PLASMA DONOR APPLICATION

IBM-Project-31855-1660205660

NALAIYA THIRAN PROJECT BASED LEARNING ON PROFESSIONAL READLINESS FOR INNOVATION, EMPLOYNMENT AND ENTERPRENEURSHIP

A PROJECT REPORT

MOHAMED SAJITH A (950819106312)

MEENAKSHI SUNDARAM S (950819106311)

SINDHU N (950819106315)

PORPANDIAN V (950819106313)

BACHELOR OF ENGINEERING IN ELECTRONICS AND COMMUNICATION ENGINEERING

Government College of Engineering

TIRUNELVELI– 627007

BONAFIDE CERTIFICATE

Certified that this project report **"PLASMA DONOR APPLICATION"** the bonafide work of **"MOHAMED SAJITH(950819106312), MEENAKSHI SUNDARAM(950819106311) , SINDHU(950819106315) , PORPANDIAN (950819104708)"** who carried out the project work under my supervision**.**

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

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.

Server less computing is the current trend in software application development. Micro services 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 Dynamo.

PURPOSE

The necessity of blood has become a significant concern in the present context all over the world. Due to a shortage of blood, people couldn’t save themselves or their friends and family members. A bag of blood can save a precious life. Statistics show that a tremendous amount of blood is needed yearly because of major operations, road accidents, blood disorders, including Anemia, Hemoph ilia, and acute viral infections likeDengue

,etc

CHAPTER 2

LITERATURE SURVEY

EXISTING PROBLEM

In the existing app we cannot upload and download the latest updates.Mostly the details of donations and donors were managed and maintained manualy.No use of Web Service and Remoting.That lead to risk in mismanagement and of data when the project is under development .Moreover it is less Secure .There is no proper co-ordination between different applications and users.It is fewer user friendly.There is less connection between the plasma authority and donors .

REFERENCE

1.)HTML-documentation:- https://html.org/docs/getting-started.html

2.)CSS-documentation:- https://css.org/dist/latest-v14.x/docs//

3.)Python-documentation:- https://pyhton.com/en/starter/l

4.)Cloud-service:- https://docs.cloud.com/manual/tutorial/getting-started/

5.)Github:- https://gist.github.com/hofmannsven/6814451

6.)W3School

7.)YouTube

PROBLEM STATEMENT DEFINITION

To identify and manage the loss of some of these substances through plasma donation can lead to an electrolyte imbalance

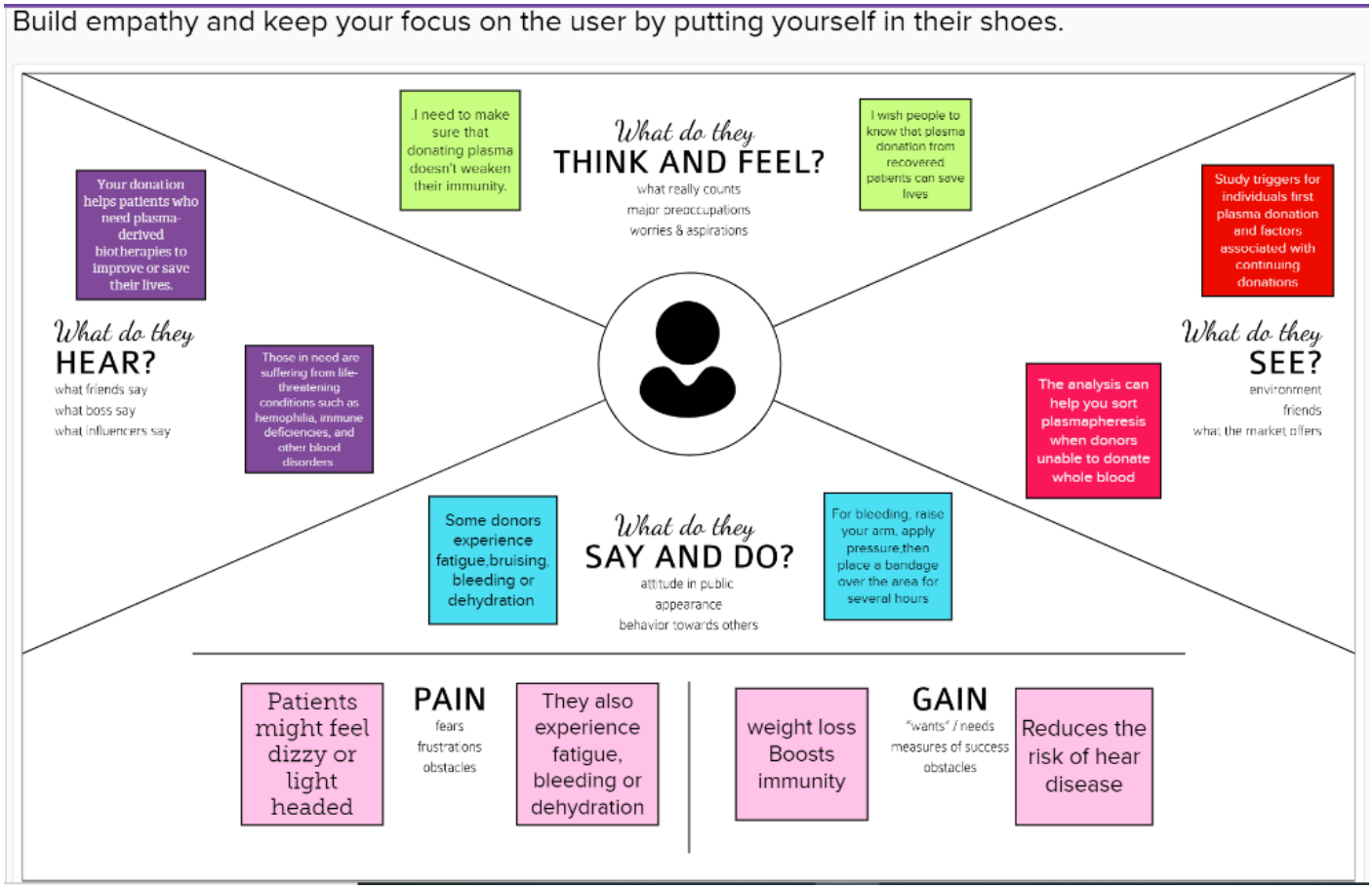
Who does the problem affect?	There can be minor side effects of plasma donor : Plasma is the liquid part of the blood. It contains proteins and antibodies that are crucial for clotting and immunity. Around 55% of the blood is plasma..
What are the boundaries of the problem?	For most people, donating plasma does not cause any side effects, but some donors can experience fatigue, bruising, bleeding, or dehydration. Additionally, you may feel dizzy or lightheaded. While not typical, fainting can also occur. It's rare, but more serious infections or reactions can occur, which can be treated.
What is the issue?	BLOODR application can resolve these issues by connecting patients promptly with a large pool of donors in the same region via an authorized clinic. When a patient needs a blood donation, the clinic (where the patient is admitted) can use the application to contact the blood donors in the vicinity or nearby city based on their location.
When does the issue occurs?	Certain chronic illnesses, such as hepatitis and HIV, automatically disqualify someone from donating. Other active conditions, such as tuberculosis, must be treated first for a certain amount of time before an individual can donate blood or plasma.
Where is the issue occurring?	It occurs if the body has low levels of nutrients and salts. Fatigue after plasma donation is another common side effect, but it’s usually mild.
Why is it important that we fix theproblem?	Your donation helps patients who need plasma-derived biotherapies to improve or save their lives. Those in need are suffering from life-threatening conditions such as hemophilia, immune deficiencies, and other blood disorders.

