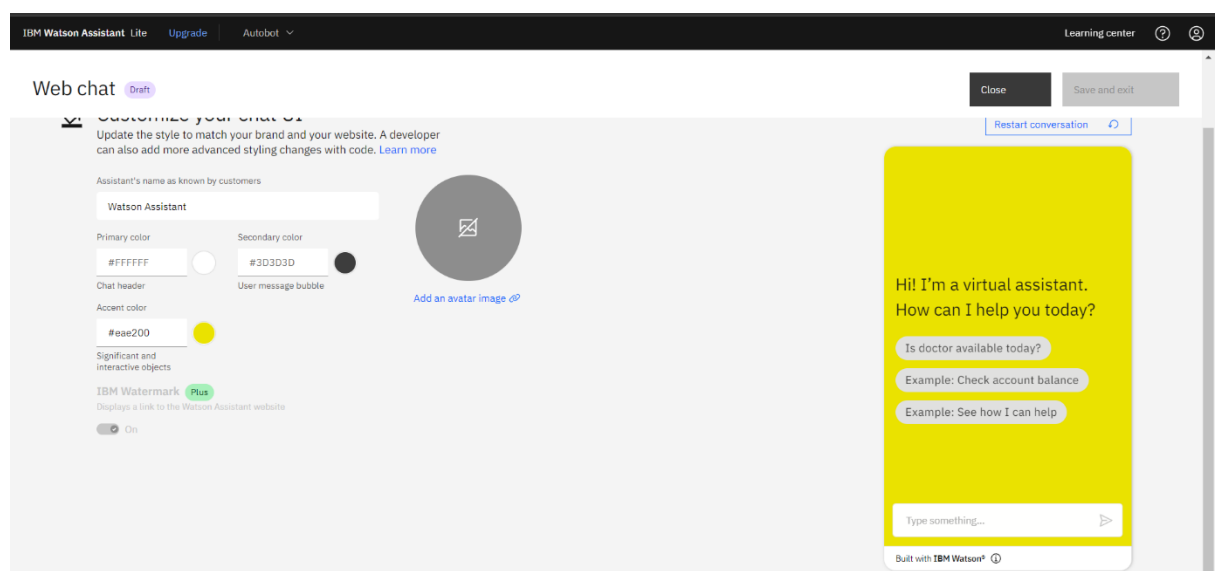


7.2 FEATURE 2

Feature 2 that implementing on this web application is Chatbot facility. Here, in this web application portal we use IBM Watson chat bot. Basically, IBM Watson is an assistant that been used on various places on technology. By implementing this feature on this portal, helps the user in more ways on various stages such as navigation, form fill up and plasma donation count and also nearby donation camps. So, this will help in making the web application a responsive and interactive to the user.

WATSON ASSISTANT CREATION AND DEPLOYMENT ON WEB PORTAL

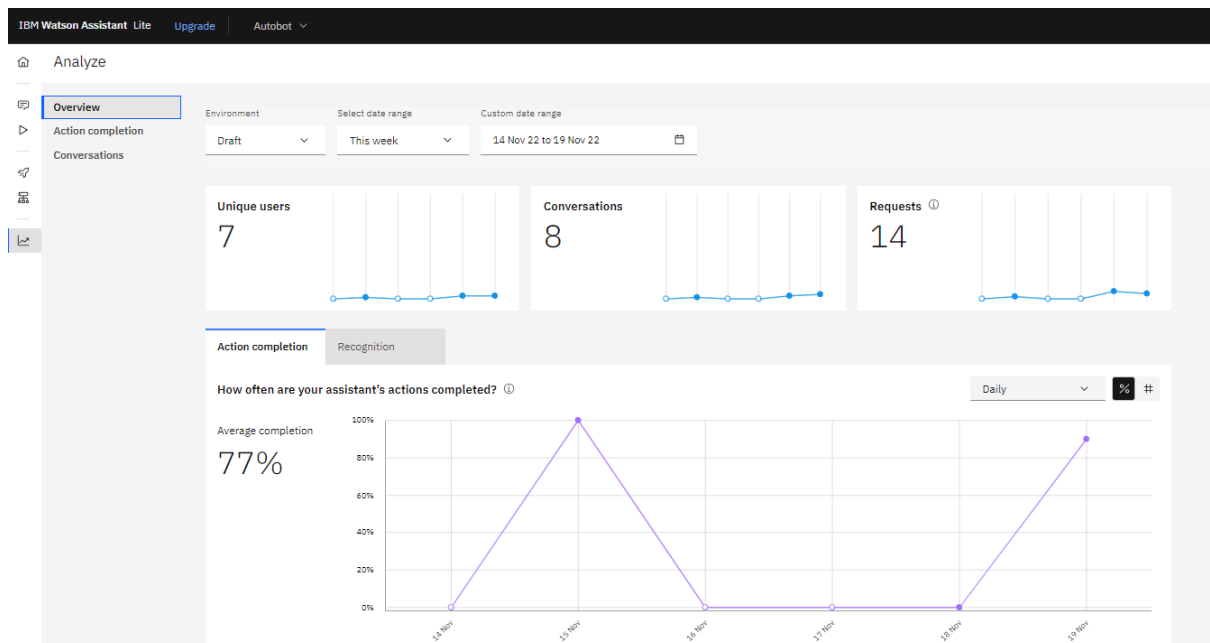
Here we named this Watson assistant as ‘AUTOBOT’.



Some of the actions that created are

Name	Last edited	Examples Count	Status
Is doctor available today?	6 minutes ago	1	Success
Request contact information	5 minutes ago	16	Success
Create an account	5 minutes ago	20	Success

Overview of Actions



Watson Assistant on our web application portal



7.3 DATABASE SCHEMA

A database schema is considered the “blueprint” of a database which describes how the data may relate to other tables or other data models. A database schema defines how data is organized within a relational database; this is inclusive of logical constraints such as, table names, fields, data types, and the relationships between these entities. Schemas commonly use visual representations to communicate the architecture of the database, becoming the foundation for an organization’s data management discipline.

