PLASMA DONAR APPLICATION

IBM-Project-1974-1658421777

NALAIYA THIRAN PROJECT BASED LEARNING ON

JEPPIAAR ENGINEERING COLLEGE CHENNAI

PROJECT REPORT BY

TAMIL THENDRAL M

ROHITH SIVAM P

RAVEEN RAJ C

RAM GANESH S

CHAPTER 1

1. INTRODUCTION

1.1 Project Overview

In recent days, it is noticed the increase in blood request posts on social media such as Facebook, Twitter, and Instagram. Interestingly there are many people across the world interested in donating blood when there is a need, but those donors don't have an access to know about the blood donation requests in their local area. This is because that there is no platform to connect local blood donors with patients. BLOODR solves the problem and creates a communication channel through authorized 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. Clinics can use this web application to maintain the blood 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.

BLOODR 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 blood 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 Blood Request Feeds. Appointments can be synchronized with Google and Outlook calendars for the ease of users.

1.2 Purpose

During the COVID 19 crisis, the requirement of plasma became a high priority and the donor count has become low. Saving the donor information and helping the needy by notifying the current donors list, would be a helping hand. 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.

Plasma is the liquid portion of blood. About 55% of our blood is plasma, and the remaining 45% are red blood cells, white blood cells and platelets that are suspended in the plasma. Plasma is about 92% water. It also contains 7% vital proteins such as albumin, gamma globulin and anti-hemophilic factor, and 1% mineral salts, sugars, fats, hormones and vitamins. Plasma is commonly given to trauma, burn and shock patients, as well as people with severe liver disease or multiple clotting

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.

CHAPTER 2

2. LITERATURE SURVEY

2.1 Existing problem

- Plasma finder face more difficult to find plasma donor
- get more panic

2.2 References

[1]. Plasma Donation Website using MERN stack - Neha Soni , Software Engineering Intern at FICO | Technical Blogger The person who wants to donate his/her plasma needs to register in our application providing required information which are name, age, blood group, phone number, and location, etc. Patients who need plasma can also fill the form to request the plasma. Patients can directly call the donor by taking his/her contact number from the application. The user can also view the total active cases, recovered cases, vaccine centres in their area, hospital location, and helpline number.

[2]. Instant Plasma Donor Recipient Connector Web Application - Ripathi S ,Kumar V, Prabhakar A The world is suffering from COVID 19 crisis, and we haven't found any vaccine yet. But there is another scientific way from which we can help to lower the death ratio or help the COVID 19 affected person is by donating Plasma from recovered patients. With no approved antiviral treatment plan for the deadly COVID-19 infection, plasma therapy is an experimental approach to treat COVID positive patients and help them recover faster. The therapy considered to be safe and promising. If a particular person is fully recovered from COVID 19 he/she is applicable to donate their plasma. In the proposed system, donors who need to donate plasma can donate by uploading covid-19 certificate and blood bank can view donors and can raise requests to donors and the hospital can register/login and can search for plasma, they can raise requests to blood bank and can get the plasma

[3]. Developing a plasma donor application using Function-as a-service in AWS - Aishwarya R Gowri,
Jain University Department of MCA, computer science A plasma is a liquid portion of the blood, over 55% of human blood is plasma. Plasma is used to treat various infectious diseases and it is 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. The services used are AWS Lambda, API gateway, Dynamo DB, AWS Elastic Compute Cloud with the help of these AWS services, it eliminates the need of configuring the servers and reduces the infrastructural costs associated with it and helps to achieve serverless computing. Situations like if the donor count is very low, it is very important to get the information about the plasma donors. Saving the donor information and notifying about the current donors would be a helping hand as it can save time and help the users to track down the necessary information about the donors.

2.3 Problem Statement Definition

Plasma is commonly given to trauma, burn and shock patients, as well as people with severe liver disease or multiple clotting factor deficiencies. It helps boost the patient's blood volume, which can prevent shock, and helps with blood clotting. With the number of people affected by COVID-19 infection, the demand for the plasma of recovered patients has also gone up tremendously. The antibodies, which are present in our body, can help someone fight the infection and emerge victorious. Our Plan: We plan to make a User-friendly application for users who are in need for plasma or who wish to donate plasma to anyone who are in need. However, areas of concern, including privacy and confidentiality, should be considered during design and development. Age was identified as a contributing factor that might decrease the likelihood of app usage among donors. The donation center staff focused on the educational features of the app and emphasized the importance of the app providing statistics and sending notifications and reminders to donors.

2.1 EXISTING PROBLEM

There are a quite good number of software packages that exist for PLASMA DONOR APPLICATION system. But when I visited most plasma donor center system portal. I found that existing system is limited only to those particular plasma center.

Problem Found In Existing System

- At the present there is no software to keep any records in plasma center.
- It becomes difficult to provide any record immediately at times of emergency.
- Required more human efforts in maintaining the branch related information .
- Manually to keep the accounts is also tedious & risky job & to maintain those accounts in ledgers for a long period is also very difficult.
- Difficult to manage and maintain the files.
- Chance of damage of files, if the data is stored in the files for duration of time.

- Time consuming is retrieving, storing and updating the data.
- It is difficult to keep track the record about the donor & receiver he has donated or received the plasma at the last time.

2.2 REFERENCES

CASE STUDY - I

TITLE: Instant Plasma donar Recipient connector web application

AUTHOR: Kalpana Devi Guntoju, Tejaswini Jalli, Sreeja Uppala, Sanjay Mallisetti

YEAR: 2022

ABSTRACT:

The world is suffering from the COVID 19 crisis and no vaccine has been found yet, but there is another scientific way in which we can help reduce mortality or help people affected by COVID19 by donating plasma from recovered patients. In the absence of an approved antiviral treatment plan for a fatal COVID19 infection, plasma therapy is an experimental approach to treat COVID19-positive patients and help them faster recovery. Therapy is considered competent. In the recommendation system, the donor who wants to donate plasma can donate by uploading their COVID19 certificate and the blood bank can see the donors who have uploaded the certificate and they can make a request to the donor and the hospital can register/login and search for the necessary things. plasma from a blood bank and they can request a blood bank and obtain plasma from the blood bank.

CASE STUDY - II

TITLE: Determinants of plasma donation: A review of the literature

AUTHOR: A.Beurel , F. Terrade, J.-P.Lebaudy

YEAR: 2017

ABSTRACT:

The major contribution of Human Sciences in the understanding of the whole blood donation behavior has been through the study of individuals' motivations and deterrents to donate. However, if whole blood donation has been very widely studied in the last sixty years, we still know very little about plasma donation in voluntary non-remunerated environments. Yet, the need for plasma- derived products has been strongly increasing for some years, and blood collection agencies have to adapt if they want to meet this demand. This article aims to review the main motivations and deterrents to whole blood donation, and to compare them with those that we already know concerning plasma donation. Current evidence shows similarities between both behaviors, but also differences that indicate a need for further research regarding plasma donation.

CASE STUDY – III

TITLE: Developing a plasma donor application using Function-as-a-service in AWS

AUTHOR: Aishwarya R Gowri

YEAR: 2020

ABSTRACT:

A plasma is a liquid portion of the blood, over 55% of human blood is plasma. Plasma is used to treat various infectious diseases and it is 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. The services used are AWS Lambda, API gateway, DynamoDB, AWS Elastic Compute Cloud with the help of these AWS services, it eliminates the need of configuring the servers and reduces the infrastructural costs associated with it and helps to achieve serverless computing. For instance, during COVID 19 crisis the requirement for plasma increased drastically as there were no vaccination found in order to treat the infected patients, with plasma therapy the recovery rates where high but the donor count was very low and in such situations it was very important to get the information about the plasma donors. Saving the donor information and notifying about the current donors would be a helping hand as it can save time and help the users to track down the necessary information about the donors.