CHAPTER 3

IDEATION AND PROPOSED SOLUTION

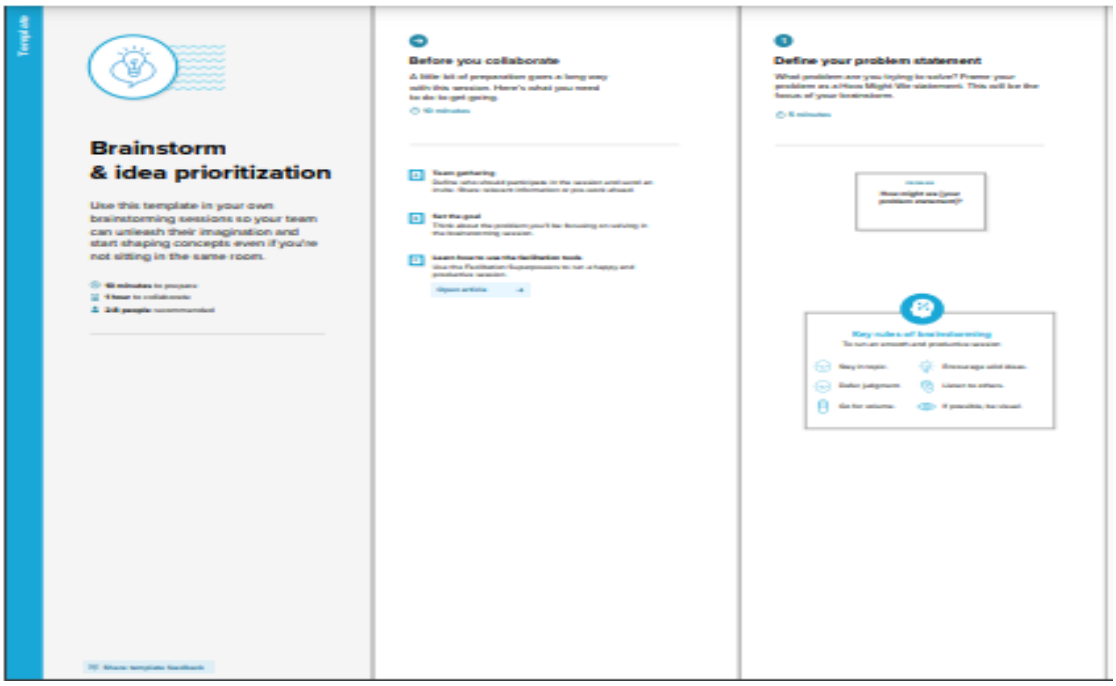
EMPATHY MAP CANVAS

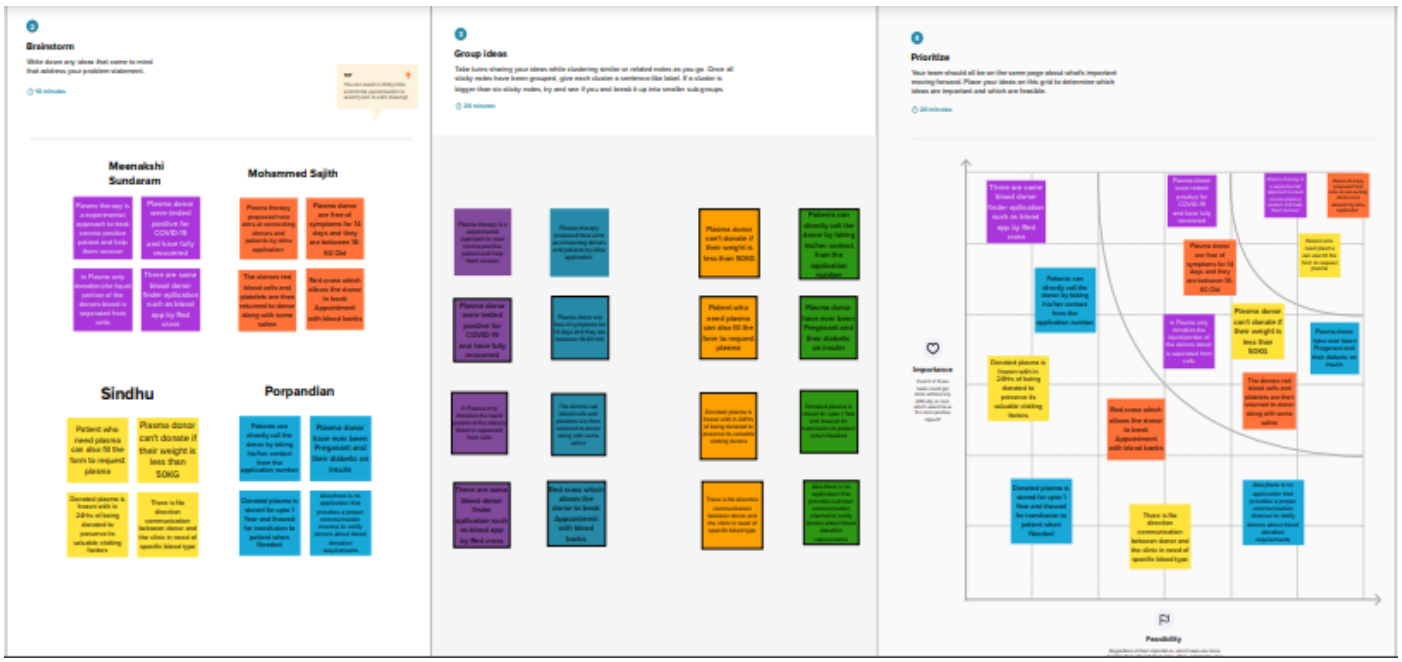
An **empathy map** is a collaborative visualization used to articulate what we know about a particular type of user. It externalizes knowledge about users in order to create a shared understanding of user needs, and aid in decision making.



IDEATION & BRAINSROMING

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.





