PLASMA DONAR APPLICATION

HX 8001-

Professional Readiness For Innovation, Employability and Entrepreneurship

IBM-Project-14173-1659543653

TEAM ID : PNT2022TMID08318

Submitted by

KOTHAMASU VENKATA RATNA SAI	(810419104054)
CHUNDURI SAI BABU	(810419104020)
JASTHI MANIKANTA	(810419104038)
BOMMISETTI PARDHA SAI	(810419104017)

in partial fulfillment for the award of the degree

of

BATCHELOR OF ENGINNERING

IN

COMPUTER SCIENCE AND ENGINEERING

DHANALAKSHMI SRINIVASAN ENGINEERING COLLEGE
(AUTONOMOUS)
PERAMBALUR-621212

TABLE OF CONTENTS

ABSTRACT :

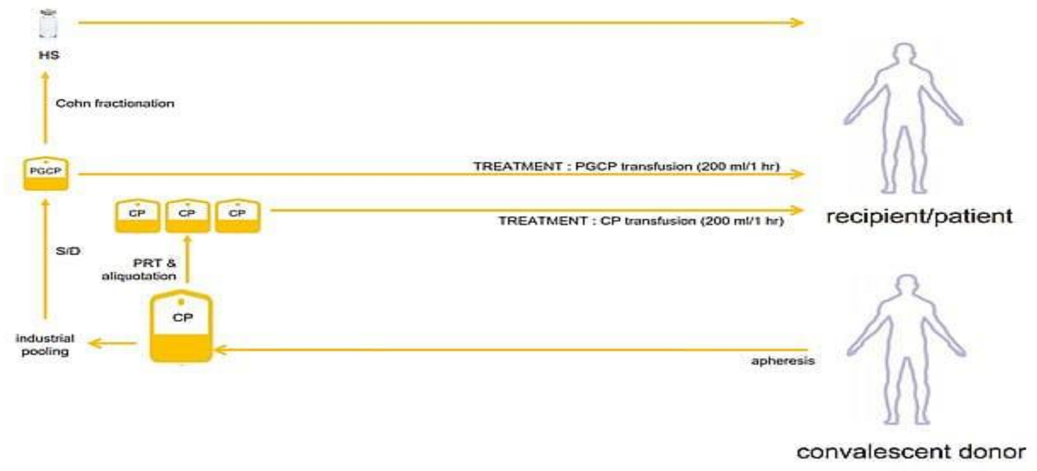
A Web based plasma application projects that act as a central database containing various plasma (with the blood groups) in dash board along with their blood groups category and database . Online plasma donar application is fast gaining ground as an accepted and used business paradigm. With rapid increase in the usage of social networks sites across the world, there is also a steady increase in blood and plasma donation requests as being noticed in the number of posts on these sites such as Facebook and twitter seeking for plasma donors. Finding plasma donor is a challenging issue in almost every country. There are some blood and plasma donor finder applications in the market such as Blood app by Red Cross and Blood and plasma Donor Finder application by Neologix. However, more reliable applications that meet the needs of users are prompted.

1. INTRODUCTION :

1.1 Project Overview :

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

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.



1.2 Purpose :

In a plasma-only donation, the liquid portion of the donor’s blood is separated from the cells. Blood is drawn from one arm and sent through a high-tech machine that collects the plasma. The donor’s red blood cells and platelets are then returned to the donor along with some saline. The process is safe and only takes a few minutes longer than donating whole blood.

Donated plasma is frozen within 24 hours of being donated to preserve its valuable clotting factors. It can be stored for up to one year and thawed for transfusion to a patient when needed. Red Cross donations are often used directly for hospital patient transfusions, rather than pharmaceutical uses.

Only a small number of people living in the U.S. who are eligible to donate blood or source plasma actually donate. What's important is that we encourage all forms of donation from those who are eligible, so that they may contribute life-saving blood and source plasma to those in need.

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 a genetic lung disease, as well as in the treatment of trauma, burns and shock. Whole blood donations most often are used locally in hospitals for transfusions required during surgery or other medical treatment. Find a donation center near you!
Plasma donation requires a commitment both in the amount of time for each donation and 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 blood 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.

Doctors can use plasma to treat different kinds of serious health problems. Some of the elements in plasma, including the antibodies and chemicals that help your blood to clot, can help in medical emergencies like burns and trauma.
Other things that plasma donation is good for include:

- Developing treatments.
- Cancer.
- Transplant surgery.
- Hemophilia.

2. LITERATURE SURVEY :

2.1 Existing problem :

As we know that we does not have any kind of website or application for the plasma donar application. So we had adopt the existing model for the blood donation management application. We had built the virtual environment for this project.

- Rishab Chakrabarti, Prof. S. M. Chitalkar - “Lifesaver E-Blood Donation App Using Cloud”, 2020: Reduction in the errors of blood bank using most eligible donor method. Direct Communication Between donor and the person in need of blood During the Emergency situation. However, this paper has the drawback that the user-provided information is still unconfirmed.
- A. Meiyappan, K. Loga Vignesh, R. Prasanna, T. Sakthivel - “D’WORLD: Blood Donation App Using Android”, 2019: When the giver gives the blood, it will naturally evacuate the contributor detail for next three months. It additionally confirms with the Department of Health and Welfare to guarantee the benefactor medical case history. However, this has the drawback that in order to utilize this program, the user must have a device running the Android operating system and a live internet connection.
- P. C. P. C. A. V. I. M. Yan - “Building a chatbot with serverless computing” IBM Watson research center, 2016: Author conducted a survey of existing serverless platform in this paper from source projects, industry, academia, use cases, and key characteristics and has described the challenges and the open problems associated with it. Authors work presented a hands-on experience of serverless technologies using different services from different cloud provides such as Amazon, Google, IBM, Microsoft Azure
- Ashlesha C. Adsul, V. K. Bhosale, R. M. Autee - “Automated blood bank system using Raspberry PI”, 2018: When there is urgent need for blood then If this model is adopted the caller is immediately connected to the donor. However, dealing with the phone users is a drawback.
- Aishwarya, R Gowri – “Developing a Plasma donor application using Function-as-a service in AWS”: A plasma is a liquid portion of the blood, over55% of human blood is plasma. Plasma is used to treat various infectious diseases and itis one of the oldest methods known as plasma therapy. Plasma therapy is a process where blood is donated by recovered patients in order to establish antibodies that fights the infection. In this project plasma donor application is being developed by using AWS services.

2.2 References:

1. https://www.html.am/html-editors/online-html-editor.cfm - Online HTML Editor for ease of creating HTML Pages
2. https://suedbroecker.net/2019/03/05/how-to-deploy-a-container-to-the-ibm-cloud-kubernetes-service/
3. https://cloud.ibm.com/docs/Registry?topic=Registry-registry_setup_cli_namespace
4. https://cloud.ibm.com/docs/Registry?topic=Registry-getting-startedd
5. https://www.ibm.com/blogs/cloud-archive/2019/04/kubernetes-deployments-get-started-fast/
6. https://cloud.ibm.com/docs/cli/reference/ibmcloud_cli/get_started.html#getting-started
7. https://kubernetes.io/docs/tasks/tools/
8. https://cloud.ibm.com/docs/cli?topic=cli-plug-ins

2.3 Problem Statement Definition :

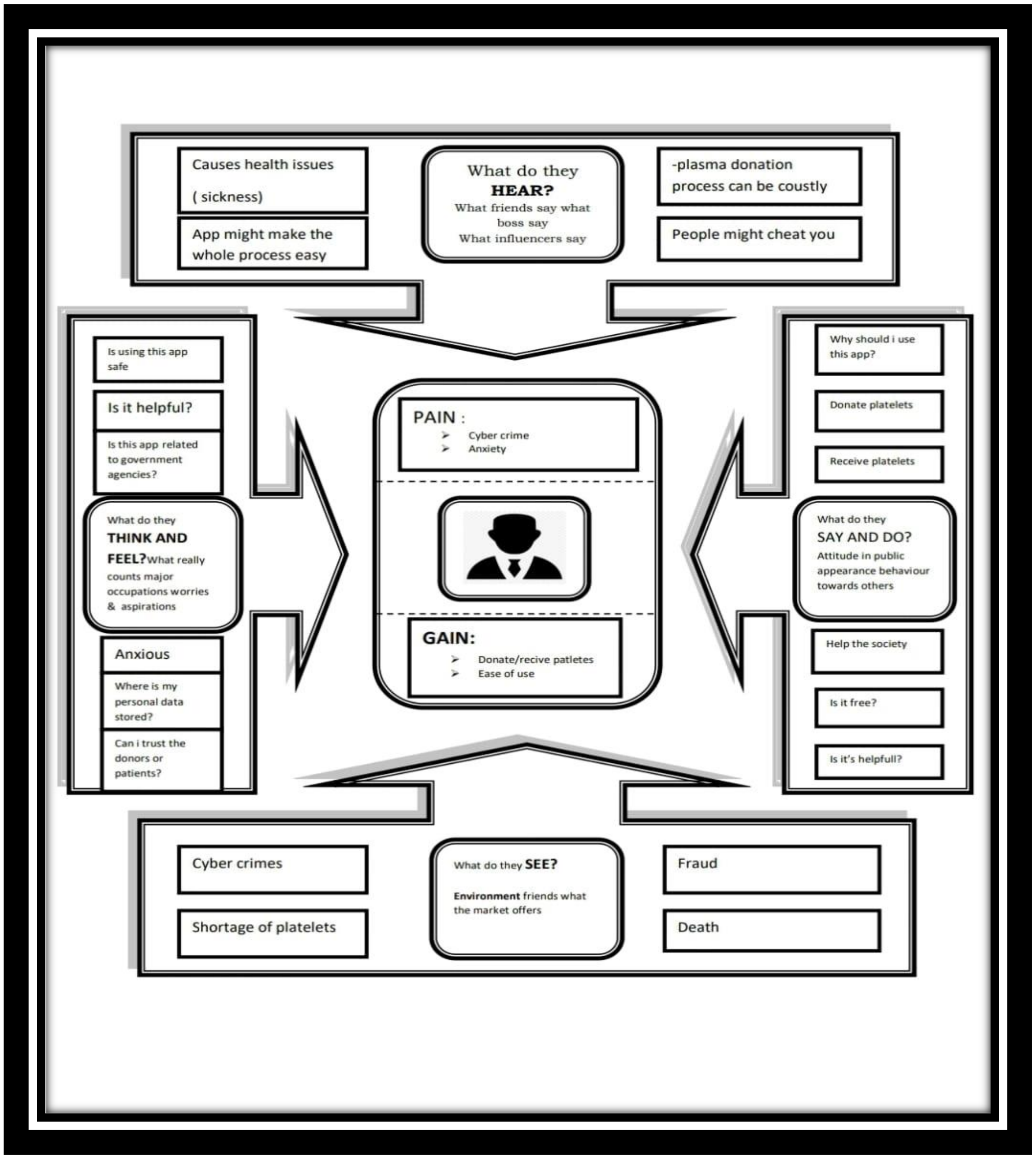
People who need plasma are increasing day by day. People who have diseases like trauma, burn, shock patients ,as well as peoples with severe liver disease or multiple clotting factors deficiencies people who have gotten into accidents and run out of plasma need constant supply of plasma to sustain their life and there is not enough plasma available for them. It is not that people do not want to donate plasma, 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 plasma donation camps ongoing in a particular area so that people who want to donate plasma will get information regarding these camps. Our System is a web application which aims to serve as a communication tool between plasma Donation camp Organizers and plasma 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, age factor(age restriction) 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 web application lways keeps updating the location of a donor. As a result, the system can automatically keep showing the nearby plasmadonation Camps to the registered donor wherever they go, and donors can easily get the idea of nearby plasma donation camps. Also, users can get information regarding the type of Plasma or blood which is available and information of past as well as
future events.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas :

An empathy map is a collaborative tool teams can use to gain a deeper insight into their customers. Much like a user persona, an empathy map can represent a group of users, such as a customer segment. The empathy map was originally created by Dave Gray and has gained much popularity within the agile community.


Causes health issues (sickness)
App might make the whole process easy
What do they HEAR? What friends say what boss say What influencers say
-plasma donation process can be coustly
People might cheat you
Is using this app safe
Is it helpful?
Is this app related to government agencies?
What do they THINK AND FEEL? What really counts major occupations worries & aspirations
Anxious
Where is my personal data stored?
Can i trust the donors or patients?
PAIN : Cyber crime Anxiety
GAIN: Donate/recive patletes Ease of use
Why should i use this app?
Donate platelets
Receive platelets
What do they SAY AND DO? Attitude in public appearance behaviour towards others
Help the society
Is it free?
Is it's helpfull?
Cyber crimes
Shortage of platelets
What do they SEE? Environment friends what the market offers
Fraud
Death

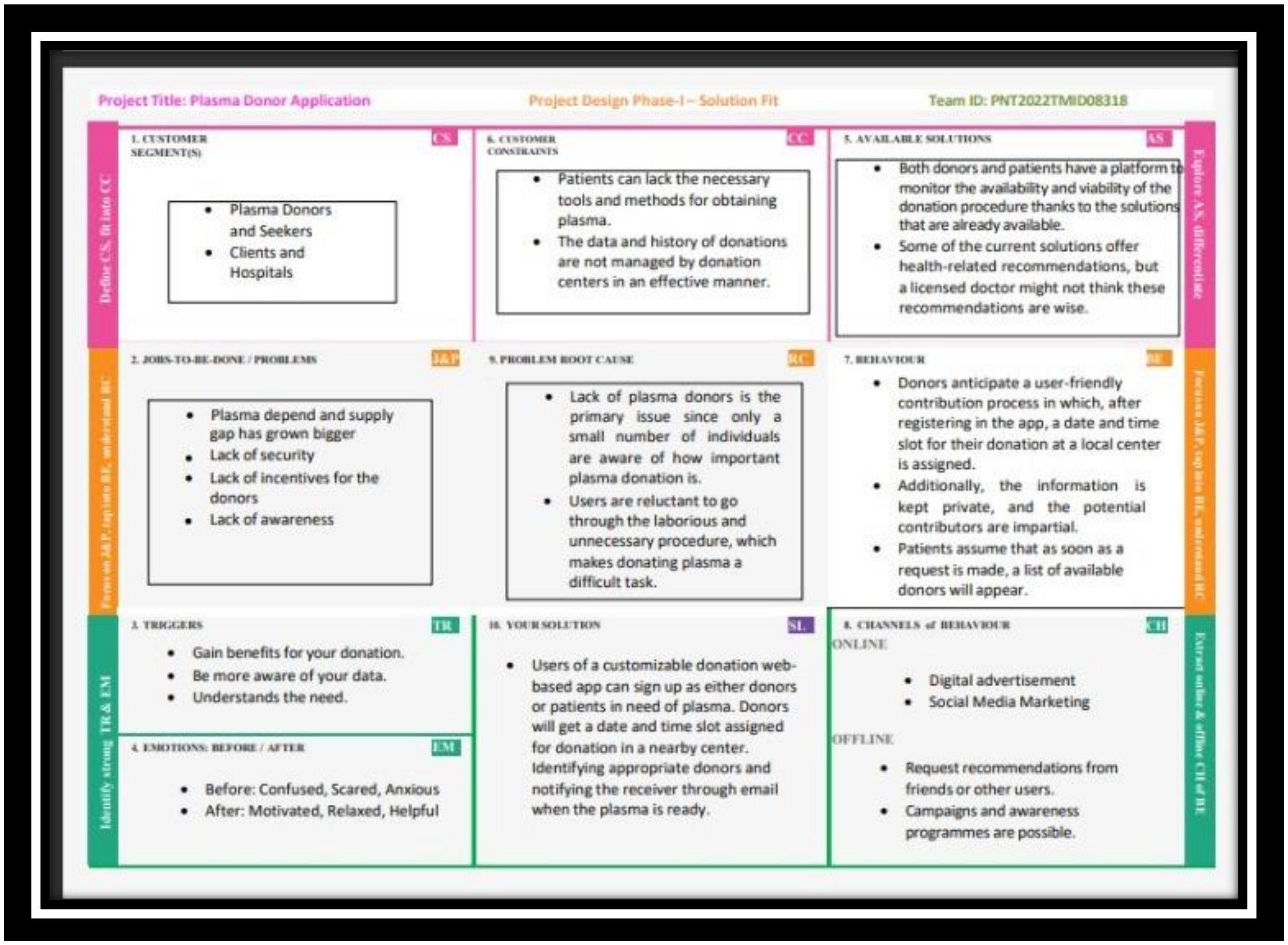
3.2 Ideation & Brainstorming :

3.3 Proposed Solution :

S.NO	Parameter	Description
1.	Problem Statement (Problem to be solved)	People who are in need of plasma are increasing day by day. Plasma is necessary to help our body to recover from injury, distribute nutrients, remove waste and prevent infection, while moving throughout our circulatory system. It is not that people don’t want to donate plasma, but they have no idea

		where they can donate. We are designing a platform which contains all the information regarding Plasma donation.
2.	Idea / Solution description	Ours is a mobile application which aims to serve as a communication tool between plasma donation organizers and plasma donors. To become a member of our system, donors need to create their profile by providing their information like name, blood group, email address, phone number, password and exact location from 'Google Map', which are integrated with this application. This mobile app always keep updating the location of the donor.
3.	Novelty / Uniqueness	Users can submit their comments if they had any difficulties during donation process. This app automatically keeps showing the plasma donors nearby. Donor will save the donor card digitally
4.	Social Impact/ Customer Satisfaction	This app will make revolutionary changes to the medical system as people will be able to donate plasma and serve the mankind. It can also help the people to know about the benefits of plasma donation, so that their small contribution can help one person to save his/her life.
5.	Business Model (Revenue Model)	There are many private sectors and NGOs, who organize plasma donation camps. Even collaboration with companies like Biolife, and other pharmaceutical companies use plasma to make treatment for conditions such as immune deficiencies and bleeding disorder in order to increase revenue.
6.	Scalability of the Solution	This application has the ability to handle more donors and provide users with good user experience. It handles the traffic, responding accurately and reacting to the growing number of requests.

3.4 Problem Solution fit :



Project Title: Plasma Donor Application

Project Design Phase-I – Solution Fit

Team ID: PNT2022TMID08318

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

- Plasma Donors and Seekers
- Clients and Hospitals

6. CUSTOMER CONSTRAINTS CC

- Patients can lack the necessary tools and methods for obtaining plasma.
- The data and history of donations are not managed by donation centers in an effective manner.

5. AVAILABLE SOLUTIONS AS

- Both donors and patients have a platform to monitor the availability and viability of the donation procedure thanks to the solutions that are already available.
- Some of the current solutions offer health-related recommendations, but a licensed doctor might not think these recommendations are wise.

Explore AS, differentiate

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

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

- Plasma depend and supply gap has grown bigger
- Lack of security
- Lack of incentives for the donors
- Lack of awareness

9. PROBLEM ROOT CAUSE RC

- Lack of plasma donors is the primary issue since only a small number of individuals are aware of how important plasma donation is.
- Users are reluctant to go through the laborious and unnecessary procedure, which makes donating plasma a difficult task.

7. BEHAVIOUR BE

- Donors anticipate a user-friendly contribution process in which, after registering in the app, a date and time slot for their donation at a local center is assigned.
- Additionally, the information is kept private, and the potential contributors are impartial.
- Patients assume that as soon as a request is made, a list of available donors will appear.

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

Identify strong TR & EM

3. TRIGGERS TR

- Gain benefits for your donation.
- Be more aware of your data.
- Understands the need.

4. EMOTIONS: BEFORE / AFTER EM

- Before: Confused, Scared, Anxious
- After: Motivated, Relaxed, Helpful

10. YOUR SOLUTION SL

- Users of a customizable donation web-based app can sign up as either donors or patients in need of plasma. Donors will get a date and time slot assigned for donation in a nearby center. Identifying appropriate donors and notifying the receiver through email when the plasma is ready.

8. CHANNELS of BEHAVIOUR CH

ONLINE

- Digital advertisement
- Social Media Marketing

OFFLINE

- Request recommendations from friends or other users.
- Campaigns and awareness programmes are possible.

