

PROJECT REPORT

Plasma Donor Application

Team ID :PNT2022TMID01682

Team Members :

- . AJAY KOUSHIK K N
- . NAVEEN PRASANTH M
- . NITHARSHAN D J
- . PRAVIN M

TABLE OF CONTENT

Sl.No	TITLE	Page No
1	INTRODUCTION	4
	1.1 Project Overview	4
	1.2 Purpose	4
2	LITERATURE SURVEY	6
3	IDEATION & PROPOSED SOLUTION	8
	3.1 Empathy Map	8
	3.2 Ideation & Brainstorming	9
	3.3 Proposed Solution	11
	3.4 Problem Solution fit	11
4	REQUIREMENT ANALYSIS	13
	4.1 Functional requirement	13
	4.2 Non-Functional requirements	14
5	PROJECT DESIGN	15
	5.1 Data Flow Diagrams	15
	5.2 Solution & Technical Architecture	16
	5.3 User Stories	16
6	PROJECT PLANNING & SCHEDULING	17
	6.1 Sprint Planning & Estimation	17
	6.2 Sprint Delivery Schedule	17
7	CODING & SOLUTIONING	18
8	TESTING	26
	8.1 Test Cases	26
	8.2 User Acceptance Testing	27
9	RESULTS	28
	9.1 Software Test Metrics	28
	9.2 Performance Metrics	28
10	ADVANTAGES & DISADVANTAGES	29

11	CONCLUSION	30
12	FUTURE SCOPE	31
13	APPENDIX	32

INTRODUCTION

1.1 PROJET OVERVIEW:

A way in which one can help the COVID 19 affected people 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 is considered to be safe and promising. If a particular person is fully recovered from COVID 19 he/she is eligible 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.

The Plasma donor app is to create details about the donor and organizations that are related to donating the blood. Through this application any person who is interested in donating blood can register himself in the same way if any organization wants to register itself with this site that can also register. Moreover if any general consumer wants to request blood online he can also take the help of this site. Admin is the main authority who can do addition" deletion" and modification if required.

1.2 PURPOSE:

The goal of the project is to develop a web application for plasma banks to manage information about their donors and plasma stock. The main objectives of this website development can be defined as follows:

- To develop a system that provides functions to support donors to view and manage their information conveniently.
- To maintain records of plasma donors, plasma donation information and plasma stock in a centralized database system.

- To support searching, matching and requesting for blood convenient for administrators.
- To provide a function to send an email directly to the donor for their user account.

LITERATURE SURVEY

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 recover 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.

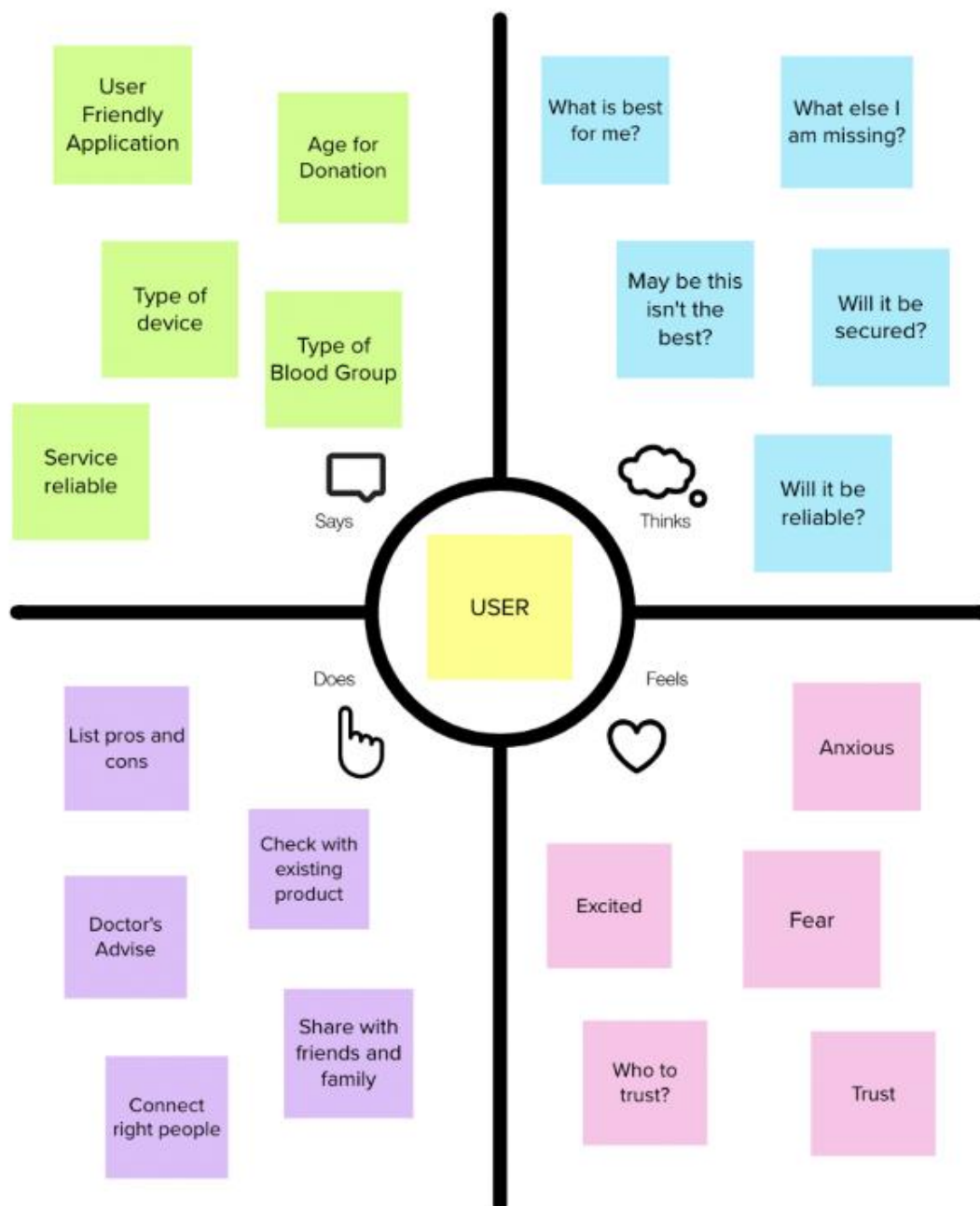
- Denuis O'Neil (1999). "Blood component" Archived from the original on June 5, 2013. Normally, certain amount of human body weight comes from blood. For adults, it is 4-6 litres of blood. This essential liquid plays an important role in transporting oxygen and nutrients to cells and removing carbon dioxide, ammonia and other waste products. Blood is a very common tissue composed of over 4000 different types of components.
- Ways to keep your plasma healthy, Original Archived November 1, 2013, Accessed November 11, 2011. Plasma donation is one of the most accepted practices for saving lives, While earning a few dollars. The whole process can take some time, but it's well worth it once you experience it a few times. Accepting money in exchange for plasma is welcome. It's a move when you feel like you're not just a hero, but you're adding value to yourself. The term "healthy" does not mean only in the absence of disease. It also means that you are healthy enough.