2.3 PROBLEM SOLUTION DEFINITION

PROBLEM STATEMENT – I

PROBLEM STATEMENT – II

PROBLEM STATEMENT – III

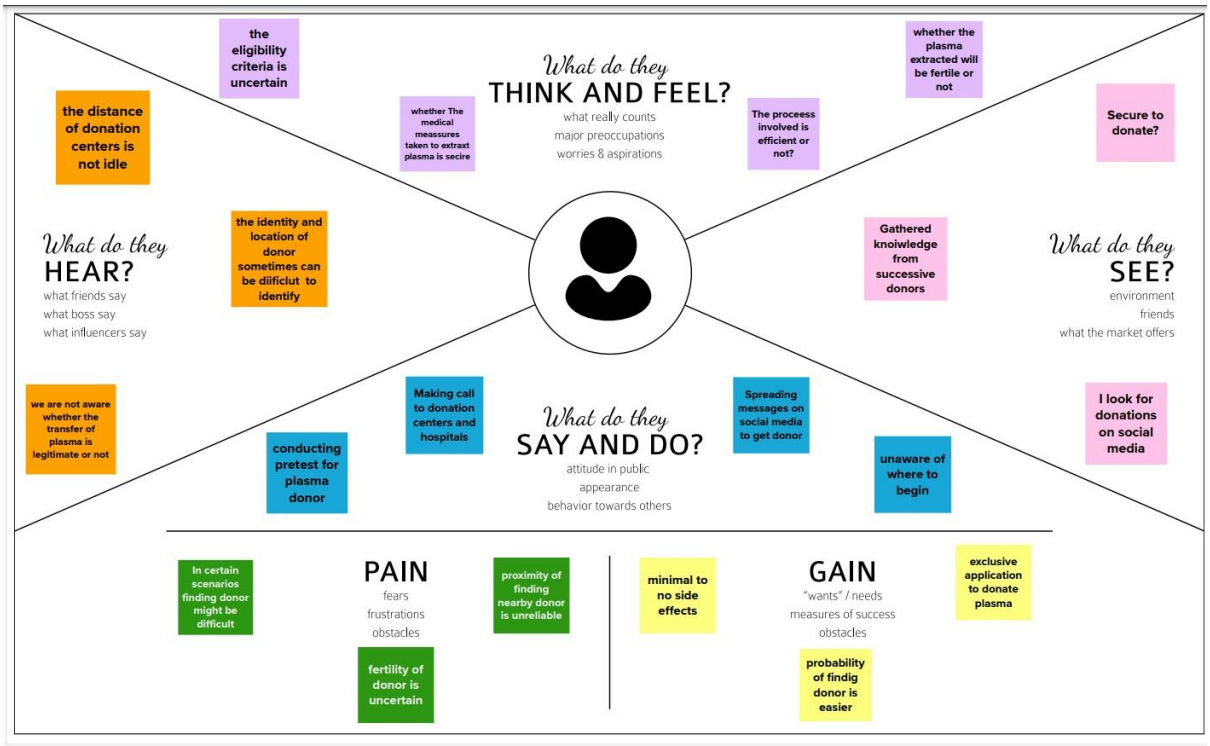
PROBLEM STATEMENT – IV

CHAPTER 3

IDEATION AND PROPOSED SOLUTION

3. IDEATION & PROPOSED SOLUTION

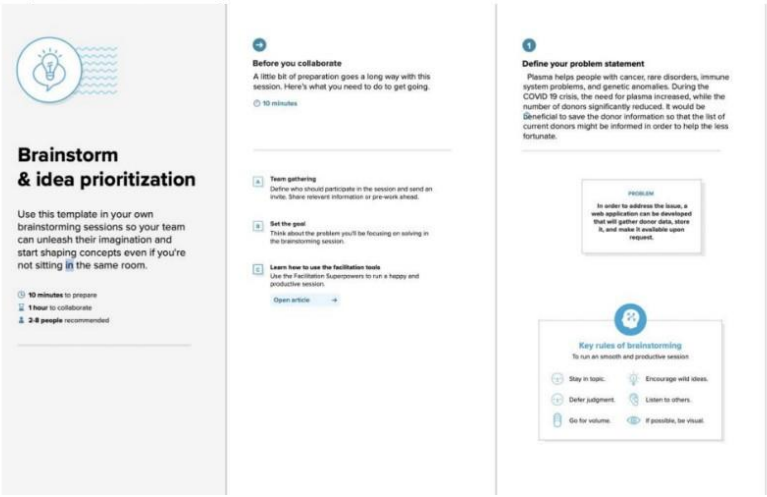
3.1 Empathy Map Canvas



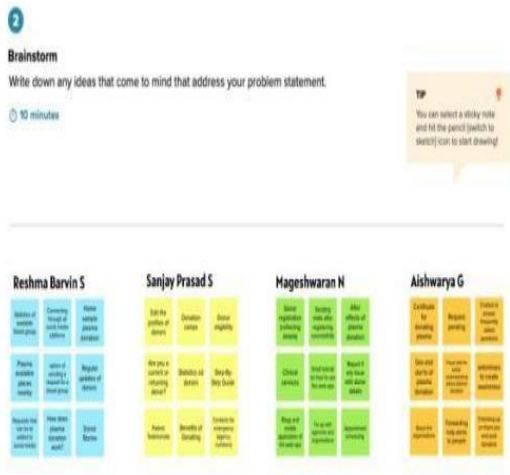
3.2 Ideation & Brainstorming

Brainstorming gives an unfastened and open environment that encourages every person inside a group to take part in the innovative questioning manner that ends in trouble solving. Prioritizing quantity over value, out-of-the-container thoughts are welcome and constructed upon, and all contributors are endorsed to collaborate, assisting every different broaden a wealthy quantity of innovative solutions.

Step-1: Team Gathering, Collaboration and Select the Problem Statement



Step-2: Brainstorm, Idea Listing and Grouping



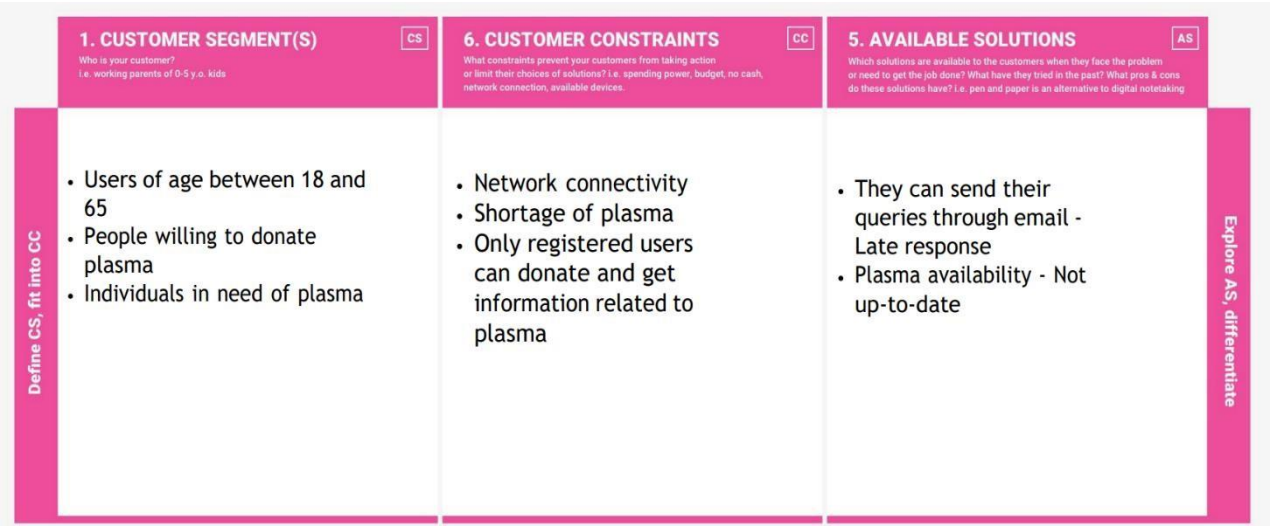
Use this template for your personal brainstorming periods so your group can harness their creativeness and begin shaping standards even supposing you're now no longer sit in inside the identical room.

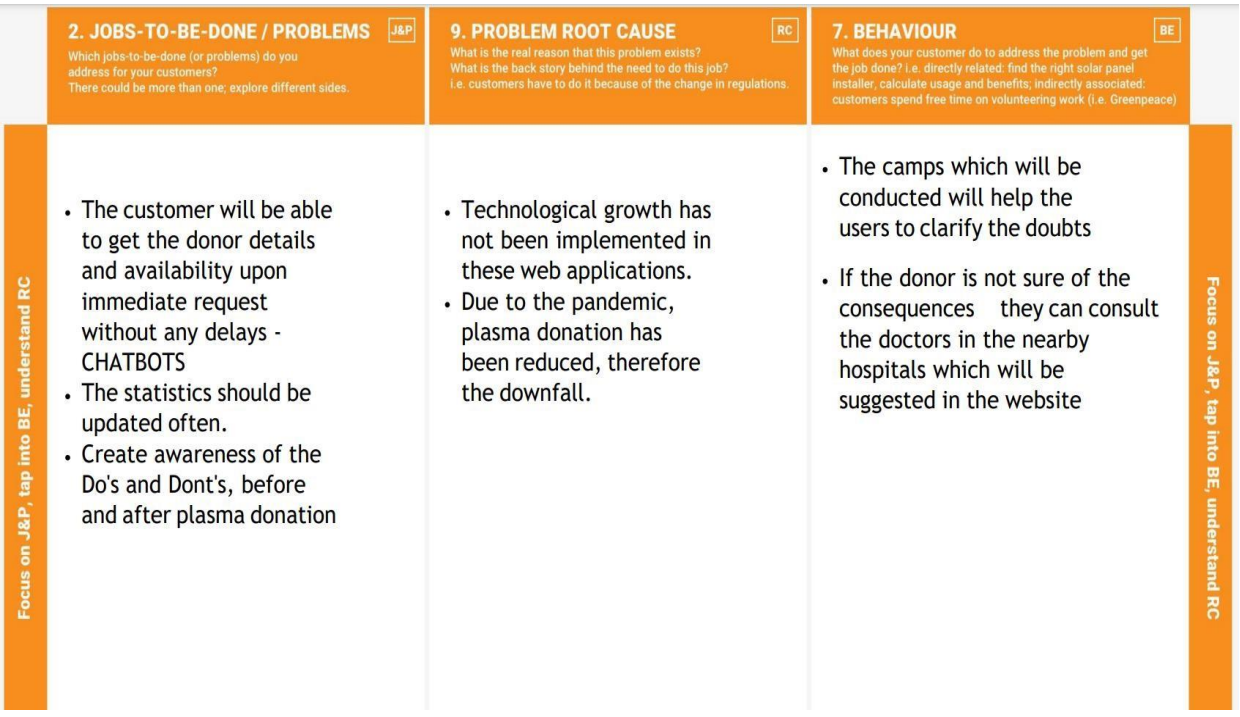
3.3 Proposed Solution

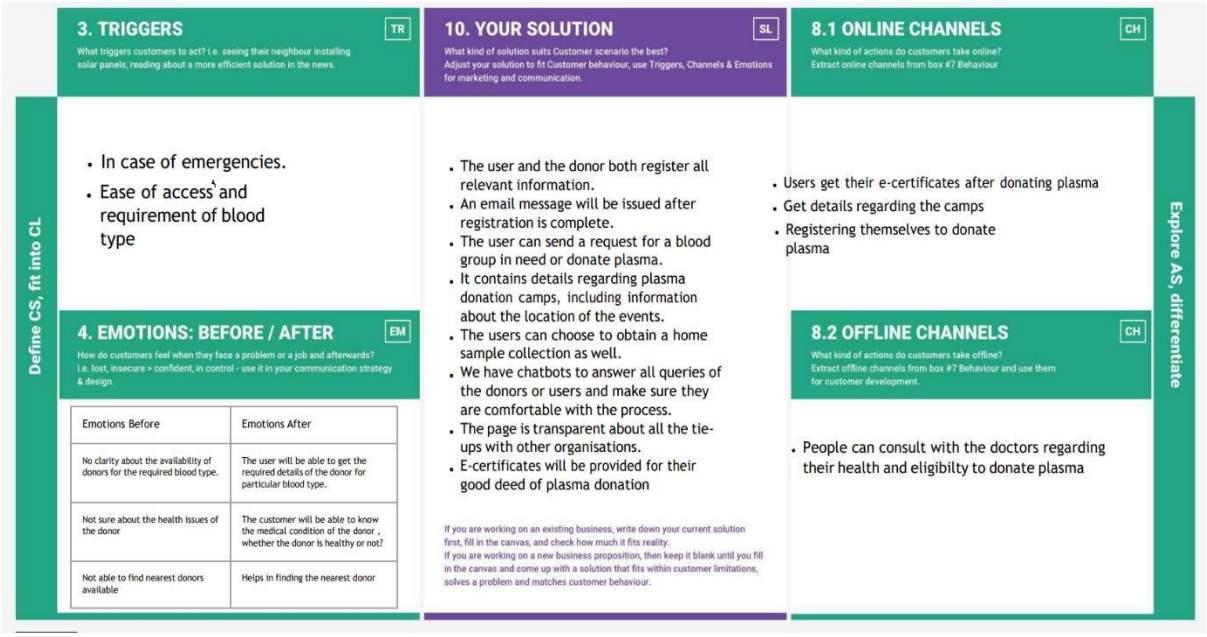
S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	During the COVID 19 crisis, the want for plasma increased, at the same time as the wide variety of donors has decreased. Plasma is vital for the survival of human beings with cancer, uncommon disorders, immunological problems, and genetic anomalies. Every blood financial institution claims to be out of blood, so we want to make human beings privy to the problem and provide support. Numerous camps, seminars, and programs may be of superb help.
2.	Idea / Solution description	Plasma donor is an application which will make things easier and efficient at crucial times and to solve our problem statement. Some of the features are : • The user and the donor both register all relevant information. • An email message will be issued after registration is complete. • The user has the option of sending a request for a blood group in need or donating plasma in this. • It contains details regarding plasma donation camps, including information about the location of the events. • The users can choose to obtain a home sample collection as well.
3.	Novelty / Uniqueness	A visual representation that is simple for users to understand will be used to display the statistics for the blood group availability data for plasma donation. The user can send a request for plasma if they are unsure about its availability in their immediate vicinity. Whether plasma is in short supply or is more readily available, users will receive an email notification within a short period of time. If individuals sign up for our application for plasma donors and decide they want to donate plasma, they can schedule an appointment. They will obtain their e-certification for donating plasma once they have completed their session according to

