IBM NALAIYATHIRAN PROJECT REPORT

PLASMA DONOR APPLICATION

Team Id	PNT2022TMID33146
Project Name	PLASMA DONOR APPLICATION
Team Members	Santhoshkumar S(922119104038) Saravanakumar G(922119104040) Shiffin Paul J(922119104042) Vishwa Bharathi J(922119104050)

SSM Institute Of Engineering And Technology

DINDIGUL-624002

CONTENT

1.INTRODUCTION

1.1 Project Overview

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

Serverless computing is the current trend in software application development. Microservices are a popular new approach for building maintainable, scalable, cloud-based applications. AWS is the perfect platform for hosting micro-services. In this project, we will be building a plasma donor app with AWS services like lambda functions, API gateway, and DynamoDB.

1.2 Purpose

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. Pharmaceutical companies use plasma to make treatments for conditions such as immune deficiencies and bleeding disorders.

2. LITERATURE SURVEY

2.1 Existing problem

Several experiments have been carried out over the years by different groups of researchers. Here are some of the following groups:

1. The "MBB: A life saving application" AUTHORS:

Ramakant Gawande, Narendra Gupta, Nikhil Thengadi. They come up with a system to link all donors and help in controlling blood transfusion process. Their system will also maintain database which hold data of donors and blood according to their city and further by their locality. They have proposed a machine so that it will hyperlink all donors. The machine will help to control the blood transfusion service and create a database to maintain records on shares of blood in every place as records on donors in every city. Moreover, human beings will be capable of see which sufferers want blood components thru the application. They will be able to check in as donors and as a result acquire a request from their nearby customers who desires blood to donate blood in instances of want.

2. In 2015, Mobile Based Healthcare Management using Artificial Intelligence

AUTHORS: Amiya Kumar Tripathy, Rebeck Carvalho, Keshav Pawaskar, Suraj Yadav, Vijay Yadav.

In this paper, the health-care management system is proposed which will consist of mobile based heart rate measurement so that the data can be transferred and diagnosis based on heart rate can be provided quickly with a click of button. The system will consist of video conferencing to connect remotely with doctor. The system will also consist of Doc-Bot and an online Blood Bank. In this implemented project, heart rate calculation differs from actual one due to noise present in input signal. So the performance is not efficient in practical. Methodology used Clustering, Text Mining, Pattern Matching, Support Vector Machine, Partitioning Algorithm and DonorHART tool used in collecting donor reaction information. Limitations are Difficulty in handling emergency situation and No proper security for personal details misuse.

3. In year 2018, "Automated blood bank system using Raspberry PI".

AUTHORS: Ashlesha C. Adsul, V. K. Bhosale, R. M. Autee . AUTHORS: Ashlesha C. Adsul, V. K. Bhosale, R. M. Autee .

"Raspberry pi based blood bank system" proposed to bring blood donors to the one place. The aim of this system is fulfil every blood request by using android application and raspberry pi. In the proposed system, data about the donors will be collected by using android application and raspberry pi by installing systems at places such as hospitals, blood banks etc. These data will be stored in the database. User/Patients needs to access application and needs to enter his requirements about the blood in the application the requirements are matched with the database and message will be to that particular blood donor through GSM modem.

4. In year 2020, "Towards an Efficient and Secure Blood Bank Management System".
AUTHORS: P.A.J. Sandaruwan, U.D.L. Dolapihilla, D.W.N.R. Karunathilaka; W.A.D.T.L. Wijayaweera, W.H. Rankoth.

A blood bank plays an important role in a hospital as well as in a country, ensuring safe and timely blood transfusions. However, there are several challenges faced by blood banks around the world, specifically when securing the blood supply chain. Reducing the supply-demand imbalance, protecting the data privacy of donors as well as receivers, are some of them. Therefore, there is a timely requirement for an effective and secure management system for the blood bank. We have proposed a management platform for the blood bank operations with the following modules: (1) forecast blood demand, (2) suggest blood donation campaign locations and (3) secure blood supply chain. The proposed platform has been implemented using techniques such as Long Short-Term Memory (LSTM), k-means clustering, Geographic Information Systems (GIS), and block chain. Our results show that using our proposed modules, we can minimize the imbalance between supply and demand of blood, find the most suitable donor in an emergency, and enhance the privacy of data

5. In year 2021, "A Systematic Review & Design of Web-Based Blood Management System".

AUTHORS: Gokul Dudani, Tanushree, Kajal Singh, Anushka Singh Chauhan

Blood is a fluid that carries oxygen and is a connective tissue that carries other substances because of its volume. Now that we understand the importance of blood, we see that it not only carries oxygen to the tissues but also clears the air between them through the heart and blood vessels. The average volume of blood donation is 470ml per person, which is only 8% of the adult volume. When blood is needed in a hospital, it is usually not available in time, leading to inconsistencies. Both patients and sponsors are unaware that the donor is being hospitalized due to a lack of communication and other services. A system like this is needed to close the communication gap between hospitals, blood banks, donors, and receptors. The main purpose of a web-based blood donation program is to ensure compliance with blood stock. In today's system, first and foremost the hand system, and when a person needs a blood type and that type is not available in that blood bank, it takes time to process blood from another blood bank, which may adversely affect the patient's health because time is critical in emergencies. Therefore, a web-based blood donation system is a good place to monitor whether a particular type of blood is available in a stack or not, as well as to provide a place where blood can be accessed.