Extract online & offline CH of BE

4. REQUIREMENT ANALYSIS :

4.1 Functional requirement :

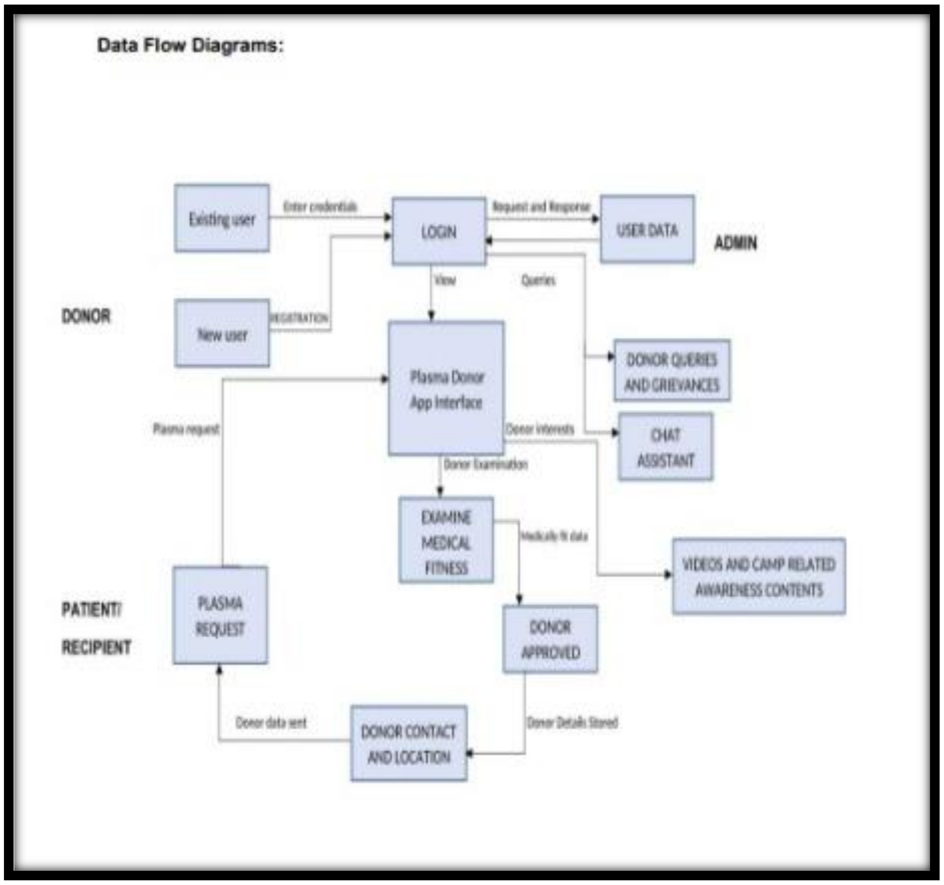
- Login of admin.
- Plasma Donor
- Change the login password of admin.
- Register the donor by himself.
- Register the donor by system admin.
- Login of the donor
- Change the login password of the donor.
- Change personal, contact details by the donor himself.
- Change personal, contact details by system admin.
- Withdraw reg. details by the donor.
- Withdraw reg. details by the admin.
- Send plasma donation details to the relevant donors.
- Send plasma testing details.
- Send plasma request.

4.2 Non-Functional requirement :

A characteristic of a quality SRS is that in addition to describing the functional requirements of a system, It will also provide detailed coverage of the non-functional requirements. In practice, this would entail detailed analysis of issues such as availability, security, usability and maintainability. However, as this document is only an outline specification, it does not contain the same degree of rig our that would normally be expected in a formal SRS. Therefore, the sections below should be seen as indicative rather than providing specific (l.e. testable) requirements.

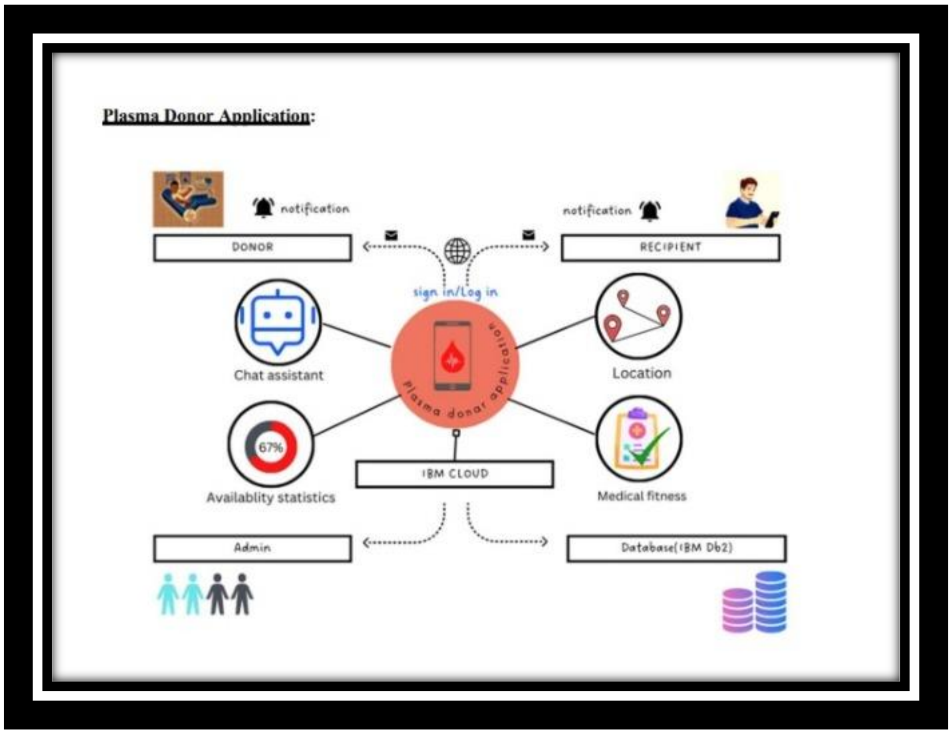
5. PROJECT DESIGN :

5.1 Data Flow Diagrams :

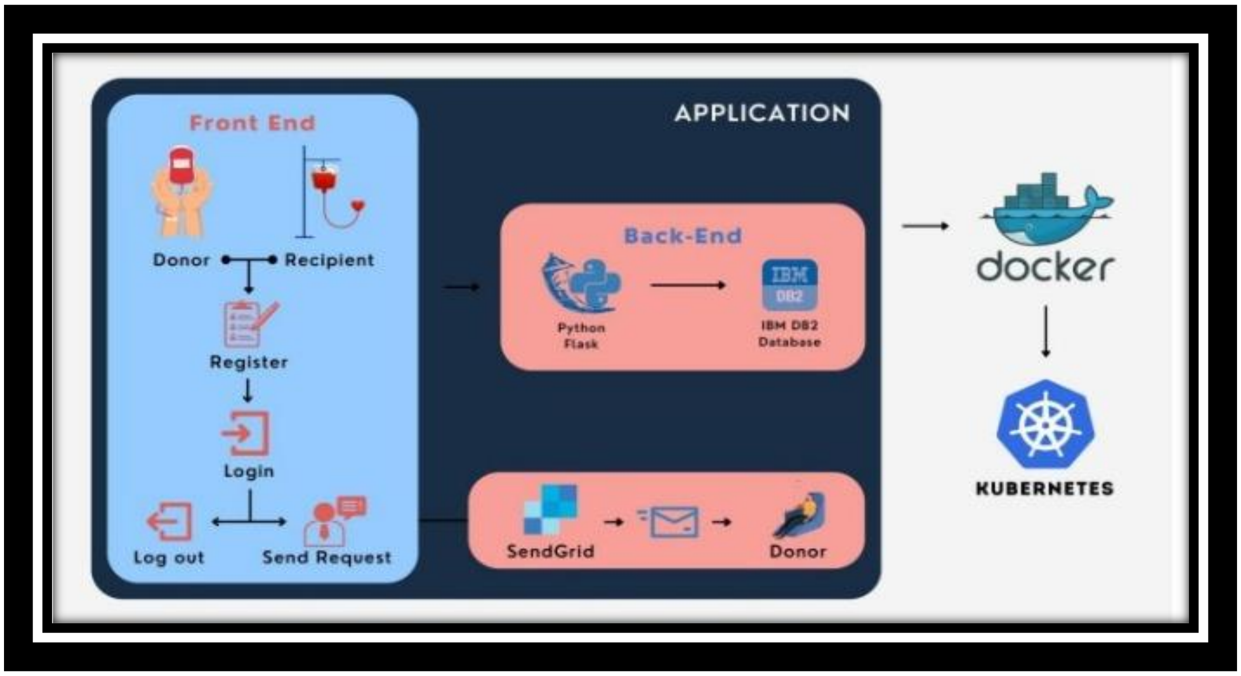


5.2 Solution & Technical Architecture :

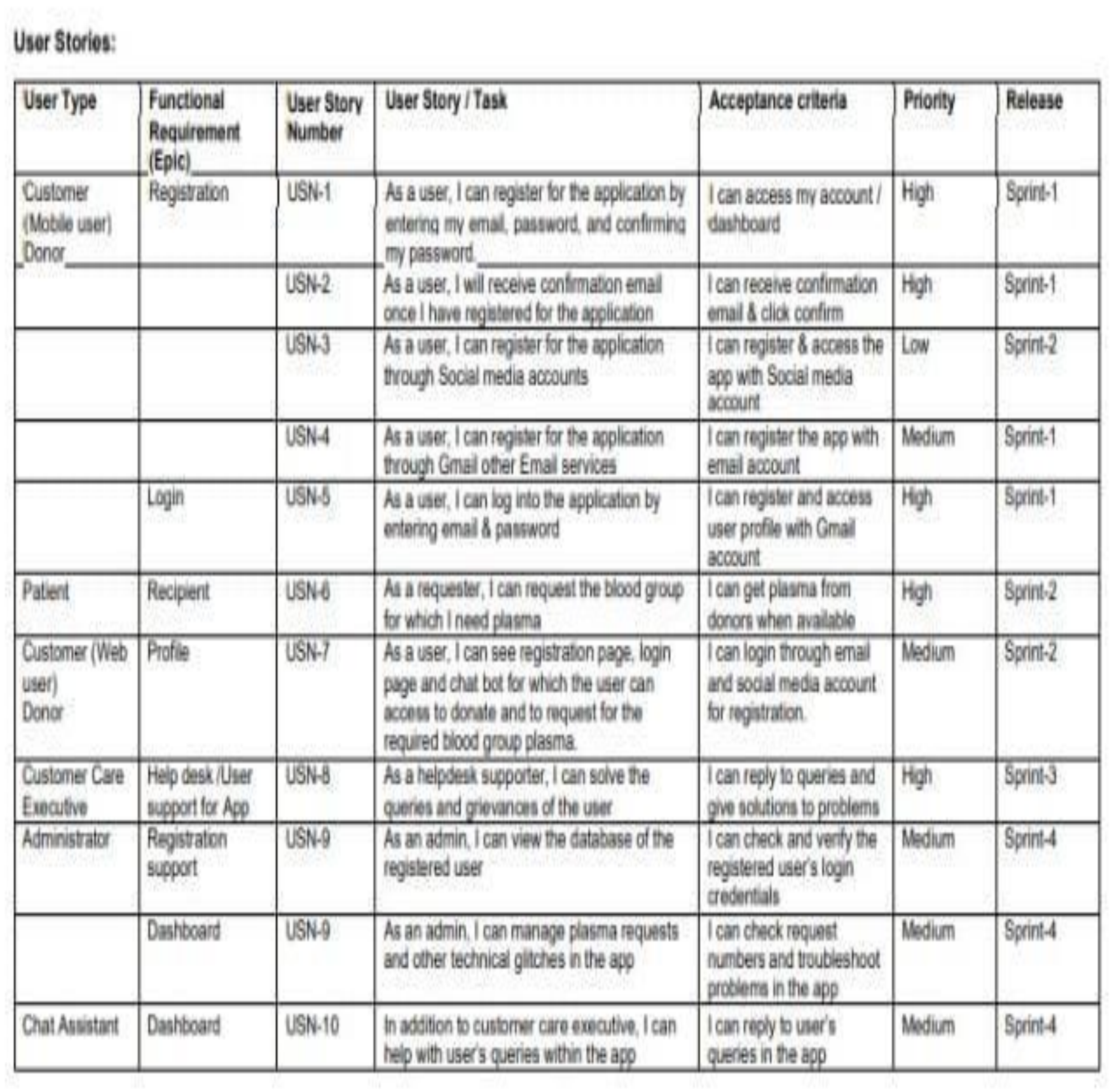
Solution Architecture :



Technical Architecture :



5.3 User Stories :



User Stories:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user) Donor	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Social media accounts	I can register & access the app with Social media account	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail other Email services	I can register the app with email account	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can register and access user profile with Gmail account	High	Sprint-1
Patient	Recipient	USN-6	As a requester, I can request the blood group for which I need plasma	I can get plasma from donors when available	High	Sprint-2
Customer (Web user) Donor	Profile	USN-7	As a user, I can see registration page, login page and chat bot for which the user can access to donate and to request for the required blood group plasma.	I can login through email and social media account for registration.	Medium	Sprint-2
Customer Care Executive	Help desk /User support for App	USN-8	As a helpdesk supporter, I can solve the queries and grievances of the user	I can reply to queries and give solutions to problems	High	Sprint-3
Administrator	Registration support	USN-9	As an admin, I can view the database of the registered user	I can check and verify the registered user's login credentials	Medium	Sprint-4
	Dashboard	USN-9	As an admin, I can manage plasma requests and other technical glitches in the app	I can check request numbers and troubleshoot problems in the app	Medium	Sprint-4
Chat Assistant	Dashboard	USN-10	In addition to customer care executive, I can help with user's queries within the app	I can reply to user's queries in the app	Medium	Sprint-4

6. PROJECT PLANNING & SCHEDULING :

6.1 Sprint Planning & Estimation :

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points Priority	Team Members
Sprint 1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	20 High	RATNA SAI SAI BABU MANIKANTA PARDHA SAI
Sprint 2	Login	USN-2	As a user, I can login into my account through the registered mail ID.	20 High	RATNA SAI SAI BABU MANIKANTA PARDHASAI
Sprint 3	Donor Information	USN-3	As a user, I can fill the information like blood pressure, blood group, address, mobile number andother information.	20 Low	RATNA SAI SAI BABU MANIKANTA PARDHASAI
Sprint 4	Finding theDonor	USN-4	The patient can find the donor by their blood groups, location.	20 Medium	RATNA SAI SAI BABU MANIKANTA PARDHASAI

6.2 Sprint Delivery Schedule :

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date *(Planned)*	Story Points Completed *(On Planned End Date)*	Sprint Release *Date (Actual)*
Sprint-1	20	5 Days	31 Oct 2022	04 Oct 2022	20	04 Oct 2022
Sprint-2	20	5 Days	05 Nov 2022	09 Nov 2022	20	09 Nov2022
Sprint-3	20	5 Days	10 Nov 2022	14 Nov 2022	20	14 Nov 2022
Sprint-4	20	5 Days	15 Nov 2022	19 Nov 2022	20	19 Nov 2022

6.3 Reports from JIRA :

VELOCITY: SPRINT - 1

Sprint duration = 5 days

Velocity of team = 20 points

Average Velocity (AV) = $\frac{\text{Velocity}}{\text{Sprint duration}}$

AV = 20/5 = 4

Average Velocity = 4

VELOCITY: Sprint 1 - 4

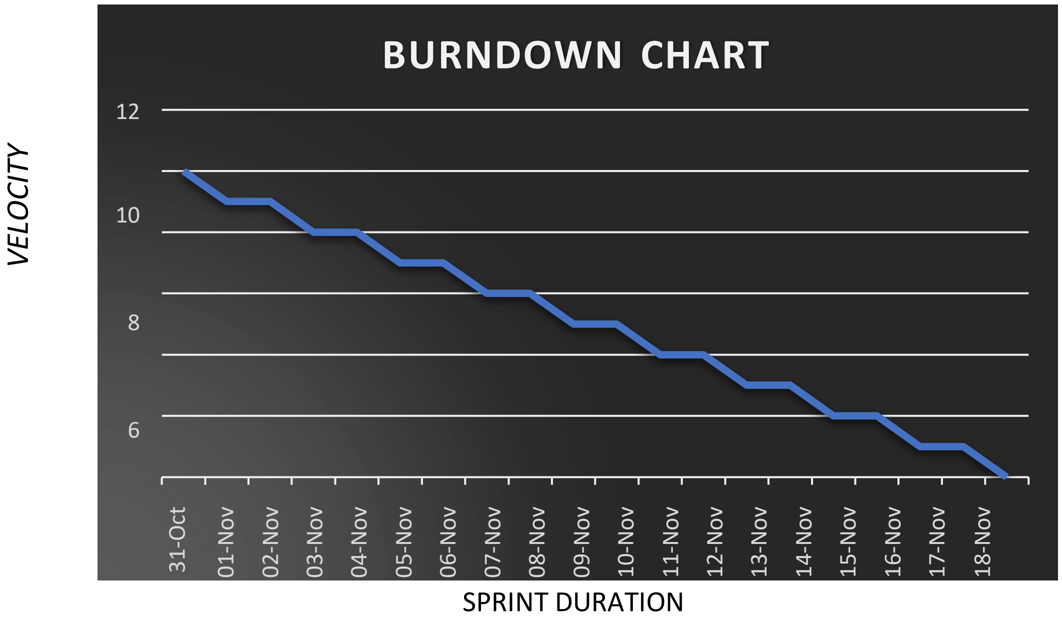
Sprint duration = 20 days

Velocity of team = 80 points

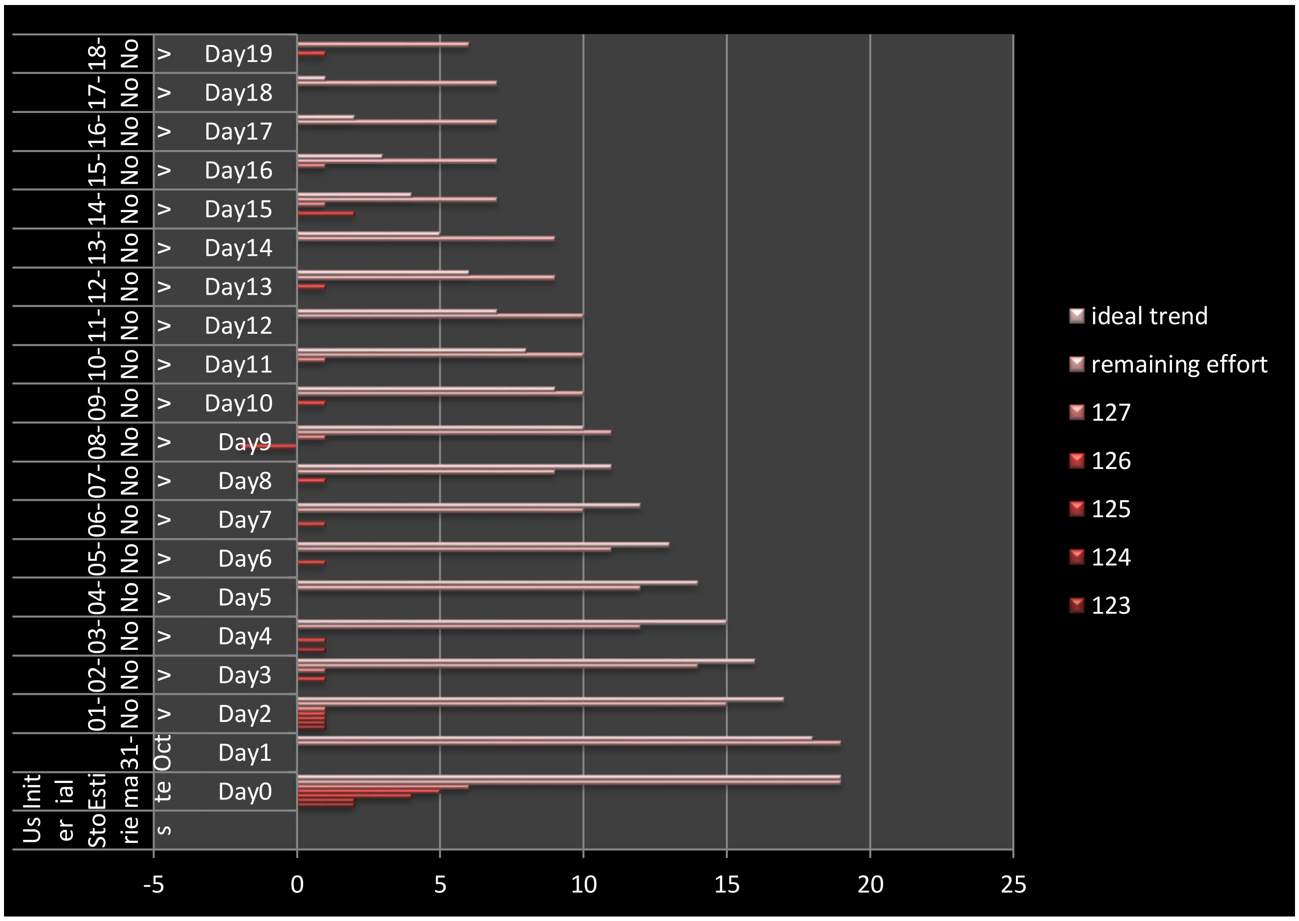
Average Velocity (AV) = $\frac{\text{Velocity}}{\text{Sprint duration}}$