		schedule.These are the innovative elements included in this.
4.	Social Impact / Customer Satisfaction	Despite the apparent abundance of resources, there are still cases where hospitals or blood banks run out of essential resources, such as specific blood type shortages. One of the major issues health facilities run into is the shortage of certain blood types. An additional problem is facilities need access to patient data as quickly as possible before beginning patient blood transfer. This application, along with all the services it provides, also helps to eradicate certain spam messages and mails circulating around regarding fake or already satisfied blood emergency situations . A single platform for maintaining all genuine blood related activities and information increases the trust of the public to get involved in these activities, and to participate in blood donations.
5.	Business Model (Revenue Model)	An unpaid application exists for plasma donors. It is readily available and accessible by all. Due to the difficulty in locating donors who match a certain blood group, this application enables users to register people who wish to donate plasma and keep their information in a database. By informing the current donors of the need, saving the donor information would assist. The need for plasma increased significantly during the COVID 19 crisis, and the number of donors is limited. In the end, working with the government can use an app to aid those in need of plasma.
6.	Scalability of the Solution	This software assists customers in locating the closest blood centre, understanding their eligibility to donate blood, receiving notifications while an pressing blood donation name comes in, and scheduling a handy appointment using temporary and/or spatial information. A cutting-edge donor profile could be used, containing info which includes the donor's gift location, blood type, and the date of their latest donation, amongst different things. The proper donors could be cleverly knowledgeable of the call for blood donations, making it less difficult to discover a nearby appropriate donor at the proper time.

3.4 Problem Solution fit






3. TRIGGERS
TR
What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.
In case of emergencies.
Ease of access and requirement of blood type
4. EMOTIONS: BEFORE / AFTER
EM
How do customers feel when they face a problem or a job and afterwards? i.e. lost, insecure > confident, in control - use it in your communication strategy & design.
Emotions Before
Emotions After
No clarity about the availability of donors for the required blood type.
The user will be able to get the required details of the donor for particular blood type.
Not sure about the health issues of the donor
The customer will be able to know the medical condition of the donor , whether the donor is healthy or not?
Not able to find nearest donors available
Helps in finding the nearest donor
10. YOUR SOLUTION
SL
What kind of solution suits Customer scenario the best? Adjust your solution to fit Customer behaviour, use Triggers, Channels & Emotions for marketing and communication.
The user and the donor both register all relevant information.
An email message will be issued after registration is complete.
The user can send a request for a blood group in need or donate plasma.
It contains details regarding plasma donation camps, including information about the location of the events.
The users can choose to obtain a home sample collection as well.
We have chatbots to answer all queries of the donors or users and make sure they are comfortable with the process.
The page is transparent about all the tie-ups with other organisations.
E-certificates will be provided for their good deed of plasma donation
If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality.
If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.
8.1 ONLINE CHANNELS
CH
What kind of actions do customers take online? Extract online channels from box #7 Behaviour
Users get their e-certificates after donating plasma
Get details regarding the camps
Registering themselves to donate plasma
8.2 OFFLINE CHANNELS
CH
What kind of actions do customers take offline? Extract offline channels from box #7 Behaviour and use them for customer development.
People can consult with the doctors regarding their health and eligibilty to donate plasma
Define CS, fit into CL
Explore AS, differentiate

CHAPTER 4

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form (Web App)
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Certification	The certification of appreciation and authentication is given by us after the donor donates the plasma.
FR-4	Statistical data	The availability of plasma is given in the app page as stats which will be helpful for the users of the app.
FR-5	User Plasma Request	Users can request to donate plasma by filling out the request form the app page. We will get an e-mail when they submit their request on the app page.
FR-6	Searching/reporting requirement	Users can use the search bar to look up information about camps and other information about us.
FR-8	Virtual Assistants	A virtual assistant is a software agent that can carry out tasks or provide services on behalf of a person in response to commands or inquiries. When users enter their inquiries, the system will respond with patient information about plasma and details of plasma donation.

4.1 FUNCTIONAL REQUIREMENT

In software engineering and systems engineering, a functional requirement defines a function of a system or its components.

Access Website:

Software operator should be capable to access web-application through either an application browser or similar service on the PC. There should not be any limitation to access web-application.

Software operator Registration:

Given that software operator has accessed web-application, then the software operator should be able to register through the web-application. The donor software operator must provide first name, gender, plasma group, location, contact, software operator name and password.

New Releases:

When a new/update/revise version of the web-application is released, the appearance will be automatically appears when the software operator access the web-application.

Software operator log-in:

Given that the software operator has registered, then the software Operator should be able to login to the web-application. The login information will be stored on the database for future use.

Search result in a list view:

Search result can be viewed in a list. Each element in the list represents a specific donor. Each element should include first name, gender, plasma group, location, contact according to the software operator position.

Request plasma:

Software operator (Clinic) should be able to request for plasma at emergency situation, software operator need to define plasma group, location, required date, contact. The plasma request requested will be sent to plasma bank and then to the Inventory to check the availability. If available, the requested plasma will be sent to the requested donor (Clinic).

View Request:

The plasma Bank should be able to view received request and then respond to them and can search requests by selecting two options select plasma group and provision.

Search plasma Bank Stock:

Receiving the blood or plasma request from Clinic, the blood or plasma stock in the Blood or plasma Bank Inventory will be searched to match the requested blood or plasma request.

View Blood or plasma request Details:

The Clinic, Blood or plasma Bank should be able to view the Blood or plasma requestId, time of the blood or plasma request placed, name of the clinic, location and the address of the clinic. In addition to this an additional feature of tracking the distribution person which includes his location and the checkpoints passed.

View Distribution Status:

The Clinic, Blood or plasma Bank should be able to view the status of the distribution time. If the distribution seems to be delayed then the clinic manager must to able to call the distribution person to get the update/revise on the distribution.

4.2 NON-FUNCTIONAL REQUIREMENTS

In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specifies behaviors.

Maintainability:

The plasma donar application System have must have high level of Maintainability.

Serviceability

If issue arises in the plasma donar application System, then then project must be programmed in such a way that developer can service it again.

Environmental

The plasma donar application System must be working in latest operating system environments like windows 7, windows 8, windows 10 and on Linux.

Data Integrity

All the data in the plasma donar application System must be accurate and reliable. **Usability**

The plasma donar application System must have a good looing user friendly interface.

Recoverability

The plasma donar application System must have a proper data backup mechanism.

Interoperability

The plasma donar application System must work with or use the parts or equipment of another system.

Capacity

The plasma donar application System must fulfill on storage requirements, today and in the future. The Blood bank Management System must be scale up for increasing volume demands.

Performance

The plasma donar application System must perform well in different scenarios.

Security

The plasma donar application System must be secured with proper user name and passwords.

Regulatory

The plasma donar application System must obey all the governmental requirements and constraints.

Availability

The plasma donar application System must be available 24 hours a day with no bandwidth issues.

Manageability

The plasma donar application System must Alerts when the system suffers from a recoverable interruption.

4.2 Non-Functional requirements

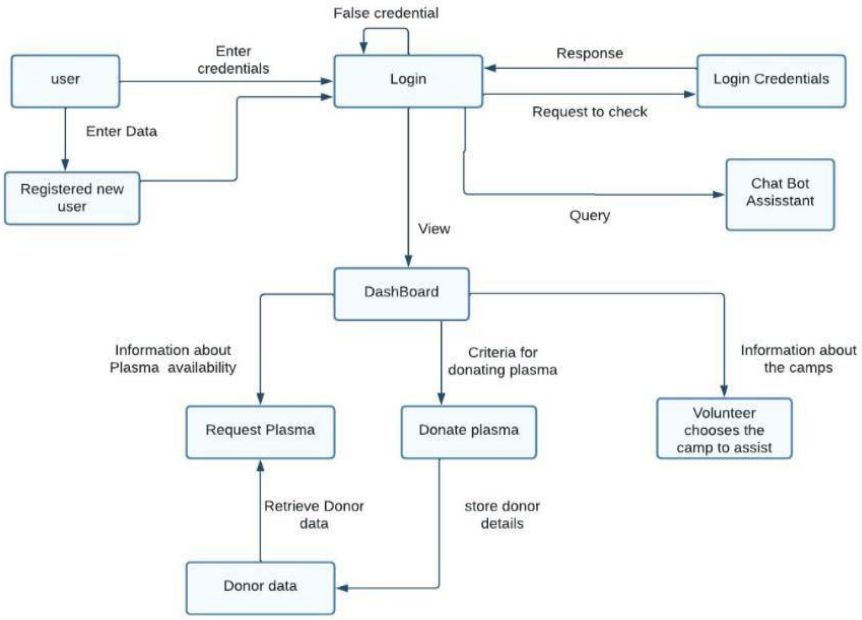
FR No.	Non-Functional Requirement	Description
NFR-1	USABILITY	Must have a goodlooking User friendly interface.
NFR-2	SECURITY	It must be secured with the proper username and password.
NFR-3	RELIABILITY	The system should be made in such a way that it is reliable in its operations and far securing the sensitive details.
NFR-4	PERFORMANCE	Users should have a proper Internet Connection.
NFR-5	AVAILABILITY	The system including the online and offline components should be available 24/7.
NFR-6	**SCALABILITY**	The application has the ability to handle growing numbers of users and load without compromising on performance and causing disruptions to user experience.