Here we use IBM DB-2 version 11.5.8.0.00000.015 and Schema name DCL74999 Db2 on cloud is used to access data we stored and view those data .and create database objects. The administrator of data base can also monitor performance and usage of that particular database. Here we used this service to store the details of the plasma requester and the plasma donor’s information. Also, the data that are been got from registration process also stored in this database for login process.

For this model, we created three database tables namely USERS, APPLIEDUSERS & REQUUSERS. In User table, the details that are collected from registration process get stored. In Applied users table, the details of the user who wish to donate plasma get stored in this table. In Requsers table, the details of the user who seek perfect plasma match get stored in this table.

By using IBM cloud cli and we can perform actions on table such as connecting the database, calling the function to execute, returning the prepared statements and so on. By using python, the cloud and our application are integrated. For integrating an IBM database and application the following parameters are required. Here `ibm_db.connect` is used. Database name, port number, SSL certificate, protocol type, UID and password of that particular service connection. The created tables are as follows as

OVERVIEW:

IBM Db2 on Cloud

OverviewIn-flight executionsConnectionsTable performance

SQL

Filter by: Tables

Search schema or table name

Pause data refresh

Refresh

Last collected: --

Schema name	Table name	Table type	Table space name	Table scans /min	Rows read /min	Hit ratio	Access /min
DCL74999	USER	USER_TABLE	DCL74999SPACE1	0	0	0.00%	0
DCL74999	REQUSERS	USER_TABLE	DCL74999SPACE1	0	0	0.00%	0
DCL74999	APPLIEDUSERS	USER_TABLE	DCL74999SPACE1	0	0	0.00%	0

USER TABLE:

```
CREATE TABLE USER(EMAIL VARCHAR(30),USERNAME CHAR(20),PASSWORD VARCHAR(100));
```

IBM Db2 on Cloud

Load DataLoad HistoryTablesViewsIndexesAliasesMQTsSequencesApplication objects

DCL74999.USER

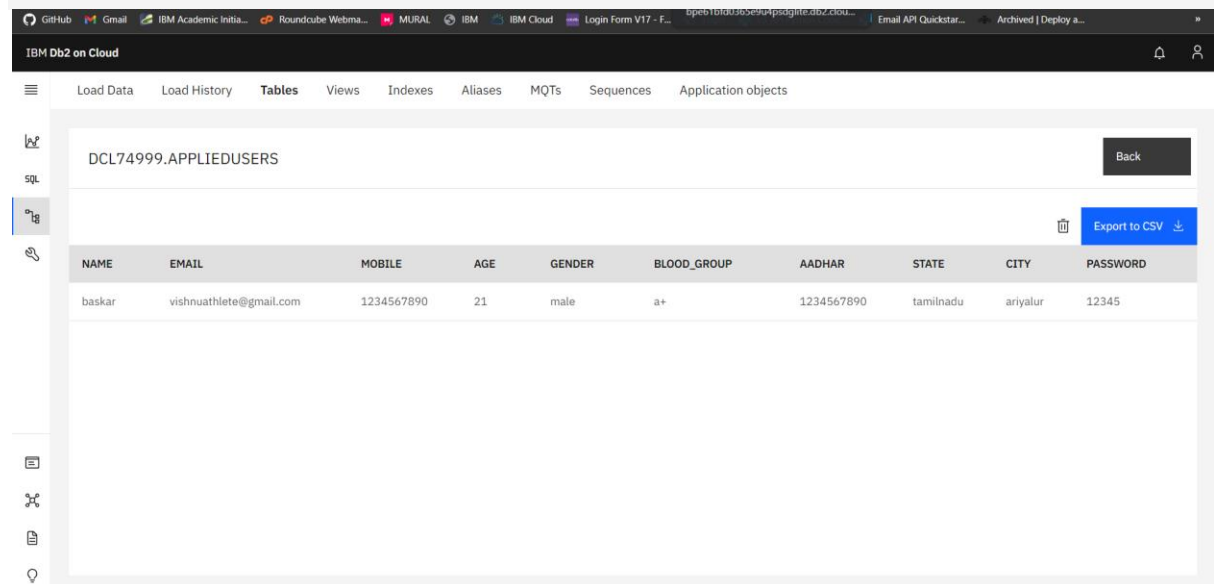
Back

Export to CSV

EMAIL	USERNAME	PASSWORD
akashbalu16@gmail.com	akash	\$2b\$12\$TNuoUEqBHE.Hns1qbNeOFuAX1Lc9j/oRr5efhsV4aFqfYJInI9PbC
dineshmalar2106@gmail.com	dinesh	\$2b\$12\$PQMohmXulPOIbWZSin95S.w2eS1qux4tquZTdu8/E.pFNiSNSlrm

APPLIED USER:

```
CREATE TABLE APPLIEDUSERS(NAME CHAR(30),EMAIL CHAR(50), MOBILE BIGINT,AGE INTEGER, GENDER CHAR(20), BLOOD_GROUP CHAR(10), AADHAR BIGINT, STATE CHAR(25), CITY CHAR(25), PASSWORD VARCHAR(50));
```