- Ripathis S, Kumar V, Prabhakar A, Joshi S, Agarwal A (2015). "Microscale Passive Plasma Separation: A Review of Design Principles and Microdevices," J. Micromech Micro 25 (8): 083001; Plasma separation is of great importance in the fields of diagnosis and healthcare. Due to the lagging transition to microscale, these recent trends are a rapid shift towards shrinking complex macro processes.
- Published in International Research Journal of Modernization in Engineering Technology and Science - e-ISSN: 2582-5208.

IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviors and attitudes. It is a useful tool to help teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.



3.2 IDEATION AND BRAINSTORMING:

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions. Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

Team page



Brainstorm & Idea prioritization

Plasma Donor Application
Team Members:
Ajay Koushik K N
Nisharshan D J
Naveen Prasanth M
Pravin M

15 minutes to prepare
1 hour to collaborate
3-5 people recommended

Show template feedback

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

10 minutes

Team gathering
Define who should participate in the session and send an invite. Share relevant information in general ahead.

Get the goal
Think about the problem you'll be focusing on solving in the brainstorming session.

Learn how to use the facilitation tools
Use the Facilitation Superpowers to run a happy and productive session.

Open article

Define your problem statement
Although the government is carrying out Covid vaccination campaigns on a large scale, the number of vaccines produced is not enough for all the population to get vaccinated at present. And with the corona positive cases rising every day, saving lives has become the prime matter of concern. As per the data provided by WHO more than 2 million people have died due to the coronavirus (H1N1) (2019-2020). However apart from vaccination, there is another scientific method by which a covid infected person can be treated and the death rate can be reduced. This plasma therapy is an experimental approach to treat corona-positive patients and help them recover. This plasma therapy is considered to be safe & promising. A person who has recovered from Covid can donate his/her plasma to a person who is infected with the coronavirus.