CHAPTER 5

5. PROJECT DESIGN

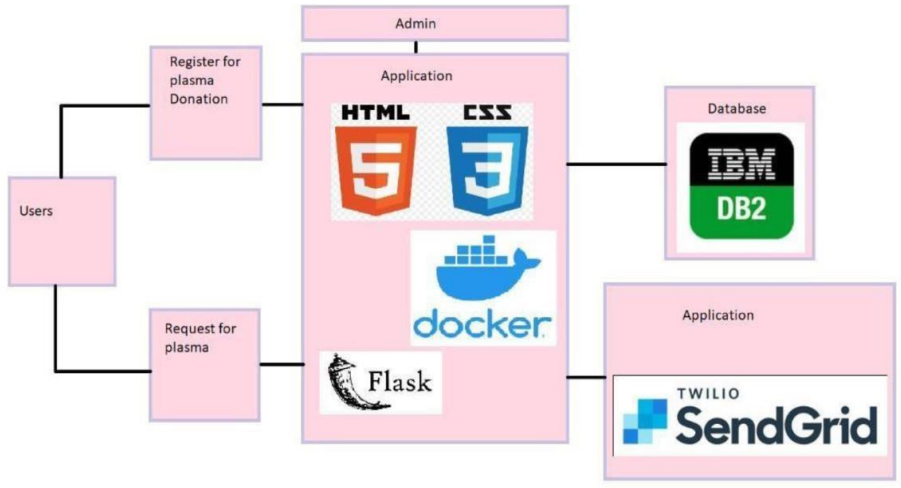
5.1 Data Flow Diagrams

Data Flow Diagrams:

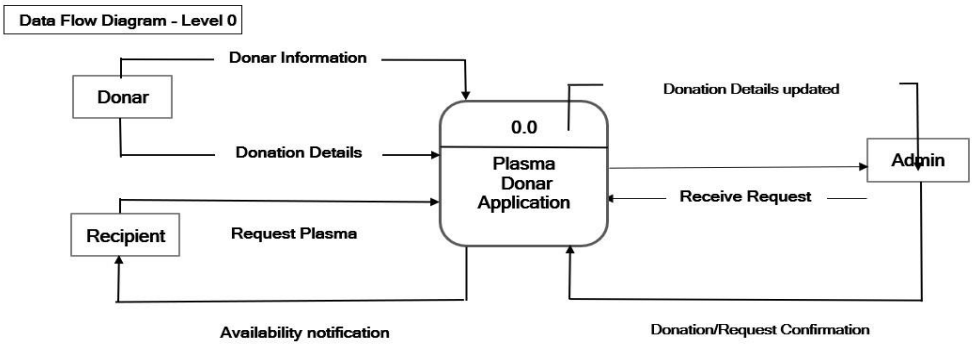


5.2 Technical Architecture

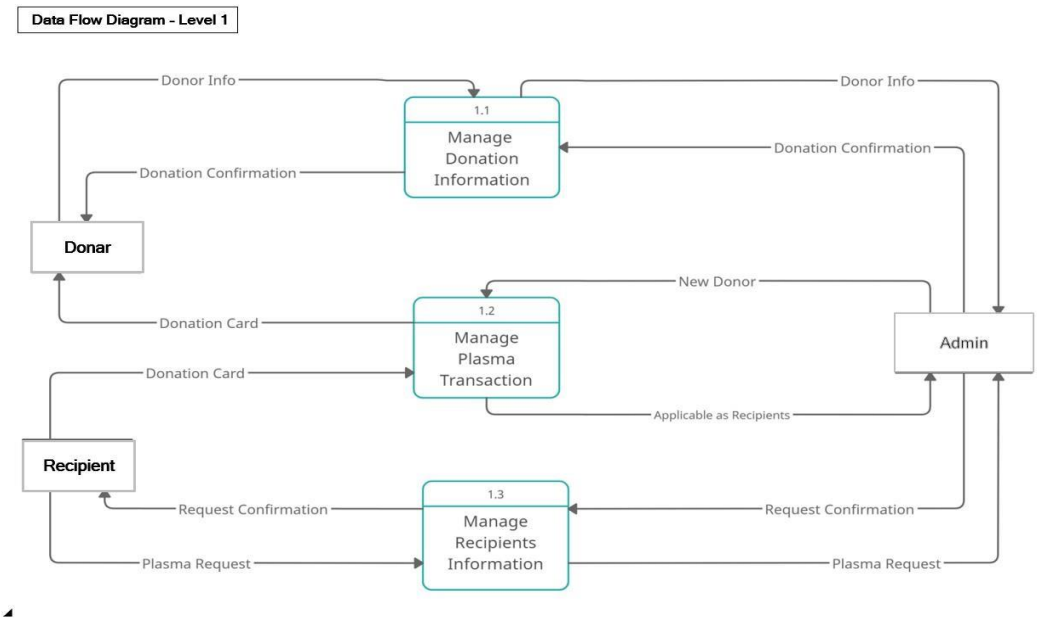
Technical Architecture:



5.1 DATA FLOW DIAGRAM



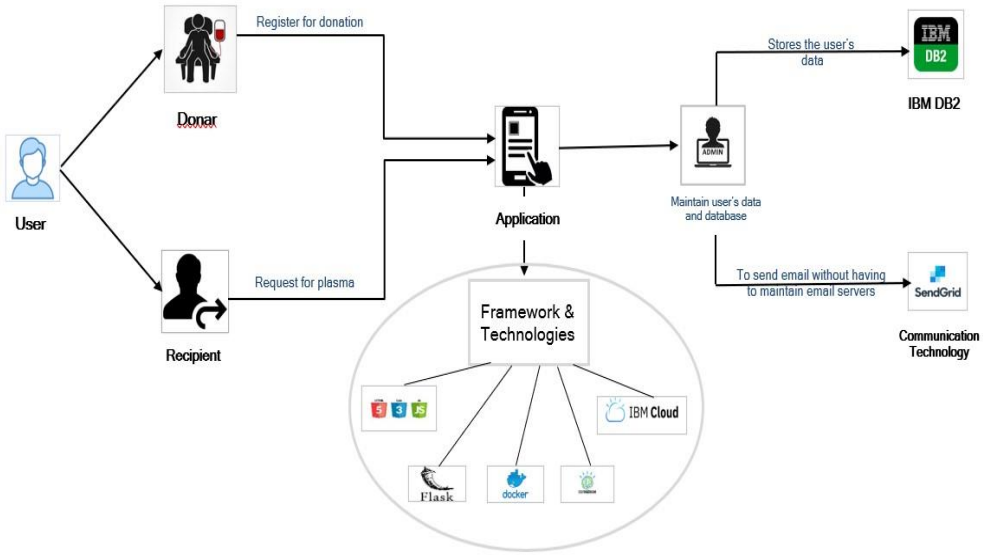
Data Flow Diagram – Level 0



Data Flow Diagram – Level 1

5.2 SOLUTION AND TECHNOLOGY ARCHITECTURE

Technical Architecture:



5.3 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority

Custome r (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, phone number, password.	I can access my account / profile.	High
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive verification email for confirmation.	High
		USN-3	As a user, I can register for the application through social media site/account.	I can register & access my account/profile with social media account.	Low
		USN-4	As a user, I can register for the application through Gmail, Yahoo mail, Outlook…	I can register the app with email account.	Medium
	Login	USN-5	As a user, I can log into the application by entering email & password.	I can register & access user profile/account with Gmail account.	High
	Requesting/ recipient	USN-6	As a recipient, I can request the blood group for which I need plasma.	I can get plasma through Donation center while plasma is available.	High
Customer (Web user)	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
Customer Care Executive	Help desk /User support	USN-8	As a customer care executive, I can solve the queries of the users.	I can reply to their queries and solve their related problems.	High
Administrator	Registration	USN-9	As an Administrator, I can view the database of the registered users.	I can check and verify the persons who are the registered their mail Id's and information's.	Medium
	Dashboard	USN-10	As an Administrator, I can view how many members requested for what kind of blood group for plasma.	I can check the number of requirements and monitor the availability.	Low

Chabot	User-Interface	USN-11	In addition to the customer care executive, I can solve all the queries of the donor as well as the recipient.	I can reply to all the Questions which are asked by the users that are related to the service we provided.	Medium

5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can sign up for the application via my email, password, and confirm my password.	I can get entry to my account / dashboard.	High	Sprint-1
		USN-2	As a user, I will obtain affirmation e mail as soon as I actually have registered for the application.	I can acquire affirmation email & click on affirm.	High	Sprint-1
		USN-3	As a user, I can sign up for the application through Gmail.	I can get affirmation notifications via email.	Medium	Sprint-1
	Login	USN-4	As a user, I can log into the application by entering email & password	I can get entry into my user profile and consider information in the dashboard.	High	Sprint-1
	Dashboard	USN-5	As a user, I can ship the right requests to donate and acquire plasma.	I can acquire suitable notifications via email.	High	Sprint-1
Customer (Web user)	Login	USN-6	As a user, I can sign up and log into the application via e mail & password to view the profile.	I can get entry into my user profile and consider information in dashboard.	High	Sprint-1
	Dashboard	USN-7	As a user, I can ship the right requests to donate and acquire plasma.	I can acquire suitable notifications via email.	High	Sprint-1
Customer Care Executive	Application	USN-8	As a customer care executive,I can try to address user's concerns and questions.	I can view and address their concerns and questions	Medium	Sprint-2
Administrator	Application	USN-9	As an administrator, I can assist with user-going through components of a website, like its appearance, navigation and use of media.	I can change the appearance and navigation in a user friendly manner	Medium	Sprint-3
		USN-10	As an administrator, I can run the technical aspects of websites.	I can assist with inclusive of troubleshooting issues, putting in place net hosts, making sure customers have entry to and programming servers.	Medium	Sprint-1

CHAPTER 6

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint-2	Search Donor	USN-7	As a user, I can search for the donor	9	Medium	Team Lead
Sprint-3	About us	USN-8	As a User, I can view the about us page which contains all contact information	5	Medium	Team Member 2
Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
Sprint-3	Modify data	USN-9	As a admin, I can modify the User data.	9	High	Team Lead
Sprint-3	Send mail	USN-10	As a user, I can send mail to donors using sendgrid.	9	High	Team Lead Team Member 3 Team Member 1
Sprint-3	Home page	USN-11	As a user I can view the home page and select the desired option.	9	Medium	Team Lead Team Member 1 Team Member 2 Team Member 3
Sprint -4	Send Query	USN-12	As a user I can ask my query through email.	9	Medium	Team Lead Team Member 3
Sprint-4	Download data	USN-13	As a admin I can download the user data	9	High	Team Lead