IBM Db2 on Cloud

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

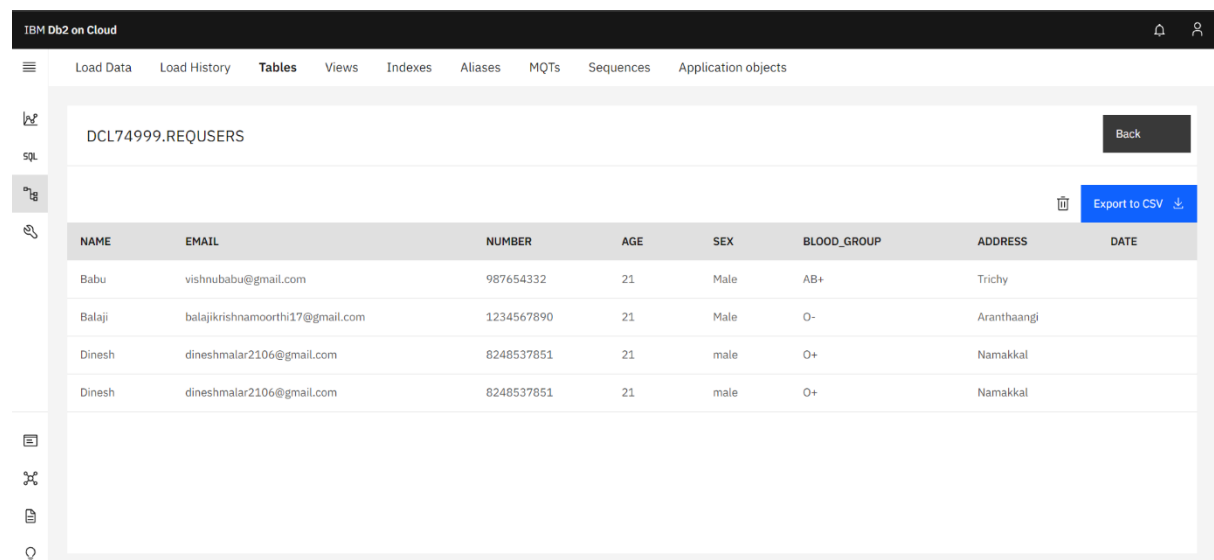
DCL74999.APPLIEDUSERS Back

Export to CSV

NAME	EMAIL	MOBILE	AGE	GENDER	BLOOD_GROUP	AADHAR	STATE	CITY	PASSWORD
baskar	vishnuathlete@gmail.com	1234567890	21	male	a+	1234567890	tamilnadu	ariyalur	12345

REQUESTED USER:

```
CREATE TABLE REQUERS (NAME CHAR(30),EMAIL CHAR(50),NUMBER BIGINT,AGE INT,SEX CHAR(25),BLOOD_GROUP CHAR(10),ADDRESS CHAR(100),DATE DATE);
```



IBM Db2 on Cloud

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

DCL74999.REQUERS Back

Export to CSV

NAME	EMAIL	NUMBER	AGE	SEX	BLOOD_GROUP	ADDRESS	DATE
Babu	vishnubabu@gmail.com	987654332	21	Male	AB+	Trichy	
Balaji	balajikrishnamoorthi17@gmail.com	1234567890	21	Male	O-	Aranthaangi	
Dinesh	dineshmalar2106@gmail.com	8248537851	21	male	O+	Namakkal	
Dinesh	dineshmalar2106@gmail.com	8248537851	21	male	O+	Namakkal	

CHAPTER 8

TESTING

8.1 TEST CASES

				Date	3-Nov-22				
				Team ID	PNT2022TMID13250				
				Project Name	Plasma Application Development				
				Maximum Marks	4 marks				
Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status
Index_TC_001	Functional	Home Page	Verify the links for redirection to the login and registration page .		1.Enter URL and click go 2.Click on login/registration button 3.Verify login/Signup page displayed or not		Login/Signup page should display	Working as expected	Pass
Index_TC_002	UI	Home Page	Verify the UI elements in Index page		1.Enter URL and click go 2.Click on My Account dropdown button 3.Verify login/Signup popup with below UI elements: a.email text box b.password text box c.Login button d.New customer? Create account link e.Last password? Recovery password link		Application should show below UI elements: a.Login button b.Registration button c.Testimonial cards	Working as expected	Pass
LoginPage_TC_001	UI	Login page	Verify the UI elements in Login/Signup popup		1.Enter URL and click go 2.Verify login/Signup popup with below UI elements: a.email text box b.password text box c.Login button		Application should show below UI elements: a.Login button b.Textbox for entering email c.Textbox for entering password	Working as expected	Pass
LoginPage_TC_002	Functional	Login page	Verify user is able to log into application with Valid credentials		1.Enter URL and click go 2.Click on Login button 3.Enter Valid email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username: barath@gmail.com password: barath	User should navigate to user account homepage	Working as expected	pass
LoginPage_TC_003	Functional	Login page	Verify user is able to log into application with Invalid credentials		1.Enter URL and click go 2.Click on Login button 3.Enter Invalid email in Email text box 4.Enter in valid password in password text box 5.Click on login button	Username: barath@gmail password: hello	Application should show 'Incorrect email or password' validation message.	Working as expected	pass
RegPage_TC_001	UI	Registration page	Verify the UI elements in the register page		1.Enter URL and click go 2.Verify login/Signup popup with below UI elements: a.email text box b.name text box c.password text box d.mobile number text box e.register button		Application should show below UI elements: a.Register button b.Textbox for entering email c.Textbox for entering password c.Textbox for entering name e.Textbox for entering mobile number	Working as expected	Pass
RegPage_TC_002	Functional	Registration page	Verify user is able to register into application with Invalid credentials		1.Enter URL and click go 2.Click on Login button 3.Enter Valid email in Email text box 4.Enter valid password in password text box 5.Enter the name 6.Enter the mobile number 7.Click on register button	Username: barath@gmail.com password: barath name: Barath mobile number: 8925195582	Application should register the user successfully.	Working as expected	pass
RegPage_TC_003	Functional	Registration page	Verify whether the user is able to register into application with Invalid credentials		1.Enter URL and click go 2.Click on Register button 3.Enter invalid email in Email text box 4.Enter Name in the Name text box 5.Enter Mobile Number in the Mobile Number text box 6.Enter Password in the Password text box 7.Click on register button	Username: barath@gmail password: barath name: Barath mobile number: 8925195582	Application should show 'Incorrect email or password' validation message.	Working as expected	pass
DonorPage_TC_001	UI	Donor Page	Verify the UI elements in the donor page		1.Enter URL and click go 2.Login to the application 3.Click on Donate in the home page 4.Verify Donor page popup with below UI elements: a.name text box b.age text box c.blood group text box d.email text box e.gender text box f.date text box g.mobile number text box h.location text box i.apply button	0	Application should show below UI elements: a.name text box b.age text box c.blood group text box d.email text box e.gender text box f.date text box g.mobile number text box h.location text box i.apply button	Working as expected	pass