Key rules of brainstorming
Run an smooth and productive session

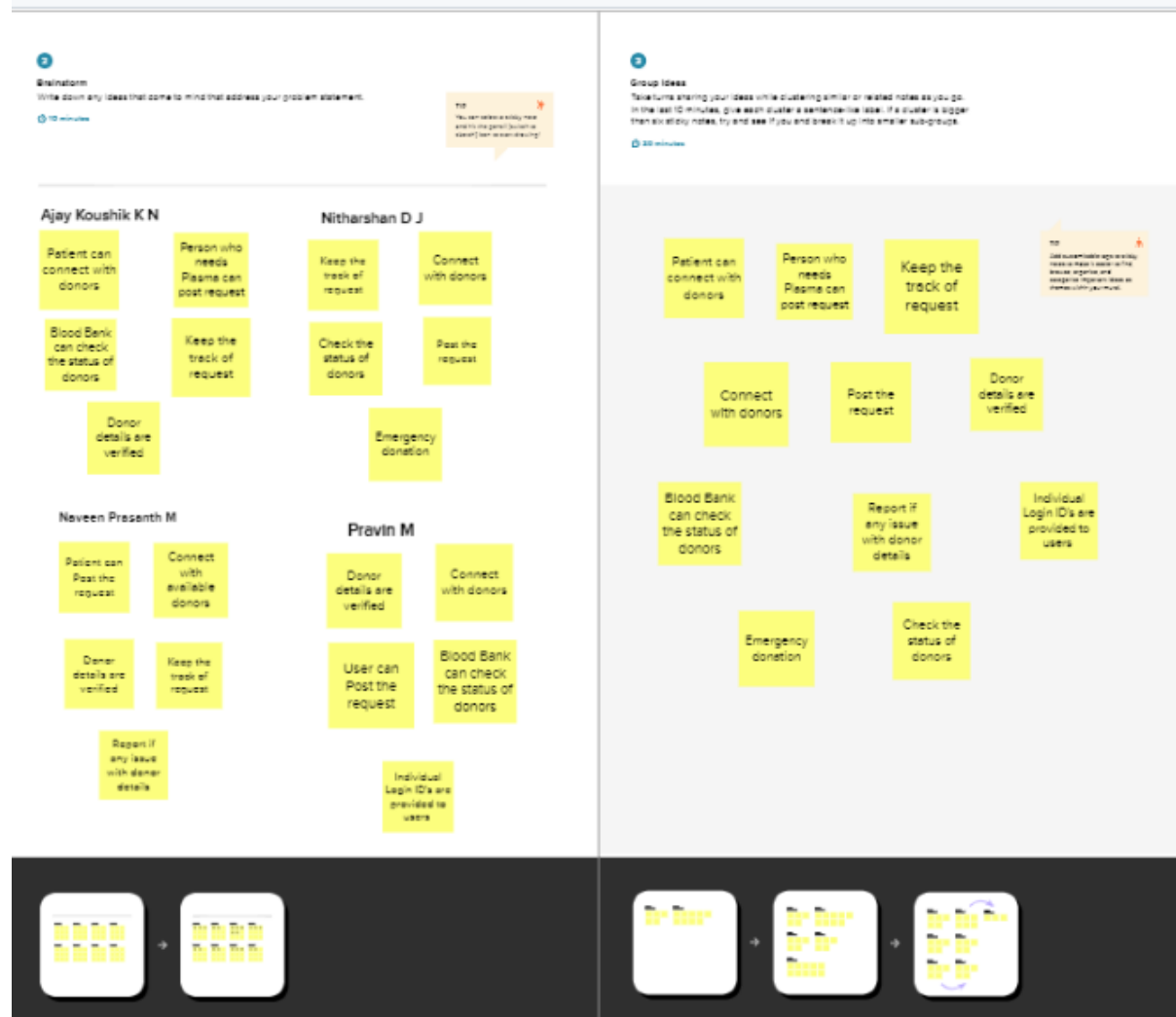
- Beginnings
- Encourage wild ideas
- Defer judgement
- Listen to others
- Defer volume
- 7 people or more



Plasma Donor Application
Show a detailed version of this template to enhance your work.

Open template

Ideation is a creative process where designers generate ideas in sessions. The participants gather with open minds to produce as many ideas as they can to address a problem statement in a facilitated, judgment-free environment.



The ideas gathered in ideation phase are then gathered, and given a priority ranking, so as to select the high priority ideas during the implementation phase.

4 Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on the grid to determine which ideas are important and which are feasible.

30 minutes

Importance
Team all agree: how do you want others to efficiently interact with you?

Feasibility
Degree of how important, which ideas are more feasible for you? (low cost, effort, complexity, etc)

Report if any issue with donor details

Emergency donation

Check the status of donors

User can Post the request

Donor details are verified

Patient can connect with donors

Keep the track of request

Connect with donors

Individual Login IDs are provided to users

Use
Developers can use their own login IDs to interact with the system. The feedback can be tracked by using the user profile tracking the ID on the system.

5 After you collaborate

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

Quick add-ons

- Share the mural**
Share a view link to the mural with stakeholders to keep them in the loop about the evolution of the vision.
- Export the mural**
Export a copy of the mural as a PNG or PDF to attach to emails, include in slides, or save in your drive.

Keep moving forward

- Strategy blueprint**
Define the components of a new idea or strategy.
[Open the template](#)
- Customer experience journey map**
Visualize customer needs, interactions, and decisions for an experience.
[Open the template](#)
- Strengths, weaknesses, opportunities & threats**
Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.
[Open the template](#)

[Share template feedback](#)

3.3 PROPOSED SOLUTION:

Our unified platform is aimed at all donors and requesters so they can communicate easily. Donors can be registered, verified and listed on the platform database. Patients or people in urgent need of plasma can use their location and other attributes to find a compatible plasma donor match right in time. Although this benefits the donors by offering them a universal location to list themselves, it poses a more significant impact to the patients who can find the resources they need in a way smaller timeframe. Patients and requesters don't have to be mentally pressured and frustrated about being able to find a compatible donor before it is too late.

3.4 PROBLEM SOLUTION FIT

Define CS, fit into CC	<div>1. CUSTOMER SEGMENT(S)<div>CS</div></div> <div>Who is your customer? i.e. working parents of 0-5 y.o. kids</div> <div>Donor And Receipients</div>	<div>6. CUSTOMER CONSTRAINTS<div>CC</div></div> <div>What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices.</div> <div>*Network Connection *Donor Health Condition *Unavailability of Plasma</div>	<div>5. AVAILABLE SOLUTIONS<div>AS</div></div> <div>Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking</div> <div>Bringing recovered patient back to hospital. And further next donation is impossible. Plasma demand and supply gap has grown even bigger.</div>	Explore AS, differentiate
	<div>2. JOBS-TO-BE-DONE / PROBLEMS<div>J&P</div></div> <div>Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one, explore different sides.</div> <div>*Difficult to find donors at the right time / at the time of emergency. *Donors not aware of plasma requirements.</div>	<div>9. PROBLEM ROOT CAUSE<div>RC</div></div> <div>What is the real reason that this problem exists? What is the back story behind the need to do this job? i.e. customers have to do it because of the change in regulations.</div> <div>*Not able to find the donors at the time of emergency. *Count of donors has been tremendously decreasing since hospital management couldn't contact them or get them notified at the right.</div>	<div>7. BEHAVIOUR<div>BE</div></div> <div>What does your customer do to address the problem and get the job done? i.e. Directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)</div> <div>he customer comes forward to *Attend plasma donation camps. *Donate plasma The hospital management/ patient is able to find plasma donors at the right time.</div>	Focus on J&P, fit into BE, understand RC
Focus on J&P, fit into BE, understand RC	<div>3. TRIGGERS<div>TR</div></div> <div>What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.</div> <div>Blood donation improves or saves lives and enhances social solidarity. It is also influenced by increasing deaths due to unavailability of plasma at required times</div>	<div>10. YOUR SOLUTION<div>SL</div></div> <div>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.</div> <div>Creating website which will provide information about available donors and plasma. If not available, the customer will be notified when plasma is available</div>	<div>8. CHANNELS of BEHAVIOUR<div>CH</div></div> <div>8.1 ONLINE What kind of actions do customers take online? Extract online channels from #7</div> <div>8.2 OFFLINE What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.</div> <div>Online: Can use the website to find donors. Offline: Can use the record maintain by the hospital</div>	Focus on BE, fit into RC, understand TR
Identify strong TR & EM	<div>4. EMOTIONS: BEFORE / AFTER<div>EM</div></div> <div>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.</div> <div>Before: Patient/ hospital find it hard to get a right resource to get plasma leaving them upset. After: The donors and customers have a feeling of satisfaction.</div>			Identify strong TR & EM