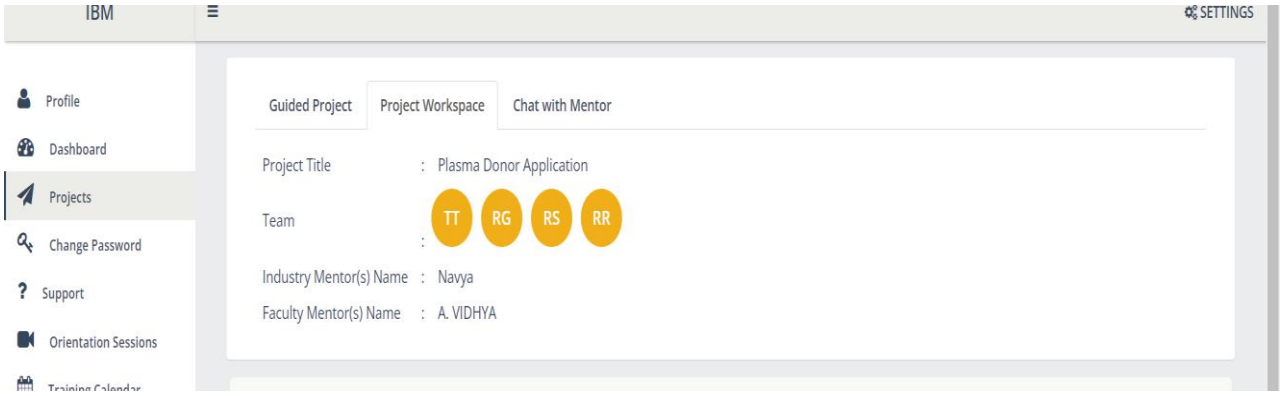
Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	5 Nov 2022

6.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint EndDate (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

6.3 Reports from JIRA



CHAPTER 7

7. CODING & SOLUTIONING

7.1 Feature

```
def sendemail(email,password):

    sg = sendgrid.SendGridAPIClient(api_key="API_KEY")
    from_email = Email("mageshwarannit@gmail.com")
    to_email = To(str(email))
    subject = "Sending with SendGrid is Fun"
    content = Content("text/plain", "your username is " + email + " and password is " + password)
    mail = Mail(from_email, to_email, subject, content)

    mail_json = mail.get()

    response = sg.client.mail.send.post(request_body=mail_json)
    print(response.status_code)
    print(response.headers)


con = ibm_db.connect("DATABASE=bludb;HOSTNAME=9938aec0-8105-433e-8bf9-0fbb7e483086.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32459;Security=SSL;SSLServerCertificate=
@app.route("/",methods=["GET"])
def main():

    return render_template("home.html")

@app.route("/register",methods=["POST"])
def register():
    name = request.form['name']
    dob = request.form['dob']
    phnum = request.form['phnum']
    email = request.form['email']
    password = request.form['pass']

    uniqid = uuid.uuid4().hex

    print(name,dob,phnum,email,password )
```

7.2 Feature

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

    username = request.form['username']
    password = request.form['password']

    sql = """SELECT * FROM "BLN37196"."USER_DETAILS" where email = '{usr}' AND passw = '{pas}';""".format(usr=username,pas=password)

    stmt = ibm_db.exec_immediate(con, sql)
    user = ""
    while ibm_db.fetch_row(stmt) != False:
        user = ibm_db.result(stmt, 2)

    name = ""
    if(user == username):

        sql1 = """SELECT * FROM "BLN37196"."USER_DETAILS" where email = '{usr}';""".format(usr=username)
        stmt1 = ibm_db.exec_immediate(con, sql1)
        uniqid = ""
        while ibm_db.fetch_row(stmt1) != False:
            uniqid = ibm_db.result(stmt1, 4)
            name = ibm_db.result(stmt1, 0)
        print(name)
        session['username'] = username
        session['name'] = name
        session['uniqid'] = uniqid

        return redirect("/dashboard")

    return render_template("login.html")


@app.route("/login",methods=["GET"])
def login_get():
```

```python
@app.route("/register",methods=["POST"])
def register():
    name = request.form['name']
    dob = request.form['dob']
    phnum = request.form['phnum']
    email = request.form['email']
    password = request.form['pass']

    uniqid = uuid.uuid4().hex

    print(name,dob,phnum,email,password )

    sql = """INSERT INTO  "BLN37196"."USER_DETAILS"  VALUES(?,?,?,?,?,?);"""
    stmt = ibm_db.prepare(con, sql)

    # Explicitly bind parameters
    ibm_db.bind_param(stmt, 1, name)
    ibm_db.bind_param(stmt, 2, dob)
    ibm_db.bind_param(stmt, 3, email)
    ibm_db.bind_param(stmt, 4, phnum)
    ibm_db.bind_param(stmt, 5, uniqid)
    ibm_db.bind_param(stmt, 6, password)
    ibm_db.execute(stmt)

    sendemail(email,password)

    return render_template("register.html")

@app.route("/register",methods=["GET"])
def register_get():
    return render_template("register.html")

```

CHAPTER 8

8. TESTING

8.1 Test Cases

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status
1	Functional	Login Page	Verify user is able to Login into the Application		1) Open the Plasma Donor Applicaion 2) Login with user Credentials	Username: Priyanka Password: test	Login Successful	Working as expected	Pass
2	Functional	Signup Page	Verify user is able to Signup in the Application		1) Open the Plasma Donor Applicaion 2) Enter the Details and Create a new User 3) Verify if user is created and	Username: Ayshu Password: test Name: Ayshu DOB: 12/9/2001 Password: test	Account Created Successfully	Working as expected	Pass
3	Functional	Personal Details page	Verify if all the user details are stored in Database		1) Open the Plasma Donor Applicaion 2) Enter the Details and Create a new User 3) Verify if user is created and	Username: chalam@gmail.com password: Testing123	User should navigate to user account homepage		
4	Functional	Login page	Verify user is able to log into application with InValid credentials		1.Enter URL(https://shopenzer.com/) and click go 2.Click on My Account dropdown button 3.Enter InValid username/email in Email text box 4.Enter valid password in password	Username: chalam@gmail password: Testing123	Application should show 'Incorrect email or password ' validation message.		
5	Functional	Login page	Verify user is able to log into application with InValid credentials		1.Enter URL(https://shopenzer.com/) and click go 2.Click on My Account dropdown button 3.Enter Valid username/email in Email text box	Username: chalam@gmail.com password: Testing123678686786876876	Application should show 'Incorrect email or password ' validation message.		

Test Scenarios

1 Verify user is able to see login page

2 Verify user is able to login to application or not?

3 Verify user is able to navigate to create your account page?

4 Verify user is able to recovery password

5 Verify login page elements

Search

1. Verify user is able to search by entering keywords in search box
2. Verify user is able to see suggestions based on keyword entered in search box
3. Verify user is able to see related auto suggestions displaying based on keyword entered in search box
4. Verify user is able to see no matches found message when no results are matching with entered keyword
5. Verify user is able to see search detailed page when nothing entered in textbox

8.2 User Acceptance Testing

Purpose of Document

1. The purpose of this document is to briefly explain the test coverage and open issues of the [Product Name] project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

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

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	5	0	0	0	5
Duplicate	1	0	0	0	1
External	0	0	0	0	0
Fixed	3	0	0	0	3
Not Reproduced	2	0	0	0	2
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	10	0	0	0	10

3. Test Case Analysis

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

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	0	0	0	0
Client Application	5	0	0	5
Security	0	0	0	0
Outsource Shipping	0	0	0	0
Exception Reporting	0	0	0	0

CHAPTER 9

9. RESULTS

9.1 Performance Metrics

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status
1	Functional	Login Page	Verify user is able to Login into the Application		1) Open the Plasma Donor Applicaion 2) Login with user Credentials	Username: Priyanka Password: test	Login Successful	Working as expected	Pass
2	Functional	Signup Page	Verify user is able to Signup in the Application		1) Open the Plasma Donor Applicaion 2) Enter the Details and Create a new User 3) Verify if user is created and	Username: Ayshu Password: test Name: Ayshu DOB: 12/9/2001 Password: test	Account Created Successfully	Working as expected	Pass
3	Functional	Personal Details page	Verify if all the user details are stored in Database		1) Open the Plasma Donor Applicaion 2) Enter the Details and Create a new User 3) Verify if user is created and	Username: chalam@gmail.com password: Testing123	User should navigate to user account homepage		
4	Functional	Login page	Verify user is able to log into application with InValid credentials		1.Enter URL(https://shopenzer.com/) and click go 2.Click on My Account dropdown button 3.Enter InValid username/email in Email text box 4.Enter valid password in password	Username: chalam@gmail password: Testing123	Application should show 'Incorrect email or password ' validation message.		
5	Functional	Login page	Verify user is able to log into application with InValid credentials		1.Enter URL(https://shopenzer.com/) and click go 2.Click on My Account dropdown button 3.Enter Valid username/email in Email text box	Username: chalam@gmail.com password: Testing123678678786876876	Application should show 'Incorrect email or password ' validation message.		

CHAPTER 10

10. ADVANTAGES & DISADVANTAGES

1. ADVANTAGES

The project is identified by the merits of the system offered to the user. The merits of this project are as follows; -

- It's a web-enabled project.
- This project offers user to enter the data through simple and interactive forms. This is very helpful for the client to enter the desired information through so much simplicity.
- The user is mainly more concerned about the validity of the data, whatever he is entering. There are checks on every stages of any new creation, data entry or updation so that the user cannot enter the invalid data, which can create problems at later date.
- Sometimes the user finds in the later stages of using project that he needs to update some of the information that he entered earlier. There are options for him by which he can update the records. Moreover there is restriction for his that he cannot change the primary data field. This keeps the validity of the data to longer e0tent.
- User is provided the option of monitoring the records he entered earlier. He can see the desired records with the variety of options provided by him.
- From every part of the project the user is provided with the links through framing so that he can go from one option of the project to other as per the requirement. This is bound to be simple and very friendly as per the user is concerned. That is" we can say that the project is user friendly which is one of the primary concerns of any good project.
- Data storage and retrieval will become faster and easier to maintain because data is stored in a systematic manner and in a single database.
- Decision making process would be greatly enhanced because of faster processing of information since data collection from information available on computer takes much less time then manual system.
- Allocating of sample results becomes much faster because at a time the user can see the records of last years.
- Easier and faster data transfer through latest technology associated with the computer and communication.
- Through these features it will increase the efficiency, accuracy and transparency.

2. DISADVANTAGES

- Wrong inputs will affect the project outputs.
- Internet Connection is mandatory.
- Reports are not Verified

CHAPTER 11

11. CONCLUSION

This project proved good for me as it provided practical knowledge of not only programming in ASP.NET and VB.NET web based application and no some extent windows Application and SQL Server, but also about all handling procedure related with "Plasma Donor Application". It also provides knowledge about the latest technology used in developing web enabled application and client server technology that will be great demand in future. This will provide better opportunities and guidance in future in developing projects independently.

CHAPTER 12

12. FUTURE SCOPE

Plasma Donor Application is a web application to build such a way that it should suits for all type of blood banks in future. One important future scope is availability of location-based blood bank details and extraction of location-based donor's detail, which is very helpful to the acceptant people. All the time the network facilities cannot be use. This time donor request does not reach in proper time, this can be avoided through adding some message sending procedure this will help to find proper blood donor in time. This will provide availability of blood in time.

SOURCE CODE