2.2 References

[1] G. Muddu Krishna; S. Nagaraju(2016),"Design and implementation of short message service (SMS) based blood bank", 2016 International Conference on Inventive Computation Technologies (ICICT)

[3] "Benefits of Management Information System in Blood Bank" by 1, Vikas Kulshreshtha, 2, Dr. Sharad Maheshwari 1, Research Scholar, 2, Associate Professor 2 1, Singhania University, Jhunjhunu, Rajasthan, India 2, Government Engineering College Jhalawar, Rajasthan, India

[4] The Optimization of Blood Donor Information and Management System by Technopedia P. Priya1, V. Saranya2, S. Shabana3, Kavitha Subramani4 Department of Computer Science and Engineering, Panimalar Engineering College, Chennai, India 1, 2, 3, 4

[5] Anish Hamlin M R, Albert Mayan J (2016), "Blood Donation And Life Saver-Blood Donation App", 2016 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)

[6] "Android Blood Bank" by Prof. Snigdha1, Varsha Anabhavane2, Pratiksha lokhande3, Siddhi Kasar4, Pranita More5 Lecturer, Information Technology, Atharva College of Engineering, Mumbai, India 1 Student, Information Technology, Atharva College of Engineering, Mumbai, India 2,3,4,5

[7] "A Study on Blood Bank Management System" by A. Clemen Teena, K. Sankar and S. Kannan, Department of MCA, Bharath University, Selaiyur, Chennai-73, Tamil Nadu, India

2.3 Problem Statement Definition

Name: Cloud Application Development.

Project Name: Plasma Donor Application.

Who does the problem affect?

Patients with severe liver disorders and numerous clotting deficiencies are given plasma.

What is the issue?

When a patient needs plasma, it can be challenging to get in contact with a donor within the patient's family and friends in a timely manner. It can also be challenging to get in contact with authorized donor centers.

What is the impact of the issue?

Due to a lack of awareness regarding plasma donation, there is a demand for plasma donors, making it challenging for the affected patients to locate donors. During the COVID19 pandemic, the need for plasma increased and the donor rates decreased in order to provide an immunity boost for COVID-affected patients.

What would happen if we didn't solve the problem?

It takes a long time for a patient to discover the proper donor, and it also takes time for the spreading about the need for plasma donors to disseminate on social media to a larger audience. As a result, patients cannot locate the right donor within a given time frame.

What would happen when it is fixed?

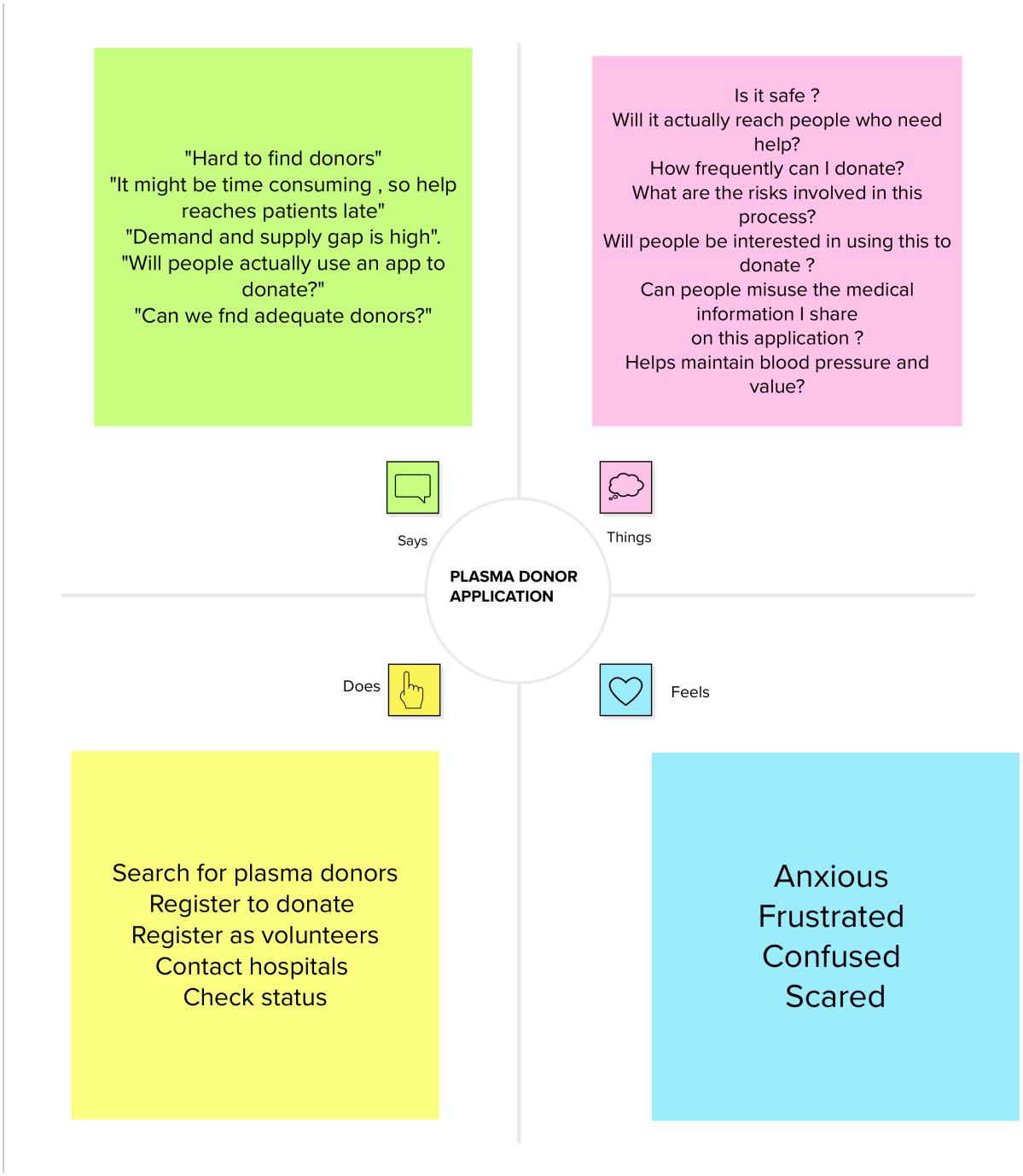
Our application enables patients with severe liver illnesses, blood clotting issues, and covid to quickly and easily locate the correct donor within the allotted period.