AV = 80/20 = 4

Total Average Velocity = 4



SPRINT BURNDOWN CHART :

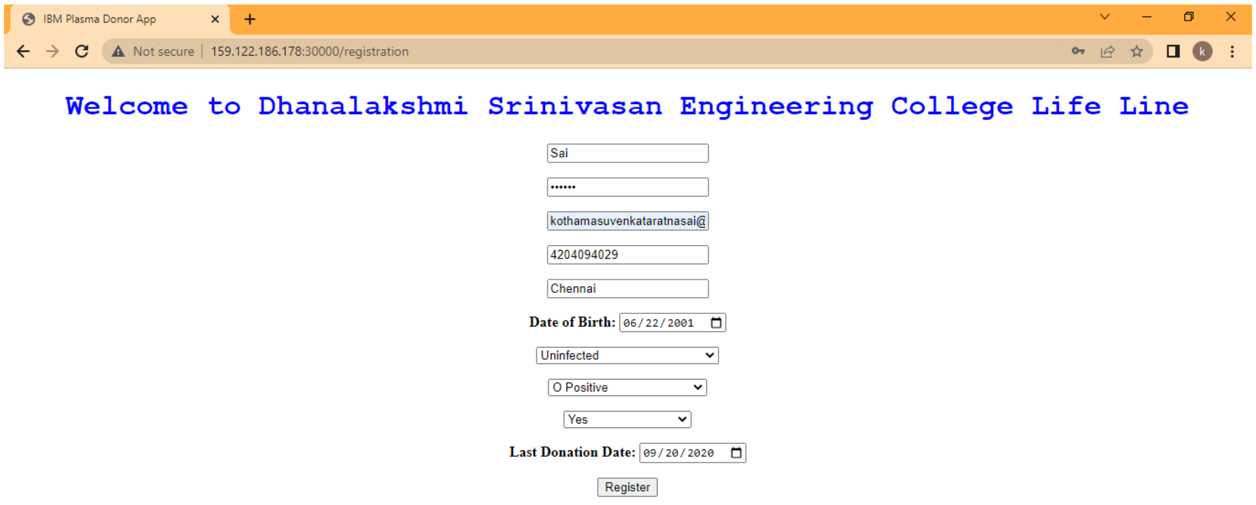


7. CODING & SOLUTIONING:

7.1 Registration Page:

CODE :

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

def registration():
    msg = ''
    if request.method == 'POST' :
        username = request.form['username'].lower()
        password = request.form['password']
        email = request.form['email']
        phone = request.form['phone']
        address = request.form['address']
        dob = datetime.strptime(request.form['dob'],'%Y-%m-%d')
        covid19_status = request.form['infect']
        bloodgroup = request.form['blood']
        last_donated_date = request.form['last_donated_date']
        is_donor =  request.form['donor']
        today = date.today()
        donation_signedup_date = date.today()
        # print(dob, file=sys.stderr)
        age = today.year - dob.year - ((today.month, today.day) <
(dob.month, dob.day))
        conn = sql.connect('plasmadatabase.db',check_same_thread=False)
        check_user_sql = f"SELECT * FROM pd_user_data WHERE pdapp_username
= '{username}'"
        user_data = conn.execute(check_user_sql)
        account = user_data.fetchone()
        conn.close()
        if account:
            msg = 'Account already exists, please go ahead and login!'
        elif not re.match(r'[A-Za-z0-9]+', username):
            msg = 'name must contain only characters and numbers !'
        elif age <= 16:
            msg = 'must be an have age greater than 16 to register into
the Plasma Donation App !'
        else:
            conn =
sql.connect('plasmadatabase.db',check_same_thread=False)
            insert_sql = f"INSERT INTO pd_user_data VALUES ('{username}',
'{email}', '{phone}', '{address}', '{dob}' , '{covid19_status}')"
            conn.execute(insert_sql)
            conn.commit()
            conn.close()
            conn =
sql.connect('plasmadatabase.db',check_same_thread=False)
            insert_sql = f"INSERT INTO pd_app_user_creds VALUES
('{username}', '{password}')"
            conn.execute(insert_sql)
            conn.commit()
            conn.close()
            if is_donor == 'Yes':
                conn =
sql.connect('plasmadatabase.db',check_same_thread=False)
                insert_sql = f"INSERT INTO pd_donors VALUES ('{username}',
'{bloodgroup}','{donation_signedup_date}','{last_donated_date}')"
                conn.execute(insert_sql)
                conn.commit()
                conn.close()
            msg = 'You have successfully registered !'
 # sendmail(email,'Plasma donor App Registration','You are successfully
Registered {}!'.format(username))

    elif request.method == 'POST':
        msg = 'Please fill out the form !'
    return render_template('landingpage.html', msg = msg)
```



Welcome to Dhanalakshmi Srinivasan Engineering College Life Line

Sai

••••••

kothamasuvenkataratnasai@

4204094029

Chennai

Date of Birth: 06/22/2001

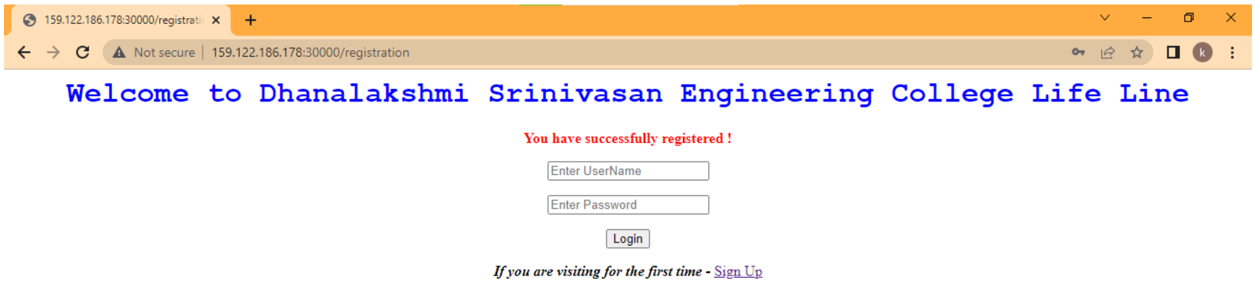
Uninfected

O Positive

Yes

Last Donation Date: 09/20/2020

Register


159.122.186.178:30000/registrati
Not secure | 159.122.186.178:30000/registration

Welcome to Dhanalakshmi Srinivasan Engineering College Life Line

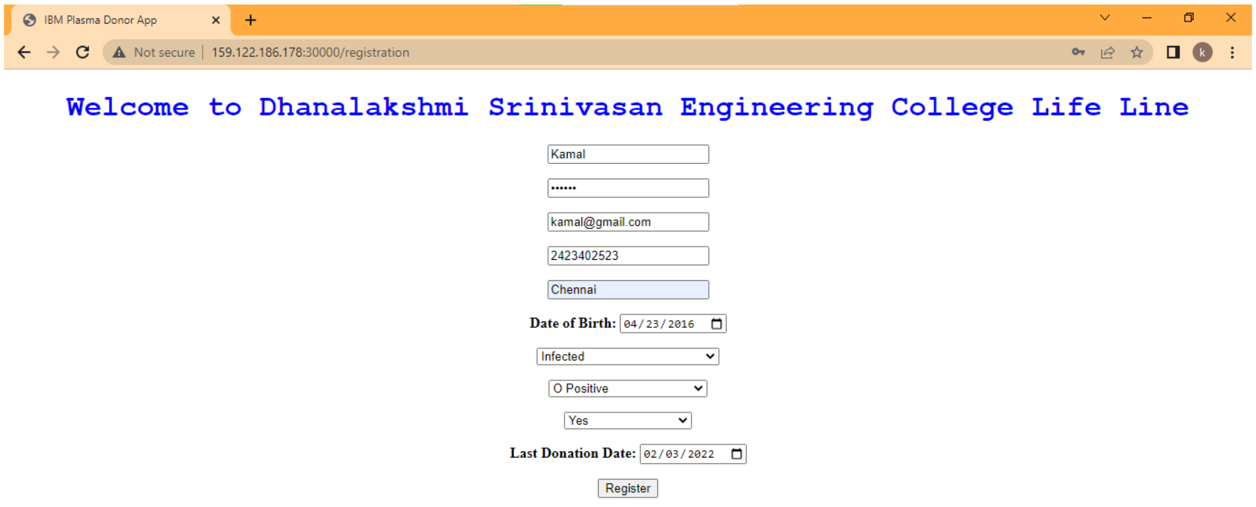
You have successfully registered !

Enter UserName

Enter Password

Login

If you are visiting for the first time - Sign Up



Welcome to Dhanalakshmi Srinivasan Engineering College Life Line

Kamal

••••••

kamal@gmail.com

2423402523

Chennai

Date of Birth: 04/23/2016

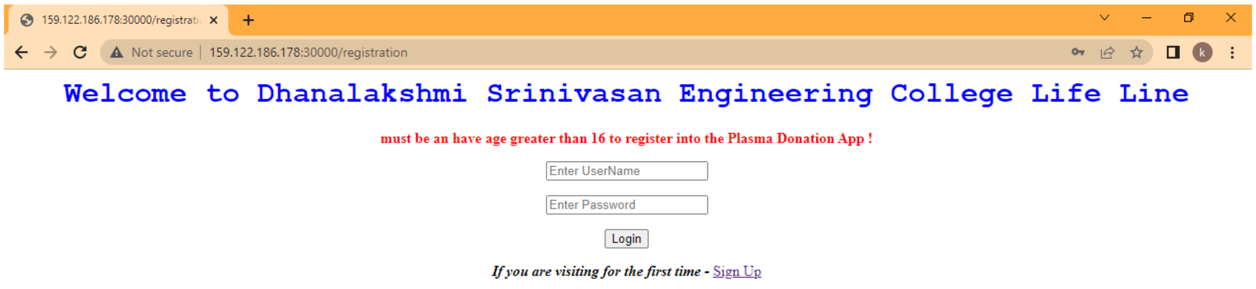
Infected

O Positive

Yes

Last Donation Date: 02/03/2022

Register


159.122.186.178:30000/registrati
Not secure | 159.122.186.178:30000/registration

Welcome to Dhanalakshmi Srinivasan Engineering College Life Line

must be an have age greater than 16 to register into the Plasma Donation App !

Enter UserName

Enter Password

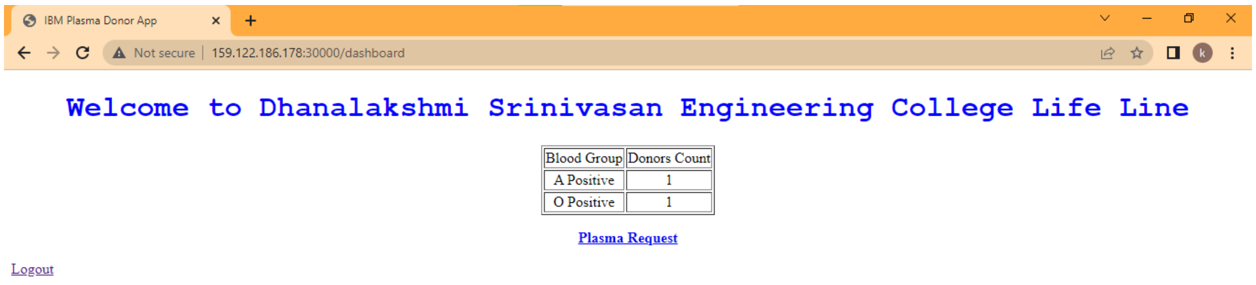
Login

If you are visiting for the first time - Sign Up

7.2 Dashboard Page :

CODE :