```
 from flask import Flask, jsonify,
request,
make_response,render_template,redirect
from flask import * import ibm_db import
uuid import hashlib import os from
sendgrid import SendGridAPIClient from
sendgrid.helpers.mail import Mail
 import sendgrid import os from sendgrid.helpers.mail
import Mail, Email, To, Content
 app= Flask(__name__) app.config['SECRET_KEY'] = 'the quick
brown fox jumps over the lazy   dog'
app.config['CORS_HEADERS'] = 'Content-Type'
```

```
def sendemail(email,password):
     sg = sendgrid.SendGridAPIClient(api_key="API_KEY")
from_email = Email("mageshwarannit@gmail.com")      to_email =
To(str(email))      subject = "Sending with SendGrid is Fun"
content = Content("text/plain", "your username is " + email + " and
password is " + password)     mail = Mail(from_email, to_email,
subject, content)
     mail_json =
mail.get()
     response =
sg.client.mail.send.post(request_body=mail_json)
print(response.status_code)     print(response.headers)
 con = ibm_db.connect("DATABASE=bludb;HOSTNAME=9938aec0-8105-
433e8bf9-
0fbb7e483086.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=324
59;Security=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=bl
n37196;PWD=HJG0wr88Ysyrv41B;","","") @app.route("/",methods=["GET"])
def main():

print(con)

print("hello")

    # sendemail("","")
     return
render_template("home.html")

@app.route("/register",methods=["POST"]) def
register():
    name = request.form['name']
dob = request.form['dob']     phnum
= request.form['phnum']     email =
request.form['email']     password
= request.form['pass']
     uniqid =
uuid.uuid4().hex
```

```
    print(name,dob,phnum,email,password )

    sql = """INSERT
INTO   "BLN37196"."USER_DETAILS"   VALUES(?,?,?,?,?,?);"""
stmt = ibm_db.prepare(con, sql)
     ibm_db.bind_param(stmt, 1, name)
ibm_db.bind_param(stmt, 2, dob)
ibm_db.bind_param(stmt, 3, email)
ibm_db.bind_param(stmt, 4, phnum)
ibm_db.bind_param(stmt, 5, uniqid)
ibm_db.bind_param(stmt, 6, password)
ibm_db.execute(stmt)

sendemail(email,password)
     return
render_template("register.html")

@app.route("/register",methods=["GET"]) def
register_get():
     return
render_template("register.html")

@app.route("/dashboard",methods=["get"]) def
dashboard():
         uid = str(session.get("uniqid")+'')     sql =
f"""select * from "BLN37196"."REQUEST" Where
"UNIQID"!='{uid}' AND "status"='waiting'"""
     sql +=
";"


    arr = []
larr = []
barr = []
harr = []


    stmt = ibm_db.exec_immediate(con, sql)
tuple = ibm_db.fetch_tuple(stmt)
```

```
    while tuple != False:
arr.append(tuple)         tuple =
ibm_db.fetch_tuple(stmt)
     print("arr")     print(arr)     print(larr)     print(harr)
print(barr)     return
render_template("dashboard.html",requestarray=arr,locarr=larr,bgarr=
barr,hosarr=harr)

@app.route("/changestatus/<id>",methods=["get"]) def
chngstatus(id):
     print(id)     uid = str(session.get("uniqid"))+""     name =
str(session.get("name"))+""     sql = f"""UPDATE
"BLN37196"."REQUEST" SET "donorid" = '{uid}',
"donorname" = '{name}',"status"='accepted' WHERE "FUNIQID" =
'{id}';"""     stmt =
ibm_db.prepare(con, sql)
ibm_db.execute(stmt)
print("suc")
     return
"success"



@app.route("/requestform",methods=["get"]) def
reqform_get():
    return render_template("form.html")

@app.route("/requestform",methods=["post"]) def
reqform_post():
    name = request.form['name']
bg = request.form['bg']     loc
= request.form['loc']     hosp
= request.form['hosp']
```

```
    formid = (hashlib.sha1((uuid.uuid4().hex +
session.get("uniqid")).encode())).hexdigest() + ""
print(formid)     uid = str(session.get("uniqid"))
+ ""
         sql = f"""INSERT INTO
"BLN37196"."REQUEST"
("UNIQID","FUNIQID","NAME","BG","LOC","HOSP","status")
VALUES('{uid}','{formid}','{name}','{bg}','{loc}','{hosp}','waiting'
);"""       stmt =
ibm_db.prepare(con, sql)
ibm_db.execute(stmt)
     return
redirect("/dashboard")



@app.route("/login",methods=["POST"]) def
login():
     username =
request.form['username']     password =
request.form['password']
     sql = """SELECT * FROM "BLN37196"."USER_DETAILS" where email
=
'{usr}' AND passw = '{pas}';""".format(usr=username,pas=password)
     stmt = ibm_db.exec_immediate(con,
sql)     user = ""     while
ibm_db.fetch_row(stmt) != False:
        user = ibm_db.result(stmt, 2)
     name = ""
if(user == username):
         sql1 = """SELECT * FROM "BLN37196"."USER_DETAILS"
where email = '{usr}';""".format(usr=username)         stmt1 =
ibm_db.exec_immediate(con, sql1)         uniqid = ""
while ibm_db.fetch_row(stmt1) != False:             uniqid =
ibm_db.result(stmt1, 4)             name = ibm_db.result(stmt1,
0)         print(name)
```

```
        session['username'] = username
session['name'] = name          session['uniqid']
= uniqid
         return
redirect("/dashboard")
     return
render_template("login.html")

@app.route("/login",methods=["GET"]) def
login_get():
     print(con)     return
render_template("login.html")


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

```
     ibm_db.bind_param(prep_stmt, 5, mnumb)
ibm_db.bind_param(prep_stmt, 6, gender)
ibm_db.bind_param(prep_stmt, 7, address)
ibm_db.bind_param(prep_stmt, 8, pin)
ibm_db.execute(prep_stmt)
       return render_template('donlogin.html', msg="Account has been
created successfully..")
   return
"success..."

@app.route('/admin') def
admin():
  return render_template('admin.html')
@app.route('/donar') def
donar():
  return render_template('donar.html')

## donar registering for donation
@app.route('/giveplasma',methods = ['POST',
'GET']) def giveplasma():   if request.method ==
'POST':
     name = request.form['name']     age =
request.form['age']     gender =
request.form['gender']     mnumb =
request.form['mnumb']     email =
request.form['email']     city =
request.form['city']     address =
request.form['address']     bloodgroup =
request.form['bloodgroup']     issue =
request.form['issue']     lastbd =
request.form['lastbd']     slot =
request.form['slot']

    sql = "SELECT * FROM donar WHERE name =?"
stmt = ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt,1,name)
ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)
     if
account:
      return render_template('donlogin.html', msg="Your request for
donation is successfully submitted..")     else:
      insert_sql = "INSERT INTO donar VALUES
(?,?,?,?,?,?,?,?,?,?,?)"
```

```
      prep_stmt = ibm_db.prepare(conn, insert_sql)
ibm_db.bind_param(prep_stmt, 1, name)
ibm_db.bind_param(prep_stmt, 2, age)
ibm_db.bind_param(prep_stmt, 3, gender)
ibm_db.bind_param(prep_stmt, 4, mnumb)
ibm_db.bind_param(prep_stmt, 5, email)
ibm_db.bind_param(prep_stmt, 6, city)
ibm_db.bind_param(prep_stmt, 7, address)
ibm_db.bind_param(prep_stmt, 8, bloodgroup)
ibm_db.bind_param(prep_stmt, 9, issue)
ibm_db.bind_param(prep_stmt, 10, lastbd)
ibm_db.bind_param(prep_stmt, 11, slot)
ibm_db.execute(prep_stmt)
         return render_template('donar.html', msg="Your request for
donation is successfully submitted..")

@app.route('/plasmadon')
def plasmadon():
  donar = []   sql = "SELECT * FROM
donar"   stmt =
ibm_db.exec_immediate(conn, sql)
dictionary = ibm_db.fetch_both(stmt)
while dictionary != False:
    # print ("The Name is : ",
dictionary)     donar.append(dictionary)
dictionary = ibm_db.fetch_both(stmt)
   if
donar:
    return render_template("plasmadon.html", donar = donar)
@app.route('/delete/<name>') def
delete(name):
  sql = f"SELECT * FROM donar WHERE
name='{escape(name)}'"   print(sql)   stmt =
ibm_db.exec_immediate(conn, sql)   donar =
ibm_db.fetch_row(stmt)   print ("The Name is : ",  donar)
if donar:
    sql = f"DELETE FROM donar WHERE name='{escape(name)}'"
print(sql)
    stmt = ibm_db.exec_immediate(conn, sql)
     donar = []     sql = "SELECT * FROM
donar"     stmt =
ibm_db.exec_immediate(conn, sql)
dictionary = ibm_db.fetch_both(stmt)
while dictionary != False:
```

```
      donar.append(dictionary)       dictionary =
ibm_db.fetch_both(stmt)     if donar:       return
render_template("plasmadon.html", donar = donar, msg="Delete
successfully")


  # # while student != False:
  # #   print ("The Name is : ",  student)

  # print(student)
return "success..."

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

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

@app.route('/takeplasma',methods = ['POST',
'GET']) def takeplasma():   if request.method ==
'POST':
     name = request.form['name']
age = request.form['age']     gender
= request.form['gender']     mnumb =
request.form['mnumb']     proof =
request.form['proof']     address =
request.form['address']     plasma =
request.form['plasma']


    sql = "SELECT * FROM recipient WHERE name =?"
stmt = ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt,1,name)
ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)
     if
account:
      return render_template('reclogin.html', msg="You are already a
member, please login using your details")     else:
      insert_sql = "INSERT INTO recipient VALUES (?,?,?,?,?,?,?)"
prep_stmt = ibm_db.prepare(conn, insert_sql)
```

```
      ibm_db.bind_param(prep_stmt, 1, name)
ibm_db.bind_param(prep_stmt, 2, age)
ibm_db.bind_param(prep_stmt, 3, gender)
ibm_db.bind_param(prep_stmt, 4, mnumb)
ibm_db.bind_param(prep_stmt, 5, proof)
ibm_db.bind_param(prep_stmt, 6, address)
ibm_db.bind_param(prep_stmt, 7, plasma)
ibm_db.execute(prep_stmt)

    return     render_template('recipient.html',     msg="Registration
succesfull for Plasma request..")


@app.route('/plasmareq') def
plasmareq():   recipient = []   sql =
"SELECT * FROM recipient"   stmt =
ibm_db.exec_immediate(conn, sql)
dictionary = ibm_db.fetch_both(stmt)
while dictionary != False:
    # print ("The Name is : ",  dictionary)
recipient.append(dictionary)
    dictionary = ibm_db.fetch_both(stmt)
   if
recipient:
    return render_template("plasmareq.html", recipient = recipient)

@app.route('/delete/<name>') def
deleted(name):
  sql = f"SELECT * FROM recipient WHERE name='{escape(name)}'"
print(sql)
  stmt = ibm_db.exec_immediate(conn, sql)   recipient =
ibm_db.fetch_row(stmt)   print ("The Name is : ",  recipient)
if recipient:     sql = f"DELETE FROM recipient WHERE
name='{escape(name)}'"     print(sql)
    stmt = ibm_db.exec_immediate(conn, sql)
     recipient = []     sql = "SELECT *
FROM recipient"     stmt =
ibm_db.exec_immediate(conn, sql)
dictionary = ibm_db.fetch_both(stmt)
while dictionary != False:
      recipient.append(dictionary)
dictionary = ibm_db.fetch_both(stmt)     if
recipient:
```

```
      return render_template("plasmareq.html", recipient = recipient,
msg="Delete successfully")
     return "Deleted
Successfully"


if __name__ == "__main__":
    app.run(port=5000, host="0.0.0.0", debug=True)
```