DonorPage_TC_002	Functional	Donor Page	Verify whether the user able to register for donation		1.Enter URL and click go 2.Login to the application 3.Click on Donate in the home page 4.Enter name in name text box 5.Enter age in age text box 6.Enter blood group in blood group text box 7.Enter email in email text box 8.Enter gender in gender text box 9.Enter date in date text box 10.Enter mobile number in mobile number text box 11.Enter location in location text box 12.Click on apply button	Username: barath@gmail.com name: Barath age: 21 blood group: O+ve gender: Male Date: 2022-11-03 mobile number: 8925195582 location: Theni	The user will get Successfully applied and also get a mail from the application.	Working as expected	Pass
DonorPage_TC_003	Functional	Donor Page	Verify whether the user able to register for donation using invalid credentials		1.Enter URL and click go 2.Login to the application 3.Click on Donate in the home page 4.Enter name in name text box 5.Enter age in age text box 6.Enter blood group in blood group text box 7.Enter invalid email in email text box 8.Enter gender in gender text box 9.Enter date in date text box 10.Enter mobile number in mobile number text box 11.Enter location in location text box 12.Click on apply button	Username: barath@gmail name: Barath age: 21 blood group: O+ve gender: Male Date: 2022-11-19 mobile number: 8925195582 location: Theni	Application should show 'Incorrect email or password' validation message.	Working as expected	pass
RequestPage_TC_001	UI	Request page	Verify the UI elements in the request page		1.Enter URL and click go 2.Login to the application 3.Click on Request in the home page 4.Verify Request page popup with below UI elements: a.name text box b.age text box c.blood group text box d.email text box e.gender text box f.date text box g.mobile number text box h.location text box i.password text box j.apply button		Application should show below UI elements: a.name text box b.age text box c.blood group text box d.email text box e.gender text box f.date text box g.mobile number text box h.location text box i.password text box j.apply button	Working as expected	Pass
RequestPage_TC_002	Functional	Request page	Verify whether the user able to request for donation		1.Enter URL and click go 2.Login to the application 3.Click on Donate in the home page 4.Enter name in name text box 5.Enter age in age text box 6.Enter blood group in blood group text box 7.Enter email in email text box 8.Enter gender in gender text box 9.Enter date in date text box 10.Enter mobile number in mobile number text box 11.Enter location in location text box 12. Enter password in password text box 13.Click on request button	Username: barath@gmail.com name: Barath age: 21 blood group: O+ve gender: Male Date: 2022-11-03 mobile number: 8925195582 location: Theni password: barath	The user will get Successfully applied and also get a mail from the application.	Working as expected	Pass
RequestPage_TC_003	Functional	Request page	Verify whether the user able to request for donation using invalid credentials		1.Enter URL and click go 2.Login to the application 3.Click on Donate in the home page 4.Enter name in name text box 5.Enter age in age text box 6.Enter blood group in blood group text box 7.Enter invalid email in email text box 8.Enter gender in gender text box 9.Enter date in date text box 10.Enter mobile number in mobile number text box 11.Enter location in location text box 12. Enter invalid password in password text box 13.Click on request button	Username: barath@gmail name: Barath age: 21 blood group: O+ve gender: Male Date: 2022-11-19 mobile number: 8925195582 location: Theni password: hello	Application should show 'Incorrect email or password' validation message.	Working as expected	Pass

8.2 USER ACCEPTANCE TESTING

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	0	0	0	0	0
Duplicate	0	0	0	0	0
External	1	0	0	1	1
Fixed	0	2	0	0	2
Not Reproduced	0	0	0	0	0
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	1	2	0	1	3

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
Login	3	0	0	3
Registration	4	0	0	4
Home	2	0	0	2
Plasma Request	3	0	1	2
Donor Application/Registration	2	0	0	2

CHAPTER 9

RESULTS

9.1 PERFORMANCE METRICS

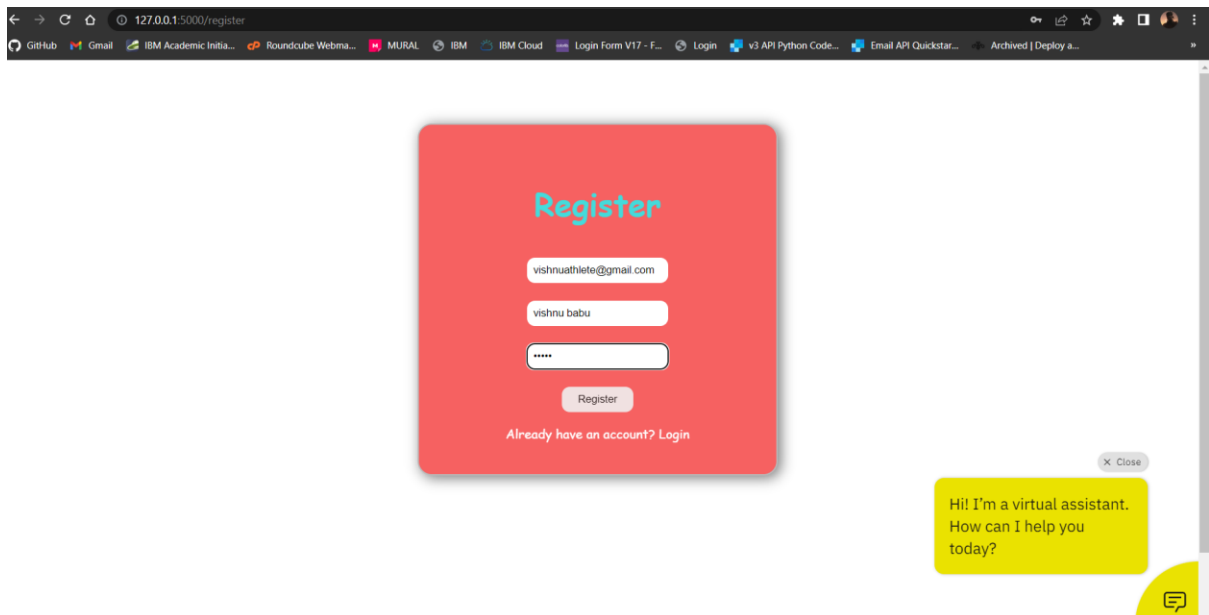
NFT - Risk Assessment									
S.No	Project Name	Scope/feature	Functional Changes	Hardware Changes	Software Changes	Impact of Downtime	Load/Volume Changes	Risk Score	Justification
1	Plasma Donor Application	Existing	Low	No Changes	Moderate		>5 to 10%	GREEN	As we have seen the changes
2	Plasma Donor Application	New	Low	No Changes	Moderate		>5 to 10%	ORANGE	As we have seen the changes
3	Plasma Donor Application	New	Low	No Changes	Moderate		>5 to 10%	GREEN	As we have seen the changes
4	Plasma Donor Application	Existing	Low	No Changes	High		>5 to 10%	GREEN	As we have seen the changes
5	Plasma Donor Application	New	Low	No Changes	Moderate		As we have seen the changes	As we have seen the changes	As we have seen the changes