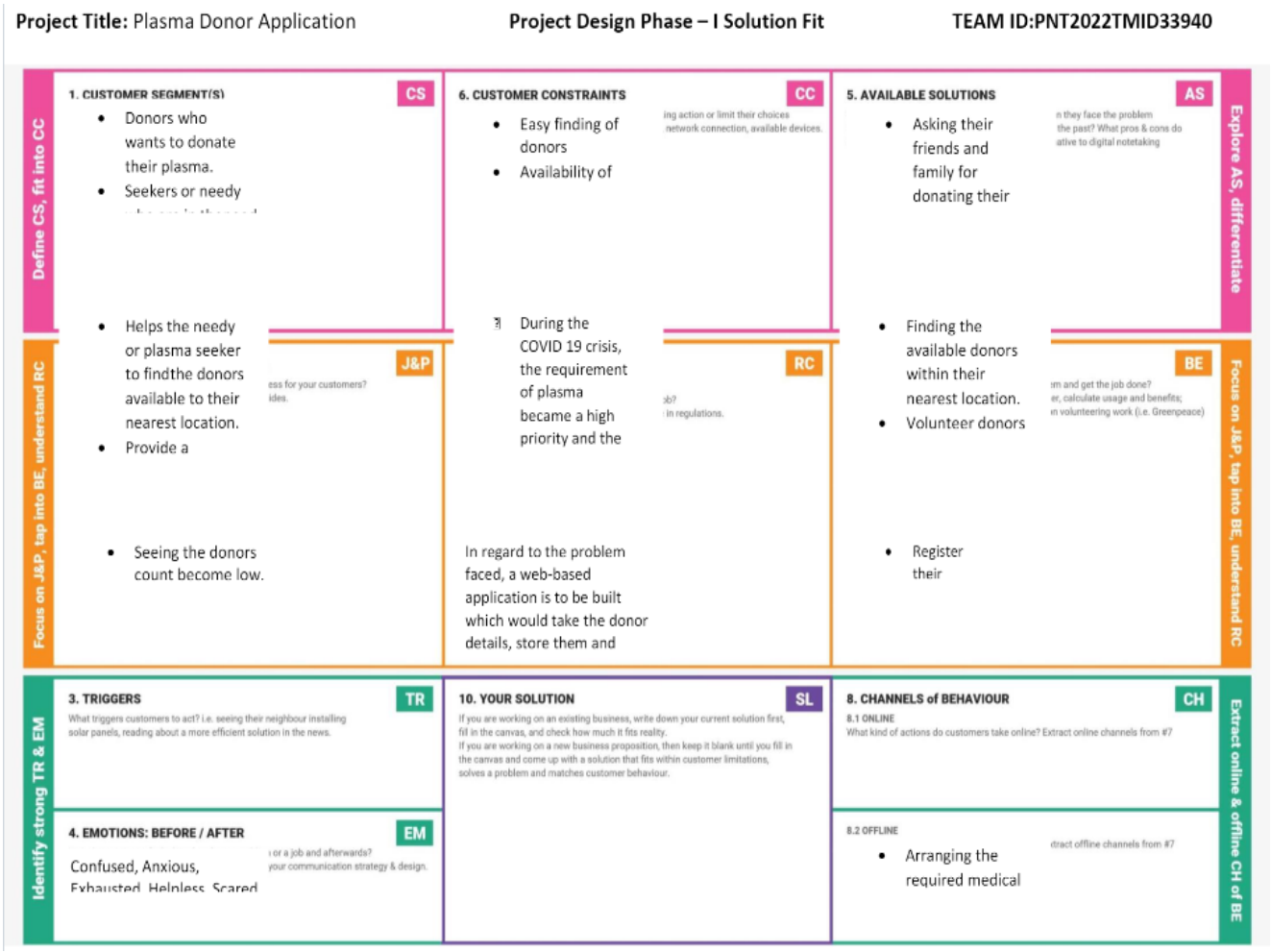
PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Plasma is a critical part of the treatment for manyserious health problems. This is why there are blood drives asking people to donate blood plasma. During the COVID 19 crisis, the requirementof plasma became a high priority and the donor count has become low. Saving the donor information and helping the needy by notifying the current donors list, would be a helping hand.
2.	Idea / Solution description	In regard to the problem faced, an application is tobe built which would take the donor details, store them and inform them upon a request. In This way,the one who in need in plasma can able to make a request, then the application can able to read the information of donors that are stored in database and informing up the donors regarding that request.
3.	Novelty / Uniqueness	This application can able to perform certain functionality and possess certain feature whichare unique. Those are listed below: • Those who want to donate their plasma can do by simply register byuploading their covid-19 recovery certificate. • It can able to find donors who are located close to the needy by usingGPS location tracking. • A chat-bot to answer frequently askedquestion about plasma donation.
4.	Social Impact / Customer Satisfaction	By using the application one can easily able to find the donor at emergency situations and theone who willing to donate their plasma can easily be connected with the needy. Since this process takes place continuously, we can builda healthy society of tomorrow.
5.	Business Model (Revenue Model)	We can provide some additional medical services in order to generate some revenue. Medical services like blood test, medical record management, medical transportation serviceand some other health care service.

PROBLEM SOLUTION FIT

The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer's problem. What do you have with a Problem-Solution fit

on Fit?



CHAPTER 4

REQUIREMENT ANALYSIS

4.1FUNCTIONAL REQUIREMNT

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form (WebApp)
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Certification	After the donor donates plasma, we will give them a certificate of appreciation and authentication.
FR-4	Statistical data	The availability of plasma is given in the page as stats, which will be helpful for the users.
FR-5	User Plasma Request	Users can request to donate plasma by filling out the request form on the page. Once the request is submitted, they will get an email
FR-6	Searching/reporting requirements	Users can use the search bar to look up information about camps and other topics.

4.2NON-FUNCTIONAL

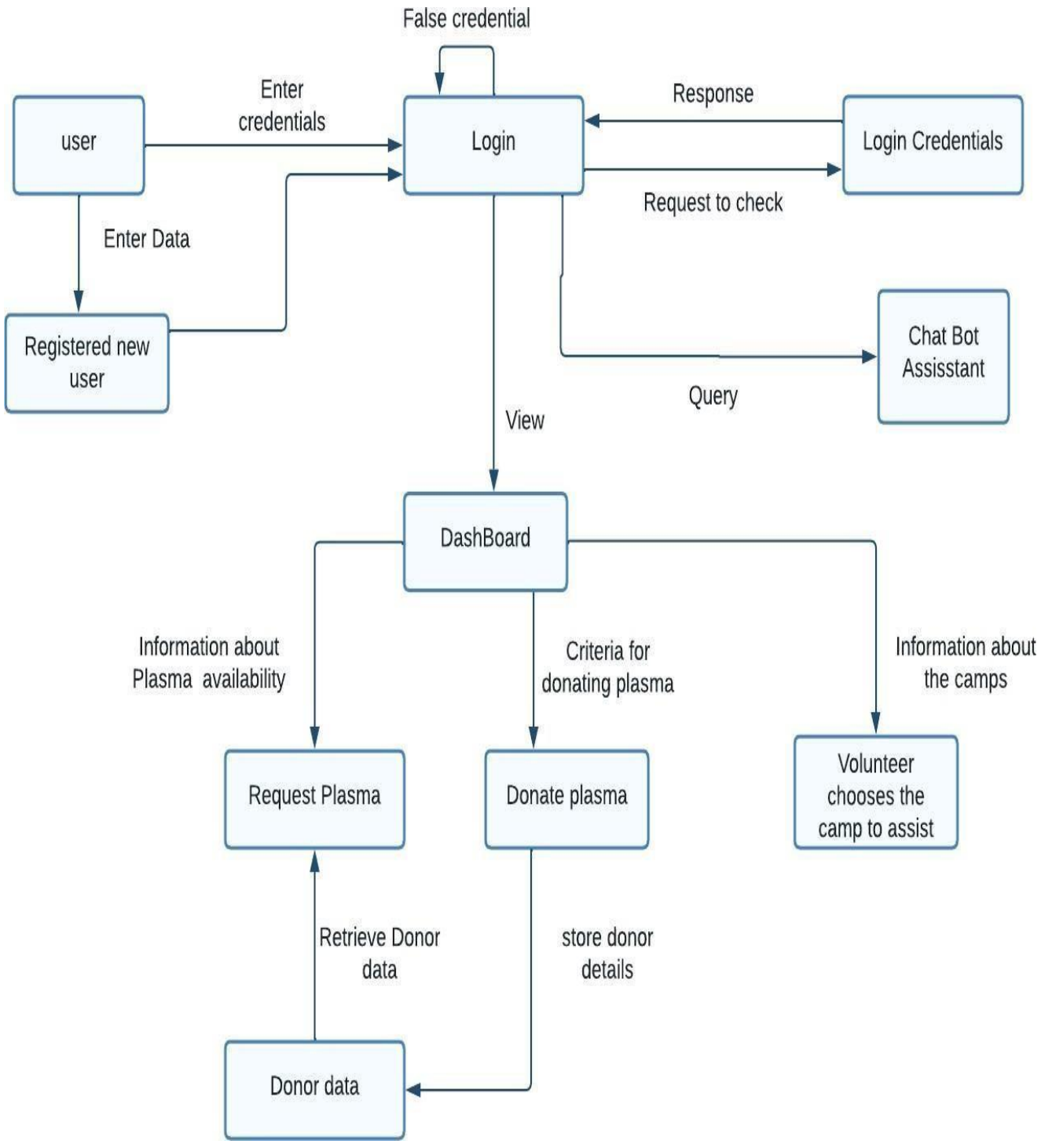
REQUIREMENTS

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Must have a good looking User friendly interface.
NFR-2	**Security**	It must be secured with the proper username andpassword.
NFR-3	**Reliability**	The system should be made in such a way that it is reliable in its operations and for securing thesensitive details.
NFR-4	**Performance**	Users should have a proper Internet Connection.
NFR-5	**Availability**	The system including the online and offlinecomponents should be available 24/7.
NFR-6	**Scalability**	The application has the ability to handle growingnumbers of users and load without compromising on performance and causing disruptions to userexperience.