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT:

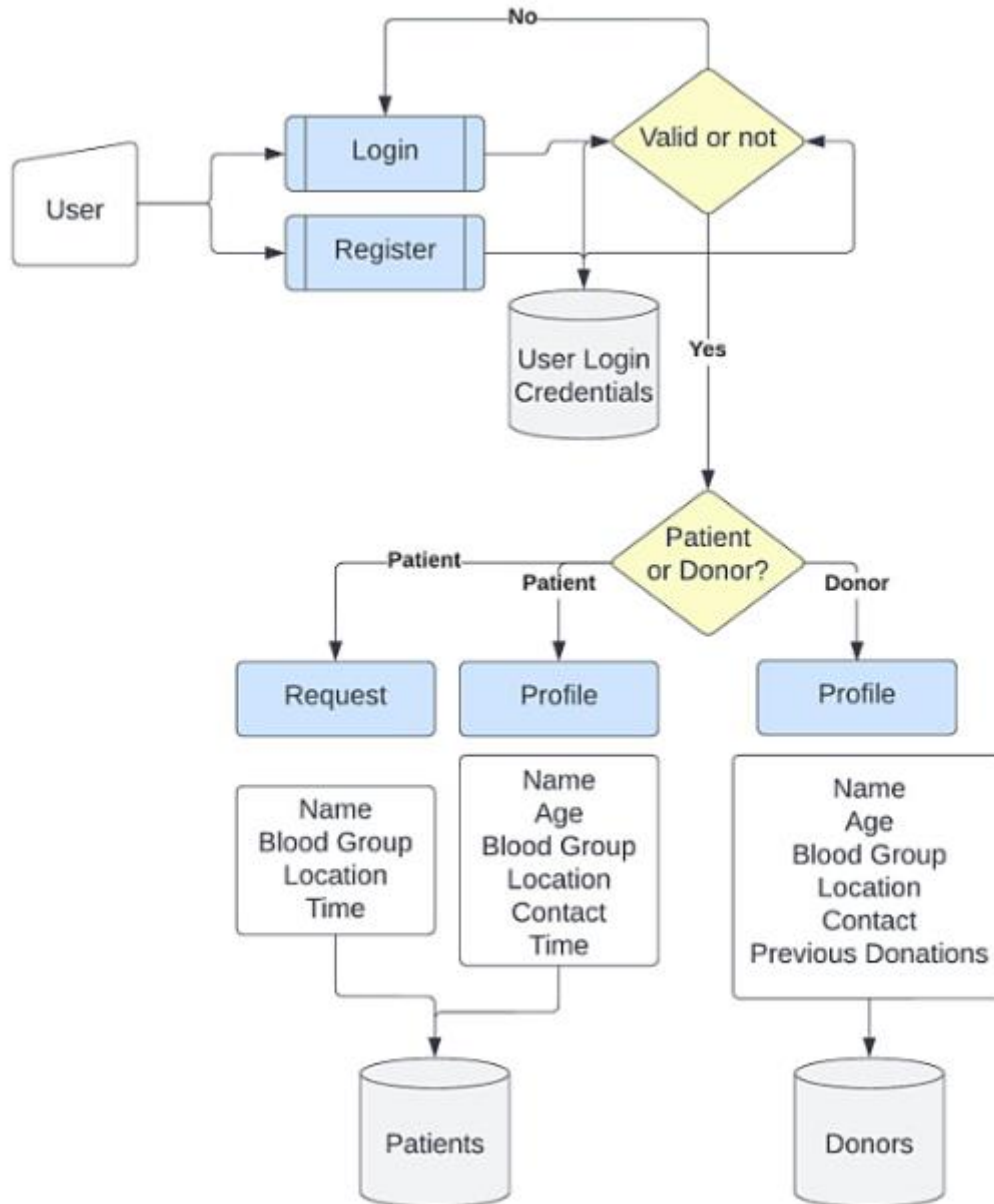
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Mobile Number Registration through Gmail
FR-2	User Registration	Confirmation via Email Confirmation via OTP
FR-3	User Validation	Donor – Check health conditions Patient – Check credibility of user
FR-4	Chatbot	Help the user understand the process and navigate the website
FR-5	Search	Patient enters details to search for compatible donors
FR-6	Request	Patient can send a donor request to obtain their plasma
FR-7	Email notification	Donor will get notified through email when a compatible patient places request
FR-8	Donation completion	Patient and Donor to confirm completion of donation