NFT - Detailed Test Plan				
S.No	Project Overview	NFT Test approach	Assumptions/Dependencies/Risks	Approvals/SignOff
1	Login Page	LOAD	Page slow down. It may not be accessible	Akash B, Dineshkumar G
2	Apply page	STRESS	Page slow down. It may not be accessible	Akash B, Barathkumar P
3	Request Page	STRESS	Page slow down. It may not be accessible	Akash B, Hariharan V M

S.No	Project Overview	NFT Test approach	NFR - Met	Test Outcome	GO/NO-GO decision	Identified Defects (Detected/Closed/Open)	Approvals/SignOff
1	Login Page	LOAD	Not met because of login the user	PASS	NO-GO	CLOSED	Akash B, Dineshkumar G
2	Apply Page	STRESS	Not met because donor able to register	PASS	NO-GO	CLOSED	Akash B, Barathkumar P
3	Request Page	STRESS	Not met because requester is able to request	PASS	NO-GO	CLOSED	Akash B, Hariharan V M

OUTPUTS:

REGISTER PAGE



127.0.0.1:5000/register

Register

vishnuathilete@gmail.com

vishnu babu

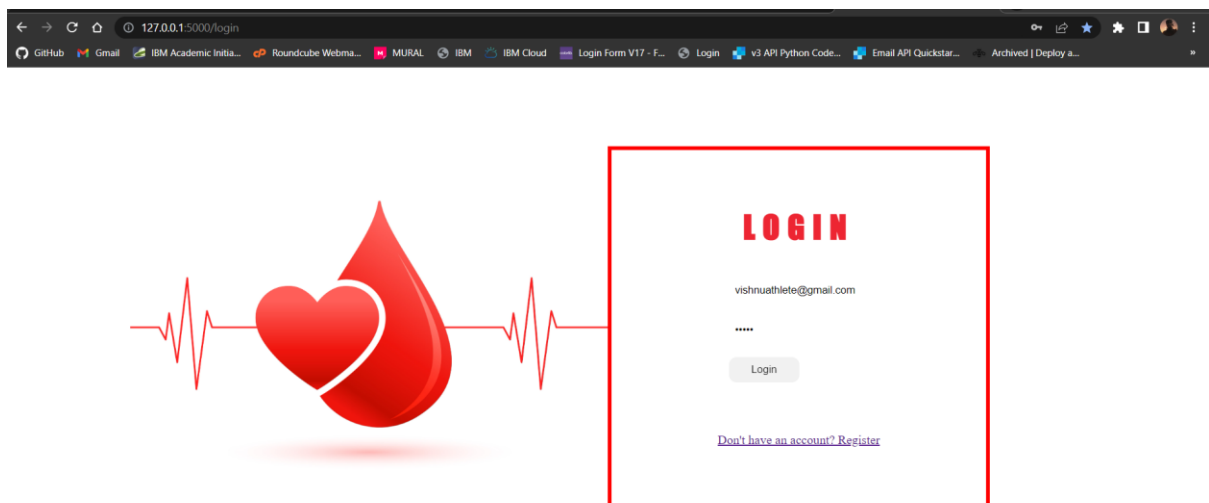
Register

Already have an account? [Login](#)