```
@app.route('/dashboard')
def dashboard():
    if session['loggedin'] == True:
        conn = sql.connect('plasmadatabase.db',check_same_thread=False)
        donations_sql = "SELECT blood_group_With_RH,COUNT(*) Donors_Cnt
FROM pd_donors where last_donated_date >= CURRENT_DATE-180 GROUP BY
blood_group_With_RH"
        con = sql.connect("plasmadatabase.db")
        con.row_factory = sql.Row
        cur = con.cursor()
        cur.execute(donations_sql)
        rows = cur.fetchall();
        conn.close()
        return render_template('dashboard.html',rows = rows)
    else:
         msg = 'Please login!'
         return render_template('landingpage.html', msg = msg)
```



Welcome to Dhanalakshmi Srinivasan Engineering College Life Line

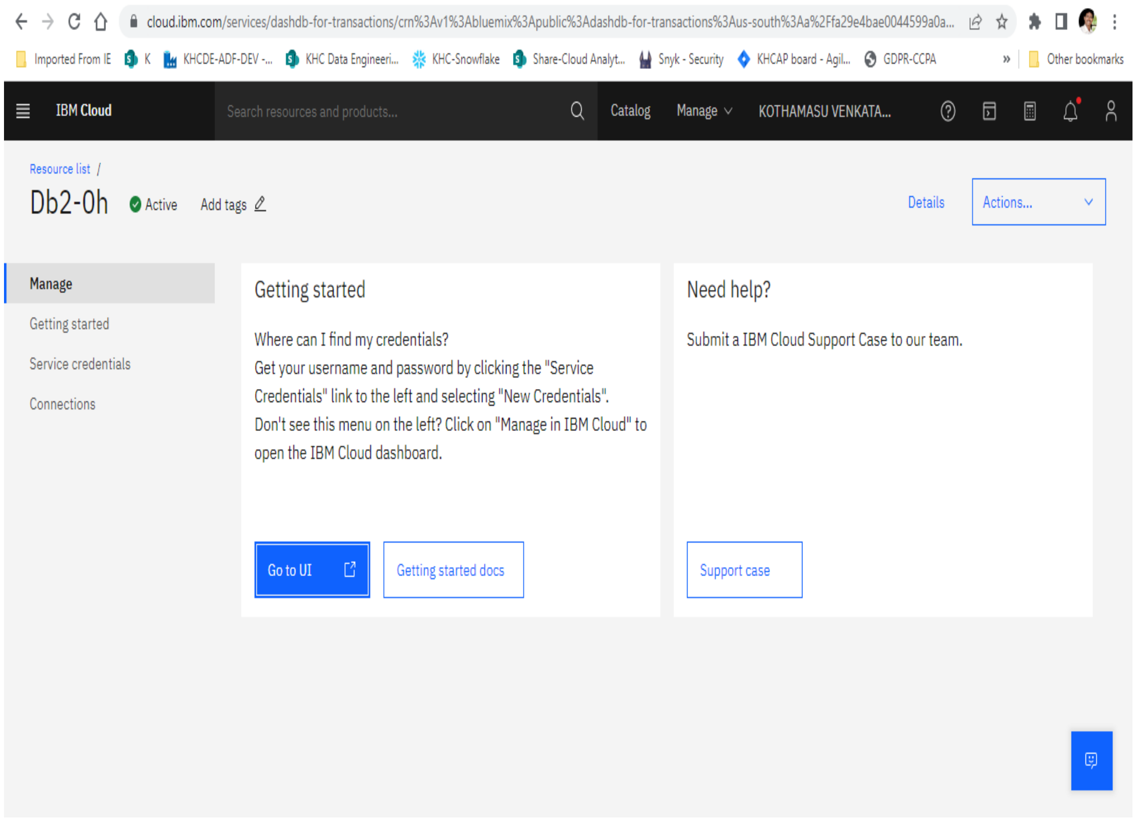
Blood Group	Donors Count
A Positive	1
O Positive	1

Plasma Request

Logout

7.3 Data base Schema:

DB 2 – Database:



cloud.ibm.com/services/dashdb-for-transactions/crn%3Av1%3Abluemix%3Apublic%3Adashdb-for-transactions%3Aus-south%3Aa%2Ffa29e4bae0044599a0a...

Imported From IE | K | KHCDE-ADF-DEV -... | KHC Data Engineeri... | KHC-Snowflake | Share-Cloud Analyt... | Snyk - Security | KHCAP board - Agil... | GDPR-CCPA | Other bookmarks

IBM Cloud | Search resources and products... | Catalog | Manage | KOTHAMASU VENKATA...

Resource list /

Db2-0h Active Add tags

Details | Actions...

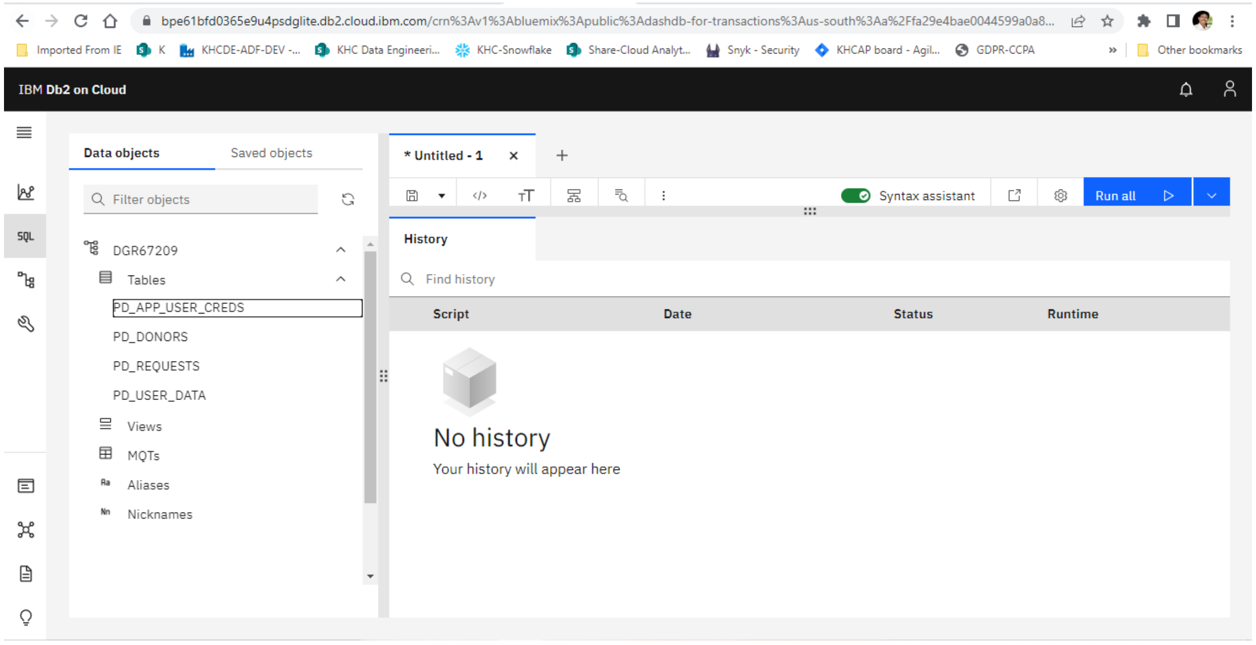
Getting started

Where can I find my credentials?
Get your username and password by clicking the "Service Credentials" link to the left and selecting "New Credentials".
Don't see this menu on the left? Click on "Manage in IBM Cloud" to open the IBM Cloud dashboard.

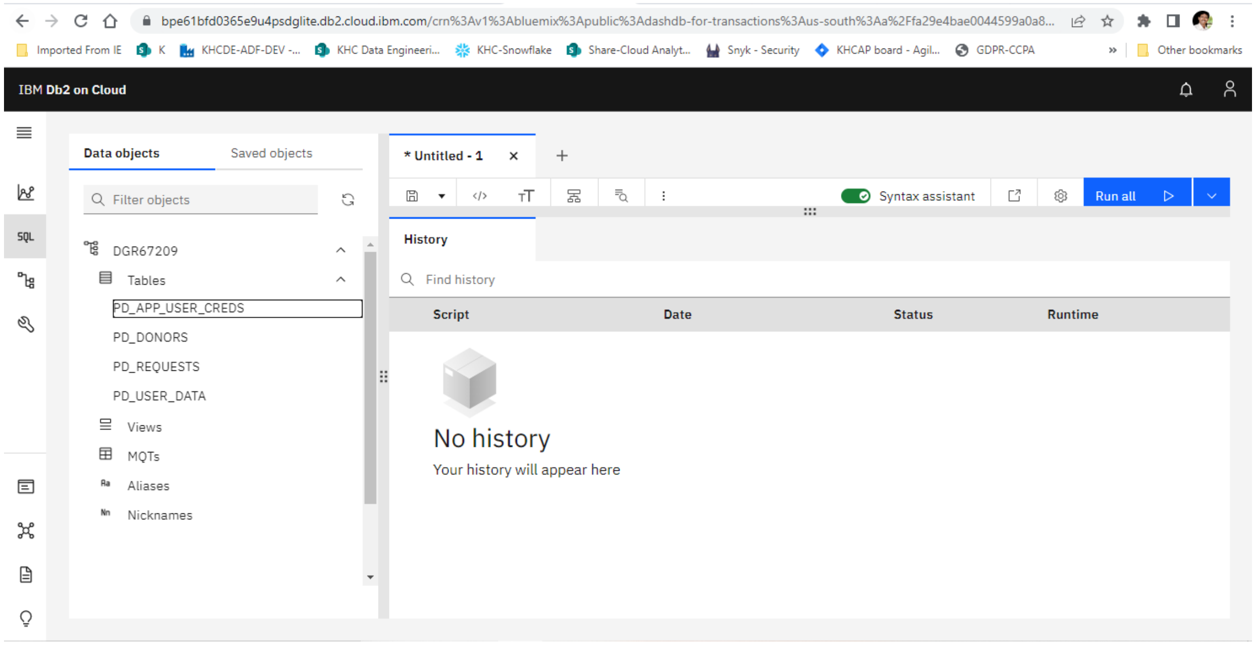
Need help?

Submit a IBM Cloud Support Case to our team.

DB 2 – Database Tables:



DB2 – Table - ***PD_USER_DATA:***



DB2 – Table - ***PD_APP_USER_CREDS:***

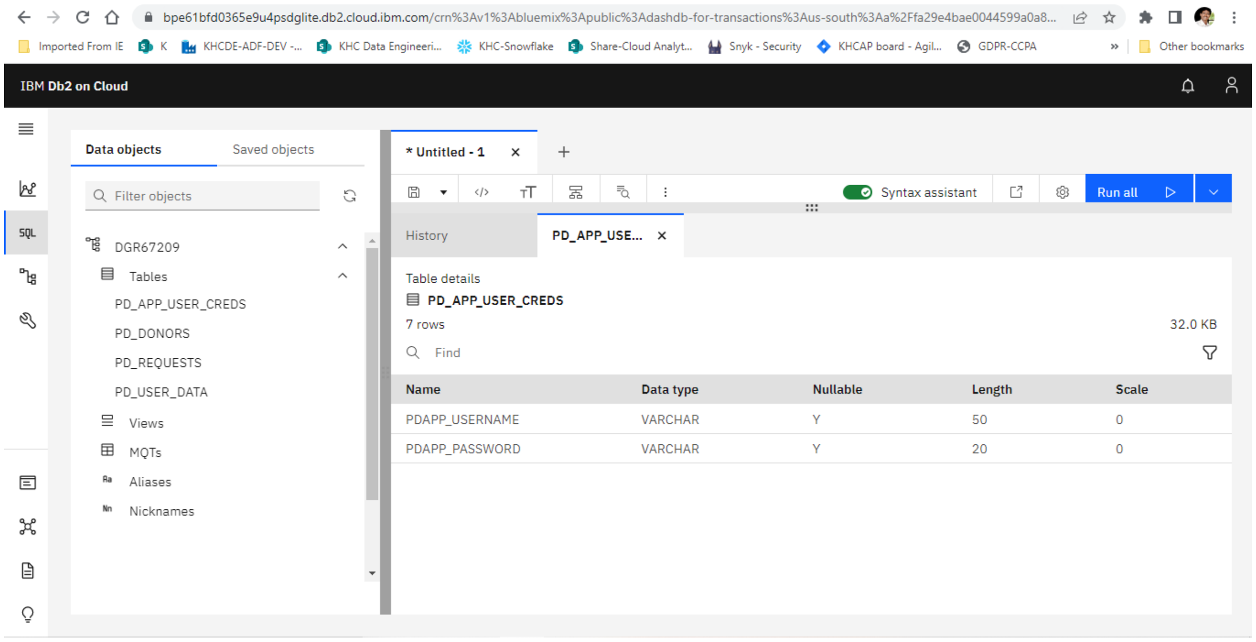


Table details

PD_APP_USER_CREDS

7 rows 32.0 KB

Name	Data type	Nullable	Length	Scale
PDAPP_USERNAME	VARCHAR	Y	50	0
PDAPP_PASSWORD	VARCHAR	Y	20	0

DB2 – Table - ***PD_DONORS:***

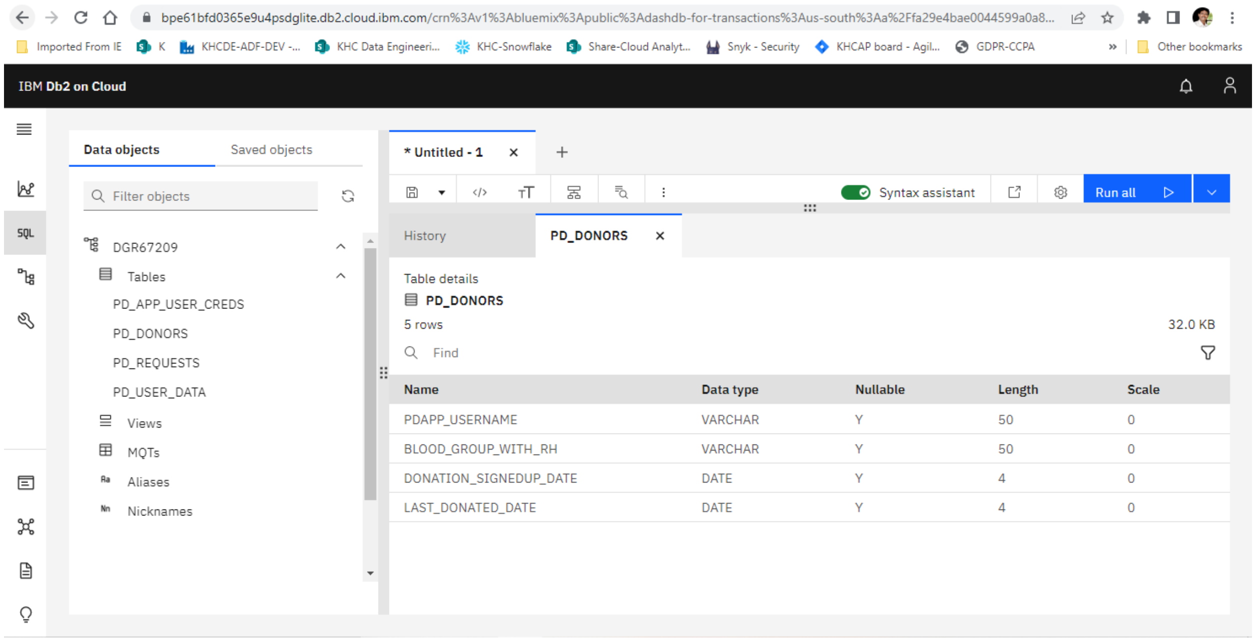


Table details

PD_DONORS

5 rows 32.0 KB

Name	Data type	Nullable	Length	Scale
PDAPP_USERNAME	VARCHAR	Y	50	0
BLOOD_GROUP_WITH_RH	VARCHAR	Y	50	0
DONATION_SIGNEDUP_DATE	DATE	Y	4	0
LAST_DONATED_DATE	DATE	Y	4	0

DB2 – Table - PD_REQUESTS:

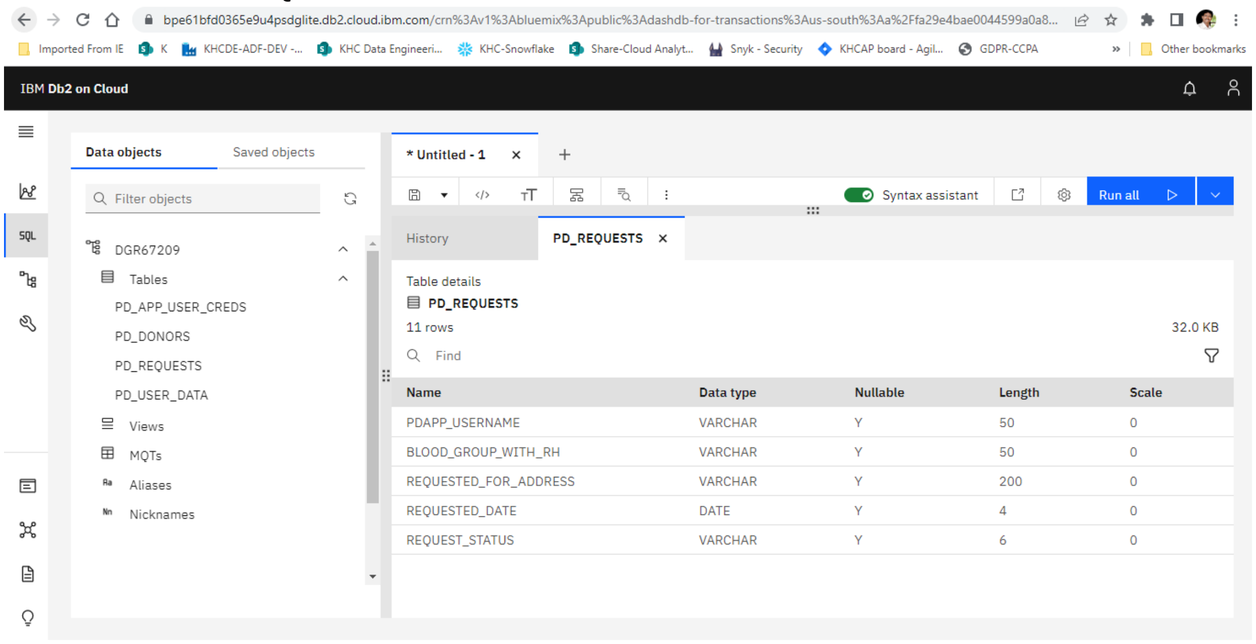
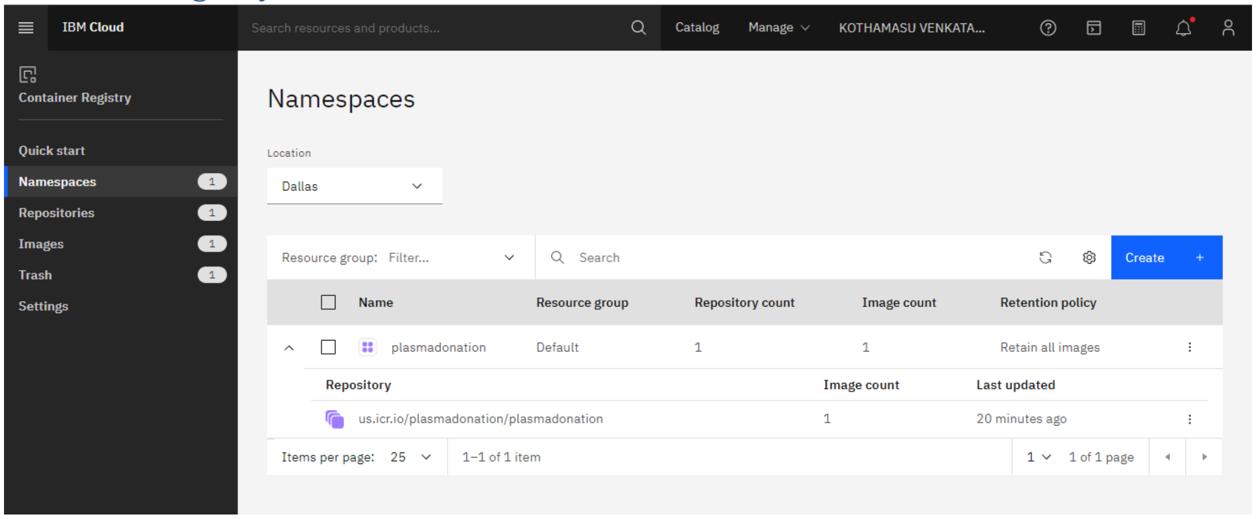


Table details
PD_REQUESTS
11 rows 32.0 KB

Name	Data type	Nullable	Length	Scale
PDAPP_USERNAME	VARCHAR	Y	50	0
BLOOD_GROUP_WITH_RH	VARCHAR	Y	50	0
REQUESTED_FOR_ADDRESS	VARCHAR	Y	200	0
REQUESTED_DATE	DATE	Y	4	0
REQUEST_STATUS	VARCHAR	Y	6	0

Container Registry:



Namespaces

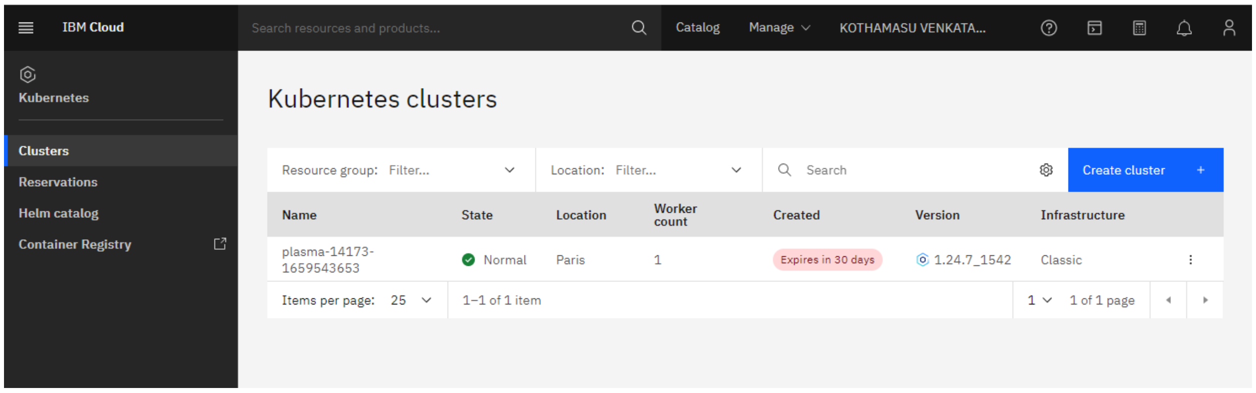
Location: Dallas

Name	Resource group	Repository count	Image count	Retention policy
plasmadonation	Default	1	1	Retain all images

Repository	Image count	Last updated
us.icr.io/plasmadonation/plasmadonation	1	20 minutes ago

Items per page: 25 1–1 of 1 item 1 1 of 1 page

Kubernetes Cluster:



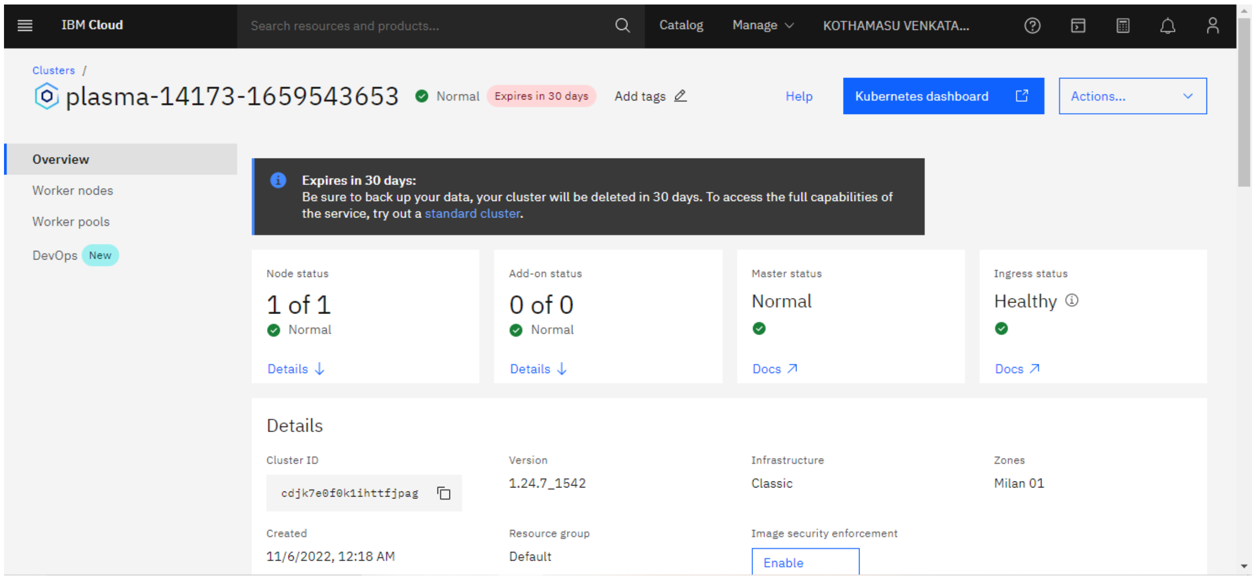
Kubernetes clusters

Resource group: Filter... | Location: Filter... | Search | Create cluster +

Name	State	Location	Worker count	Created	Version	Infrastructure
plasma-14173-1659543653	Normal	Paris	1	Expires in 30 days	1.24.7_1542	Classic

Items per page: 25 | 1–1 of 1 item | 1 | 1 of 1 page

Cluster Overview:



plasma-14173-1659543653 Normal Expires in 30 days Add tags

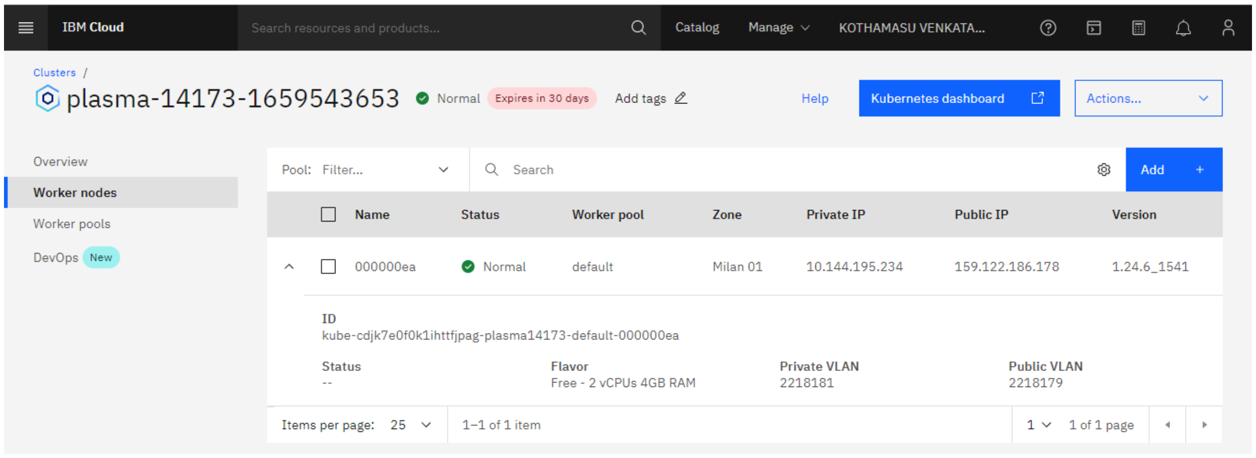
Expires in 30 days:
Be sure to back up your data, your cluster will be deleted in 30 days. To access the full capabilities of the service, try out a standard cluster.

Node status: 1 of 1 Normal — Details
Add-on status: 0 of 0 Normal — Details
Master status: Normal — Docs
Ingress status: Healthy — Docs

Details

Cluster ID: cdjk7e0f0k1ihttfjpag
Version: 1.24.7_1542
Infrastructure: Classic
Zones: Milan 01
Created: 11/6/2022, 12:18 AM
Resource group: Default
Image security enforcement: Enable

Worker Nodes:



plasma-14173-1659543653 Normal Expires in 30 days Add tags

Pool: Filter... | Search | Add +

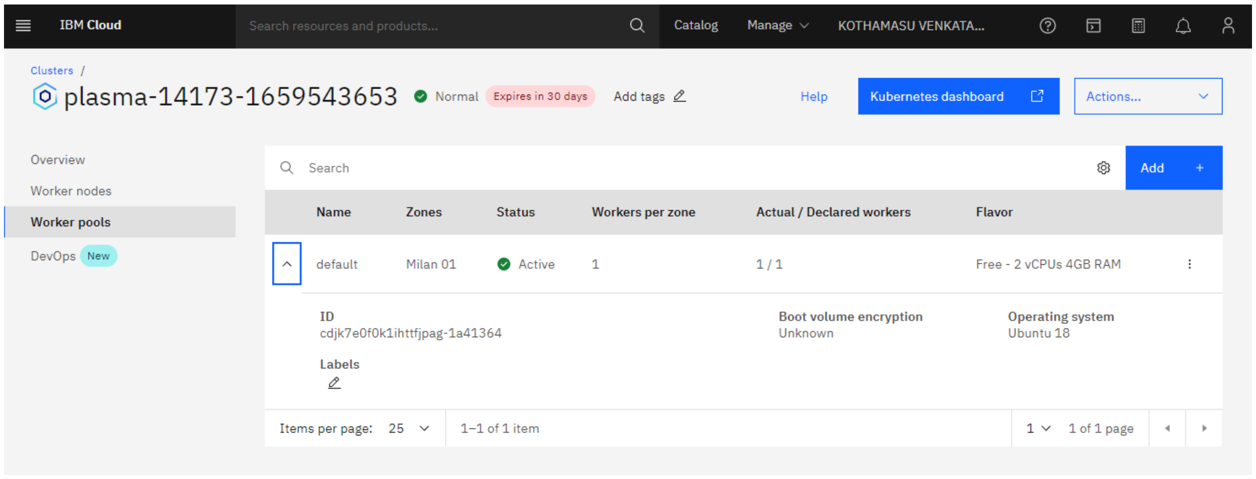
Name	Status	Worker pool	Zone	Private IP	Public IP	Version
000000ea	Normal	default	Milan 01	10.144.195.234	159.122.186.178	1.24.6_1541

ID
kube-cdjk7e0f0k1ihttfjpag-plasma14173-default-000000ea

Status	Flavor	Private VLAN	Public VLAN
--	Free - 2 vCPUs 4GB RAM	2218181	2218179

Items per page: 25 | 1–1 of 1 item | 1 | 1 of 1 page

Worker Pods:



Name	Zones	Status	Workers per zone	Actual / Declared workers	Flavor
default	Milan 01	Active	1	1 / 1	Free - 2 vCPUs 4GB RAM

ID
cdjk7e0f0k1ihttfjpag-1a41364

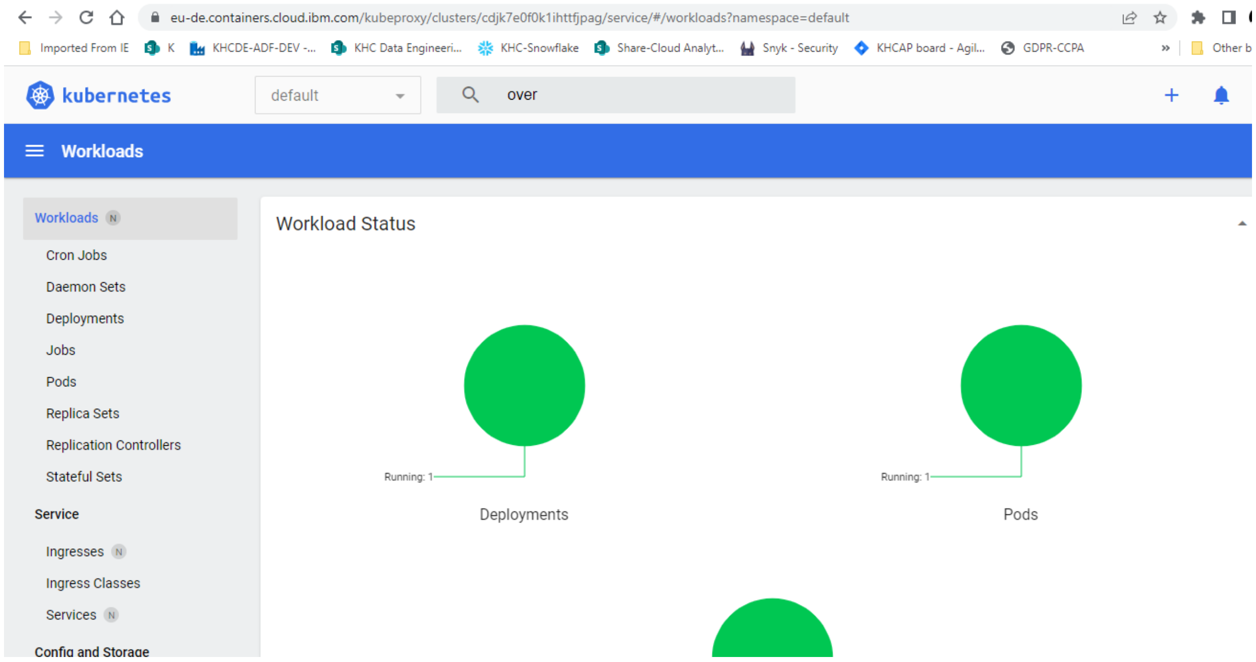
Boot volume encryption
Unknown

Operating system
Ubuntu 18

Labels

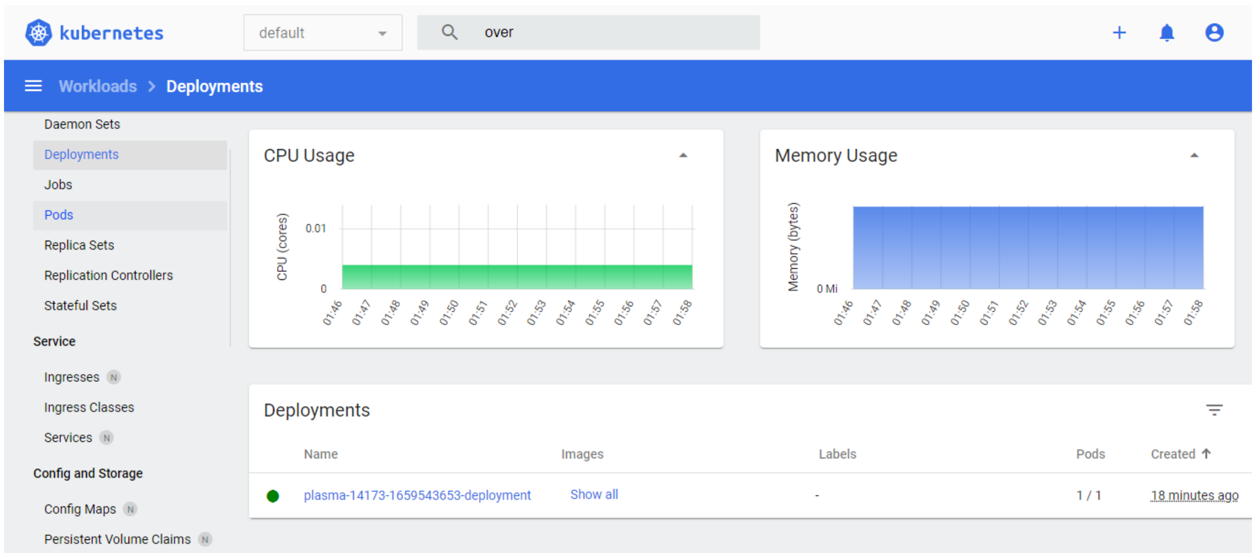
Items per page: 25 1–1 of 1 item 1 1 of 1 page

Kubernetes Dashboard:



eu-de.containers.cloud.ibm.com/kubeproxy/clusters/cdjk7e0f0k1ihttfjpag/service/#/workloads?namespace=default

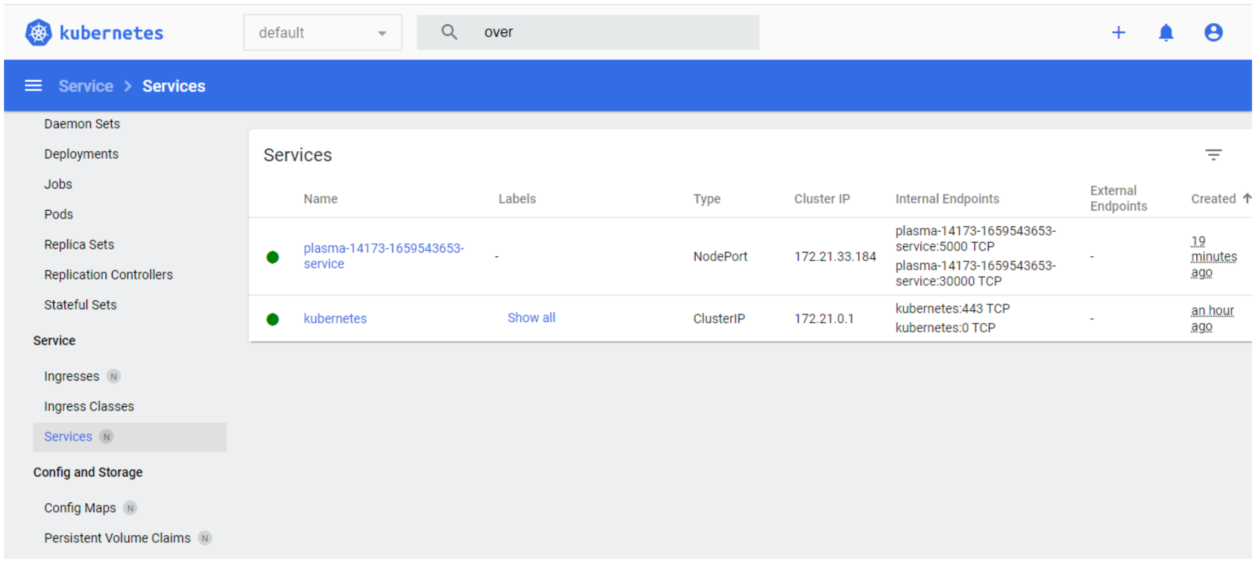
Deployment:



Deployments

	Name	Images	Labels	Pods	Created ↑
●	plasma-14173-1659543653-deployment	Show all	-	1 / 1	18 minutes ago

Service:



Services

	Name	Labels	Type	Cluster IP	Internal Endpoints	External Endpoints	Created ↑
●	plasma-14173-1659543653-service	-	NodePort	172.21.33.184	plasma-14173-1659543653-service:5000 TCP plasma-14173-1659543653-service:30000 TCP	-	19 minutes ago
●	kubernetes	Show all	ClusterIP	172.21.0.1	kubernetes:443 TCP kubernetes:0 TCP	-	an hour ago

SQL_LITE3:

CODE :