4.2 NON FUNCTIONAL REQUIREMENT:

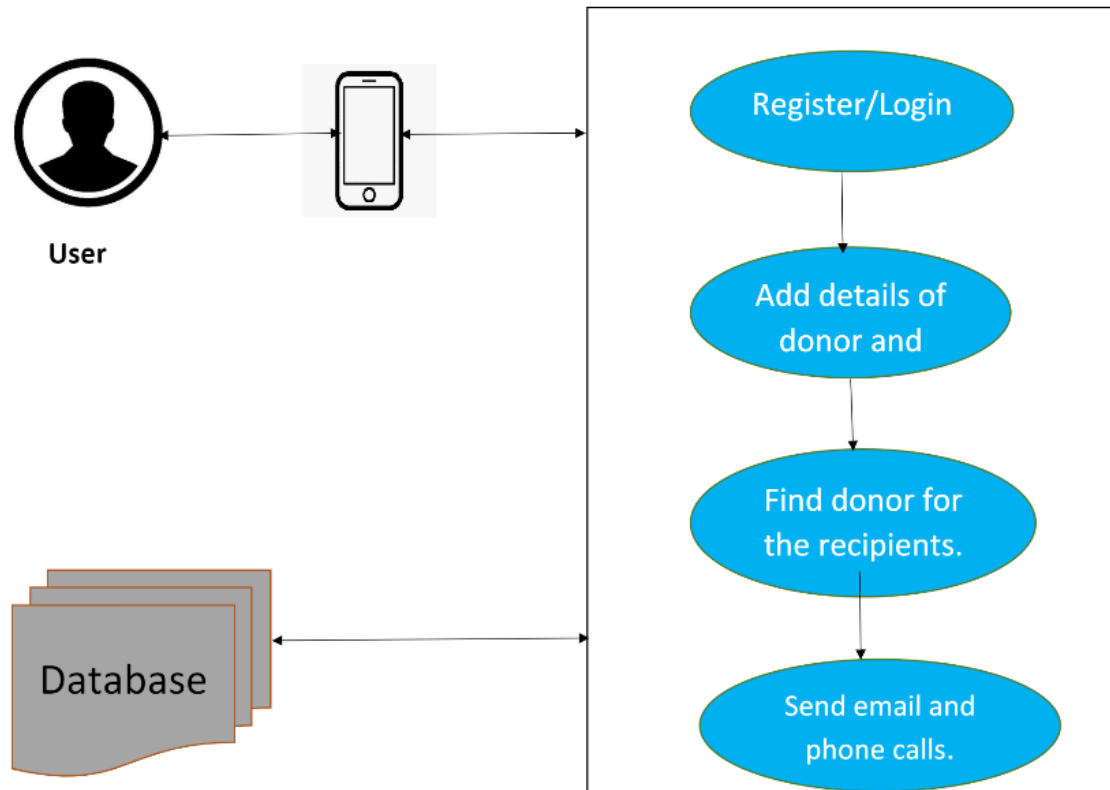
NFR No.	Non-Functional Requirement	Description
NFR-1	Usability	The user experience must be simple and the user must be able to perform all actions available on the platform
NFR-2	Security	The database layer and the logic layer are hosted using IBM services, hence the security of all data involved in the process is kept secure
NFR-3	Reliability	All processes involved in the platform must be uniform whenever accessed. It must also function without any bugs and errors
NFR-4	Performance	Immediate feedback from the platform is provided to user, so they are not discouraged to use the application
NFR-5	Availability	The application must be usable at all time, the database and servers need to be available and reachable anytime from anywhere
NFR-6	Scalability	The platform must adapt so that it can support a high volume of concurrent users. Meanwhile, the platform must also be loosely couples to ensure it can scale vertically too, by adding more functionality

PROJECT DESIGN

5.1 DATA FLOW DIAGRAM:



5.2 SOLUTION AND TECHNICAL ARCHITECTURE:



5.3 USER STORIES:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a donor, I can register for the application by entering my email /Phone number, password, and confirming my password.	4	High	Ajay Koushik KN (Leader)
Sprint-1	Login	USN-2	Registered donor can log into the application by entering donor email & password	3	High	Pravin M (Member 3)

Sprint-2	Verification	USN-3	As a donor, I can enter my details to check the donor eligibility criteria,	10	Medium	Naveen Prasanth M (Member 2)
Sprint-3	Dashboard	USN-4	User can provide their personal details and location	7	Low	Nitharshan DJ (Member 1)

Sprint-4	Acceptance	USN-5	User can accept their willingness to donate plasma	10	Medium	Ajay Koushik KN (Leader)
----------	------------	-------	--	----	--------	-----------------------------

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a receiver, I can register for the application by entering my email /Phone number, password, and confirming my password	4	High	Pravin M (Member 3)
Sprint-1	Login	USN-2	Registered receiver can log into the application by entering receiver email & password	3	High	Naveen Prasanth M (Member 2)
Sprint-2	Verification	USN-3	As a receiver, I can enter my details to check the receiver eligibility criteria	10	Medium	Nitharshan DJ (Member 1)