Close

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

LOGIN PAGE



127.0.0.1:5000/login

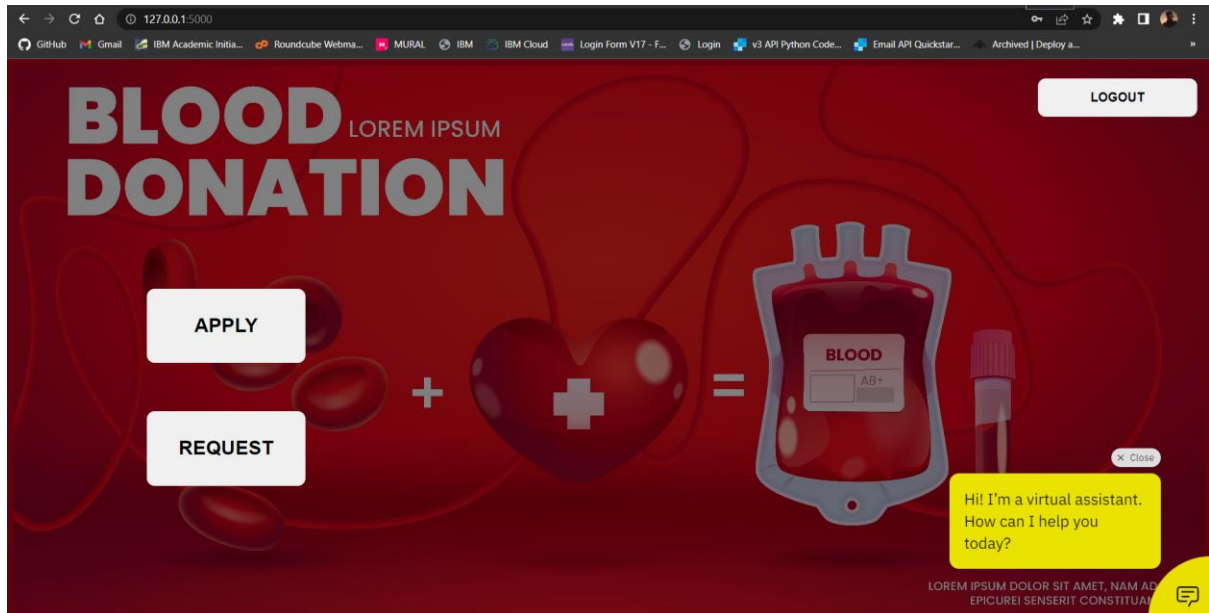
LOGIN

vishnuathilete@gmail.com

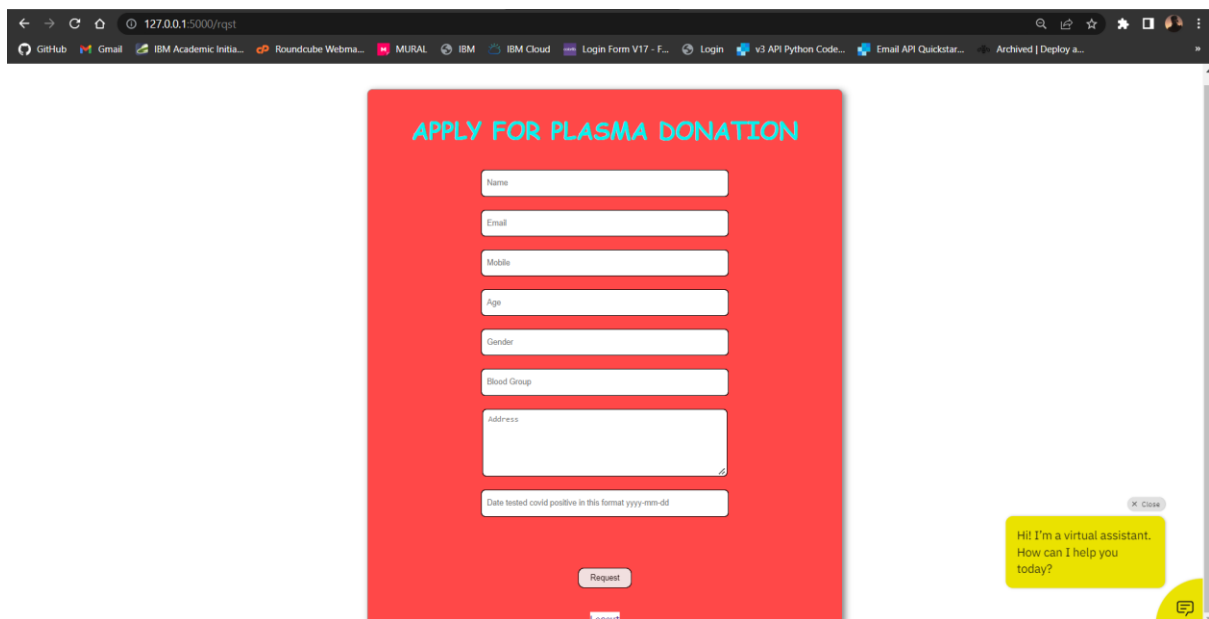
Login

[Don't have an account? Register](#)

HOME PAGE



APPLY PAGE



REGISTER PAGE

The screenshot shows a web browser window with the address bar displaying '127.0.0.1:5000/apply'. The browser's tab bar includes several open tabs: 'GitHub', 'Gmail', 'IBM Academic Initia...', 'Roundcube Webma...', 'MURAL', 'IBM', 'IBM Cloud', 'Login Form V17 - F...', 'Login', 'v3 API Python Code...', 'Email API Quickstar...', and 'Archived | Deploy a...'. The main content of the page is a red rectangular form titled 'PLASMA REQUEST' in bold red text. The form contains two columns of input fields. The left column includes fields for 'Name', 'Email', 'Mobile', 'Age', and 'Gender'. The right column includes fields for 'Blood Group', 'Aadhar', 'State', 'City', and 'Password'. Below the input fields, there is a light gray 'Apply' button. At the bottom center of the form, there is a red 'Logout' button. On the right side of the browser window, there is a vertical scrollbar and a small circular chat icon at the bottom.

PLASMA REQUEST

Name	Blood Group
Email	Aadhar
Mobile	State
Age	City
Gender	Password

Apply

Logout

CHAPTER 10

ADVANTAGES & DISADVANTAGES

10.1 ADVANTAGES

- The work load that was needed when visiting and locating the blood banks was difficult. So, by this application, that load can be minimized.
- The time consumption during in search of blood camps in person is maximum. So, by this application, the details of the requested person send out to donor in a short period.
- Another advantage over other medium that it provides a personal account for the user to login, apply and register. So, he can use this application whenever the user wants.
- Having a integrated chatbot service is a another advantage of this web application over other applications.

10.2 DISADVANTAGES

- Only problem that may caused is due to donor's late reception or something related to him. This web application contributes its services in a best way to its users.

CHAPTER 11

CONCLUSION

In recent days, it is noticed the increase in plasma request posts on social media such as Facebook, Twitter, and Instagram. Interestingly there are many people across the world interested in donating plasma when there is a need, but those donors don't have an access to know about the plasma donation requests in their local area. This is because that there is no platform to connect local blood donors with patients. Our application solves the problem and creates a communication channel through authorized clinics whenever a patient needs plasma donation. It is a useful tool to find compatible plasma donors who can receive plasma request posts in their local area. Clinics can use this web application to maintain the plasma donation activity. Collected data through this application can be used to analyse donations to requests rates in a local area to increase the awareness of people by conducting donations camps. The web application will send a notification to the registered donor's if there is any request for plasma. By sending the notification the donor will know whether there are any requests in the application it saves the time of the donor and it also share the request to the donor. After receiving the notification, the donor can visit the clinic for plasma donation. The application makes the searching of donor easier and more efficient by making it online the application saves the time of the requester and it allows the requester to find multiple donors. The right mail at the right time will change the lives of many with this motto the project model was working towards the process of the further development of the application towards the donation of the body organs also...

CHAPTER 12

FUTURE SCOPE

- This web application still needs some features which will make this app a very responsive one.
- For now, only mail notification system was implemented. In future, notification will send through different mediums such as making an automated post on several social medias, making an automated call to users regarding their services etc.
- Also in coming days, the history of the user's request and application will shown on a separate dashboard.
- By publishing the locations of the various blood camps and donation centres to users will help them to get needy help in time if they are nearby to those landmarks.