TEMPLATES >

Index.html

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

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- CSS only -->
    <link
href="https://fonts.googleapis.com/css?family=Merriweather&display=swap"
rel="stylesheet">
    <link rel="shortcut icon" href="assets/images/fav.jpg">
    <link rel="stylesheet" href="../static/bootstrap.min.css">
    <link rel="stylesheet" href="../static/fontawsom-all.min.css">
    <link rel="stylesheet" href="../static/grid-gallery.min.css">
    <link rel="stylesheet" href="../static/grid-gallery.css">
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.cs
s" rel="stylesheet"
integrity="sha384iYQeCzEYFbKjA/T2uDLTpkwGzCiq6soy8tYaI1GyVh/UjpbCx/TYkiZhlZB
6+fzT" crossorigin="anonymous">
    <link rel="stylesheet" type="text/css" href="../static/style.css" />
    <title>Home page</title>

</head>

<body>
    <div class="loader_bg">
        <div class="loader"></div>
      </div>



    <header class="p-3 text-bg-dark">
        <div class="container">
```

```
            <div class="d-flex flex-wrap align-items-center justify-
contentcenter justify-content-lg-start">
                <a href="/" class="d-flex align-items-center mb-2 mb-
lg-0 text-white text-decoration-none">
                    <svg class="bi me-2" width="40" height="32"
role="img" aria-label="Bootstrap"><use xlink:href="#bootstrap"/></svg>
</a>

                <ul  class="nav  col-12  col-lg-auto  me-lg-auto  mb-2
justifycontent-center mb-md-0">
                    <li><a href="/" class="nav-link px-2
textwhite"><b>Home</b></a></li>   
                    <li><a href="#" class="nav-link px-2
textwhite"><b>About</b></a></li>   
                    <li><a href="#" class="nav-link px-2
textwhite"><b>Blogs</b></a></li>   
                    <li><a href="#" class="nav-link px-2
textwhite"><b>Camps</b></a></li>
                </ul>
                <div class="text-end">
                    <button type="button" class="btn btn-outline-
secondary"><a href="/adminlogin">Admin Login</a></button>
                    <button   type="button"   class="btn   btn-outline-
info"><a              href="/recipientlogin">Looking              for
plasma?</a></Looking></button>
                    <button type="button" class="btn btn-outline-
warning"><a href="/donarlogin">Donate Now!</a></button>
                </div>
            </div>
        </div>
    </header>
    <div class="slider-detail">

        <div id="carouselExampleIndicators" class="carousel slide"
dataride="carousel">
            <ol class="carousel-indicators">
                <li   data-target="#carouselExampleIndicators"   data-
slide-to="0" class="active"></li>
                <li   data-target="#carouselExampleIndicators"   data-
slideto="1"></li>

            </ol>
            <div class="carousel-inner">
                <div class="carousel-item active">
                    <img class="d-block w-100"
src="https://model001.s3.jptok.cloud-object-
storage.appdomain.cloud/slide-03.jpg" alt="First slide">
<div class="carousel-caption d-none d-md-block">
                        <h5 class=" bounceInDown">Donate Plasma & Save
```

```
        </div>


    </div>

    <!--*************** About Us Starts Here ***************-->
   <section id="about" class="contianer-fluid about-us">
       <div class="container">
           <div class="row session-title">
<h2><u>About Us</u></h2>
               <p> text will  be added</p>
           </div>
            <div class="row">
                <div class="col-md-6 text">
                    <h2>About Plasma Donars</h2>
                    <p>when a patient needs plasma, he/she has to
contact a Medical center or a compatible blood group of a donor in
their circle, family, and friends. However, it is difficult to find
suitable donor within a limited group of people in a given time. In
addition, there is no guarantee that Medical center will have
compatible plasma in stock. There is also steady increase in plasma
donation requests posts in social networking sites (like
Facebook, twitter, Instagram, etc.) requesting for donation.</p>
                    <p>Ease of access, requirements of plasma, and
the plasma donation statistics are taken into consideration while
researching the topic.
There is a steady need for plasma.</p>
                    <p>Although this application helps finding
donors, but the ease of communication with those donors is not prompt
and it requires man power as the requester (patient or clinic) has to
contact each donor individually. Also, there is no application that
provides a proper communication channel to notify donors about the
plasma donation requirements.</p>
                    <p>Our application provides donors with
functionalities including “plasma request”, “Ask for donation”, “share
with friend”,(slot alloted to donate plasma), at the same time the
recipient can send requests and use this application to maintain the
donation activities.</p>                </div>
                <div class="col-md-6 image">
                    <img        src="https://model001.s3.jp-tok.cloud-
objectstorage.appdomain.cloud/about.jpg" alt="">
                </div>
            </div>
       </div>
   </section>



     <!-- ################## Gallery Start Here ########################--->
```

```
<div class="container">
            <div class="row session-title">
                <h2><u>Checkout Our Gallery</u></h2>
            </div>
            <div class="gallery-row row">
                    <div id="gg-screen"></div>
                    <div class="gg-box">
                            <div class="gg-element">
                                <img src="https://model001.s3.jp-
tok.cloudobject-storage.appdomain.cloud/g1.jpg">
                            </div>
                            <div class="gg-element">
                                <img src="https://model001.s3.jp-
tok.cloudobject-storage.appdomain.cloud/g2.jpg">
                            </div>
                            <div class="gg-element">
                                <img src="https://model001.s3.jp-
tok.cloudobject-storage.appdomain.cloud/g3.jpg">
                            </div>
                            <div class="gg-element">
                                <img src="https://model001.s3.jp-
tok.cloudobject-storage.appdomain.cloud/g4.jpg">
                            </div>
                            <div class="gg-element">
                                <img src="https://model001.s3.jp-
tok.cloudobject-storage.appdomain.cloud/g5.jpg">
                            </div>
                            <div class="gg-element">
                                <img src="https://model001.s3.jp-
tok.cloudobject-storage.appdomain.cloud/g6.jpg">
                            </div>
                            <div class="gg-element">
                                <img src="https://model001.s3.jp-
tok.cloudobject-storage.appdomain.cloud/g7.jpg">
                            </div>
                            <div class="gg-element">
                                <img src="https://model001.s3.jp-
tok.cloudobject-storage.appdomain.cloud/g8.jpg">
                            </div>
                            <div class="gg-element">
                                <img src="https://model001.s3.jp-
tok.cloudobject-storage.appdomain.cloud/g9.jpg">
                            </div>
                            <div class="gg-element">
                                <img src="https://model001.s3.jp-
tok.cloudobject-storage.appdomain.cloud/g10.jpg">
                            </div>
```

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

```
    <!-- ################## Donation Process Start Here
########################--->
```

```
    <section id="process" class="donation-care">
        <div class="container">
          <div class="row session-title">
              <h2><u>Donation Process</u></h2>
              <!-- <p><b>The donation process from the time you arrive center until
the time you leave.</b></p> -->
          </div>
           <div class="row">
                <div class="col-md-3 col-sm-6 vd">
                   <div class="bkjiu">
                    <img src="https://model001.s3.jp-tok.cloud-
objectstorage.appdomain.cloud/g1.jpg" alt="">
                    <h4><b>1 - </b>Registration</h4>
                    <p>When you arrive at a plasma center, you will check in at the
front desk. You will need to show a valid photo ID, proof of address, and proof of
social security.</p>
                    <button class="btn btn-sm btn-danger"><a href="#">Readmore </a><i
class="fas fa-arrow-right"></i></button>
                    </div>
                </div>
                <div class="col-md-3 col-sm-6 vd">
                   <div class="bkjiu">
                    <img src="https://model001.s3.jp-tok.cloud-
objectstorage.appdomain.cloud/g2.jpg" alt="">
                       <h4><b>2 - </b>Screening</h4>
                    <p>During the screening, you will give a blood sample and get your
vital signs checked, including your blood pressure, pulse, and temperature</p>
                    <button class="btn btn-sm btn-danger">Readmore <i class="fas fa-
arrow-right"></i></button>
                    </div>
                </div>
                <div class="col-md-3 col-sm-6 vd">
                   <div class="bkjiu">
                    <img src="https://model001.s3.jp-tok.cloud-
objectstorage.appdomain.cloud/g4.jpg" alt="">
                        <h4><b>3 - </b>Physical Exam</h4>
                            <p> The first time you give plasma, you will receive a
```