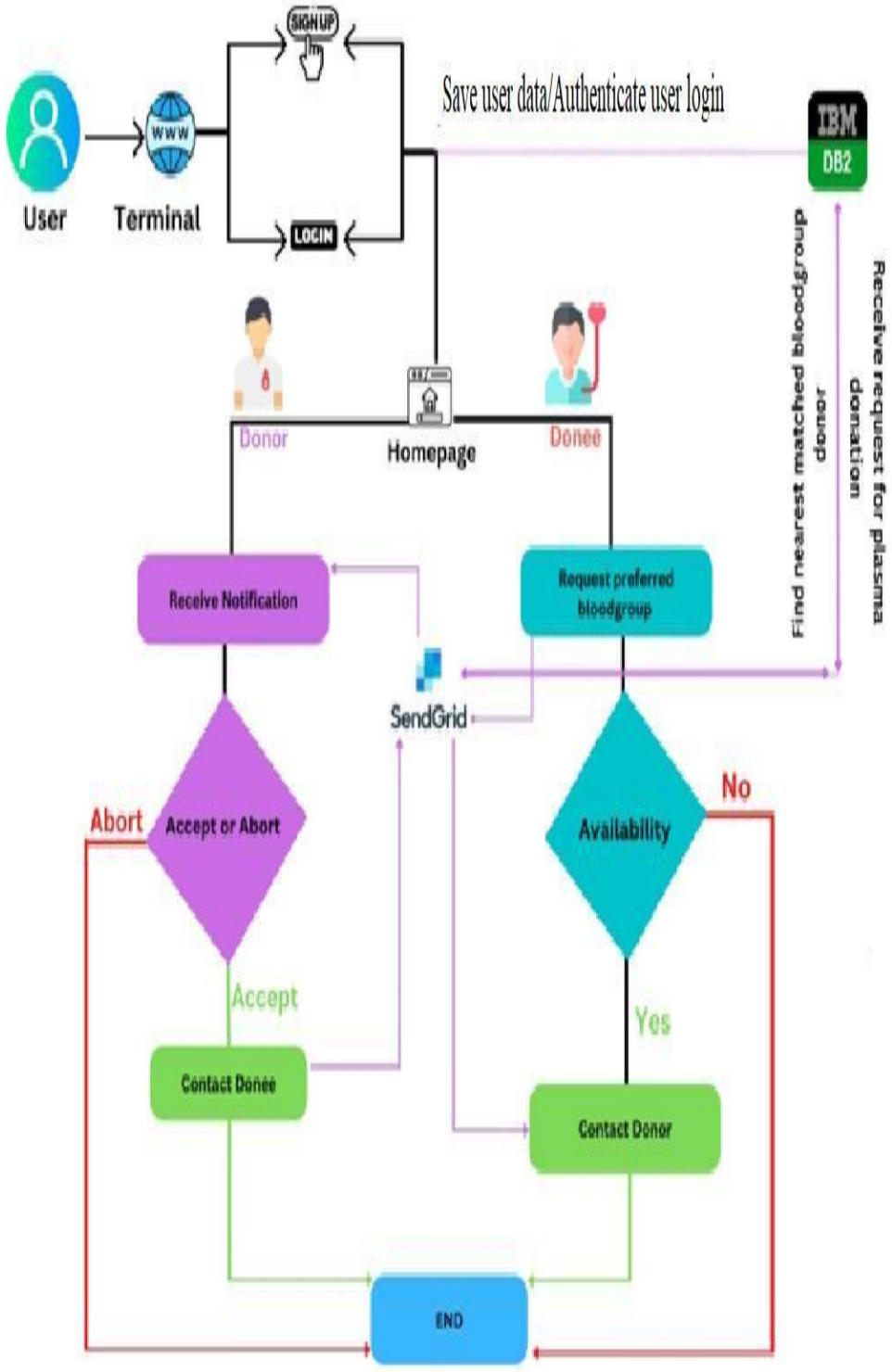
CHAPTER 5

PROJECT DESIGN

5.1DATA FLOW DIAGRAMS



SOLUTION AND TECHNICAL ARCHITECHTURE



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 register for the application by entering my email, password, and confirmingmy password.	I can access my account /dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation emailonce I have registered for the application	I can receive confirmationemail & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the applicationthrough Gmail	I can receive confirmation notifications through Gmail	Medium	Sprint-1
	Login	USN-4	As a user, I can log into the application by entering email & password	I can access into my User profile and view details indashboard	High	Sprint-1
	Dashboard	USN-5	As a user,I can send the proper requests todonate and obtain plasma.	I can receive appropriate notifications through email	High	Sprint-1
Customer (Web user)	Login	USN-6	As a user,I can register and application by entering email & password toview the profile	I can access into my User profile and view details indashboard	High	Sprint-1
	Dashboard	USN-7	As a user,I can send the proper requests todonate and obtain plasma.	I can receive appropriate notifications through 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 help with user-facing aspects of a website, like its appearance,navigation and use of media.	I can change the appearance navigation in	Medium	Sprint-3
		USN-10	As an administrator, I can involve working withthe technical side of websites.	I can help with such as troubleshooting issues, setting up web hosts, ensuring users have access and programming servers	Medium	Sprint-1
Chatbot	Dashboard	USN-11	In addition the Customer care executive,chatbot can try to address user's concerns and questions	I can reply to all the queries related to our application	Medium	Sprint-3