Sprint-3	Dashboard	USN-4	User can search the list of available donor	7	Low	Ajay Koushik KN (Leader)
Sprint-4	Access	USN-5	User can access the available donors list ,then they can choose the donor who is nearby to receiver	10	Medium	Pravin M (Member 3)
Sprint-1	Registration	USN-1	Third Party user can register for the application by entering my email /Phone number, password, and confirming my password.	3	High	Naveen Prasanth M (Member 2)
Sprint-1	Login	USN-2	Registered user can log into the application by entering user email & password	3	High	Nitharshan DJ (Member 1)
Sprint-3	Query System	USN-3	User can ask their queries via Chabot which is available 24/7 to sort user issues	6	Medium	Ajay Koushik KN (Leader)

PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION:

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

6.2 SPRINT DELIVERY SCHEDULE:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application
Sprint-2		USN-3	As a user, I can register for the application through phone number and log in using it
Sprint-1	Login	USN-4	As a user, I can log into the application using my registered email & password
Sprint-2	Dashboard	USN-5	As a user, I want to enter/update my medical and contact information
Sprint-4	Chatbot	USN-6	As a user, I can ask questions to the chatbot
Sprint-3	Receive Alerts	USN-7	As a donor, I want to receive immediate alerts upon requests from patient
Sprint-2	Request Plasma	USN-8	As a patient, I want a list of donors
Sprint-4		USN-9	As a patient, I want to sort out donor list
Sprint-3		USN-10	As a patient, I want to request for plasma

CODING AND SOLUTIONING

▼ SOURCECODE


> __pycache__

> assets


> js


> static


> templates

 app.py

! deployment.yaml

 DigiCertGlobalRootCA.crt

 dockerfile

 mail.py

▼ js

JS bootstrap.min.js

JS bootstrap.min.js.map

JS grid-gallery.js

JS grid-gallery.min.js

JS jquery-3.2.1.min.js

JS jquery-scrolltofixed-min.js

JS popper.js

JS popper.min.js

JS popper.min.js.map

JS script.js

> static

▼ static

- # adminlogin.css
- # adminreg.css
- # animate.css
- # bootstrap.min.css
- # donar.css
- # fontawsom-all.min.css
- # grid-gallery.css
- # grid-gallery.min.css
- # logincss.css
- # recipient.css
- # reciptreg.css
- # style.css

▼ templates

- <> admin.html
- <> adminlogin.html
- <> adminreg.html
- <> donar.html
- <> donlogin.html
- <> donregistration.html
- <> index.html
- <> mail.html
- <> plasmadon.html
- <> plasmareq.html
- <> recipient.html
- <> recipregistration.html
- <> reclogin.html

```

app.py > ...
1  from flask import Flask,render_template,request,url_for,redirect
2  from flask_mail import *
3  from markupsafe import escape
4
5  import ibm_db
6  conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cm
7
8
9  app = Flask(__name__)
10
11
12  @app.route('/')
13  def index():
14      return render_template('index.html')      # index - home page
15
16  # admin credentials
17
18  @app.route('/adminlogin')
19  def adminlogin():
20      return render_template('adminlogin.html')  # admin log in page
21
22  @app.route('/adminreg')
23  def adminreg():
24      return render_template('adminreg.html')  # admin sign up page
25

```

```

95  @app.route('/donrec',methods = ['POST', 'GET'])
96  def donrec():
97      if request.method == 'POST':
98
99          fname = request.form['fname']
100         lname = request.form['lname']
101         dob = request.form['dob']
102         email = request.form['email']
103         mnumb = request.form['mnumb']
104         gender = request.form['gender']
105         address = request.form['address']
106         pin = request.form['pin']
107
108         sql = "SELECT * FROM donarrec WHERE fname =?"
109         stmt = ibm_db.prepare(conn, sql)
110         ibm_db.bind_param(stmt,1,fname)
111         ibm_db.execute(stmt)
112         account = ibm_db.fetch_assoc(stmt)
113

```

```

56     if account:
57         return render_template('reclogin.html', msg="Already your account exists, please try to log in")
58     else:
59         insert_sql = "INSERT INTO recipientrec VALUES (?, ?, ?, ?, ?, ?, ?, ?)"
60         prep_stmt = ibm_db.prepare(conn, insert_sql)
61         ibm_db.bind_param(prepare_stmt, 1, fname)
62         ibm_db.bind_param(prepare_stmt, 2, lname)
63         ibm_db.bind_param(prepare_stmt, 3, dob)
64         ibm_db.bind_param(prepare_stmt, 4, email)
65         ibm_db.bind_param(prepare_stmt, 5, mnumb)
66         ibm_db.bind_param(prepare_stmt, 6, gender)
67         ibm_db.bind_param(prepare_stmt, 7, address)
68         ibm_db.bind_param(prepare_stmt, 8, pin)
69         ibm_db.execute(prepare_stmt)
70
71     return render_template('reclogin.html', msg="Account has been created successfully..")
72
73     return "success..."
74

```

```

27 @app.route('/recipregistration')
28 def recipregistration():
29     return render_template('recipregistration.html')  ## recipient signup page uh
30
31 @app.route('/recipientlogin')
32 def recipientlogin():
33     return render_template('reclogin.html')  ## recipt login page
34
35
36 @app.route('/recipientrec', methods = ['POST', 'GET'])
37 def recipientrec():
38     if request.method == 'POST':
39
40         fname = request.form['fname']
41         lname = request.form['lname']
42         dob = request.form['dob']
43         email = request.form['email']
44         mnumb = request.form['mnumb']
45         gender = request.form['gender']
46         address = request.form['address']
47         pin = request.form['pin']
48
49         sql = "SELECT * FROM recipientrec WHERE fname =?"
50         stmt = ibm_db.prepare(conn, sql)
51         ibm_db.bind_param(stmt, 1, fname)
52         ibm_db.execute(stmt)
53         account = ibm_db.fetch_assoc(stmt)
54