Why is it important that we fix the problem?

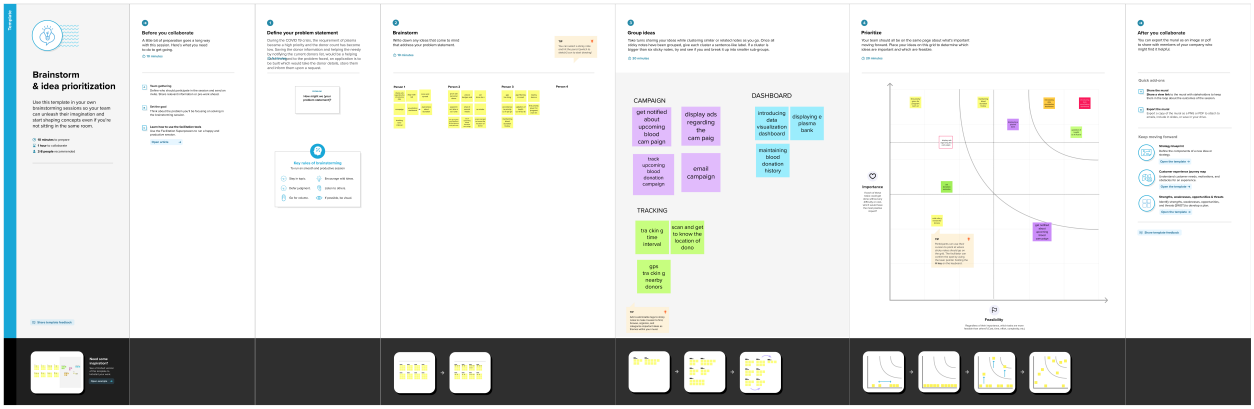
The condition of the affected patient may suffer and perhaps result in death, if the appropriate plasma donor cannot be found within a certain amount of time. IDEATION & PROPOSED SOLUTION.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming



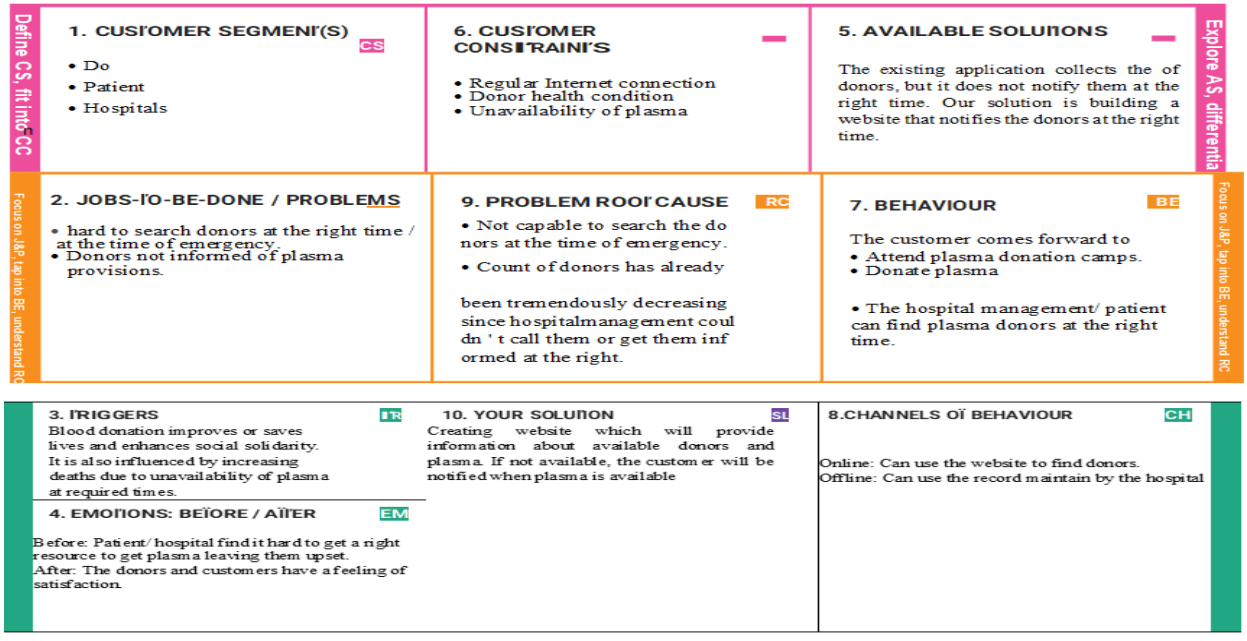
3.3 Proposed Solution

Project Design Phase-I
Proposed Solution

S .No.	Parameter	Description
1.	Problem Statement	To developa mobile application that allows users to donate their plasma to those who need it mostly in times of emergency.
2.	Idea / Solution description	The application will allow usersto register themselves on the portalfor plasma donation so that recipients who require it can view their information so that they can get the plasma.
3.	Novelty / Uniqueness	Whenthe user requests for plasma transmission, if there is lack of plasma atthe time of request, automatically user will be marked in hold back list. Later when there is availability of plasma, the receiver waiting in hold back list will be alerted via calling system.