```
# Call SQL Lite DB to setup the DB and Tables
sql_lite_db()

# Connect to SQL Lite
conn = sql.connect('plasmadatabase.db',check_same_thread=False)
sql_query = """SELECT name FROM sqlite_master WHERE type='table';"""
cursor = conn.cursor()
cursor.execute(sql_query)
conn.close()
# print(cursor.fetchall(), file=sys.stderr)
import sqlite3

def sql_lite_db():
    # Create DB if not exists and Connect to the
DB
    conn = sqlite3.connect('plasmadatabase.db')

    # Table DDL Scripts
    create_tables =  ['create table  if not
exists pd_user_data ( pdapp_username TEXT ,email
TEXT ,phone TEXT ,user_addess TEXT ,dob date,
covid19_status TEXT );',
    'create table  if not exists pd_donors
(pdapp_username TEXT ,blood_group_With_RH TEXT
,donation_signedup_date date ,last_donated_date
date );',
    'create table  if not exists pd_requests (
pdapp_username TEXT ,blood_group_With_RH
TEXT,requested_for_address TEXT ,requested_date
date ,request_status TEXT);',
    'create table  if not exists
pd_app_user_creds ( pdapp_username TEXT
,pdapp_password TEXT );']

    # Inset Data
    insert_data = [
    "Insert into pd_requests values
('srinu57','A RhD positive (A+)','H.No: 34
Mambalam Chennai 600023','2022-10-
6','Open');","Insert into pd_requests values
('srinu57','AB RhD negative (AB-)','H.No: 100
Poondamalli Chennai 600025','2022-10-
4','Open');","Insert into pd_requests values
('suresh120','B RhD negative (B-)','H.No: 198
Koembedu Chennai 600037','2022-10-
4','Open');","Insert into pd_requests values
('suresh120','A RhD negative (A-)','H.No: 120 T
Narag Chennai 600039','2022-10-
6','Open');","Insert into pd_requests values
('sitar975','O RhD negative (O-)','H.No: 228 T
Narag Chennai 600016','2022-10-
7','Open');","Insert into pd_requests values
('balaji23','O RhD negative (O-)','H.No: 114
Poondamalli Chennai 600018','2022-10-
7','Open');","Insert into pd_requests values
```

```
('mahesh01','O RhD negative (O-)','H.No: 56
Koembedu Chennai 600057','2022-10-
1','Closed');","Insert into pd_requests values
('balaji23','A RhD negative (A-)','H.No: 138 T
Narag Chennai 600004','2022-10-
6','Open');","Insert into pd_requests values
('mahesh01','A RhD positive (A+)','H.No: 234 T
Narag Chennai 600033','2022-10-
8','Open');","Insert into pd_requests values
('sitar975','B RhD negative (B-)','H.No: 117 T
Narag Chennai 600067','2022-10-
7','Closed');","Insert into pd_requests values
('sitar975','AB RhD negative (AB-)','H.No: 71
Mambalam Chennai 600064','2022-10-
8','Open');","","Insert into pd_donors
values('mahesh01','AB RhD negative (AB-)','2022-
10-27','2020-9-13');","Insert into pd_donors
values('suresh120','AB RhD negative (AB-
)','2022-10-11','2022-10-26');","Insert into
pd_donors values('balaji23','O RhD positive
(O+)','2022-10-9','2022-9-15');","Insert into
pd_donors values('srinu57','AB RhD positive
(AB+)','2022-10-11','2020-10-12');","Insert into
pd_donors values('balu76','A RhD positive
(A+)','2022-11-1','2019-1-5');","","insert into
pd_app_user_creds values
('mahesh01','VQ300A');","insert into
```

```
pd_app_user_creds values
('suresh120','NI446K');","insert into
pd_app_user_creds values
('balaji23','RF477R');","insert into
pd_app_user_creds values
('srinu57','WD546Z');","insert into
pd_app_user_creds values
('balu76','JB481O');","insert into
pd_app_user_creds values
('sitar975','PL840Q');","insert into
pd_app_user_creds values
('hafeez12','ZU563A');","","Insert into
pd_user_data Values
('mahesh01','mahesh01@Yahoo.com','974-744-
4068','H.No: 78 Mambalam Chennai 600077','1997-
12-22');","Insert into pd_user_data Values
('suresh120','suresh120@gmail.com','886-540-
7410','H.No: 53 Koembedu Chennai 600095','1980-
5-22');","Insert into pd_user_data Values
('balaji23','balaji23@gmail.com','763-664-
7317','H.No: 123 Mambalam Chennai 600017','1992-
1-10');","Insert into pd_user_data Values
('srinu57','srinu57@live.com','771-396-
8496','H.No: 230 T Narag Chennai 600087','1988-
11-22');","Insert into pd_user_data Values
('balu76','balu76@live.com','976-159-
2142','H.No: 24 T Narag Chennai 600021','1984-
10-23');","Insert into pd_user_data Values
('sitar975','sitar975@live.com','710-181-
9979','H.No: 178 Koembedu Chennai 600004','1997-
8-16');","Insert into pd_user_data Values
('hafeez12','hafeez12@hotmail.com','844-148-
2828','H.No: 66 Koembedu Chennai 600037','1988-
11-10');"]

    # Create tables
    for ddl in create_tables:
        conn.execute(ddl)

    # Insert Data into table for tetsing
    # for dml in insert_data:
        # conn.execute(dml)

    # Close Connection
    conn.close()
```

8 TESTING :

8.1 Test Cases:

A test case has components that describe input, action and an expected response, in order to determine if a feature of an application is working correctly. A test case is a set of instructions on "HOW" to validate a particular test objective/target, which when followed will tell us if the expected behavior of the system is satisfied or not.

Characteristics of a good test case:

- Accurate: Exacts the purpose.
- Economical: No unnecessary steps or words.
- Traceable: Capable of being traced to requirements.
- Repeatable: Can be used to perform the test over and over.
- Reusable: Can be reused if necessary.

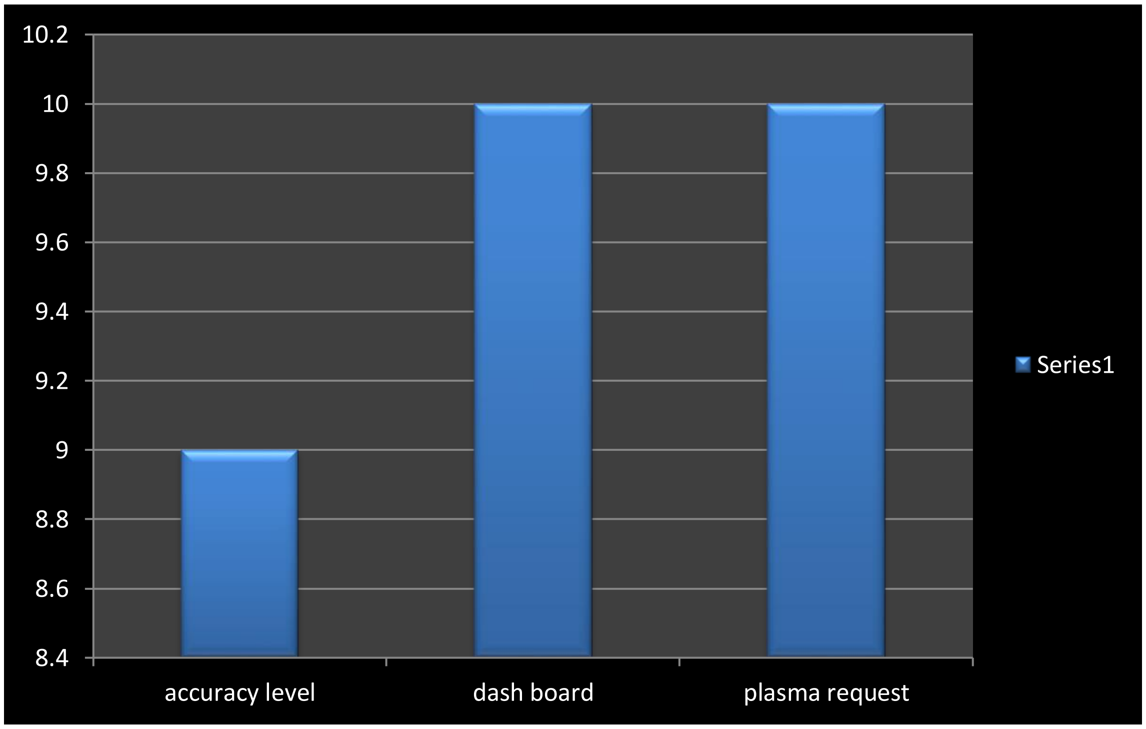
S.NO	**Scenario**	**Input**	**Excepted output**	**Actual output**
1	User login	User name and password	Login	Login success.
2.	Sign up	User should register by their deatails	Login to landing page	Login success
2	Search Plasma	Show plasma donar list	Request for plasma	User details are stored in a database.
3	Plasma request	Asking request to donar based on location	Get location of the donar to be donate	Details are stored in a database.

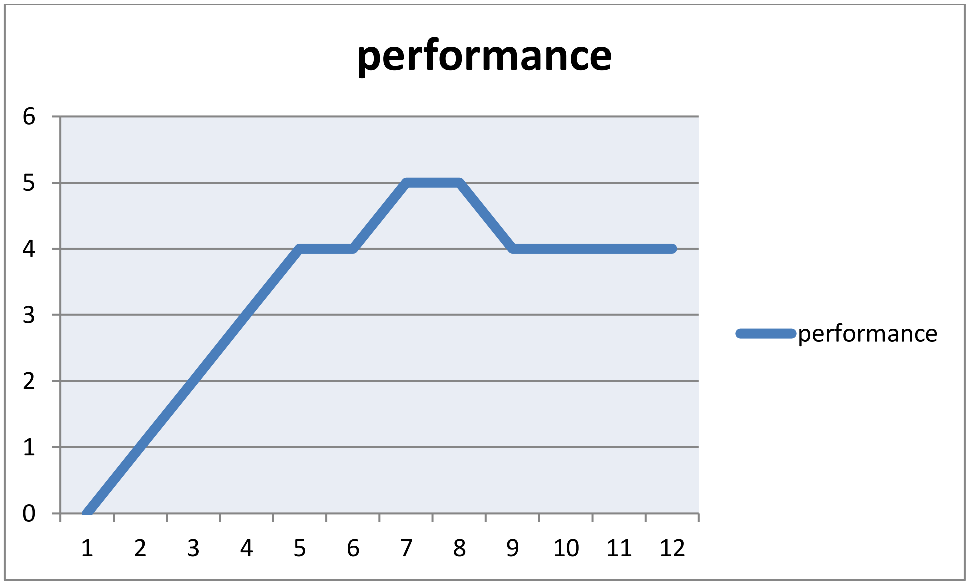
8.2 User Acceptance Testing :

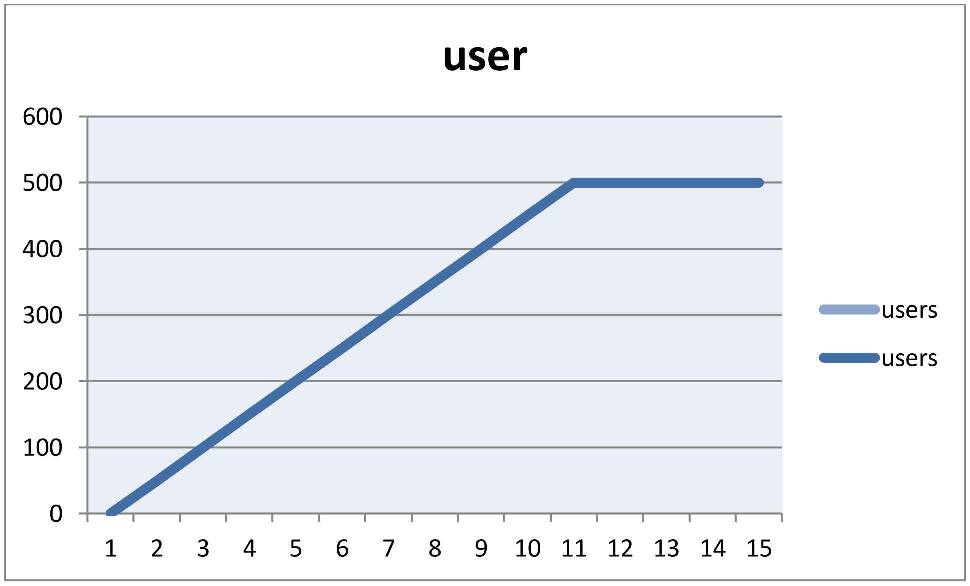
This sort of testing is carried out by users, clients, or other authorised bodies to identify the requirements and operational procedures of an application or piece of software. The most crucial stage of testing is acceptance testing since it determines whether or not the customer will accept the application or programmer. It could entail the application's U.I., performance, usability, and usefulness. It is also referred to as end-user testing, operational acceptance testing, and user acceptance testing (UAT).

9 RESULTS :

9.1 Performance Metrics :




performance
6
5
4
3
2
1
0
1
2
3
4
5
6
7
8
9
10
11
12
performance


user
600
500
400
300
200
100
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
users
users

10 ADVANTAGES & DISADVANTAGES:

ADVANTAGES :

- It is a user-friendly application.
- It will help people to find plasma easily.
- Simple User Interface
- It alleviates the burden of coordinator to manage Users and resources easily.
- Compared to all other mobile applications, it incorporates provisions for Plasma and mother's milk donation.
- Attracts more, number of users as it is available in the form of Mobile application instead of What's app group.
- Usage of this application will greatly reduce time in selecting the right donor.

DISADVANTAGES :

- It cannot auto verify user genuineness.
- It requires an active internet connection.
- Due to some wrong location the application will get confused

11 CONCLUSION :

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

Plamsa Application can be developed to further improve user accessibility via integrating this application with various social networks application program interfaces (APIs). Consequently, users can login and sign up using various social networks. This would increase number of donors and enhances the process of plasma donation.

User interface (UI) can be improved in future to accommodate global audience by supporting different languages across countries. Data scraping can be done from different social networks and can be shown in the plasmaRequest Feeds. Appointments can be synchronized with Google and Outlook calendars for the ease of users.

Plasma application provides a reliable platform to connect local blood and plasma donors with patients. plasma creates a communication channel through authenticated clinics whenever a patient needs blood and as well as plasma donation. It is a useful tool to find compatible plasma donors who can receive plasma request posts in their local area. Clinics can use this web application to maintain the plasma donation activity. Future improvement of the plasma is explained.

Keywords: Mobile and web application, m-health, e-health, application, plasma donation

12 FUTURE SCOPE:

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

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

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

13 APPENDIX:

Source Code:

App.py:

```
from flask import Flask, render_template, request, session,
redirect, url_for
from sql_lite_db import sql_lite_db
import sqlite3 as sql
from datetime import date,datetime
import sys,re
from distutils.log import debug
from sendgridmail import sendmail
from dotenv import load_dotenv

#Initialize the flask app
app = Flask(__name__)
app.secret_key = 'a'

# Call SQL Lite DB to setup the DB and Tables
sql_lite_db()

# Connect to SQL Lite
conn = sql.connect('plasmadatabase.db',check_same_thread=False)
sql_query = """SELECT name FROM sqlite_master WHERE
type='table';"""
cursor = conn.cursor()
cursor.execute(sql_query)
conn.close()
# print(cursor.fetchall(), file=sys.stderr)

@app.route('/')
def home():
   return render_template('landingpage.html')

@app.route('/loginpage',methods=['GET', 'POST'])
def loginpage():
    global userid
    msg = ''
    if request.method == 'POST' :
        username = request.form['username'].lower()
        password = request.form['password']
        conn = 
sql.connect('plasmadatabase.db',check_same_thread=False)
        check_user_sql = f"SELECT pdapp_username FROM 
pd_app_user_creds WHERE pdapp_username='{username}'"
        user_data = conn.execute(check_user_sql)
        account = user_data.fetchone()
        # print (account)
        if account:
            check_user_pw_sql = f"SELECT * FROM pd_app_user_creds 
WHERE pdapp_username='{username}' AND 
pdapp_password='{password}'"
            conn = 
sql.connect('plasmadatabase.db',check_same_thread=False)
            user_pw_data = conn.execute(check_user_pw_sql)
            account_pw = user_pw_data.fetchone()
            conn.close()
            # print (account_pw)
            if account_pw:
                session['loggedin'] = True
                session['id'] = account[0]
                userid = account[0]
                session['username'] = account[0]
                msg = 'Logged in successfully !'
                #sendmail(account['email'],'Plasma donor App 
login','You are successfully logged in!')
                return redirect(url_for('dashboard'))
            else:
                msg = 'Logged in Failed, re-try with correct 
password !'
                #sendmail(account['email'],'Plasma donor App 
login','You are successfully logged in!')
```

```
        else:
            msg = 'User Not Found, Please Sign Up !'
    return render_template('landingpage.html', msg = msg)

@app.route('/registration')
def register():
    return render_template('register.html')

@app.route('/registration',methods=['GET', 'POST'])
def registration():
    msg = ''
    if request.method == 'POST' :
        username = request.form['username'].lower()
        password = request.form['password']
        email = request.form['email']
        phone = request.form['phone']
        address = request.form['address']
        dob = datetime.strptime(request.form['dob'],'%Y-%m-%d')
        covid19_status = request.form['infect']
        bloodgroup = request.form['blood']
        last_donated_date = request.form['last_donated_date']
        is_donor =  request.form['donor']
        today = date.today()
        donation_signedup_date = date.today()
        # print(dob, file=sys.stderr)
        age = today.year - dob.year - ((today.month, today.day) <
(dob.month, dob.day))
        conn =
sql.connect('plasmadatabase.db',check_same_thread=False)
        check_user_sql = f"SELECT * FROM pd_user_data WHERE
pdapp_username = '{username}'"
        user_data = conn.execute(check_user_sql)
        account = user_data.fetchone()
        conn.close()
        if account:
            msg = 'Account already exists, please go ahead and
login!'
        elif not re.match(r'[A-Za-z0-9]+', username):
            msg = 'name must contain only characters and numbers
!'
        elif age <= 16:
            msg = 'must be an have age greater than 16 to
register into the Plasma Donation App !'
        else:
            conn =
sql.connect('plasmadatabase.db',check_same_thread=False)
            insert_sql = f"INSERT INTO pd_user_data VALUES
('{username}', '{email}', '{phone}', '{address}', '{dob}' ,
'{covid19_status}')"
            conn.execute(insert_sql)
            conn.commit()
            conn.close()
            conn =
sql.connect('plasmadatabase.db',check_same_thread=False)
            insert_sql = f"INSERT INTO pd_app_user_creds VALUES
('{username}', '{password}')"
            conn.execute(insert_sql)
            conn.commit()
            conn.close()
            if is_donor == 'Yes':
                conn =
sql.connect('plasmadatabase.db',check_same_thread=False)
                insert_sql = f"INSERT INTO pd_donors VALUES
('{username}',
'{bloodgroup}','{donation_signedup_date}','{last_donated_date}')"
                conn.execute(insert_sql)
                conn.commit()
                conn.close()
            msg = 'You have successfully registered !'
            # sendmail(email,'Plasma donor App Registration','You
are successfully Registered {}!'.format(username))
```

```
    elif request.method == 'POST':
        msg = 'Please fill out the form !'
    return render_template('landingpage.html', msg = msg)

@app.route('/dashboard')
def dashboard():
    if session['loggedin'] == True:
       conn =
sql.connect('plasmadatabase.db',check_same_thread=False)
       donations_sql = "SELECT blood_group_With_RH,COUNT(*)
Donors_Cnt FROM pd_donors where last_donated_date >=
CURRENT_DATE-180 GROUP BY blood_group_With_RH"
       con = sql.connect("plasmadatabase.db")
       con.row_factory = sql.Row
       cur = con.cursor()
       cur.execute(donations_sql)
       rows = cur.fetchall();
       conn.close()
       return render_template('dashboard.html',rows = rows)
    else:
        msg = 'Please login!'
        return render_template('landingpage.html', msg = msg)

@app.route('/plasmarequestform')
def plasmarequest():
    return render_template('plasmarequest.html')

@app.route('/plasmarequestform',methods=['GET', 'POST'])
def plasmarequestform():
    msg = ''
    if request.method == 'POST' :
       username = userid.lower()
       bloodgroup = request.form['blood']
       requested_for_address = request.form['address']
       requestdate = date.today()
```

```
        request_status = 'Open'
        conn = 
sql.connect('plasmadatabase.db',check_same_thread=False)
        insert_sql = f"INSERT INTO pd_requests VALUES 
('{username}', '{bloodgroup}', '{requested_for_address}', 
'{requestdate}','{request_status}')"
        conn.execute(insert_sql)
        conn.commit()
        conn.close()
        msg= 'Request Placed'
        return render_template('plasmarequest.html', msg = msg)
    else:
         msg = 'Please login!'
         return render_template('landingpage.html', msg = msg)

@app.route('/logout')
def logout():
   session.pop('loggedin', None)
   session.pop('id', None)
   session.pop('username', None)
   return render_template('landingpage.html')

if __name__ == '__main__':
   app.run(host='0.0.0.0',debug='TRUE')
```

sendgrid mail.py

```
# using SendGrid's Python Library
# https://github.com/sendgrid/sendgrid-python
import os
from dotenv import load_dotenv

load_dotenv()
from sendgrid import SendGridAPIClient
from sendgrid.helpers.mail import Mail

def sendmail(usermail,subject,content):
    message =
Mail(from_email='kothamasuvenkataratnasai@gmail.com',to_
emails=usermail,subject=subject,html_content='<strong>
{} </strong>'.format(content))
    try:
        sg =
SendGridAPIClient(os.getenv('SENDGRID_API_KEY'))
        response = sg.send(message)
        print(response.status_code)
        print(response.body)
        print(response.headers)
    except Exception as e:
        print(e.message)
```

sql lite db.py

```
import sqlite3

def sql_lite_db():
    # Create DB if not exists and Connect to the DB
    conn = sqlite3.connect('plasmadatabase.db')

    # Table DDL Scripts
    create_tables =  ['create table  if not exists
pd_user_data ( pdapp_username TEXT ,email TEXT ,phone
TEXT ,user_addess TEXT ,dob date, covid19_status TEXT
);',
    'create table  if not exists pd_donors
(pdapp_username TEXT ,blood_group_With_RH TEXT
,donation_signedup_date date ,last_donated_date date
);',
    'create table  if not exists pd_requests (
pdapp_username TEXT ,blood_group_With_RH
TEXT,requested_for_address TEXT ,requested_date date
,request_status TEXT);',
    'create table  if not exists pd_app_user_creds (
pdapp_username TEXT ,pdapp_password TEXT );']

    # Inset Data
    insert_data = [
    "Insert into pd_requests values ('srinu57','A RhD
positive (A+)','H.No: 34 Mambalam Chennai 600023','2022-
10-6','Open');","Insert into pd_requests values
('srinu57','AB RhD negative (AB-)','H.No: 100
Poondamalli Chennai 600025','2022-10-
4','Open');","Insert into pd_requests values
('suresh120','B RhD negative (B-)','H.No: 198 Koembedu
Chennai 600037','2022-10-4','Open');","Insert into
pd_requests values ('suresh120','A RhD negative (A-
)','H.No: 120 T Narag Chennai 600039','2022-10-
6','Open');","Insert into pd_requests values
('sitar975','O RhD negative (O-)','H.No: 228 T Narag
Chennai 600016','2022-10-7','Open');","Insert into
pd_requests values ('balaji23','O RhD negative (O-
)','H.No: 114 Poondamalli Chennai 600018','2022-10-
7','Open');","Insert into pd_requests values
('mahesh01','O RhD negative (O-)','H.No: 56 Koembedu
Chennai 600057','2022-10-1','Closed');","Insert into
```

```
pd_requests values ('balaji23','A RhD negative (A-
)','H.No: 138 T Narag Chennai 600004','2022-10-
6','Open');","Insert into pd_requests values
('mahesh01','A RhD positive (A+)','H.No: 234 T Narag
Chennai 600033','2022-10-8','Open');","Insert into
pd_requests values ('sitar975','B RhD negative (B-
)','H.No: 117 T Narag Chennai 600067','2022-10-
7','Closed');","Insert into pd_requests values
('sitar975','AB RhD negative (AB-)','H.No: 71 Mambalam
Chennai 600064','2022-10-8','Open');","","Insert into
pd_donors values('mahesh01','AB RhD negative (AB-
)','2022-10-27','2020-9-13');","Insert into pd_donors
values('suresh120','AB RhD negative (AB-)','2022-10-
11','2022-10-26');","Insert into pd_donors
values('balaji23','O RhD positive (O+)','2022-10-
9','2022-9-15');","Insert into pd_donors
values('srinu57','AB RhD positive (AB+)','2022-10-
11','2020-10-12');","Insert into pd_donors
values('balu76','A RhD positive (A+)','2022-11-1','2019-
1-5');","","insert into pd_app_user_creds values
('mahesh01','VQ300A');","insert into pd_app_user_creds
values ('suresh120','NI446K');","insert into
pd_app_user_creds values ('balaji23','RF477R');","insert
into pd_app_user_creds values
('srinu57','WD546Z');","insert into pd_app_user_creds
values ('balu76','JB481O');","insert into
pd_app_user_creds values ('sitar975','PL840Q');","insert
into pd_app_user_creds values
('hafeez12','ZU563A');","","Insert into pd_user_data
Values ('mahesh01','mahesh01@Yahoo.com','974-744-
4068','H.No: 78 Mambalam Chennai 600077','1997-12-
22');","Insert into pd_user_data Values
('suresh120','suresh120@gmail.com','886-540-7410','H.No:
53 Koembedu Chennai 600095','1980-5-22');","Insert into
pd_user_data Values
('balaji23','balaji23@gmail.com','763-664-7317','H.No:
123 Mambalam Chennai 600017','1992-1-10');","Insert into
pd_user_data Values ('srinu57','srinu57@live.com','771-
396-8496','H.No: 230 T Narag Chennai 600087','1988-11-
22');","Insert into pd_user_data Values
('balu76','balu76@live.com','976-159-2142','H.No: 24 T
Narag Chennai 600021','1984-10-23');","Insert into
pd_user_data Values
('sitar975','sitar975@live.com','710-181-9979','H.No:
178 Koembedu Chennai 600004','1997-8-16');","Insert into
pd_user_data Values
('hafeez12','hafeez12@hotmail.com','844-148-2828','H.No:
66 Koembedu Chennai 600037','1988-11-10');"]

    # Create tables
    for ddl in create_tables:
        conn.execute(ddl)

    # Insert Data into table for tetsing
    # for dml in insert_data:
        # conn.execute(dml)

    # Close Connection
    conn.close()
```

deployment.yaml :

```
apiVersion: apps/v1
 kind: Deployment
 metadata:
   name: plasma-14173-1659543653-deployment
 spec:
   replicas: 1
   selector:
     matchLabels:
       app: plasmadonationnode
   template:
     metadata:
       labels:
         app: plasmadonationnode
     spec:
       containers:
       - name: plasmadonationnode
         image: us.icr.io/plasmadonation/plasmadonation
         imagePullPolicy: Always
         ports:
         - containerPort: 5000
```

Service.yaml

```
apiVersion: v1
 kind: Service
 metadata:
   name: plasma-14173-1659543653-service
 spec:
```

```yaml
  type: NodePort
  ports:
  - name: appport
    port: 5000
    targetPort: 5000
    nodePort: 30000
  selector:
    app: plasmadonationnode
```

LANDING PAGE.HTML

```html
<html>
<head>
    <title></title>
</head>
<body>
<h1 style="text-align: center;"><span
style="color:#0000FF;"><span style="font-family:
&quot;Courier New&quot;, courier;">Welcome to Dhanalakshmi
Srinivasan Engineering College Life
Line</span></span></h1>

<p style="text-align: center;"><span
style="color:#FF0000;"><strong>{{ msg
}}</strong></span></p>

<div class="login"><!-- Main Input For Receiving Query to
our ML -->
<form action="{{ url_for('loginpage')}}" method="post">
<p style="text-align: center;"><input name="username"
placeholder="Enter UserName" required="required"
style="color:black" type="text" /></p>
```

```
<p style="text-align: center;"><input name="password"
placeholder="Enter Password" required="required"
style="color:black" type="password" /></p>

<p style="text-align: center;"><button class="btn btn-
primary btn-block btn-large"
type="submit">Login</button></p>
</form>

<p style="text-align: center;"><em><strong>If you are
visiting for the first time -</strong></em>&nbsp;<a
href="/registration" style="text-align: center;">Sign
Up</a></p>
</div>

</body>
</html>
```

REGISTER.HTML

```
<!DOCTYPE html>
<html><!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
    <meta charset="UTF-8" />
    <title>IBM Plasma Donor App</title>
    <link
href="https://fonts.googleapis.com/css?family=Pacifico"
rel="stylesheet" type="text/css" />
    <link
href="https://fonts.googleapis.com/css?family=Arimo"
rel="stylesheet" type="text/css" />
```

```
    <link
href="https://fonts.googleapis.com/css?family=Hind:300"
rel="stylesheet" type="text/css" />
    <link
href="https://fonts.googleapis.com/css?family=Open+Sans+Co
ndensed:300" rel="stylesheet" type="text/css" />
    <link href="{{ url_for('static', filename='style.css')
}}" rel="stylesheet" />
    <style type="text/css">.login{
top: 20%;
}
    </style>
</head>
<body>
<div class="header">
<h1 style="text-align: center;"><span
style="color:#0000FF;"><span style="font-family:
&quot;Courier New&quot;, courier;">Welcome to Dhanalakshmi
Srinivasan Engineering College Life
Line</span></span></h1>

<p style="text-align: center;"><span
style="color:#FF0000;"><strong>{{ msg
}}</strong></span></p>
</div>

<div class="login"><!-- Main Input For Receiving Query to
our ML -->
<form action="{{ url_for('registration')}}" method="post">
<p style="text-align: center;"><input name="username"
placeholder="Enter Your Name" required="required"
style="color:black" type="text" /></p>
```

```
<p style="text-align: center;"><input name="password"
placeholder="Enter Password" required="required"
style="color:black" type="password" /></p>

<p style="text-align: center;"><input name="email"
placeholder="Enter Email" required="required"
style="color:black" type="email" /></p>

<p style="text-align: center;"><input maxlength="10"
name="phone" placeholder="Enter 10-digit mobile number"
required="required" size="10" style="color:black"
type="number" /></p>

<p style="text-align: center;"><input name="address"
placeholder="Enter Your Address" required="required"
style="color:black" type="city" /></p>

<p style="text-align: center;"><strong>Date of
Birth:</strong>&nbsp;<input name="dob" placeholder="Enter
Date of Birth" required="required" style="color:black"
type="date" /></p>

<p style="text-align: center;"><select
name="infect"><option disabled="disabled"
selected="selected" value="select">Select COVID infection
status</option><option
value="infected">Infected</option><option
value="uninfected">Uninfected</option> </select></p>

<p style="text-align: center;"><select
name="blood"><option disabled="disabled"
```