```

```

56     if account:
57         return render_template('reclogin.html', msg="Already your account exists, please try to log in")
58     else:
59         insert_sql = "INSERT INTO recipientrec VALUES (?, ?, ?, ?, ?, ?, ?, ?)"
60         prep_stmt = ibm_db.prepare(conn, insert_sql)
61         ibm_db.bind_param(prepare_stmt, 1, fname)
62         ibm_db.bind_param(prepare_stmt, 2, lname)
63         ibm_db.bind_param(prepare_stmt, 3, dob)
64         ibm_db.bind_param(prepare_stmt, 4, email)
65         ibm_db.bind_param(prepare_stmt, 5, mnumb)
66         ibm_db.bind_param(prepare_stmt, 6, gender)
67         ibm_db.bind_param(prepare_stmt, 7, address)
68         ibm_db.bind_param(prepare_stmt, 8, pin)
69         ibm_db.execute(prepare_stmt)
70
71     return render_template('reclogin.html', msg="Account has been created successfully..")
72
73     return "success..."
74

```

```

77
78     @app.route('/donregistration')
79     def donregistration():
80         return render_template('donregistration.html')    ## donar signup page uh
81
82
83     @app.route('/donarlogin')
84     def donarlogin():
85         return render_template('donlogin.html')    ## donar login page
86
87

```

```

95 @app.route('/donrec',methods = ['POST', 'GET'])
96 def donrec():
97     if request.method == 'POST':
98
99         fname = request.form['fname']
100         lname = request.form['lname']
101         dob = request.form['dob']
102         email = request.form['email']
103         mnumb = request.form['mnumb']
104         gender = request.form['gender']
105         address = request.form['address']
106         pin = request.form['pin']
107
108         sql = "SELECT * FROM donarrec WHERE fname =?"
109         stmt = ibm_db.prepare(conn, sql)
110         ibm_db.bind_param(stmt,1,fname)
111         ibm_db.execute(stmt)
112         account = ibm_db.fetch_assoc(stmt)
113

```

```

115     if account:
116         return render_template('donlogin.html', msg="Already your account exists, please try to log in")
117     else:
118         insert_sql = "INSERT INTO donarrec VALUES (?,?,,?,,?,,?)"
119         prep_stmt = ibm_db.prepare(conn, insert_sql)
120         ibm_db.bind_param(prepare_stmt, 1, fname)
121         ibm_db.bind_param(prepare_stmt, 2, lname)
122         ibm_db.bind_param(prepare_stmt, 3, dob)
123         ibm_db.bind_param(prepare_stmt, 4, email)
124         ibm_db.bind_param(prepare_stmt, 5, mnumb)
125         ibm_db.bind_param(prepare_stmt, 6, gender)
126         ibm_db.bind_param(prepare_stmt, 7, address)
127         ibm_db.bind_param(prepare_stmt, 8, pin)
128         ibm_db.execute(prepare_stmt)
129
130     return render_template('donlogin.html', msg="Account has been created successfully..")
131
132     return "success..."

```

```

135 @app.route('/admin')
136 def admin():
137     return render_template('admin.html')
138
139 @app.route('/donar')
140 def donar():
141     return render_template('donar.html')
142
143
144 ## donar registering for donation
145 @app.route('/giveplasma',methods = ['POST', 'GET'])
146 def giveplasma():
147     if request.method == 'POST':

```



```

49     name = request.form['name']
50     age = request.form['age']
51     gender = request.form['gender']
52     mnumb = request.form['mnumb']
53     email = request.form['email']
54     city = request.form['city']
55     address = request.form['address']
56     bloodgroup = request.form['bloodgroup']
57     issue = request.form['issue']
58     lastbd = request.form['lastbd']
59     slot = request.form['slot']
60
61     sql = "SELECT * FROM donar WHERE name =?"
62     stmt = ibm_db.prepare(conn, sql)
63     ibm_db.bind_param(stmt, 1, name)
64     ibm_db.execute(stmt)
65     account = ibm_db.fetch_assoc(stmt)
66

```

```

7     if account:
8         return render_template('donlogin.html', msg="You are already a member, please login using your account")
9     else:
10         insert_sql = "INSERT INTO donar VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)"
11         prep_stmt = ibm_db.prepare(conn, insert_sql)
12         ibm_db.bind_param(prepare_stmt, 1, name)
13         ibm_db.bind_param(prepare_stmt, 2, age)
14         ibm_db.bind_param(prepare_stmt, 3, gender)
15         ibm_db.bind_param(prepare_stmt, 4, mnumb)
16         ibm_db.bind_param(prepare_stmt, 5, email)
17         ibm_db.bind_param(prepare_stmt, 6, city)
18         ibm_db.bind_param(prepare_stmt, 7, address)
19         ibm_db.bind_param(prepare_stmt, 8, bloodgroup)
20         ibm_db.bind_param(prepare_stmt, 9, issue)
21         ibm_db.bind_param(prepare_stmt, 10, lastbd)
22         ibm_db.bind_param(prepare_stmt, 11, slot)
23         ibm_db.execute(prepare_stmt)