4.	Social Impact / Customer Satisfaction	The application is user friendly so that anyone with basic knowledge can access it. The application seamlessly connects the donor and the person who need it and also hospitals who have availability of the plasma.
5.	Business Model (RevenueModel)	The application is free to use and it comesunder healthcare domain.It helps peoplewho want to donate plasma to the people who need it. Data can be storedin IBM DB2in cloud whichreduces the overallcost incurred for developing the application.
6.	Scalability of the Solution	Since the app is going to store its data in cloud, it will continue to be efficient when large number of people uses it. Also when the number of requests for plasma increases, the call notification system will work fine withoutany disruption.

3.4 Problem Solution fit



4. REQUIREMENT ANALYSIS

4.1 Functional Requirements:

Following are the functional requirements of the proposedsolution

FR No.	Functional Requirement (Epic)	Sub Requirement (Story/ Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through Linked IN
FR-2	User Confirmation	Confirmation via EmailConfirmation via OTP
FR-3	Objective	Describe what the product does
FR-4	End result	Define product features
FR-5	Focus	Focus on user requirements
FR-6	Documentation	Captured in use case

4.2 Non-functional Requirements:

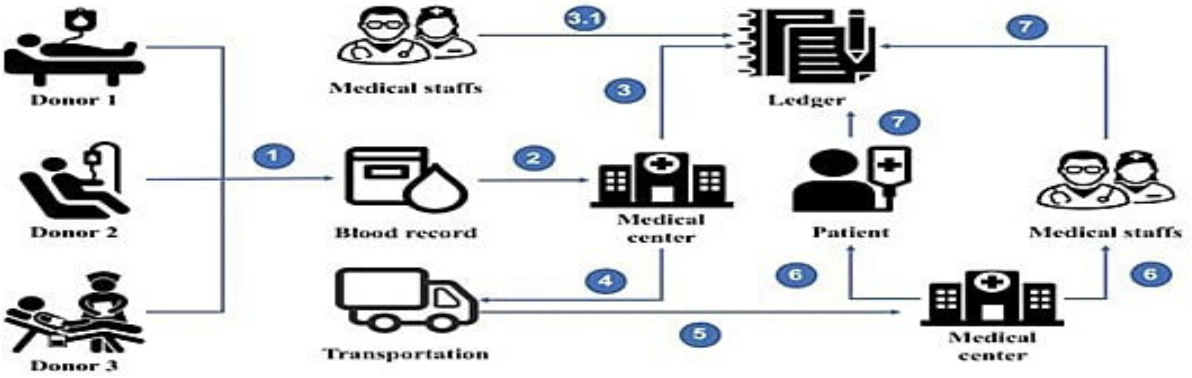
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Human Factors, overall aesthetics , consistency anddocumentation.
NFR-2	**Security**	A system's ability to prohibit unauthorized access , usage or behavior modification whileproviding service toauthorized users.
NFR-3	**Reliability**	Frequency andseverity of failure, recoverability, predictability , accuracy and mean time betweenfailures(MTBF).
NFR-4	**Performance**	Processing speed , response time ,resource consumption,throughput and efficiency.

NFR-5	**Availability**	How long a system is available forusers . This is timethe system is not down due to outages or maintenance activities. Mean Time Between Failure(MTBF) is one metric that helps us characterize systemavailability.
NFR-6	**Scalability**	The ability of solution or system to increase its capacity to serveclients and/or increase processing rates to matchdemand.