CHAPTER 6

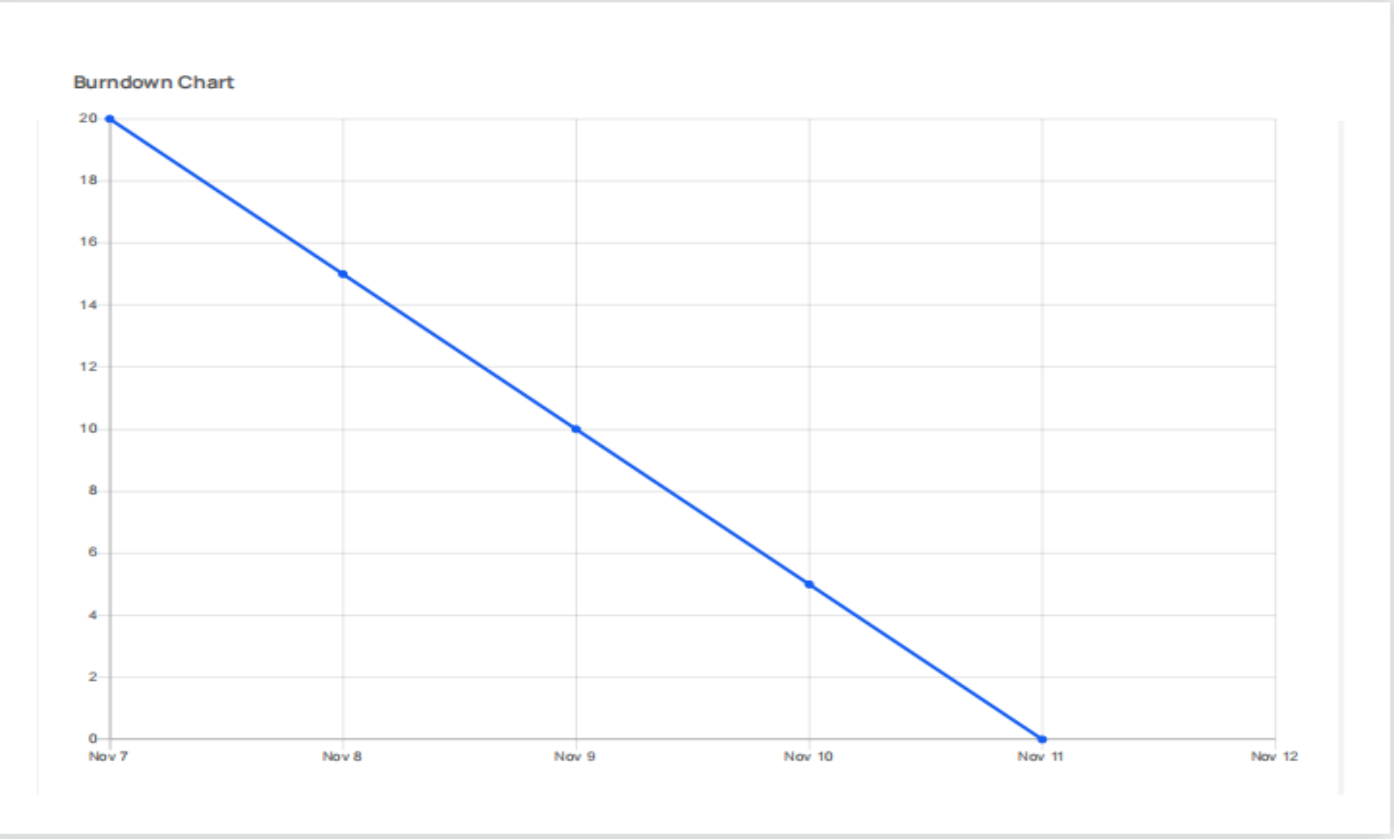
PROJECT PLANNING AND SCHEDULING

SPRINT PLANNING AND ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Setting the required environment	USN-1	Gathering information for the user .	2	High	Meenakshi Sundaram S Mohamed Sajith A Porpandian V Sindhu N
Sprint-1		USN-2	Creating account in ibm cloud,docker,kubernets	1	High	Meenakshi Sundaram S MohamedSajith A Porpandian V Sindhu N
Sprint-1		USN-3	Creating container registery.	2	High	Meenakshi Sundaram S MohamedSajith A Porpandian V Sindhu N
Sprint-2		USN-4	Creating the ibm db 2 to store the data of theuser.	2	Medium	Meenakshi Sundaram S MohamedSajith A Porpandian V Sindhu N
Sprint-2		USN-5	Creating simple flask application .	1	Low	Meenakshi Sundaram S MohamedSajith A Porpandian V Sindhu N
Sprint-2	Login	USN-6	As a user, I can log into the application by entering email & password	2	High	Meenakshi Sundaram S MohamedSajith A Porpandian V Sindhu N
Sprint-3		USN-7	As a user, I can log into the application by entering GMAIL.	2		Meenakshi Sundaram S MohamedSajith A Porpandian V Sindhu N
Sprint-3		USN-8	As a user, I can log into the application by entering facebook.	1	Low	Meenakshi Sundaram S MohamedSajith A Porpandian V Sindhu N
Sprint-3	Mail confirmation	USN-9	As a user, I will receive confirmation mail once Ilogged into it.	2	High	Meenakshi Sundaram S MohamedSajith A Porpandian V Sindhu N
Sprint-4		USN-10	As a user, I can log into application by enteringmy login credentials.	2	High	Meenakshi Sundaram S MohamedSajith A Porpandian V Sindhu N

Sprint-4	Local host	USN-11	Deploying my application in my local host bysetting up the system environment.	2	High	Meenakshi Sundaram S MohamedSajith A Porpandian V Sindhu N

Sprint-4	Ibm cloud	USN-12	Deploying my application in the ibm cloud bysetting up the cloud environment.	2	High	Meenakshi Sundaram S MohamedSajith A Porpandian V Sindhu N



REPORT FROM JIRA

CHAPTER 7

CODING &SOLUTIONING

7.1FEATURE 1

```
from flask import Flask,render_template,request,session,redirect,url_for
import ibm_db
import os
app=Flask(__name__)
app.secret_key='hidden'
conn = ibm_db.connect(
    f"DATABASE={os.environ.get('DATABASE')};"
    f"HOSTNAME={os.environ.get('HOSTNAME')};"
    f"PORT={os.environ.get('PORT')};"
    f"USERNAME={os.environ.get('DB_USERNAME')};"
    f"PASSWORD={os.environ.get('PASSWORD')};"
    "SECURITY=SSL;"
    f"SSLSERVERCERTIFICATE={os.environ.get('SSLSERVERCERTIFICATE')};",
    '',
    ''
)
print(conn)
@app.route("/")
def front():
    return render_template("front.html")
@app.route("/login",methods=["POST","GET"])
def login():
    return render_template("login.html")
@app.route("/signin",methods=["POST","GET"])
def signin():
    return render_template("signin.html")
```

```python
@app.route("/signin/details/stats",methods=["POST","GET"])
def s_stats():
    if request.method == "POST":
        global user
        user=""
        user_=request.form['user']
        name_ = request.form['name']
        father_ = request.form['father']
        age_ = request.form['age']
        gender_=request.form['gender']
        blood_=request.form['blood']
        phone_ = request.form['phone']
        mail_ = request.form['mail']
        address_ = request.form['address']
        city_ = request.form['city']
        state_ = request.form['state']
        pin_ = request.form['pin']
        query1 = "INSERT INTO details (username,name,father,age,gender,blood,phone,mail,address,city,state,pin) values (?,?,?,?,?,?,?,?,?,?,?,?)"
        insert_stmt1 = ibm_db.prepare(conn, query1)
        ibm_db.bind_param(insert_stmt1, 1, user_)
        ibm_db.bind_param(insert_stmt1, 2,name_)
        ibm_db.bind_param(insert_stmt1, 3,father_)
        ibm_db.bind_param(insert_stmt1, 4,age_)
        ibm_db.bind_param(insert_stmt1, 5,gender_)
        ibm_db.bind_param(insert_stmt1, 6,blood_)
        ibm_db.bind_param(insert_stmt1, 7,phone_)
```

```
        query1 = "INSERT INTO details (username,name,father,age,gender,blood,phone,mail,address,city,state,pin) values (?,?,?,?,?,?,?,?,?,?,?,?)"
        insert_stmt1 = ibm_db.prepare(conn, query1)
        ibm_db.bind_param(insert_stmt1, 1, user_)
        ibm_db.bind_param(insert_stmt1, 2,name_)
        ibm_db.bind_param(insert_stmt1, 3,father_)
        ibm_db.bind_param(insert_stmt1, 4,age_)
        ibm_db.bind_param(insert_stmt1, 5,gender_)
        ibm_db.bind_param(insert_stmt1, 6,blood_)
        ibm_db.bind_param(insert_stmt1, 7,phone_)
        ibm_db.bind_param(insert_stmt1, 8,mail_)
        ibm_db.bind_param(insert_stmt1, 9,address_)
        ibm_db.bind_param(insert_stmt1, 10,city_)
        ibm_db.bind_param(insert_stmt1, 11,state_)
        ibm_db.bind_param(insert_stmt1, 12,pin_)
        ibm_db.execute(insert_stmt1)
        print("success")
        user=user+user_
        return render_template("stats.html")
@app.route("/login/stats",methods=["POST","GET"])
def l_stats():
    if request.method == "POST":
        global user
        user=""
        username = request.form['username']
        password = request.form['password']
```

```
        sql = "SELECT * FROM Admin WHERE username = ? and password = ?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, username)
        ibm_db.bind_param(stmt, 2, password)
        result = ibm_db.execute(stmt)
        print(result)
        account = ibm_db.fetch_row(stmt)
        print(account)
        param = "SELECT * FROM Admin WHERE username = " + "\'" + username + "\'" + " and password = " + "\'" + password + "\'"
        print(param)
        res = ibm_db.exec_immediate(conn, param)
        print(res)
        dictionary = ibm_db.fetch_assoc(res)
        print(dictionary)
        # sendmail("hello sakthi","sivasakthisairam@gmail.com")
        msg=""
        if account:
            session['loggedin'] = True
            # session['id'] = dictionary["ID"]
            # userid = dictionary["ID"]
            session['username'] = dictionary["USERNAME"]
            # session['email'] = dictionary["EMAIL"]
            user=user+username
            return render_template('stats.html')
        else:
            msg = msg+'Incorrect username / password ! Try again'
```

```
@app.route("/login/stats/plasmarequest",methods=["POST","GET"])
def plasmareq():
    if request.method == "POST":
        param = "SELECT * FROM donors"
        result = []
        print(param)
        res = ibm_db.exec_immediate(conn, param)
        print(res)
        dictionary = ibm_db.fetch_assoc(res)
        print(dictionary)
        while dictionary != False:
            result.append(dictionary)
            dictionary = ibm_db.fetch_assoc(res)
        data_=(tuple(result))
        print(data_)
        return render_template("plasmarequest.html", datas=data_)
@app.route("/login/stats/plasmadonate",methods=["POST","GET"])
def plasmadonate():
    if request.method == "POST":
        para = "SELECT * FROM donors WHERE username = " + "\'" + user + "\'"
        re = ibm_db.exec_immediate(conn, para)
        dict = ibm_db.fetch_assoc(re)
        print(re)
        print(dict)
        if(dict==False):
            param1 = "SELECT * FROM details WHERE username = " + "\'" + user + "\'"

    return render_template('login.html',message=msg)


@app.route("/signin/details",methods=["POST","GET"])
def details():
    if request.method == "POST":
        user_name=request.form['username']
        pass_word=request.form['password']
        c_pass_word = request.form['confirm_password']
        if pass_word==c_pass_word:
            query="INSERT INTO Admin (username,password) values (?,?)"
            insert_stmt = ibm_db.prepare(conn, query)
            ibm_db.bind_param(insert_stmt, 1, user_name)
            ibm_db.bind_param(insert_stmt, 2, pass_word)
            ibm_db.execute(insert_stmt)
            msg='Account Created Successfully'
            return render_template("details.html",msg=msg)
        else:
            return render_template("signin.html",message="Check the password")
@app.route("/login_success/stats",methods=["POST","GET"])
def lo_stats():
        return render_template("stats.html")
@app.route("/login/stats/plasmarequest",methods=["POST","GET"])
def plasmareq():
```