```

```

87 @app.route('/plasmadon')
88 def plasmadon():
89     donar = []
90     sql = "SELECT * FROM donar"
91     stmt = ibm_db.exec_immediate(conn, sql)
92     dictionary = ibm_db.fetch_both(stmt)
93     while dictionary != False:
94         # print ("The Name is : ", dictionary)
95         donar.append(dictionary)
96         dictionary = ibm_db.fetch_both(stmt)
97
98     if donar:
99         return render_template("plasmadon.html", donar = donar)
100

```

```

277 @app.route('/plasmareq')
278 def plasmareq():
279     recipient = []
280     sql = "SELECT * FROM recipient"
281     stmt = ibm_db.exec_immediate(conn, sql)
282     dictionary = ibm_db.fetch_both(stmt)
283     while dictionary != False:
284         # print ("The Name is : ", dictionary)
285         recipient.append(dictionary)
286         dictionary = ibm_db.fetch_both(stmt)
287
288     if recipient:
289         return render_template("plasmareq.html", recipient = recipient)
290

```

```

@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="Accepted successfully")

    return "Accepted Successfully"

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

```

TESTING

Testing was managed on a platform called Testlodge. This website helps us create test suites and individual test cases for accounting purposes. We can describe individual test cases, their pass and fail criteria. These test cases can be then assigned to individual team members for them to carry out the test and report the result on the platform. We performed unit testing, integration testing and acceptance testing on our final application. Unit testing was done by creating testcases for individual modules to work in separation from the rest of the application. Each of the html pages, their scripting and their input and output was tested. Integration testing was performed to check if a complete logical process can be done without any errors in the workflow. As some of our functions and features work together, they need to be assessed if the testcases pass.

8.1 TEST CASES:

Database: Entering invalid data into html forms should be prevented by using appropriate scripting to validate data. Retrieval of data should be accurate for mission critical functions.

Login: Enter invalid credentials to test if login is allowed.

Registration: Date of negative covid test must be at least 14 days before current date.

8.2 ACCEPTANCE TESTING:

Acceptance testing is testing the overall service flow and checking if each logical function works with all ranges of input from the perspective of an end user. This is similar to blackbox testing, as the user doesn't know the internal functioning of the application. They only provide their input and we need to validate if the necessary output has been acquired as a result.

RESULT

9.1 SOFTWARE TEST METRICS

1. Test design efficiency = Number of tests designed / Total time
 - $25 \text{ Tests designed} / 10 \text{ Hours} = 2.5 \text{ Tests designed per hour}$
2. Passed Test Cases Percentage = (Number of Passed Tests/Total number of tests executed) X 100
 - $25 \text{ passed Test cases} / 25 \text{ Test cases} \times 100 = 100\%$
3. Failed Test Cases Percentage = (Number of Failed Tests/Total number of tests executed) X 100
 - $0 \text{ failed Test cases} / 25 \text{ Test cases} \times 100 = 0\%$
4. Schedule slippage = (Actual end date – Estimated end date) / (Planned End Date – Planned Start Date) X 100
 - $(19/11/2022 - 17/11/2022) / (17/11/2022 - 10/09/2022) \times 100 = 3 \text{ days} / 70 \text{ days} \times 100 = 4.28\%$

9.2 PERFORMANCE METRICS

Our project uses the IBM Lite plan for all the cloud services like DB2, Watson Assistant, Container Registry, Kubernetes which provides minimal access and performance for free of cost. There are performance constraints for the deployed application. All requests by the user must be processed and rendered on screen before 5 seconds of initiation. If this is not possible due to errors from any side, a buffering screen must be displayed. Database needs to be online at all times to facilitate information transfer whenever a user logs on and accesses the platform.

ADVANTAGES & DISADVANTAGES

The vision of this project is to provide a unified platform where all the requests and information can be shared for higher visibility and faster correspondence. By using an application to serve both donors and patients who are in need of plasma, we are bridging the gap in between them. Donors can register themselves so they may be contacted when a certain request is placed. Patients can also place a request on the platform so their needs can be fulfilled as soon as possible.

Disadvantages of this platform include a transparency about sensitive information provided by both patients and donors. Administrators who are responsible for managing requests would be able to see information regarding the users health. Verification also needs to be implemented to make sure that the credibility of both patients and donors are upheld, to increase the reliability of the platform for public usage.

CONCLUSION

In conclusion, we have created an application and provided an interface with which donors can contact hospitals and hospitals can in turn find donors to meet the plasma requirements. This will allow hospitals to meet the demands of plasma for the patients undergoing treatment. Plasma from Covid recovered patients can be used to treat covid patients. This allows an easy way for donors to interact with hospitals and to donate plasma when needed.

Donors who need to donate plasma can donate by uploading their details 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.

People need not have feelings of anxiety and fear that they might not receive the help they seek. Using our platform they can gather information about the donors available near them and compatible so the donation can be done.

Such an application will be beneficial to the public who currently don't have a common and dedicated platform to share such emergencies and contact details.

FUTURE SCOPE

There is scope for more features to be integrated in our application for a better user experience and more efficient process.

A messenger service can be included on the platform for the donors and patients to enable direct communication without the need for a medium.

Verification using any photo ID can be done as to automate the process by integrating government service to increase the credibility and reliability of the application.

A ticketing system can be used to collect, track and manage user queries and problems they face during the usage of the application.

Although the app is already cross platform supported due to it being a website, a mobile application can be designed and published for users to have instant access to our application on their smartphones.

APPENDIX

Github link:

<https://github.com/IBM-EPBL/IBM-Project-18807-1659690339>

Demo link:

https://drive.google.com/file/d/1cvk5umtO4UOPXVJ-8Tv1LCto2YsurTVu/view?usp=share_link