5. PROJECT DESIGN

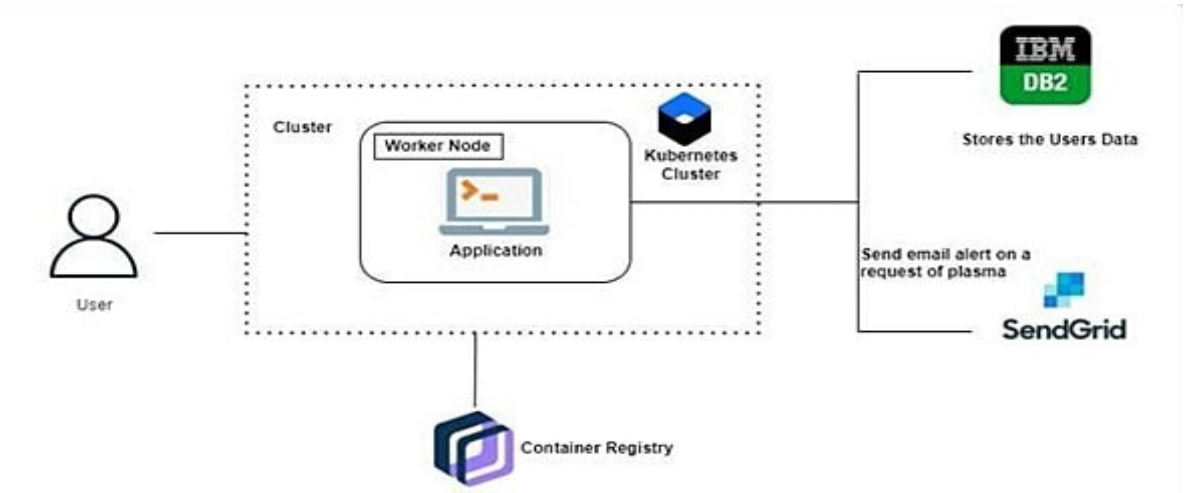
5.1 Data Flow Diagrams



5.2 Solution & Technical Architecture

Python, Flask, Docker

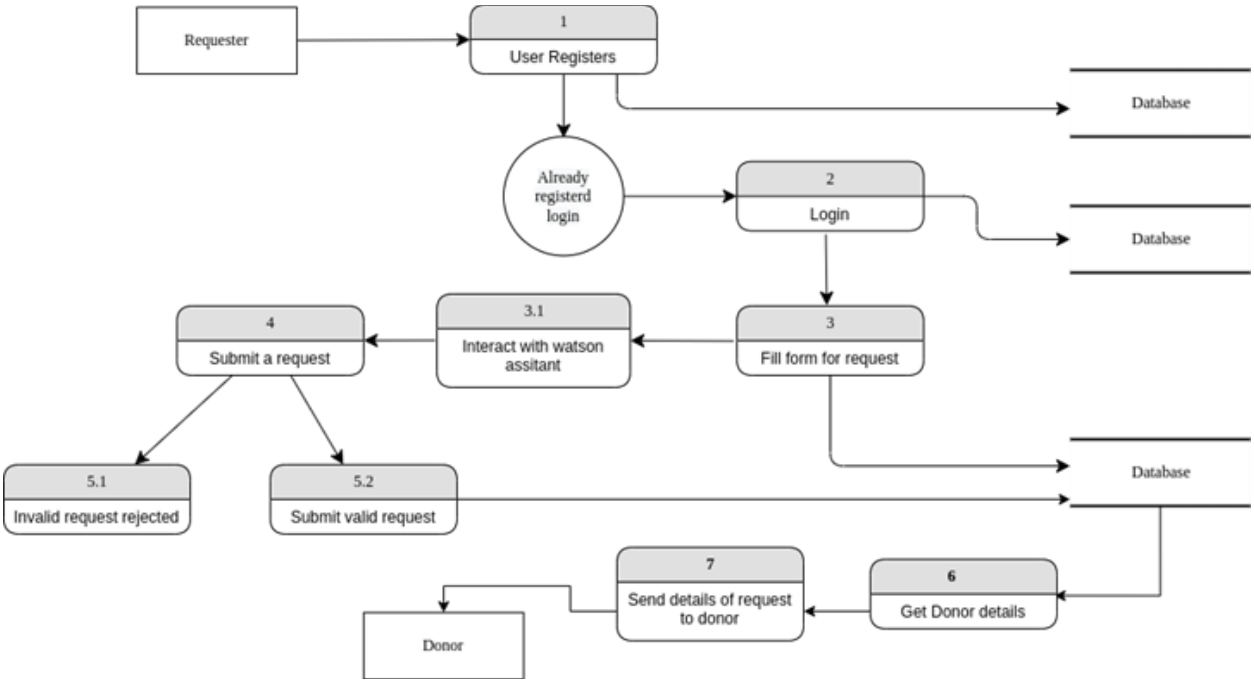
ARCHITECTURE



PROJECT WORKFLOW

- The user interacts with the application.282828
- Registers by giving the detailsas a donor.
- The database will have all the details and if a user posts a request, then the concernedbloodgroup donors will get notified about it.

5.3 User Stories



Use the below template to list all the user storiesfor the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	App Registration	USN-1	As a user, I can register forthe application by entering my email, password, and confirmingmy password.	I can access my account / dashboard	High	Sprint-1

Customer	Confirmation	USN-2	As a user, I will receive confirmation emailonce I have registered for the application	I can receive confirmation email &click confirm	High	Sprint-1
Customer	Registration	USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
Customer	Registration	USN-4	As an Administrator, I can view the database ofthe registered users.	I can see who are the persons registered here and their mailids.	Medi um	Sprint-1
Customer	Login	USN-5	As a user, I can log into the applicati on by entering email &passwo rd	I can register & access the dashboa rd with Gmail Login	High	Sprint-1
Customer	Dashboard	USN-6	As a user, I can view and manage my profile, donation history and download the receipts.	I can view and manage mydata at each section of the dashboard.	High	
Customer (Web user)	Website	USN-7	Act as an interface between the customer and donors		High	Sprint-4
Custom er Care Executive	Help/support		As a customer care executive, I can solve the queries of the users.	I can reply to their queries and solvetheir related problems.	High	Sprint-5

Administrator	Admin Dashboard	USN-1	As an Admin, I can view and manage thecustomer details	I can access the admin dashboard and perform functions like adding new donor, deleting the	High	Sprint- 3

Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
Administrator	Admin Dashboard	USN-2	As an Admin, I can manage plasma collectiondetails	I can add the new plasma stock, delete the existing stock andview the plasma details	High	Sprint - 4
User Type	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Acceptance criteria**	**Priority**	**Release**
Administrator	Admin Dashboard	USN-3	As an Administrator, I can view the databaseof the registered users.	I can see who are thepersons registered here and their mail ids.	Medium	Sprint - 4
Administrator	Manage Contactus Query	USN-4	In this section, the admin can manage the query which is received from users.	I can view and answer the queries raisedby the userand manage theirqueries	Low	Sprint - 3