CHAPTER 8

TEST CASE

Test Case ID	Purpose	TestCases	Result
TC1	Authentication	Password with length less than 4 characters	Password cannot be less than 4 characters
TC2	Authentication	User name with length less than 2 characters	User name cannot be less than 2 characters
TC3	Authentication	Valid user name with minimum 2 characters	User name accepted

TC4	Authentication	User name left blank	User name cannot be less than 2 characters
TC5	Authentication	Password field left blank	Password cannot be empty
TC6	Authentication	Minimum 4 characters valid password	Password accepted
TC7	Authentication	Password and Confirm Password did not match	Please enter same password

USER ACCEPTANCE TESTING

TEST CASE ID	TEST CASE DESCRIPTION
TC_001	Verify if user is able to login .
TC_002	Verify if user is able to create account.
TC_003	Verify if user can request for plasma donation.
TC_004	Verify if user can see the donors details.
TC_005	Verify if the registered voluntees details are valid one.
TC_006	Verify if the details are correctly stored in the database
TC_007	Verify if there is required storage space to store upcoming users details.

CHAPTER 9

RESULTS

9.1PERFORMANCE MATRICES

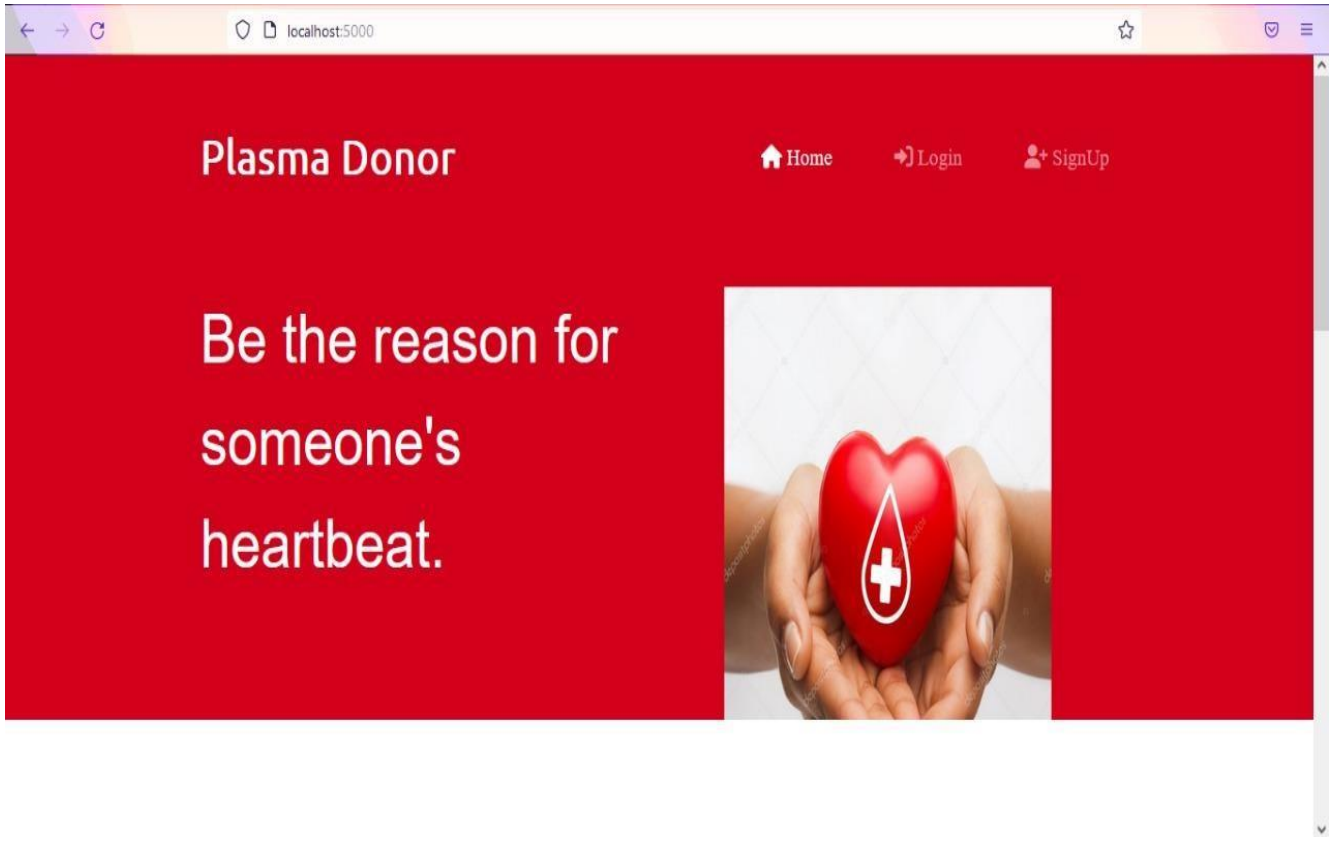


FIG.HOME PAGE

FIG.SIGIN PAGE,LOGIN PAGE

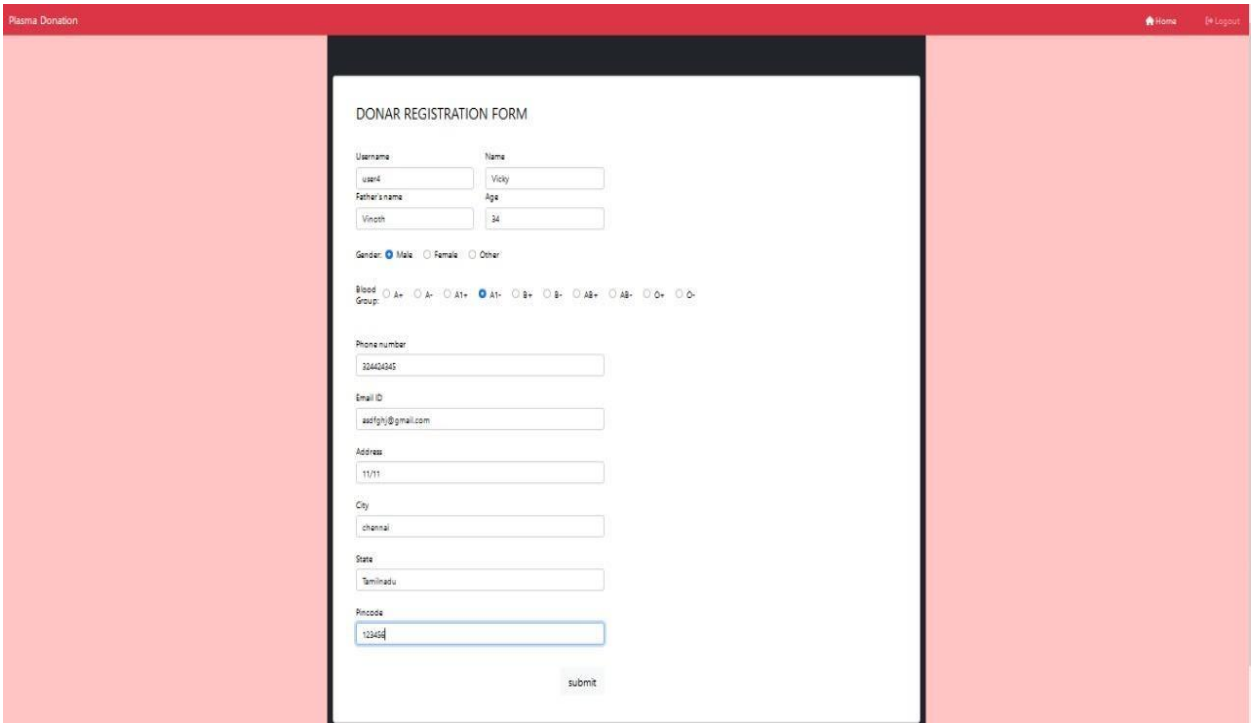


FIG.ACCOUNT CREATION

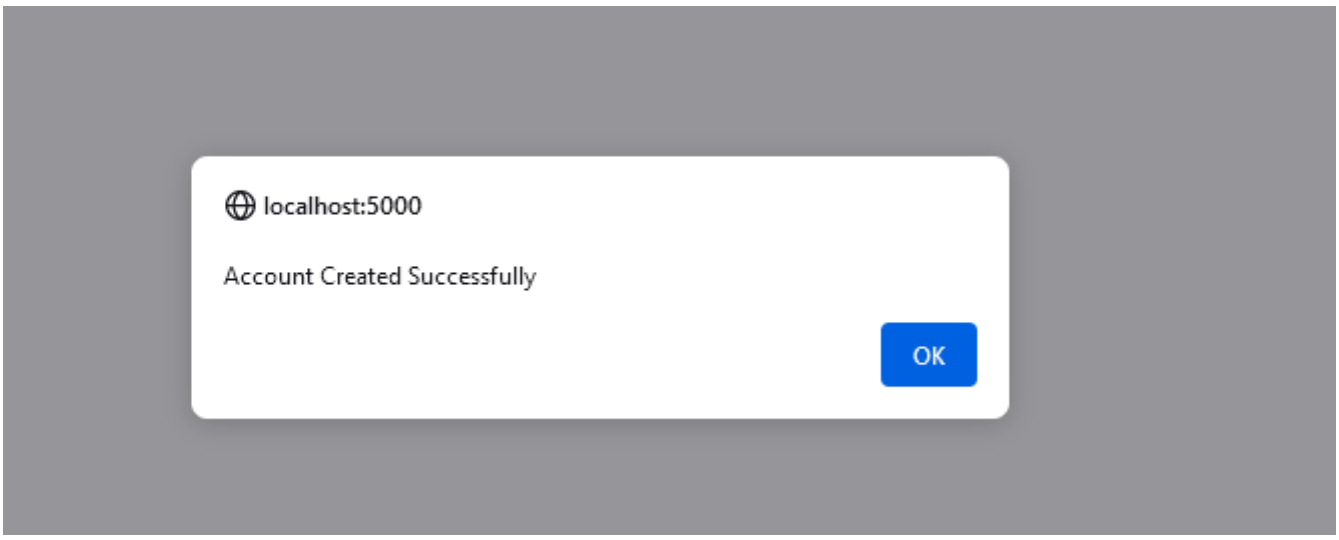


FIG.ACCOUNT CREATION SUCCESSFULL

FIG.DONATION STATISTICS PAGE