```
selected="selected" value="select">Choose your blood
group</option><option value="O Positive">O
Positive</option><option value="A Positive">A
Positive</option><option value="B Positive">B
Positive</option><option value="AB Positive">AB
Positive</option><option value="O Negative">O
Negative</option><option value="A Negative">A
Negative</option><option value="B Negative">B
Negative</option><option value="AB Negative">AB
Negative</option> </select></p>

<p style="text-align: center;"><select
name="donor"><option disabled="disabled"
selected="selected" value="select">Want to be a
Donor</option><option value="Yes">Yes</option><option
value="No">No</option></select></p>

<p style="text-align: center;"><strong>Last Donation
Date:</strong><em><strong>&nbsp;</strong></em><input
name="last_donated_date" placeholder="Enter Last Donated
Date" required="required" style="color:black" type="date"
/></p>

<p style="text-align: center;"><button class="btn btn-
primary btn-block btn-large"
type="submit">Register</button></p>
</form>
</div>
</body>
</html>
```

DASH BOARD .HTML

```
<!DOCTYPE html>
<html><!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
    <meta charset="UTF-8" />
    <title>IBM Plasma Donor App</title>
    <link
href="https://fonts.googleapis.com/css?family=Pacifico"
rel="stylesheet" type="text/css" />
    <link
href="https://fonts.googleapis.com/css?family=Arimo"
rel="stylesheet" type="text/css" />
    <link
href="https://fonts.googleapis.com/css?family=Hind:300"
rel="stylesheet" type="text/css" />
    <link
href="https://fonts.googleapis.com/css?family=Open+Sans+Co
ndensed:300" rel="stylesheet" type="text/css" />
    <link href="{{ url_for('static', filename='style.css')
}}" rel="stylesheet" />
    <style type="text/css">.login{
top: 20%;
}
    </style>
</head>
<body>
<div class="header">
<h1 style="text-align: center;"><span
style="color:#0000FF;"><span style="font-family:
&quot;Courier New&quot;, courier;">Welcome to Dhanalakshmi
Srinivasan Engineering College Life
Line</span></span></h1>
</div>
```

```
<table align="center" border="1">
    <thead>
        <tr>
            <td style="text-align: center">Blood
Group</td>
            <td style="text-align: center">Donors
Count</td>
        </tr>
    </thead>
    {% for row in rows %}
        <tr>
            <td style="text-align:
center">{{row["blood_group_With_RH"]}}</td>
            <td style="text-align:
center">{{row["Donors_Cnt"]}}</td>
        </tr>
    {% endfor %}
</table>

<p style="text-align: center;"><strong><a
href="/plasmarequestform" style="text-align:
center;">Plasma Request</a></strong></p>
</div>
<p><a class="active" href="/logout">Logout</a></p>
</body>
</html>
```

PLASMA REQUEST.HTML

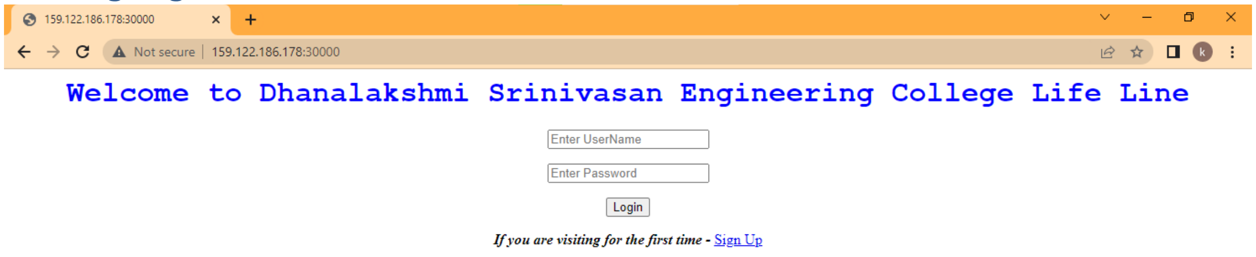
```
<!DOCTYPE html>
<html><!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
```

```
    <meta charset="UTF-8" />
    <title>IBM Plasma Donor App</title>
    <link
href="https://fonts.googleapis.com/css?family=Pacifico"
rel="stylesheet" type="text/css" />
    <link
href="https://fonts.googleapis.com/css?family=Arimo"
rel="stylesheet" type="text/css" />
    <link
href="https://fonts.googleapis.com/css?family=Hind:300"
rel="stylesheet" type="text/css" />
    <link
href="https://fonts.googleapis.com/css?family=Open+Sans+Co
ndensed:300" rel="stylesheet" type="text/css" />
    <link href="{{ url_for('static', filename='style.css')
}}" rel="stylesheet" />
    <style type="text/css">.login{
top: 20%;
}
    </style>
</head>
<body>
<div class="header">
<h1 style="text-align: center;"><span
style="color:#0000FF;"><span style="font-family:
&quot;Courier New&quot;, courier;">Welcome to Dhanalakshmi
Srinivasan Engineering College Life
Line</span></span></h1>

<p style="text-align: center;"><span
style="color:#FF0000;"><strong>{{ msg
}}</strong></span></p>
```

```
</div>
<div>
<form action="{{ url_for('plasmarequestform')}}"
method="post">

<p style="text-align: center;"><select
name="blood"><option disabled="disabled"
selected="selected" value="select">Choose your blood
group</option><option value="O Positive">O
Positive</option><option value="A Positive">A
Positive</option><option value="B Positive">B
Positive</option><option value="AB Positive">AB
Positive</option><option value="O Negative">O
Negative</option><option value="A Negative">A
Negative</option><option value="B Negative">B
Negative</option><option value="AB Negative">AB
Negative</option> </select></p>

<p style="text-align: center;"><input name="address"
placeholder="Enter Your Address" required="required"
style="color:black" type="city" /></p>

<p style="text-align: center;"><button class="btn btn-
primary btn-block btn-large" type="submit">Place the
Request</button></p>
</form>
</div>
<p><a class="active" href="/logout">Logout</a></p>
<p><a href="/dashboard">Go back to Dashboard</a></p>
</body>
</html>
```

Web App:

Landing Page:


159.122.186.178:30000

Not secure | 159.122.186.178:30000

Welcome to Dhanalakshmi Srinivasan Engineering College Life Line

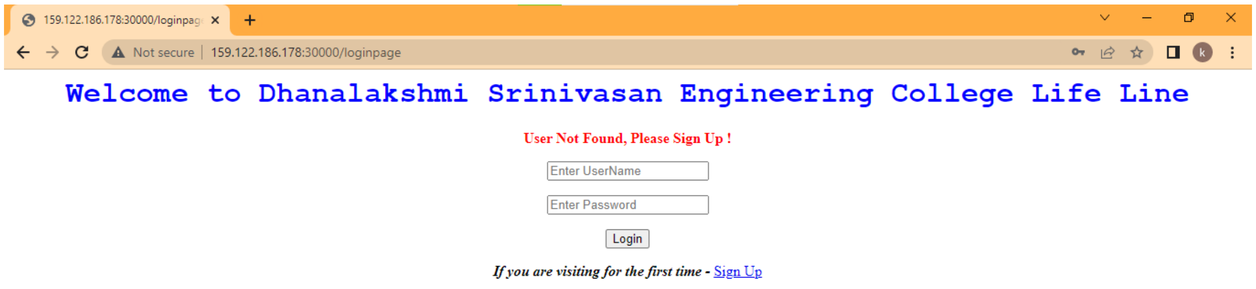
Enter UserName

Enter Password

Login

If you are visiting for the first time - Sign Up

User Validation:


159.122.186.178:30000/loginpage

Not secure | 159.122.186.178:30000/loginpage

Welcome to Dhanalakshmi Srinivasan Engineering College Life Line

User Not Found, Please Sign Up !

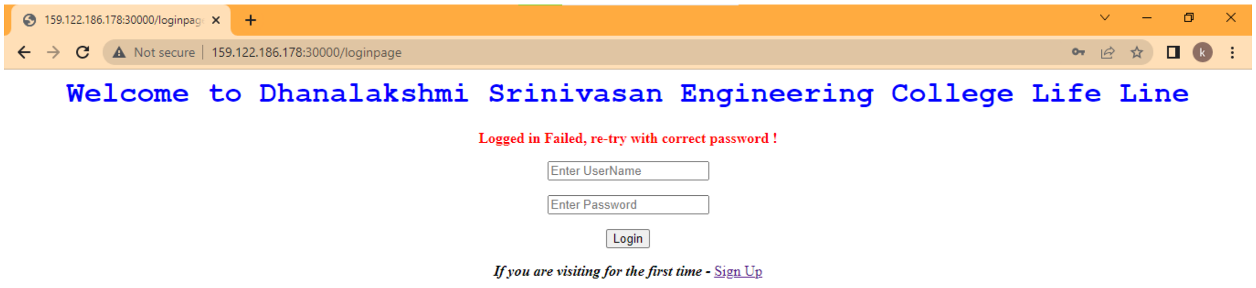
Enter UserName

Enter Password

Login

If you are visiting for the first time - Sign Up

User Password Validation:


159.122.186.178:30000/loginpage

Not secure | 159.122.186.178:30000/loginpage

Welcome to Dhanalakshmi Srinivasan Engineering College Life Line

Logged in Failed, re-try with correct password !

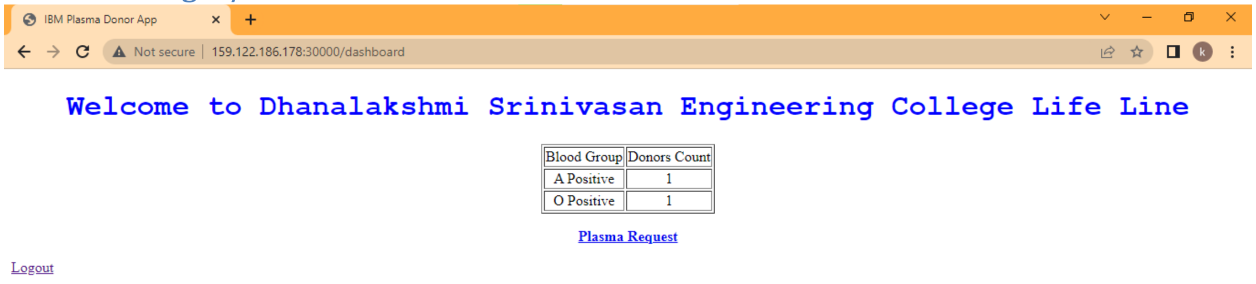
Enter UserName

Enter Password

Login

If you are visiting for the first time - Sign Up

Successful Login / Dashboard:


IBM Plasma Donor App

Not secure | 159.122.186.178:30000/dashboard

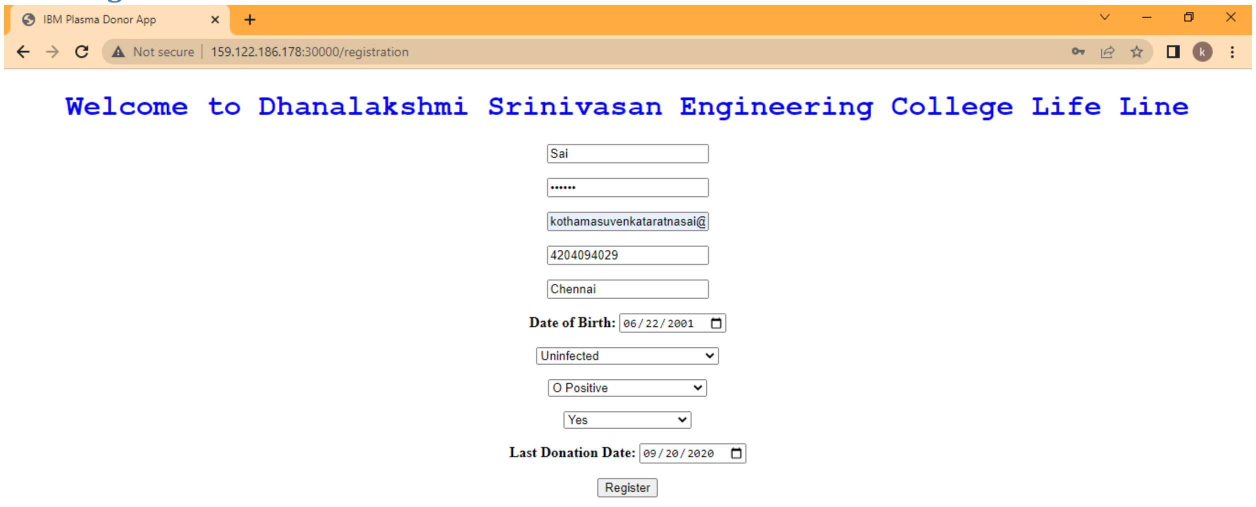
Welcome to Dhanalakshmi Srinivasan Engineering College Life Line

Blood Group	Donors Count
A Positive	1
O Positive	1

Plasma Request

Logout

User Registration:


IBM Plasma Donor App

Not secure | 159.122.186.178:30000/registration

Welcome to Dhanalakshmi Srinivasan Engineering College Life Line

Sai

••••••

kothamasuvenkataratnasai@

4204094029

Chennai

Date of Birth: 06/22/2001

Uninfected

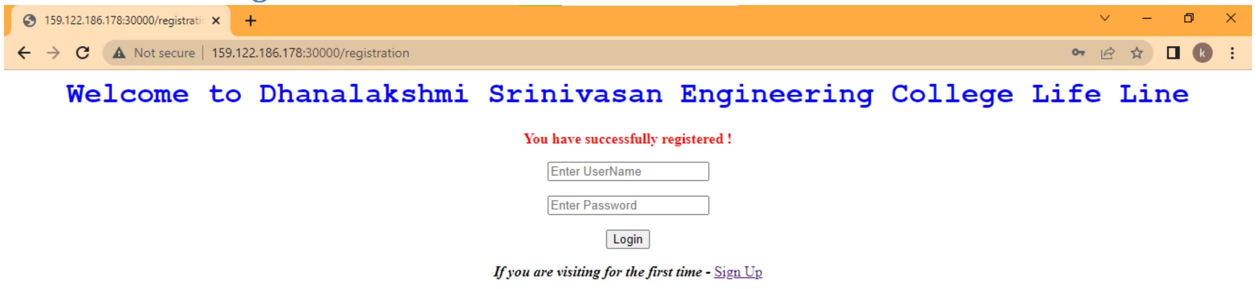
O Positive

Yes

Last Donation Date: 09/20/2020

Register

User Successful Registration:


159.122.186.178:30000/registrati

Not secure | 159.122.186.178:30000/registration

Welcome to Dhanalakshmi Srinivasan Engineering College Life Line

You have successfully registered !

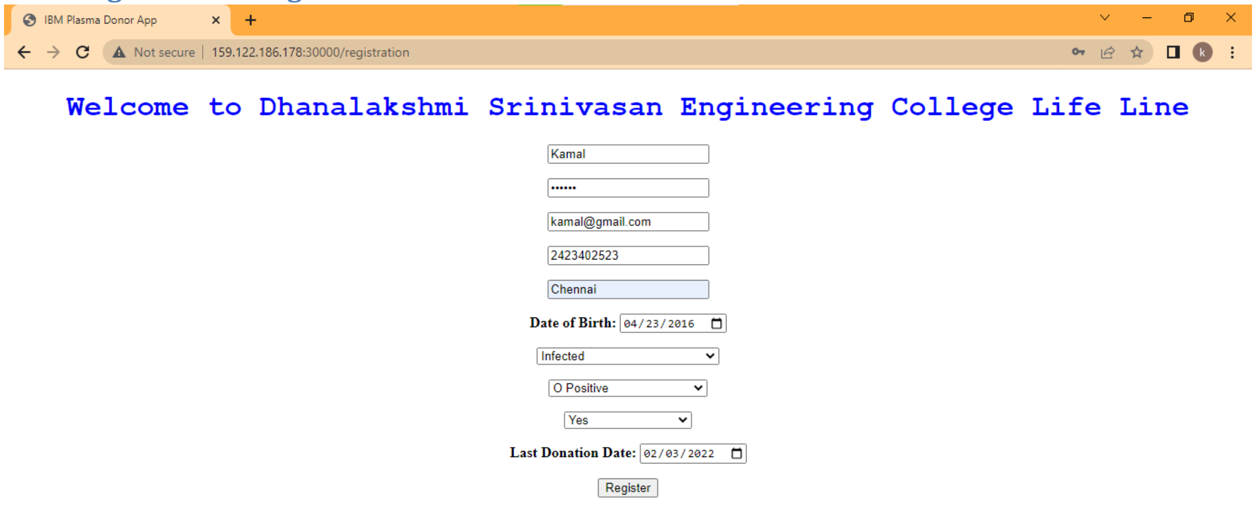
Enter UserName

Enter Password

Login

If you are visiting for the first time - Sign Up

User Registration – Age Validation:


IBM Plasma Donor App

Not secure | 159.122.186.178:30000/registration

Welcome to Dhanalakshmi Srinivasan Engineering College Life Line

Kamal

••••••

kamal@gmail.com

2423402523

Chennai

Date of Birth: 04/23/2016

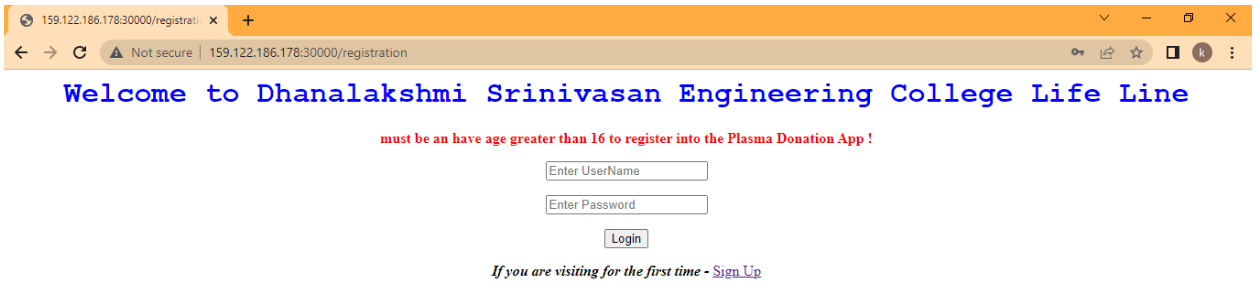
Infected

O Positive

Yes

Last Donation Date: 02/03/2022

Register



159.122.186.178:30000/registration

Not secure | 159.122.186.178:30000/registration

Welcome to Dhanalakshmi Srinivasan Engineering College Life Line

must be an have age greater than 16 to register into the Plasma Donation App !

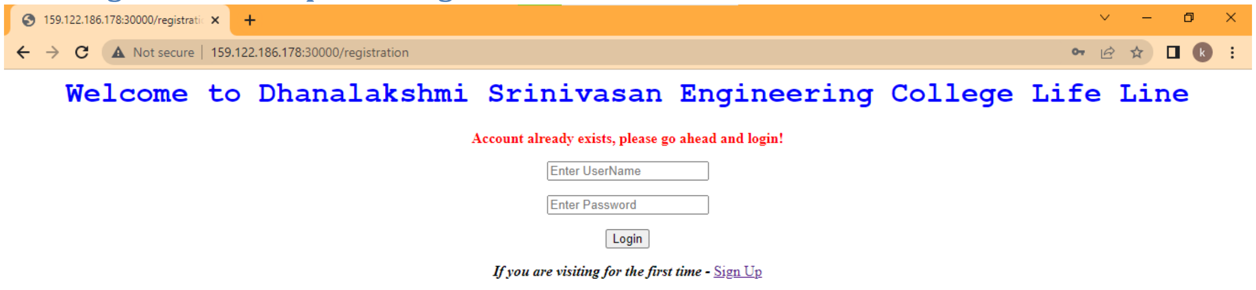
Enter UserName

Enter Password

Login

If you are visiting for the first time - Sign Up

User Registration – Duplicate Registration:



159.122.186.178:30000/registration

Not secure | 159.122.186.178:30000/registration

Welcome to Dhanalakshmi Srinivasan Engineering College Life Line

Account already exists, please go ahead and login!

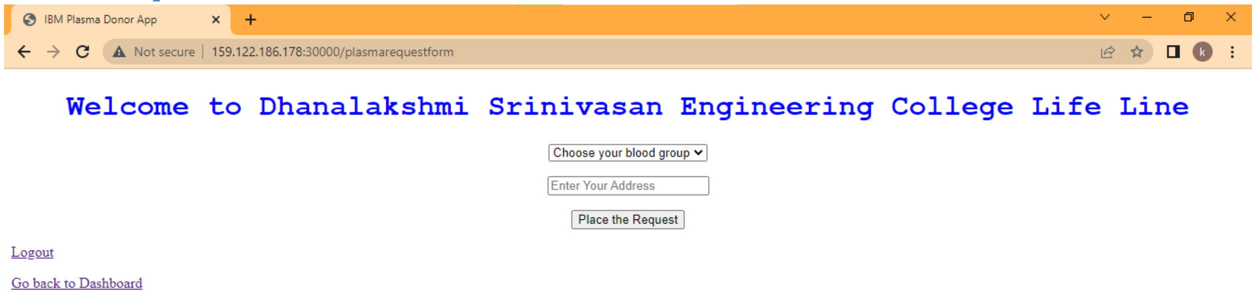
Enter UserName

Enter Password

Login

If you are visiting for the first time - Sign Up

Plasma Request Form:



IBM Plasma Donor App

Not secure | 159.122.186.178:30000/plasmarequestform

Welcome to Dhanalakshmi Srinivasan Engineering College Life Line

Choose your blood group

Enter Your Address

Place the Request

Logout

Go back to Dashboard

Commands:

Git:

Add Code to Repo:

```
ratnasai@DESKTOP-BIBS72I MINGW64 ~/desktop/IBM-Project-14173-1659543653 (main)
$ git add -A
```

Check the Status to Validate the Changes:

```
ratnasai@DESKTOP-BIBS72I MINGW64 ~/desktop/IBM-Project-14173-1659543653 (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
new file:   Final deliverable/Plasma_Donor_App/Dockerfile
new file:   Final deliverable/Plasma_Donor_App/app.py
new file:   Final deliverable/Plasma_Donor_App/deployment.yaml
new file:   Final deliverable/Plasma_Donor_App/requirements.txt
new file:   Final deliverable/Plasma_Donor_App/sendgridmail.py
new file:   Final deliverable/Plasma_Donor_App/service.yaml
new file:   Final deliverable/Plasma_Donor_App/sql_lite_db.py
new file:   Final deliverable/Plasma_Donor_App/templates/dashboard.html
new file:   Final deliverable/Plasma_Donor_App/templates/landingpage.html
new file:   Final deliverable/Plasma_Donor_App/templates/plasmarequest.html
new file:   Final deliverable/Plasma_Donor_App/templates/register.html
```

Add Commit with Message

```
ratnasai@DESKTOP-BIBS72I MINGW64 ~/desktop/IBM-Project-14173-1659543653 (main)
$ git commit -m "Added Code to Git Repo"
[main 31f61b7] Added Code to Git Repo
 11 files changed, 407 insertions(+)
 create mode 100644 Final deliverable/Plasma_Donor_App/Dockerfile
 create mode 100644 Final deliverable/Plasma_Donor_App/app.py
 create mode 100644 Final deliverable/Plasma_Donor_App/deployment.yaml
 create mode 100644 Final deliverable/Plasma_Donor_App/requirements.txt
 create mode 100644 Final deliverable/Plasma_Donor_App/sendgridmail.py
 create mode 100644 Final deliverable/Plasma_Donor_App/service.yaml
 create mode 100644 Final deliverable/Plasma_Donor_App/sql_lite_db.py
 create mode 100644 Final deliverable/Plasma_Donor_App/templates/dashboard.html
 create mode 100644 Final deliverable/Plasma_Donor_App/templates/landingpage.html
 create mode 100644 Final deliverable/Plasma_Donor_App/templates/plasmarequest.html
 create mode 100644 Final deliverable/Plasma_Donor_App/templates/register.html
```

Push Code from local to Remote (GitHub.com)

```
ratnasai@DESKTOP-BIBS72I MINGW64 ~/desktop/IBM-Project-14173-1659543653 (main)
$ git push origin main
Enumerating objects: 20, done.
Counting objects: 100% (20/20), done.
Delta compression using up to 2 threads
Compressing objects: 100% (16/16), done.
Writing objects: 100% (18/18), 6.67 KiB | 1.11 MiB/s, done.
Total 18 (delta 5), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (5/5), completed with 1 local object.
To https://github.com/IBM-EPBL/IBM-Project-14173-1659543653.git
   0394569..f3057fa  main -> main
```

Docker & Container Registry:

Docker:

Docker build:

```
$ docker build -t plasmadonation .

Sending build context to Docker daemon  45.57kB

Step 1/7 : FROM python:3.9

 ---> ab0d2f900193

Step 2/7 : WORKDIR /app

 ---> Using cache

 ---> a03b16aa12ff

Step 3/7 : ADD . /app

 ---> 56ba053e6159

Step 4/7 : COPY requirements.txt /app

 ---> cf06d9a1d4c4

Step 5/7 : RUN pip install -r requirements.txt

 ---> Running in c9618b0c2a9e

Collecting Flask

  Downloading Flask-2.2.2-py3-none-any.whl (101 kB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━

101.5/101.5 KB 2.9 MB/s eta 0:00:00

Collecting ibm_db

  Downloading ibm_db-3.1.3.tar.gz (1.4 MB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━

1.4/1.4 MB 2.0 MB/s eta 0:00:00

  Installing build dependencies: started

  Installing build dependencies: finished with status 'done'

  Getting requirements to build wheel: started

  Getting requirements to build wheel: finished with status 'done'

  Installing backend dependencies: started
  Installing backend dependencies: finished with status 'done'
  Preparing metadata (pyproject.toml): started
  Preparing metadata (pyproject.toml): finished with status 'done'
Collecting sendgrid
  Downloading sendgrid-6.9.7-py3-none-any.whl (101 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
101.1/101.1 KB 2.5 MB/s eta 0:00:00
Collecting python-dotenv
  Downloading python_dotenv-0.21.0-py3-none-any.whl (18 kB)
Collecting click>=8.0
  Downloading click-8.1.3-py3-none-any.whl (96 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
96.6/96.6 KB 5.9 MB/s eta 0:00:00
Collecting importlib-metadata>=3.6.0
  Downloading importlib_metadata-5.0.0-py3-none-any.whl (21 kB)
Collecting Werkzeug>=2.2.2
  Downloading Werkzeug-2.2.2-py3-none-any.whl (232 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
232.7/232.7 KB 2.4 MB/s eta 0:00:00
Collecting itsdangerous>=2.0
  Downloading itsdangerous-2.1.2-py3-none-any.whl (15 kB)
Collecting Jinja2>=3.0
  Downloading Jinja2-3.1.2-py3-none-any.whl (133 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
133.1/133.1 KB 3.4 MB/s eta 0:00:00
Collecting starkbank-ecdsa>=2.0.1
  Downloading starkbank-ecdsa-2.2.0.tar.gz (14 kB)
  Preparing metadata (setup.py): started
  Preparing metadata (setup.py): finished with status 'done'
Collecting python-http-client>=3.2.1
```

```
  Downloading python_http_client-3.3.7-py3-none-any.whl (8.4 kB)
Collecting zipp>=0.5
  Downloading zipp-3.10.0-py3-none-any.whl (6.2 kB)
Collecting MarkupSafe>=2.0
  Downloading MarkupSafe-2.1.1-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (25
kB)
Building wheels for collected packages: ibm_db, starkbank-ecdsa
  Building wheel for ibm_db (pyproject.toml): started
  Building wheel for ibm_db (pyproject.toml): finished with status 'done'
  Created wheel for ibm_db: filename=ibm_db-3.1.3-cp39-cp39-linux_x86_64.whl size=41499651
sha256=c6421d6bbda0f3b87144f0d493a8ed10150c64ef9a4500ec664b19aebe8093ac
  Stored in directory:
/root/.cache/pip/wheels/3d/6e/19/64e70ce3dde2ccda5c9b35bd6a313a39e46f6af0222c75cc5f
  Building wheel for starkbank-ecdsa (setup.py): started
  Building wheel for starkbank-ecdsa (setup.py): finished with status 'done'
  Created wheel for starkbank-ecdsa: filename=starkbank_ecdsa-2.2.0-py3-none-any.whl size=15986
sha256=7866bb8cd33b5354dc5c7d9659887b991be3c77730708cc6694e9ab3631f1c80
  Stored in directory:
/root/.cache/pip/wheels/ff/e0/b9/210b1c0209f93792f212d6e61553624523e49aac6cf284151f
Successfully built ibm_db starkbank-ecdsa
Installing collected packages: starkbank-ecdsa, ibm_db, zipp, python-http-client, python-dotenv,
MarkupSafe, itsdangerous, click, Werkzeug, sendgrid, Jinja2, importlib-metadata, Flask
Successfully installed Flask-2.2.2 Jinja2-3.1.2 MarkupSafe-2.1.1 Werkzeug-2.2.2 click-8.1.3 ibm_db-3.1.3
importlib-metadata-5.0.0 itsdangerous-2.1.2 python-dotenv-0.21.0 python-http-client-3.3.7 sendgrid-
6.9.7 starkbank-ecdsa-2.2.0 zipp-3.10.0
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour
with the system package manager. It is recommended to use a virtual environment instead:
https://pip.pypa.io/warnings/venv
WARNING: You are using pip version 22.0.4; however, version 22.3.1 is available.
You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.
Removing intermediate container c9618b0c2a9e
 ---> 32f2f82e0e21
```

```
Step 6/7 : EXPOSE 5000

 ---> Running in a33d08a5d85a

Removing intermediate container a33d08a5d85a

 ---> bfb591489549

Step 7/7 : CMD ["python","app.py"]

 ---> Running in 6363c1614c47

Removing intermediate container 6363c1614c47

 ---> 2bdf31a28da2

Successfully built 2bdf31a28da2

Successfully tagged plasmadonation:latest

SECURITY WARNING: You are building a Docker image from Windows against a non-Windows Docker
host. All files and directories added to build context will have '-rwxr-xr-x' permissions. It is
recommended to double check and reset permissions for sensitive files and directories.


Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
```

Docker Images:

```
$ docker images

REPOSITORY                                TAG       IMAGE ID       CREATED             SIZE

plasmadonation                              latest      7df0a7eac614   About a minute ago   1.09GB

python                                 3.9       ab0d2f900193   11 days ago         915MB
```

Docker Run (Detached Mode):

```
$ docker run -d -p 5000:5000 plasmadonation:latest

7e86122b4701f40ff0741a9d6584329083119879783f760cbc02dea7e550fcab
```

Docker local containers:

```
$ docker container ls

CONTAINER ID   IMAGE               COMMAND        CREATED           STATUS          PORTS
NAMES

7e86122b4701   plasmadonation:latest   "python app.py"   About a minute ago   Up About a minute
0.0.0.0:5000->5000/tcp   competent_hugle
```

Delete the Local Container

```
$ docker kill 7e86122b4701
```

```
7e86122b4701
```

IBM Cloud Container Registry:

IBM Cloud Login (ibmcloud cli):

```
$ ibmcloud login

API endpoint: https://cloud.ibm.com

Region: us-south


Email> kothamasuvenkataratnasai@gmail.com

Password>

Authenticating...

OK

Targeted account KOTHAMASU VENKATA RATNA SAI's Account (fa29e4bae0044599a0a816aa5d4720d7)

API endpoint:     https://cloud.ibm.com

Region:           us-south

User:             kothamasuvenkataratnasai@gmail.com

Account:          KOTHAMASU VENKATA RATNA SAI's Account (fa29e4bae0044599a0a816aa5d4720d7)

Resource group:   No resource group targeted, use 'C:\Program Files\IBM\Cloud\bin\ibmcloud.exe
target -g RESOURCE_GROUP'

CF API endpoint:

Org:

Space:
```

IBM Cloud Registry Login & Set Client as Docker (ibmcloud cli):

```
$ ibmcloud cr login --client docker

Logging 'docker' in to 'us.icr.io'...

Logged in to 'us.icr.io'.

OK
```

IBM Cloud Registry NameSpace (ibmcloud cli):

```
$ ibmcloud cr namespace-assign

OK
```

IBM Cloud Registry NameSpace List:

```
$ ibmcloud cr namespace-list

Listing namespaces for account 'KOTHAMASU VENKATA RATNA SAI's Account' in registry 'us.icr.io'...

Namespace

plasmadonation
```

IBM Cloud Registry Add Docker tag:

```
$ docker tag plasmaappdocker:latest us.icr.io/plasmadonation/plasmaappdocker:latest
```

IBM Cloud Registry Add Docker tag:

```
$ docker tag plasmaappdocker:latest us.icr.io/plasmadonation/plasmaappdocker:latest
```

IBM Cloud Registry Push Image to Cloud:

```
$ docker push us.icr.io/plasmadonation/plasmadonation:latest

The push refers to repository [us.icr.io/plasmadonation/plasmadonation]

f24e84e8aba1: Pushed

84dcd59995e7: Pushed

6749a6446e3a: Pushed

733c9e138ffe: Mounted from plasmadonation/plasmaappdocker

98c01aa6c3e4: Mounted from plasmadonation/plasmaappdocker

782cce4c7b7f: Mounted from plasmadonation/plasmaappdocker

dde9ab8bf12a: Mounted from plasmadonation/plasmaappdocker

6b183c62e3d7: Mounted from plasmadonation/plasmaappdocker

882fd36bfd35: Mounted from plasmadonation/plasmaappdocker

d1dec9917839: Mounted from plasmadonation/plasmaappdocker

d38adf39e1dd: Mounted from plasmadonation/plasmaappdocker

4ed121b04368: Mounted from plasmadonation/plasmaappdocker

d9d07d703dd5: Mounted from plasmadonation/plasmaappdocker

latest: digest: sha256:9b38b5e59ca8f0596f1e5cb57a16a94a6961dcb18bbd685f890ca577c4b0e96f size:
3052
```

IBM Cloud Registry List Images:

```
$ ibmcloud cr image-list

Listing images...

Repository                  Tag     Digest     Namespace     Created     Size     Security status
```

```
us.icr.io/plasmadonation/plasmadonation   latest   9b38b5e59ca8   plasmadonation   1 hour ago   441
MB   -

OK
```

Kubernetes:

List Clusters:

```
$ ibmcloud ks cluster ls

OK

Name                ID                State      Created        Workers   Location   Version      Resource Group
Name   Provider

plasma-14173-1659543653   cdjk7e0f0k1ihttfjpag   deploying   17 seconds ago   1       mil01
1.24.7_1542   Default            classic
```

Set Context:

```
$ kubectl config current-context

nsaz203kubercluster
```

Set the Kubeconfig for export:

```
$ export KUBECONFIG=$(mktemp)
```

Export the Kubernetes Config:

```
$ ibmcloud ks cluster config -c plasma-14173-1659543653

OK

The configuration for plasma-14173-1659543653 was downloaded successfully.

Added context for plasma-14173-1659543653 to the current kubeconfig file.

You can now execute 'kubectl' commands against your cluster. For example, run 'kubectl get nodes'.

If you are accessing the cluster for the first time, 'kubectl' commands might fail for a few seconds while
RBAC synchronizes.
```

Echo & Cat and see the Config:

```
$ echo $KUBECONFIG

/tmp/tmp.uK5in6M7uU

$ cat $KUBECONFIG

apiVersion: v1

clusters:

- cluster:
   certificate-authority: C:\Users\mmm04\.bluemix\plugins\container-service\clusters\plasma-14173-
1659543653-cdjk7e0f0k1ihttfjpag\ca-aaa00-plasma-14173-1659543653.pem
```

Get Nodes:

```
$ kubectl get nodes

NAME           STATUS   ROLES    AGE     VERSION

10.144.195.234   Ready    <none>   5m45s   v1.24.6+IKS
```

Create Deployment:

```
$ kubectl create -f deployment.yaml

deployment.apps/plasma-14173-1659543653-deployment created
```

Get Deployment:

```
$ kubectl get deployment

NAME                             READY   UP-TO-DATE   AVAILABLE   AGE

plasma-14173-1659543653-deployment   1/1     1            1           13m
```

Describe Deployment:

```
$ kubectl describe deployments plasma-14173-1659543653-deployment

Name:                plasma-14173-1659543653-deployment

Namespace:           default

CreationTimestamp:   Sun, 06 Nov 2022 00:41:08 -0500

Labels:              <none>

Annotations:         deployment.kubernetes.io/revision: 1

Selector:            app=plasmadonationnode

Replicas:            1 desired | 1 updated | 1 total | 1 available | 0 unavailable

StrategyType:        RollingUpdate

MinReadySeconds:     0

RollingUpdateStrategy:  25% max unavailable, 25% max surge

Pod Template:

  Labels:  app=plasmadonationnode

  Containers:

   plasmadonationnode:

    Image:       us.icr.io/plasmadonation/plasmaappdocker
   Port:        5000/TCP
   Host Port:    0/TCP
   Environment:  <none>
   Mounts:      <none>
  Volumes:       <none>
Conditions:
  Type          Status  Reason
  ----          ------  ------
  Available     True    MinimumReplicasAvailable
  Progressing   True    NewReplicaSetAvailable
OldReplicaSets:  <none>
NewReplicaSet:   plasma-14173-1659543653-deployment-d9767b59c (1/1 replicas created)
Events:
  Type    Reason           Age   From                Message
  ----    ------           ----  ----                -------
  Normal  ScalingReplicaSet  14m   deployment-controller  Scaled up replica set plasma-14173-
1659543653-deployment-d9767b59c to 1
```

Get Pods:

```
$ kubectl get pods
NAME                                              READY   STATUS    RESTARTS   AGE
plasma-14173-1659543653-deployment-d9767b59c-fhlkc   1/1     Running   0          55s
```

Create Service:

```
$ kubectl create -f service.yaml
service/plasma-14173-1659543653-service created
```

Get Service:

```
$ kubectl get services
NAME                          TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
kubernetes                    ClusterIP   172.21.0.1      <none>        443/TCP          156m
plasma-14173-1659543653-service   NodePort    172.21.33.184   <none>        5000:30000/TCP   78m
```

Describe Service:

```
$ kubectl describe services plasma-14173-1659543653-deployment

Name:           plasma-14173-1659543653-deployment

Namespace:      default

Labels:         <none>

Annotations:    <none>

Selector:       app=plasmadonationnode

Type:           ClusterIP

IP:             172.21.11.226

Port:           <unset>  5000/TCP

TargetPort:     5000/TCP

Endpoints:      172.30.85.75:5000

Session Affinity:  None

Events:         <none>
```

Get Replica Sets:

```
$ kubectl get replicasets

NAME                                         DESIRED   CURRENT   READY   AGE

plasma-14173-1659543653-deployment-d9767b59c   1         1         1       15m
```

Describe Replica Sets:

```
$ kubectl describe replicasets

Name:         plasma-14173-1659543653-deployment-d9767b59c

Namespace:    default

Selector:     app=plasmadonationnode,pod-template-hash=d9767b59c

Labels:       app=plasmadonationnode

              pod-template-hash=d9767b59c

Annotations:  deployment.kubernetes.io/desired-replicas: 1

              deployment.kubernetes.io/max-replicas: 2

              deployment.kubernetes.io/revision: 1

Controlled By:  Deployment/plasma-14173-1659543653-deployment

Replicas:     1 current / 1 desired
```

```
Pods Status:    1 Running / 0 Waiting / 0 Succeeded / 0 Failed

Pod Template:

  Labels:  app=plasmadonationnode

           pod-template-hash=d9767b59c

  Containers:

   plasmadonationnode:

    Image:       us.icr.io/plasmadonation/plasmaappdocker

    Port:        5000/TCP

    Host Port:    0/TCP

    Environment:  <none>

    Mounts:       <none>

  Volumes:        <none>

Events:

  Type    Reason            Age   From                 Message

  ----    ------            ----  ----                 -------

  Normal  SuccessfulCreate  15m   replicaset-controller  Created pod: plasma-14173-1659543653-
deployment-d9767b59c-fhlkc
```

Check the Ingress Health:

```
$ ibmcloud ks ingress status -c  plasma-14173-1659543653

OK

Ingress Status:   healthy

Message:         Ingress is not supported for free clusters
```

GitHub & Project Demo Link:

GitHub:

https://github.com/IBM-EPBL/IBM-Project-14173-1659543653

Project Demo Link:

EXECUTION LINK (ONLY EXECTION):

Google Drive Link :

https://drive.google.com/file/d/1WWTOzv5dTpLvmOD4b7deOOKg0qJe9ecz/view?usp=drivesdk

YouTube link :

https://youtu.be/n4GWkjQg_28

FULL VIDEO LINK (FULL EXECUTION LINK WITH COMMANDS):

Google Drive Link :

https://drive.google.com/file/d/1Vtg3ZAAQ19OyPS7e1-Je6rXb8jwYZpa4/view?usp=share_link

YouTube link :

https://youtu.be/Vu2igNVOnUU

SERVER LINK:

http://159.122.186.178:30000/