6. PROJECT PLANNING & SCHEDULING

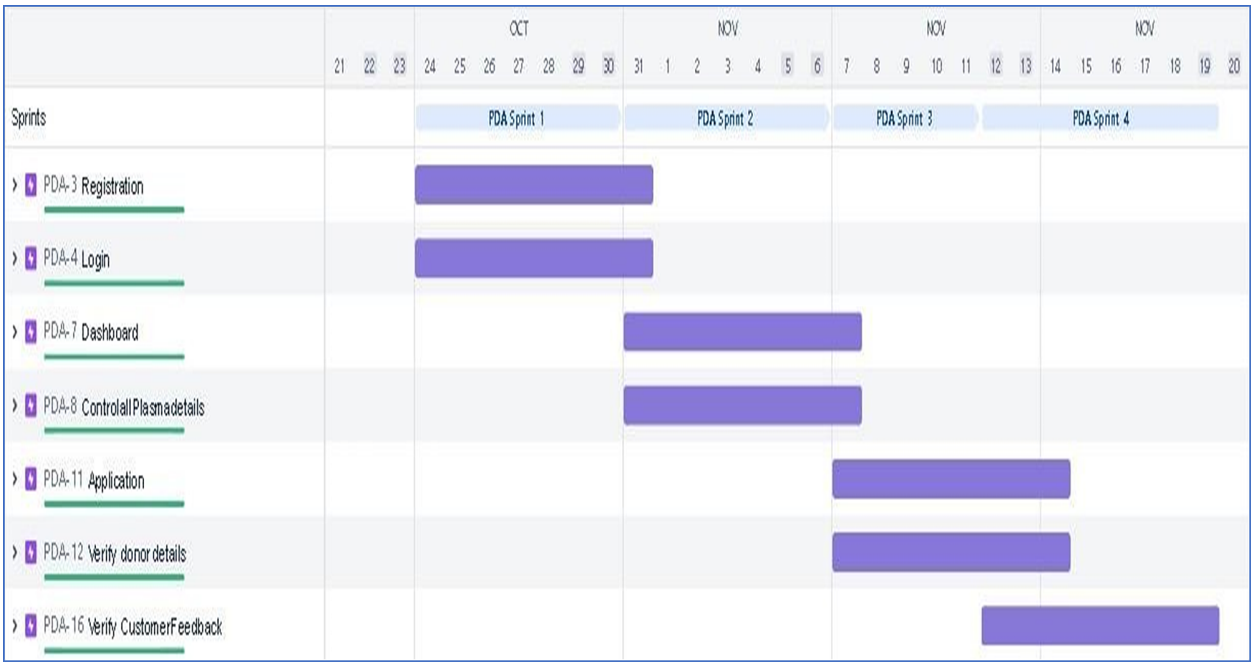
6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User StoryNumber	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Simulation creation	USN-1	Connect withpython code	2	High	Santhoshkumar SSaravanakumar GShiffin Paul J Vishwa Bharathi J
Sprint-2	Software	USN-2	Creating an IBM Watsonin Cloud platform	2	High	Santhoshkumar S Saravanakumar GShiffin Paul J Vishwa Bharathi J
Sprint-3	MIT AppInventor	USN-3	Develop an Plasma donorapplication	2	High	Santhoshkumar SSaravanakumar GShiffin Paul J Vishwa Bharathi J
Sprint-4	Dashboard	USN-4	Design theModules and test the app	2	High	Santhoshkumar S Saravanakumar GShiffin Paul J Vishwa Bharathi J
Sprint-5	Web UI	USN-5	To make the user to interact with software.	2	High	Santhoshkumar S Saravanakumar G Shiffin Paul J Vishwa Bharathi J

6.2 Sprint delivery schedule

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

6.3 Reports from JIRA



7. CODING & SOLUTIONING

7.1 Feature 1

Python

- Python is a widely-used, interpreted, object-oriented, and high-level programming language with dynamicsemantics, used for generalpurpose programming. It's everywhere, and people use numerous Python-powered devices on a daily basis, whether they realize it or not.
- Python was created by Guido van Rossum, and first releasedon February 20, 1991.
- Python is derived from many other languages, including ABC, Modula- 3, C, C++, Algol-68, Smalltalk, and Unix shell and other scripting languages.
- Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL)
- It is easy to learn – the time needed to learn Python is shorter than for many other languages; this means that it's possible to start the actual programming fast
- It is easy to use for writing new software – it's often possible to write code faster when using Python.
- It is easy to obtain, install and deploy – Python is free, open and multiplatform; not all languagescan boast that.
- Programming skills prepare you for careers in almost any industry and are required if you want to continueto more advanced and higherpayingsoftware development and engineering roles.
- Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

7.2 Feature 2

Flask

- **Flask** is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries.
- It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide

common functions. However,Flask supports extensions that can add application features as if they were implemented in Flask itself.

- Extensions exist for object-relational mappers, form validation, upload handling, variousopen authentication technologies and several common framework related tools.
- Applications that use the Flask framework include Pinterest and LinkedIn.

7.3 Database Scheme

IBM Db2

- DB2 is a databaseproduct from IBM.
- It is a Relational Database Management System (RDBMS). DB2 is designedto store, analyzeand retrieve the data efficiently.
- DB2 product is extended with the support of Object-Oriented features andnon-relational structures with XML.
- Provide a massively parallelprocessing (MPP) architecture Exploits Hive, HBase and Apache Spark concurrently for best-in-class analyticcapabilities.
- Provides low latency supportfor ad-hoc and complex queries,high performance, and federation capabilities Understands dialects from other vendors and variousproducts from Oracle,IBM® Db2® and IBM Netezza®Enables advanced row and column security

Kubernates

- **Kubernetes** is also known as **'k8s'.**
- **Kubernetes** is an extensible, portable, and open-source platform designedby **Google** in **2014**.
- It is mainly used to automate the deployment, scaling, and operations of the container-based applications across the clusterof nodes.
- Kubernetes helps to manage containerised applications in various types ofphysical, virtual, and cloud environments.
- Google Kubernetes is a highly flexible containertool to consistently deliver complexapplications running on clusters of hundreds to thousandsof individual servers
- Kubernetes is the Linux kernel whichis used for distributed systems.
- It helps you to be abstract the underlying hardware of the nodes(servers) and offers a consistent interface for applications that consume the shared pool of resources.