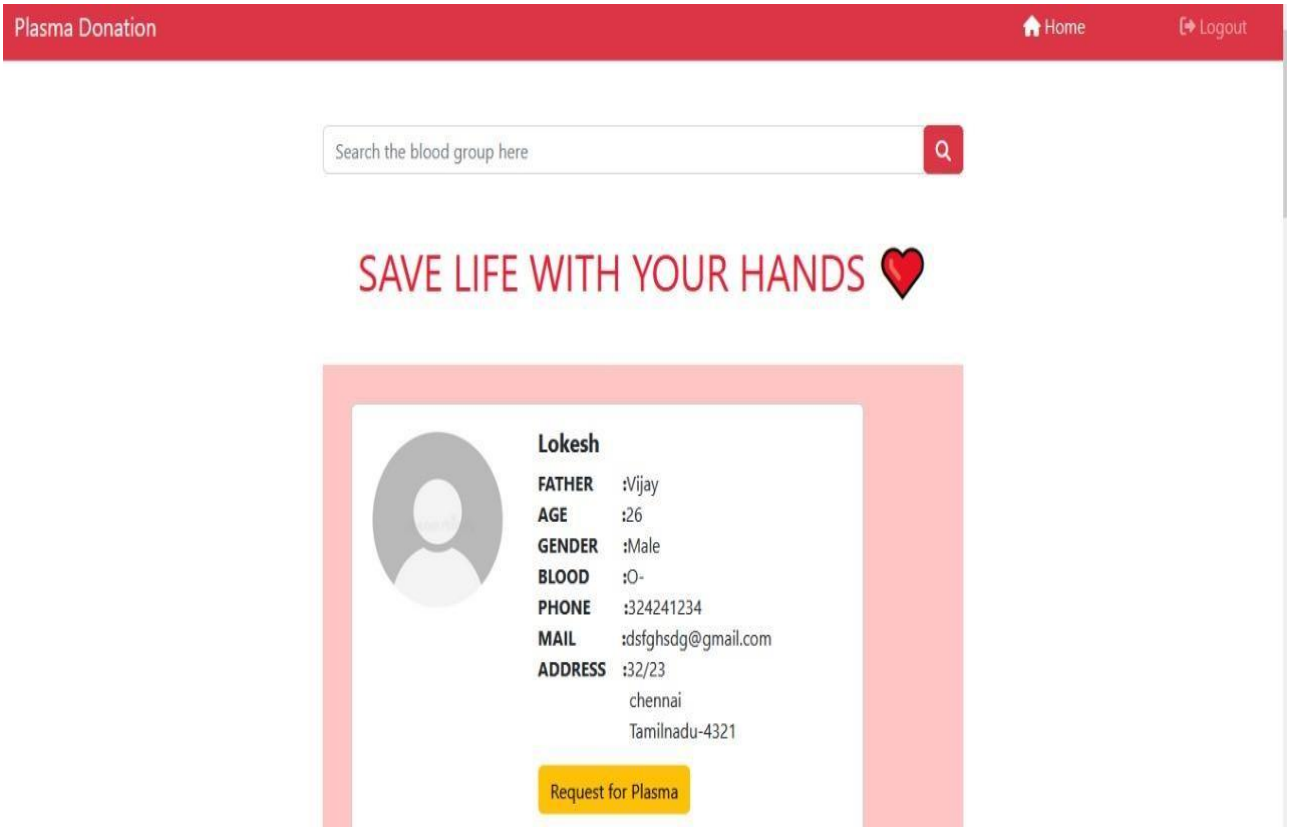


FIG. SEARCH & REQUEST PLASMA FOR DONOR PAGE

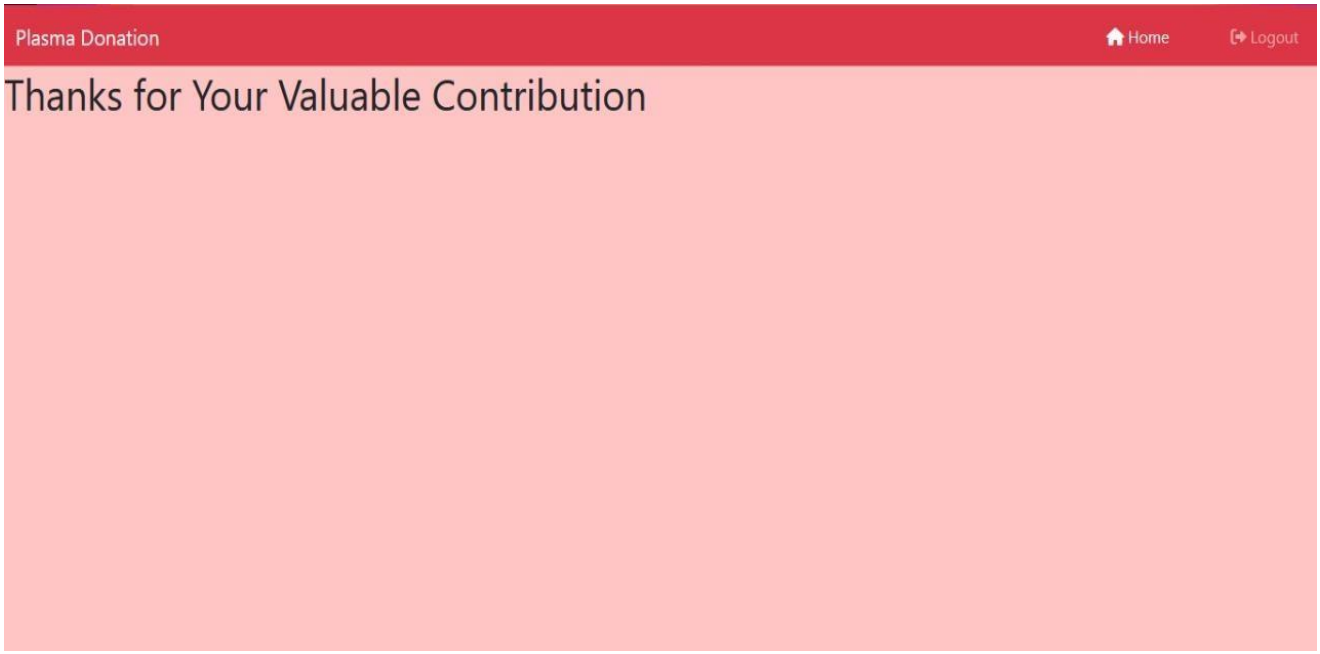
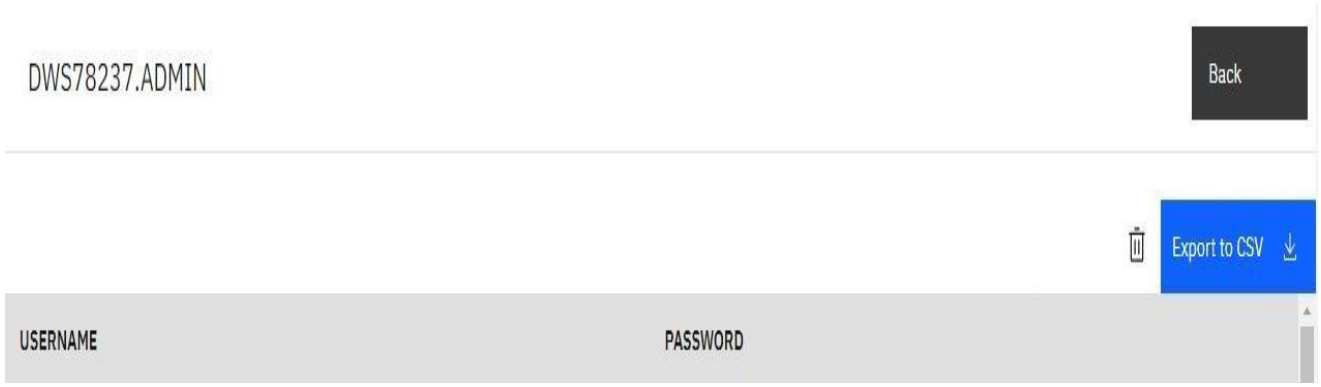
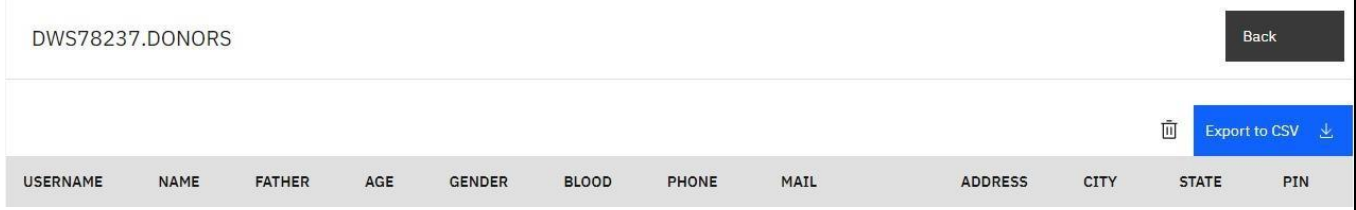


FIG .FEEDBACK PAGE

FIG.MESSAGE DELIVERABLE PAGE

ADMIN SIDE ACTIVITIES





DWS78237.DETAILS

Back

Export to CSV

USERNAME	NAME	FATHER	AGE	GENDER	BLOOD	PHONE	MAIL	ADDRESS	CITY	STATE	PIN
Hardik	H.Pandya	Krunal	25	Male	B+	325424345	asdfghjkl@mail.com	11/111	chennai	Tamilnadu	123456

CHAPTER 10

ADVANTAGES

- User friendliness provided in the application with the various controls.
- The system makes the overall project management much easier and flexible.
- Readily upload the latest updates ,allows user to download the alertsby clicking the url.
- It provides high level of security with different level of authentication.

DISADVANDAGES

- Cannot upload and download the latest updates
- .Mostly the details of donations and donors were managed and maintained manualy.
- No use of Web Service and Remoting.That lead to risk in mismanagement and of data when the project is under development .
- Moreover it is less Secure .There is no proper co-ordination between different applications and users.
- It is fewer user friendly.There is less connection between the plasma authority and donors .

CHAPTER 11

CONCLUSION