CHAPTER 13

APPENDIX

13.1 SOURCE CODE

APP.PY

```
from flask import Flask, render_template, request, redirect, url_for, session
from flask import Flask
import ibm_db
import bcrypt
import os
from mailing import *

conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=ba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=31321;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;PROTOCOL=TCPIP;UID=dc174999;PWD=Px6tqEmXL9AcQigs ","")

# url_for('static', filename='style.css')

app = Flask(__name__)
app.secret_key = b'_5#y2L"F4Q8z\n\xec]/'

@app.route("/", methods=['GET'])
def home():
    if 'email' not in session:
        return redirect(url_for('login'))
    return render_template('Home.html', name='Home')
```

```

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

        if not email or not username or not password:
            return render_template('Register.html',error='Please fill all fields')

        hash=bcrypt.hashpw(password.encode('utf-8'),bcrypt.gensalt())

        query = "SELECT * FROM USER WHERE email=? "
        stmt = ibm_db.prepare(conn, query)
        ibm_db.bind_param(stmt,1,email)
        ibm_db.execute(stmt)
        isUser = ibm_db.fetch_assoc(stmt)

        if not isUser:
            insert_sql = "INSERT INTO USER(EMAIL, USERNAME,PASSWORD)
VALUES (?,?,?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prepare_stmt, 1, email)
            ibm_db.bind_param(prepare_stmt, 2, username)
            ibm_db.bind_param(prepare_stmt, 3, hash)
            ibm_db.execute(prepare_stmt)
            register_notify(email)
            return render_template('Register.html',success="You can login")
        else:

```

```

    return render_template('Register.html',error='Invalid Credentials')

return render_template('Register.html',name='Home')

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

        if not email or not password:
            return render_template('Login.html',error='Please fill all fields')
        query = "SELECT * FROM USER WHERE email=?"
        stmt = ibm_db.prepare(conn, query)
        ibm_db.bind_param(stmt,1,email)
        ibm_db.execute(stmt)
        isUser = ibm_db.fetch_assoc(stmt)
        print(isUser,password)

        if not isUser:
            return render_template('Login.html',error='Invalid Credentials')

        isPasswordMatch = bcrypt.checkpw(password.encode('utf-8'),isUser['PASSWORD'].encode('utf-8'))

        if not isPasswordMatch:
            return render_template('Login.html',error='Invalid Credentials')

        session['email'] = isUser['EMAIL']

```

```

    return redirect(url_for('home'))

    return render_template('Login.html',name='Home')

@app.route("/apply",methods=['GET','POST'])
def apply():
    if request.method == 'POST':
        name = request.form.get('name')
        email = request.form['email']
        mobile = request.form['mobile']
        age = request.form['age']
        gender = request.form['gender']
        blood_group = request.form['blood_group']
        aadhar = request.form['aadhar']
        state = request.form['state']
        city = request.form['city']
        password = request.form['password']

        if not name or not email or not mobile or not age or not gender or not
        blood_group or not aadhar or not state or not city or not password:
            return render_template('Apply.html',error='Please fill all fields')

        hash=bcrypt.hashpw(password.encode('utf-8'),bcrypt.gensalt())

        query = "SELECT * FROM appliedusers WHERE email=? OR password=?;"
        stmt = ibm_db.prepare(conn, query)
        ibm_db.bind_param(stmt, 1, email)
        ibm_db.bind_param(stmt, 2, password)
        ibm_db.execute(stmt)

```

```

isUser = ibm_db.fetch_assoc(stmt)

if not isUser:
    insert_sql = "INSERT INTO appliedusers(NAME, EMAIL, MOBILE, AGE,
    GENDER, BLOOD_GROUP, AADHAR, STATE, CITY, PASSWORD)
    VALUES(?,?,?,?,?,?,?,?,?,?);"

    prep_stmt = ibm_db.prepare(conn, insert_sql)
    ibm_db.bind_param(prepare_stmt, 1, name)
    ibm_db.bind_param(prepare_stmt, 2, email)
    ibm_db.bind_param(prepare_stmt, 3, mobile)
    ibm_db.bind_param(prepare_stmt, 4, age)
    ibm_db.bind_param(prepare_stmt, 5, gender)
    ibm_db.bind_param(prepare_stmt, 6, blood_group)
    ibm_db.bind_param(prepare_stmt, 7, aadhar)
    ibm_db.bind_param(prepare_stmt, 8, state)
    ibm_db.bind_param(prepare_stmt, 9, city)
    ibm_db.bind_param(prepare_stmt, 10, password)
    ibm_db.execute(prepare_stmt)
    apply_notify(email)
    return render_template('Apply.html',success="Successfully Registered")
else:
    return render_template('Apply.html',error='Please check the details you have
    entered')

return render_template('Apply.html',name='Home')

@app.route("/rqst",methods=['GET','POST'])
def rqst():
    if request.method == 'POST':
        name = request.form.get('name')
        email = request.form['email']

```

```

mobile = request.form['mobile']
age = request.form['age']
sex = request.form.get('sex')
blood_group = request.form.get('blood_group')
address = request.form['address']
date = request.form.get('date')

```

```

insert_sql = "insert into requsers
(name,email,number,age,sex,blood_group,address,date) values (?,?,?,?,?,?,?);"
prep_stmt = ibm_db.prepare(conn, insert_sql)
ibm_db.bind_param(prepare_stmt, 1, name)
ibm_db.bind_param(prepare_stmt, 2, email)
ibm_db.bind_param(prepare_stmt, 3, mobile)
ibm_db.bind_param(prepare_stmt, 4, age)
ibm_db.bind_param(prepare_stmt, 5, sex)
ibm_db.bind_param(prepare_stmt, 6, blood_group)
ibm_db.bind_param(prepare_stmt, 7, address)
ibm_db.bind_param(prepare_stmt, 8, date)
ibm_db.execute(prepare_stmt)
request_notify(email)
return render_template('request.html',success="Successfully Registered")
return render_template('request.html',name='Home')

```

```

@app.route('/logout')
def logout():
    session.pop('email', None)
    return redirect(url_for('login'))

```



```

if __name__ == "__main__":
    # port=int(os.environ.get('PORT',8000))
    # app.run(host='0.0.0.0',port=port)
    app.run(debug=True)