8. Test case

8.1.TESTING

a. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectation and does notfail in an unacceptable manner.
b. There are various types of test. Each test type addresses a specific testing requirement.

Test case ID	Feature Type	Component	Test Scenario	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	BUG ID	Executed By
LoginPage_TC_OO1	UI	Admin Login Page	Verify user is able to see the Login/Singup popup when user clicked on My account button	1.Enter URL http://127.0.0.1:8000/ and click go 2.Click on My Account dropdown button 3.Verify login/Singup popup displayed or not	Username: rit password : rit123	Login/Singup popup should display and navigate to Admin dashboard	Working as expected	Pass		Y		Admin
LoginPage_TC_OO2	Functional	Patient Login page	Verify user is able to log into application with InValid credentials	1.Enter URL http://127.0.0.1:8000/ and click go 2.Click on 3.Verify login/Singup popup with below Patient elements: a.username text box b.password text box c.Login button	Username: shriram password : 20190112 80	Application should show 'Incorrect Username or password ' validation message.	Working as expected	Fail	Steps are not clear to follow	N	BUG-1234	Patient
LoginPage_TC_OO3	Functional	Donor Login Page	Verify user is able to log into application with Valid credentials	1.Enter URL http://127.0.0.1:8000/and click go 2.Click on 3.Enter Valid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username: sathish password: 201901120	User should navigate to user Donor Home Page	Working as expected	Pass		Y		Donor
LoginPage_TC_OO4	Functional	Patient Login page	Verify user is able to log into application with InValid credentials	1.Enter URL http://127.0.0.1:8000/and click go 2.Click on 3.Enter Valid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username: shriram password: 201901128	User should navigate to user Donor Home Page	Working as expected	Pass		Y		Patient

8.2.User Acceptance Testing

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets
LoginPage_TC_001	Functional	Home Page	Verify user is able to see the Login/Signup popup when user clicked on Login/Signup button		1.Enter URL and click go 2.Click on Login/Signup button 3.Verify login/Singup popup displayed or not		Login/Signup page popup should display	Working as expected	Pass	
LoginPage_TC_002	UI	Home Page	Verify the UI elements in Login/Signup popup		1.Enter URL and click go 2.Click on Login/Signup button 3.Verify login/Singup popup with below UI elements: a.email text box b.password text box c.Login button d.New customer? Create account link		Application should show below UI elements: a.email text box b.password text box c.Login button. d.New customer? Create account link	Working as expected	Pass	Recover Password Feature not yet added
LoginPage_TC_003	Functional	Home page	Verify user is able to log into application with Valid credentials		1.Enter URL and click go 2.Click on Login/Signup button 3.Enter Valid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username: charan@gmail.com password: Testing123	User should navigate to user account homepage	Working as expected	Pass	

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status
HomePage_TC_006	Functional	Home page	Verify User is able to Sign in With his Details		1.Enter URL and click go 2.Click on Sign in button 3.Redirected to Sign in page 4.Enter valid password and username 5.Click on login button	Username: charan@gmail.c	Application must redirect to proper webpage without delay	Working as expected	Pass
HomePage_TC_007	Functional	Home page	Verify User is able to Register With his Details		1.Enter URL and click go 2.Click on Login/Signup button 3.Enter Valid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username: charan@gmail.com password: Testing123 Email : abc@gmail.com PhoneNo: 123456789 Sex: -M Blood:B+ Address:123 street ,abc	Application must redirect to proper webpage after verifying the details	Working as expected	Pass
Register_TC_008	UI	Register Page	Verify the UI elements in Login/Signup popup		1.Enter URL and click go 2.Click on Login/Signup button 3.Verify login/Singup popup with below UI elements: a.Name b.email text box c.password text box d.Phone No e.Sex f:Age g:Blood	Username: charan@gmail.com password: Testing123 Email : abc@gmail.com PhoneNo: 123456789 Sex: -M Blood:B+ Address:123 street ,abc nagar,india	Application should show below UI elements: a.Name b.email text box c.password text box d.Phone No e.Sex f:Age g:Blood h:Address Sign up Button	Working as expected	Pass

9. RESULTS

9.1.Performance Metrics

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

ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- **Speed**: This website is fast and offers great accuracy as compared tomanual registered keeping.
- **Maintenance:** Less maintenance is required
- **User Friendly:** It is very easy to use and understand. It is easily workableand accessible for everyone.

- **Fast Results:** It would help you to provide plasma donorseasily dependingupon the availability of it.

DISADVANTAGES:

- **Internet:** It would requirean internet connection for the workingof thewebsite.
- **Auto- Verification:** It cannot automatically verifythe genuine users.

11. CONCLUSION

- The efficientway of finding plasma door for the infected people is implemented using the plasmadonor website that is hostedon IBM Cloudplatform.

- To ensure the smooth functioning of the web site operation. I have hosted the website in IBM Db2 & Kubernates Cluster to make sure the operations are running successfully Cloud lambda function is used and to deploy the application IBM Db2 service is used.

12. FUTURE SCOPE

- Upgrading the UI that is more user-friendly which will help many users to accessthe website and also ensuresthat many plasmadonors can be addedinto the community.

- Using elastic load balancer, it helps to handle multiplerequests at the same time which will maintain the uptime of the websitewith negligible downtime.