It has been a great pleasure to work on this exciting and challenging project. This project proved good for us,as it provided practical knowledge of not only programming in web development,python and cloud From this project, we are able to manage and get details about the palsma donors.While making this project, wegained a lot of experience of working as a team. We discovered Plasma Donor Application

IBM-Project-31855-1660205660 various

predicted andunpredicted problems and we enjoyed alot solving them as a team. We adopted things like video tutorials, text tutorials, internet and learning materials to make

our project complete.

CHAPTER12

FUTURE SCOPE

The project assists well to get details about the plasma donors and individuals can make volunteer themselves by providing their details un our app

However, this project has some limitations:

The application is unable to maintain the backup of data once it

Is uninstalled.

Plasma Donor Application

IBM-Project-31855-1660205660

This application does not provide higher decision capability.

To further enhance the capability of this application, we recommend the following.

- Multiple language interface.
- Provide backup and recovery of data.
- Provide better user interface for user.
- Mobile apps advantage.

CHAPTER 13

APPENDIX

Source Code Github Link :

Plasma Donor Application
IBM-Project-31855-1660205660

https://github.com/IBM-EPBL/
IBM-Project-31855-1660205660

Project Demo Link :

https://drive.google.com/file/d/1uib-HzjUPB6GbCbDygRkx-NpjUW0FARq/view?usp=drivesdk