```

MAILING.PY

```

import sendgrid
import os
from sendgrid.helpers.mail import *
def register_notify(email):
    sg = sendgrid.SendGridAPIClient(api_key=os.environ.get('SENDGRID_API_KEY'))
    from_email = Email("ibmproject00@gmail.com")
    to_email = To(email)
    subject = "Plasma-Registration Success"
    content = Content("text/plain", "Registration for opening an account was successfully completed. Now, you can login with your credentials.")
    mail = Mail(from_email, to_email, subject, content)
    response = sg.client.mail.send.post(request_body=mail.get())
    print(response.status_code)
    print(response.body)
    print(response.headers)

```

```

def apply_notify(email):
    sg = sendgrid.SendGridAPIClient(api_key=os.environ.get('SENDGRID_API_KEY'))

```

```

from_email = Email("ibmproject00@gmail.com")
to_email = To(email)
subject = "Plasma-Regarding your plasma donation application "
content = Content("text/plain","Your application for donation of plasma was
successfully done. You will get notified once a matched request found...")
mail = Mail(from_email, to_email, subject, content)
response = sg.client.mail.send.post(request_body=mail.get())
print(response.status_code)
print(response.body)
print(response.headers)
def request_notify(email):
    sg = sendgrid.SendGridAPIClient(api_key=os.environ.get('SENDGRID_API_KEY'))
    from_email = Email("ibmproject00@gmail.com")
    to_email = To(email)
    subject = "Plasma-Regarding your plasma Request"
    content = Content("text/plain", "Your request for needed plasma was
successfully submitted. A perfect matched donor will contact you soon...")
    mail = Mail(from_email, to_email, subject, content)
    response = sg.client.mail.send.post(request_body=mail.get())
    print(response.status_code)
    print(response.body)
    print(response.headers)

```

HOME.html

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```

<title>Home</title>

<script>
  window.watsonAssistantChatOptions = {
    integrationID: "86ce9fb6-2ca4-48ec-aeb3-ddbed260132b", // The ID of this
integration.
    region: "us-south", // The region your integration is hosted in.
    serviceInstanceID: "77eeda2-2b53-4a2e-bfcd-3b68cd38e464", // The ID of
your service instance.

    onLoad: function (instance) { instance.render(); }
  };
  setTimeout(function () {
    const t = document.createElement('script');
    t.src = "https://web-chat.global.assistant.watson.appdomain.cloud/versions/"
+ (window.watsonAssistantChatOptions.clientVersion || 'latest') +
"/WatsonAssistantChatEntry.js";
    document.head.appendChild(t);
  });
</script>

<style>
  body {
    overflow: hidden;
    background-color: #60010e;
  }

  img {
    width: 100%;
    height: 100vh;
    object-fit: cover;

    filter: brightness(0.5);

```

```
z-index: -1;  
}
```

```
button a {  
  text-decoration: none;  
  color: black;  
  font-weight: 600;  
}
```

```
button {  
  border-radius: 10px;  
  padding: 15px 44px;  
  outline: none;  
  border: none;  
  width: 203px;  
  font-size: 16px;  
  z-index: 2;  
;  
}
```

```
button:hover{  
  background-color: #c4e7ff  
}
```

```
.btn1 {  
  position: absolute;  
  top: 25px;  
  right: 18px;
```

```

        z-index: 2;
    }
    .btn_cont{
position: absolute;
top:35%;
    }
    .btn_cont button {
margin-top: 43px;
margin-left: 170px;
padding: 33px;
font-size: 25px;
    }
</style>
</head>

<body>
    
    <center>
        <h1> WELCOME!!</h1><Br>
        <div class="btn_cont">
            <button><a href="/rqst">APPLY</a></button><br><br>
            <button><a href="/apply">REQUEST</a></button><br><br>
        </div>
        <button class="btn1"><a href="/logout">LOGOUT</a></button>
    </center>
</body>

</html>

```

LOGIN.HTML

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
<title>Login</title>
```

```
<style>
```

```
  /* body {
```

```
    background: linear-gradient(white, white, rgb(156, 244, 238), rgb(60, 255, 242), rgb(25, 255, 240));
```

```
    height: 600px;
```

```
    text-align: center;
```

```
    font-family: cursive;
```

```
  } */
```

```
  h3 {
```

```
    color: rgb(237, 37, 50);
```

```
    padding-top: 7%;
```

```
    font-size: 40px;
```

```
    font-family: fantasy;
```

```
    letter-spacing: 10px;
```

```
  }
```

```
  input {
```

```
    border: 2px;
```

```
    border-radius: 7px;
```

```
    padding-top: 8px;
```

```
    padding-left: 8px;
```

```
    padding-bottom: 8px;
```

```

        padding-right: 8px;
    }

    button {
        border: 2px;
        border-radius: 10px;
        padding-top: 8px;
        padding-left: 8px;
        padding-bottom: 8px;
        padding-right: 8px;
        width: 90px;
        opacity: 90%;
        cursor: pointer;
    }

    .login_cont{
        display: flex;
        width: 80%;
    }

    .image img {
width: 100%;

}

    .image{
        width: 50%;
    }

    .login_text_cot{
        width:40%;
    }

```

```

.login_text_cot_main{
    display: flex;
    flex-direction: column;
    align-items: center;
    border:5px solid red;
}

.login_cont_main{
    width: 100%;
    display: flex;
    justify-content: center;
    margin-top:100px;

}
</style>
</head>

<body>
<div class="login_cont_main">
<div class="login_cont">

<div class="image">
    
</div>

<div class="login_text_cot">
<div class="login_text_cot_main">
    <h3>LOGIN</h3>
    <form method="POST">

```



```
<input type="email" name="email" placeholder="Email" required
/><br><br>
```

```
<input type="password" name="password" placeholder="Password"
required /><br><br>
```

```
<button type="submit">Login</button>
```

```
</form>
```

```
<p style="color: rgb(27, 161, 27);">{{ success }}</p>
```

```
<p style="color: rgb(214, 25, 25)">{{ error }}</p>
```

```
<a href="/register" style="margin-bottom:80px">Don't have an account?
Register</a>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</body>
```

```
</html>
```

13.2 GITHUB/PROJECT DEMO LINK

GITHUB REPO LINK:

<https://github.com/IBM-EPBL/IBM-Project-2805-1658483149>

PROJECT DEMO LINK:

<https://youtu.be/ZZ8bsYkJSX8>