13. APPENDIX

Source Code:

```
from flask import Flask,redirect,url_for,render_template,request,make_response,jsonify,request
import ibm_db
from flask import request
import json
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=55fbc997-9266-4331-afd3-
888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=31929;SECURITY=SS
L;SSLServerCertificate=PDA.crt;UID=tlk28900;PWD=uq84umWJgJHoEaMY","","")
print(conn)
```

```
print("connection successful...")
app = Flask(__name__,template_folder='template')
@app.route('/')
def home():
    return render_template("landing.html")

@app.route('/home')
def dash():
    return render_template("dashboard.html")

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

    if request.method=='POST':
        username = request.form['username']
        password = request.form['password']
        sql = "select * from user where username=? and password=?"
        stmt = ibm_db.prepare(conn,sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.bind_param(stmt, 2, password)
        ibm_db.execute(stmt)
        dic = ibm_db.fetch_assoc(stmt)
        print(dic)

        requests = []
        if dic:
            role = dic['ROLE']
            # sql = "select * from user where blood_group=?"
            # stmt = ibm_db.prepare(conn, sql)
            # ibm_db.bind_param(stmt, 1, username)
            # ibm_db.execute(stmt)
            # dic = ibm_db.fetch_assoc(stmt)


            # while dic != False:
            #     single_request = {
            #         'name': dic['NAME'],
            #         'age': dic['AGE'],
            #         'sex': dic['SEX'],
            #         'blood_type': dic['BLOOD_TYPE']
```

```
            #    }
            #    print(single_request)
            #    requests.append(single_request)
                # dic = ibm_db.fetch_assoc(stmt)

            return render_template('dashboard.html',username=username,role=role)



        else:

            return render_template('login.html')
        return redirect(url_for('home'))
    else:
        print("else")
        return render_template('login.html')

@app.route('/signup',methods=['POST','GET'])
def signup():
    if request.method=='POST':
        username = request.form['username']
        email = request.form['email']
        password = request.form['password']
        roll_no = request.form['roll_no']
        sex = request.form['sex']
        age = request.form['age']
        address = request.form['address']
        blood_group = request.form['blood_group']
        sql = "insert into user values(?,?,?,?,?,?,?,?,?)"
        prep_stmt = ibm_db.prepare(conn,sql)
        ibm_db.bind_param(prep_stmt,1,username)
        ibm_db.bind_param(prep_stmt,2,email)
        ibm_db.bind_param(prep_stmt,3,password)
        ibm_db.bind_param(prep_stmt,4,roll_no)
        ibm_db.bind_param(prep_stmt,5,sex)
        ibm_db.bind_param(prep_stmt,6, age)
        ibm_db.bind_param(prep_stmt,7, "USER")
        ibm_db.bind_param(prep_stmt,8, address)
        ibm_db.bind_param(prep_stmt,9, blood_group)
        ibm_db.execute(prep_stmt)
        #db post operation
```

```
        return redirect(url_for('login'))
    elif request.method=='GET':
        return render_template('signup.html')

@app.route('/toggle',methods=['POST'])
def toggle_user():


    data =  request.get_json(force=True)

    username =data['username']
    role = data['role']
    print(username)
    print(role)
    sql = "update user set role=? where username=?"
    prep_stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(prep_stmt, 1, role)
    ibm_db.bind_param(prep_stmt, 2, username)
    ibm_db.execute(prep_stmt)
    return jsonify(
        status = "success",
        role = role
    )

@app.route('/requestPlasma',methods=['POST'])
def requestBloodPlasma():
    #fetch mail address of the donors
    username = request.form['username']
    name = request.form['name']
    age = request.form['age']
    sex = request.form['sex']
    blood_type = request.form['bloodtype']
    sql = "select email from user where blood_group=?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt, 1, blood_type)
    ibm_db.execute(stmt)
    dic = ibm_db.fetch_assoc(stmt)
    while dic!=False:
        print(dic['email'])
    #send mail
```

```
    #insert data into requests table
    sql = "insert into bloodrequests(username,name,age,sex,blood_type) values (?,?,?,?,?)"
    prep_stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(prep_stmt, 1, username)
    ibm_db.bind_param(prep_stmt, 2, name)
    ibm_db.bind_param(prep_stmt, 3, age)
    ibm_db.bind_param(prep_stmt, 4, sex)
    ibm_db.bind_param(prep_stmt, 5, blood_type)
    ibm_db.execute(prep_stmt)

    return jsonify(
        name = name,
        age = age,
        sex = sex,
        bloodtype = blood_type,
        status = "yes"
    )

@app.route('/getrequests',methods=['POST'])
def getBloodRequests():
    data =  request.get_json(force=True)

    username =data['username']
    sql = "select * from bloodrequests where username=?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt, 1, username)
    ibm_db.execute(stmt)
    dic = ibm_db.fetch_assoc(stmt)
    requests = []
    print(dic)
    while dic != False:
        single_request = {
            'name':dic['NAME'],
            'age':dic['AGE'],
            'sex':dic['SEX'],
            'blood_type':dic['BLOOD_TYPE']
        }
        print(single_request)
        requests.append(single_request)
        dic = ibm_db.fetch_assoc(stmt)
```

```
    return jsonify(
        username = username,
        requests = requests
    )
if __name__=='__main__':
    app.run(host="0.0.0.0",debug = True)
```

GitHub Link

https://github.com/IBM-EPBL/IBM-Project-1111-1658372594

Project Demo Link

https://drive.google.com/file/d/14jBY7ZJK8y8ASFvKjyhB1-CabLVodiFg/view?usp=drivesdk