```
brief physical exam given by a trained medical specialist to make sure you stay in
good health.</p>
                    <button class="btn btn-sm btn-danger">Readmore <i class="fas fa-
arrow-right"></i></button>
                    </div>
                </div>
                <div class="col-md-3 col-sm-6 vd">
                    <div class="bkjiu">
                        <img src="https://model001.s3.jp-tok.cloud-
objectstorage.appdomain.cloud/g1.jpg" alt="">
                        <h4><b>4 - </b>Donation</h4>
                         <p>After approval, plasma center staff will set you up at a
plasmapheresis machine that collects whole blood from a vein in your arm and it
separates out the plasma.</p>
                         <button class="btn btn-sm btn-danger">Readmore <i class="fas
fa-arrow-right"></i></button>
                    </div>
                </div>
            </div>
```

```
        </div>
    </section>
            <!--################### Our Blog Starts Here
#######################--->
            <div id="blog" class="blog-container contaienr-fluid">
              <div class="container">
                  <div class="session-title row">
                    <h2><u>Latest Blog</u></h2>
                    <!-- <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Fusce fringilla vel nisl a dictum. Donec ut est arcu. Donec hendrerit velit
consectetur adipiscing elit.</p> -->
                  </div>
                  <div class="row news-row">
                      <div class="col-md-6">
                          <div class="news-card">
                              <div class="image">
                                  <img src="https://model001.s3.jptok.cloud-object-
storage.appdomain.cloud/blog_01.jpg" alt="">
                              </div>
                              <div class="detail">
                                  <h3>Give Thanks, Give Blood</h3>
<p>Blood donors share life. And for that, thousands of people are thankful that blood
donors give generously. After donating blood,we wants to thank our loyal platelet
donors with a t-shirt they can wear loud and proud... </p>
                                  <p class="footp">
                                  10 Comments <span>/</span>
                                  Blog Design <span>/</span>
                                  Read More
                                  </p>
```

```
                            </div>
                        </div>
                    </div>
                    <div class="col-md-6">
                        <div class="news-card">
                            <div class="image">
                                <img src="https://model001.s3.jptok.cloud-object-
storage.appdomain.cloud/blog_02.jpg" alt="">
                            </div>
                            <div class="detail">
                                <h3>Donar Celebrate Milestone</h3>
<p>A few Greenwood donors have gone above and beyond in their blood donation journeys
to save hundreds of local lives! These donors are great examples of loyal, local
lifesavers Thank you for your continuous blood donations!</p>
                                <p class="footp">
                                    17 Comments <span>/</span>
                                    Blog Design <span>/</span>
                                    Read More
                                </p>
                            </div>
                        </div>
                    </div>
                    <div class="col-md-6">
                        <div class="news-card">
                            <div class="image">
                                <img src="https://model001.s3.jptok.cloud-object-
storage.appdomain.cloud/blog_03.jpg" alt="">
                            </div>
                            <div class="detail">
                                <h3>Plasma Donation Do’s and Don’ts</h3>
<p>Whether you’re a new or returning plasma donor, or someone who is curious about
giving plasma, you probably have some questions about the donation process. In this
blog, we’re breaking down the do’s and don’ts of plasma donation... </p>
                                <p class="footp">
                                    09 Comments <span>/</span>
                                    Blog Design <span>/</span>
                                    Read More
                                </p>
                            </div>
                        </div>
                    </div>
                    <div class="col-md-6">
                        <div class="news-card">
```

```
  position: relative;   display: inline-
block;   cursor: pointer;   text-align:
center;
} input[type="submit"]:hover {   background-
color: #329dd5;   -webkit-transition: all
0.2s ease;   transition: all 0.2s ease;
}

#create-account-wrap {   background-
color: #eeedf1;   color: #8a8b8e;
font-size: 14px;   width: 100%;
padding: 10px 0;   border-radius: 0 0
4px 4px;
}  button a:link {
  text-decoration: none;
} button a:visited { text-decoration: none; color:#ffffff;}
 button a:hover { text-decoration: none; color:#ffffff; }
 a:active { text-decoration: none; }
 button{   background-color: #3ca9e2;
border-color: #3ca9e2;   border-
radius: 5px;   color: white;
}  h4{   color: white;
}
```

```
  background-color: none;
border-collapse: collapse;
border: none;   border-
radius: 15px;
}
table.inner{
border: 0px
}  form{   margin-top :30px;   width: 35rem;
height: 35rem;   display: flex;   flex-
direction: column;   background:
rgba(255,255,255,0.06);   box-shadow: 0 8px
32px 0 rgba(31,38,135,.40);   border-radius:
35px;   border: 1px solid
rgba(255,255,255,0.3);

}

button a:visited { text-decoration: none; color:#000000;}
 button a:hover { text-decoration: none; color:#000000;
}
 a:active { text-decoration: none;
}
 a:link { text-decoration: none;} a:visited {
text-decoration: none; color:#000000;} a:hover {
text-decoration: none; color:#000000; }
 p{   color:
white;
}
```

Js > Grid-gallery.js

```
$(document).on('click','.gg-
element',function(){   var selected=$(this);
var prev=$(this).prev().find('img');   var
next=$(this).next().find('img');
  $('#gg-screen').show();   var l=$(".gg-element").length-1;   var
p=$(".gg-element").index(selected);   function buttons(){     if (l >
1) {       if (p == 0){         return '<div class="gg-close gg-
bt">&times</div><div class="gg-nxt ggbt">&rarr;</div>';
      }       else if (p == l) {         return '<div class="gg-close gg-
bt">&times</div><div class="gg-prev gg-bt">&larr;</div>';
      }          else{             return  '<div  class="gg-close  gg-
bt">&times</div><div  class="gg-nxt  ggbt">&rarr;</div><div  class="gg-
prev gg-bt">&larr;</div>';
      }     }     else{       return '<div class="gg-
close gg-bt">&times</div>';     }   }   buttons();
var content=buttons();
  $("#gg-screen").html('<div class="gg-image"></div>' + content);
  $(".gg-image").html('<img src="'+ $('img', this).attr('src')
+'">');
  $("body").css('overflow','hidden');
  $(document).on('click','.gg-close',function(){
    $("#gg-screen").hide();
    $("body").css('overflow','auto');
  });
  $("#gg-screen").on('click', function(e) {
if (e.target == this){
      $("#gg-screen").hide();
      $("body").css('overflow','auto');
    }
  });
  $(document).on('click','.gg-prev',function(){
selected=selected.prev();
prev=selected.find('img');
```

```
    var previmg='<img src="'+ prev.attr('src') +'">';
    $(".gg-image").html(previmg);     p=$(".gg-
element").index(selected);     buttons();
    content=buttons();
    $("#gg-screen").html('<div class="gg-image">'+ previmg + '</div>'
+ content);
  });
  $(document).on('click','.gg-nxt',function(){
selected=selected.next();
next=selected.find('img');     var nxtimg='<img src="'+
next.attr('src') +'">';
    $(".gg-image").html(nxtimg);     p=$(".gg-
element").index(selected);     buttons();
    content=buttons();
    $("#gg-screen").html('<div class="gg-image">'+ nxtimg + '</div>' +
content);
  });
  $(document).on('keydown',function(e) {     if(e.keyCode
== 37 && p>0) {       selected=selected.prev();
prev=selected.find('img');       var previmg='<img src="'+
prev.attr('src') +'">';
      $(".gg-image").html(previmg);       p=$(".gg-
element").index(selected);       buttons();
      content=buttons();
      $("#gg-screen").html('<div class="gg-image">'+ previmg + '</div>' +
content);     }     else if(e.keyCode == 39 && p < l) {
selected=selected.next();       next=selected.find('img');
      var nxtimg='<img src="'+ next.attr('src') +'">';
      $(".gg-image").html(nxtimg);       p=$(".gg-
element").index(selected);       buttons();
      content=buttons();
      $("#gg-screen").html('<div class="gg-image">'+ nxtimg + '</div>'
+ content);
    }
  });
});
```

Grid-gallery-min.js

```
$(document).on("click",".gg-element",function(){var c=$(this);var
f=$(this).prev().find("img");var
b=$(this).next().find("img");$("#gg-
screen").show();var a=$(".gg-element").length-1;var g=$(".gg-
element").index(c);function e(){if(a>1){if(g==0){return'<div class="gg-close
gg-bt">&times</div><div class="gg-nxt gg-
bt">&rarr;</div>'}else{if(g==a){return'<div class="gg-close gg-
bt">&times</div><div class="gg-prev gg-
bt">&larr;</div>'}else{return'<div
class="gg-close gg-bt">&times</div><div class="gg-nxt gg-bt">&rarr;</div><div
class="gg-prev gg-bt">&larr;</div>'}}}else{return'<div class="gg-
close gg-
bt">&times</div>'}}e();var d=e();$("#gg-screen").html('<div
class="gg-
image"></div>'+d);$(".gg-image").html('<img
src="'+$("img",this).attr("src")+'">');$("body").css("overflow","hidden");$(do
cument).on("click",".gg-close",function(){$("#gg-
screen").hide();$("body").css("overflow","auto")});$("#gg-
screen").on("click",function(h){if(h.target==this){$("#gg-
screen").hide();$("body").css("overflow","auto")}});$(document).on("click",".g
g-prev",function(){c=c.prev();f=c.find("img");var h='<img
src="'+f.attr("src")+'">';$(".gg-image").html(h);g=$(".gg-
element").index(c);e();d=e();$("#gg-screen").html('<div class="gg-
image">'+h+"</div>"+d)});$(document).on("click",".gg-
nxt",function(){c=c.next();b=c.find("img");var h='<img
src="'+b.attr("src")+'">';$(".gg-image").html(h);g=$(".gg-
element").index(c);e();d=e();$("#gg-screen").html('<div class="gg-
image">'+h+"</div>"+d)});$(document).on("keydown",function(j){if(j.keyCode==37
&&g>0){c=c.prev();f=c.find("img");var i='<img
src="'+f.attr("src")+'">';$(".gg-image").html(i);g=$(".gg-
element").index(c);e();d=e();$("#gg-screen").html('<div class="gg-
image">'+i+"</div>"+d)}else{if(j.keyCode==39&&g<a){c=c.next();b=c.find("img");
var h='<img src="'+b.attr("src")+'">';$(".gg-
image").html(h);g=$(".gg-
element").index(c);e();d=e();$("#gg-screen").html('<div class="gg-
image">'+h+"</div>"+d)}}})});
```

Script.js

```
$( document ).ready(function() {
var w = window.innerWidth;

    if(w > 767){
        $('#menu-jk').scrollToFixed();
    }else{
        $('#menu-jk').scrollToFixed();
    }



})
```

CHAPTER 13

APPENDIX

GITHUB LINK - https://github.com/IBM-EPBL/IBM-Project- 1974-1658421777

PROJECT DEMO LINK : https://drive.google.com/file/d/1KLRklK4-f4UHSoCOWEfwZIIE1-Yb2Udg/view?usp=share_link