



PLASMA DONOR APPLICATION

IBM-Project-31592-1660203275

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

A PROJECT REPORT

**PRIYA M (950819104037)
GLARA PUSHPAM (95081910416)
VENKATESHWARI M (950819104049)
SUBBIAH@SURESH (950819104305)**

**BACHELOR OF ENGINEERING IN COMPUTER SCIENCE
AND ENGINEERING**

Government College of Engineering

TIRUNELVELI– 627007

BONAFIDE CERTIFICATE

Certified that this project report “**PLASMA DONOR APPLICATION**”
the bonafide work of **PRIYA M(950819104037)**, **GLARA PUSHAM G**
(950819104016), **VENKATESHWARI M(950819104049)**
SUBBIAH@SURESH (950819104708)” who carried out the project
work under my supervision.

TABLE OF CONTENTS

CHAPTER	TITLE
1	INTRODUCTION 1.1 PROJECT OVERVIEW 1.2 PURPOSE
2	LITERATURE SURVEY 2.1 EXISTING PROBLEM 2.2 REFERENCES 2.3 PROBLEM STATEMENT DEFINITION
3	IDEATION & PROPOSED SOLUTION 3.1 EMPATHY MAP CANVAS 3.2 IDEATION & BRAINSTORMING 3.3 PROPOSED SOLUTION 3.4 PROBLEM SOLUTION FIT
4	REQUIREMENT ANALYSIS 4.1 FUNCTIONAL REQUIREMENT

4.2NON-FUNCTIONAL
REQUIREMENTS

5 PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS

5.2 SOLUTION & TECHNICAL
ARCHITECTURE

5.3 USER STORIES

**6 PROJECT PLANNING &
SCHEDULING**

6.1 SPRINT PLANNING &
ESTIMATION

6.2 SPRINT DELIVERY SCHEDULE

6.3REPORT FROM JIRA

7 CODING &SOLUTIONING

7.1 FEATURE -1

7.2FEATURE -2

7.3DATABASE SCHEMA

(if applicable)

8 TESTING

8.1 TEST CASES

8.2 USER ACCEPTANCE TESTING

9 RESULTS

9.1PERFORMANCE METRICES

10 **ADVANTAGES &
DISADVANTAGES**

11 **CONCLUSION**

12 **FUTURE SCOPE**

13 **APPENDIX**
Source Code GitHub &
Project Demo Link

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

The Plasma Donation Application is to create an e-Information about the donor and organization that are related to donating the plasma. Through this application any person who is interested in donating the blood can register himself in the same way and if any organization wants to register itself with this application that can also register. Moreover if any general consumer wants to make request plasma online he/she can also take the help of this application. Admin is the main authority who can do any modification if required.

1.2 PURPOSE

This project is mainly towards persons who are willing to donate plasma to the patients. Through this app it will be easier to find a donor for exact plasma and easy to build the connection between donor and plasma bank authorities. The main intend of building this software is to formal the procedure of plasma donation and motivate donors in order to donate plasma. We have tried to maintain all information of donor which is easily understandable to the doctors which makes them easy to find the donor.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING PROBLEM

In the existing application,

The details of donations and donors were managed and maintained manually. 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. There is less connection between the plasma authority and donors .

2.2 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/>
- 4.) Cloud-service:- <https://docs.cloud.com/manual/tutorial/getting-started/>
- 5.) Github:- <https://gist.github.com/hofmannsven/6814451>
- 6.) nevonprojects.com/instant-plasma-donor-recipient-connector-app

2.3 PROBLEM STATEMENT DEFINITION

We saw the world suffering from the COVID 19 crisis. There is a 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. 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. And they can request a blood bank and obtain plasma from the blood bank.

CHAPTER 3

IDEATION AND PROPOSED SOLUTION

3.1 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.



3.2 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.

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

⌚ 20 minutes

THINGS TO KNOW WHILE DONATING PLASMA



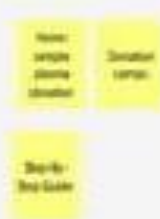
TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas or themes within your mind.

PROCESS



METHODS



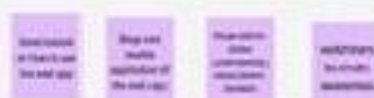
BACKEND/ADMIN WORKS



SUPPORT



AWARENESS USING DIFFERENT FORUMS



STATISTICS AND UPDATES

3.3 PROPOSED SOLUTION

This method helps the users to check the availability of donors. The user and the donor both register all relevant information. A donor has to register on the website by providing their details. The registered users can get information about the donor count of each blood group. Here donor or Recipient no need to pay any money for registering or plasma donation. This application Shows plasma related Doubts and benefits in the description Section. This system can be used by any User who wants to donate or find a donor for Plasma. This could be used in Hospitals, Labs, and Health Clinics.

This application has the potential to create a deep positive impact on society regarding plasma donation or blood donation in general. This will encourage people to donate blood or plasma. Connecting with blood banks to the regular customer through the app also makes sure of transparency. Moreover, the chances of wastage of blood or plasma in blood banks is reduced.

3.4 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?



CHAPTER 4

REQUIREMENT ANALYSIS

4.1FUNCTIONAL REQUIREMNT

The functional requirements of Plasma donor application are:

<u>S.No</u>	<u>Functional requirement</u>	<u>Sub Requirement(Story)</u>
FR-1	User Registration	After downloading the app,the user has to sign up (i.e) register through a new account or gmail or linkedin and so on.
FR-2	User Confirmation	Once the User signs up,confirmation message will be sent via OTP or gmail.
FR-3	Statistics about plasma	The availability of plasma/blood and their complete detail is given.
FR-4	Donor Camps	Information about plasma donor camps conducted across the areas.
FR-5	User Request	If any User is in need of plasma,they have to enter the requirements about the plasma.If the needed type is available,they get the return info.
FR-6	Receiver- donor communication	The donor info will be available in the app with the respective donor's consent.The user and the donor can contact if needed.
FR-7	Chatbots	Chatbots are like virtual assistants and users can communicate with them in case of help or queries.

4.2NON-FUNCTIONAL

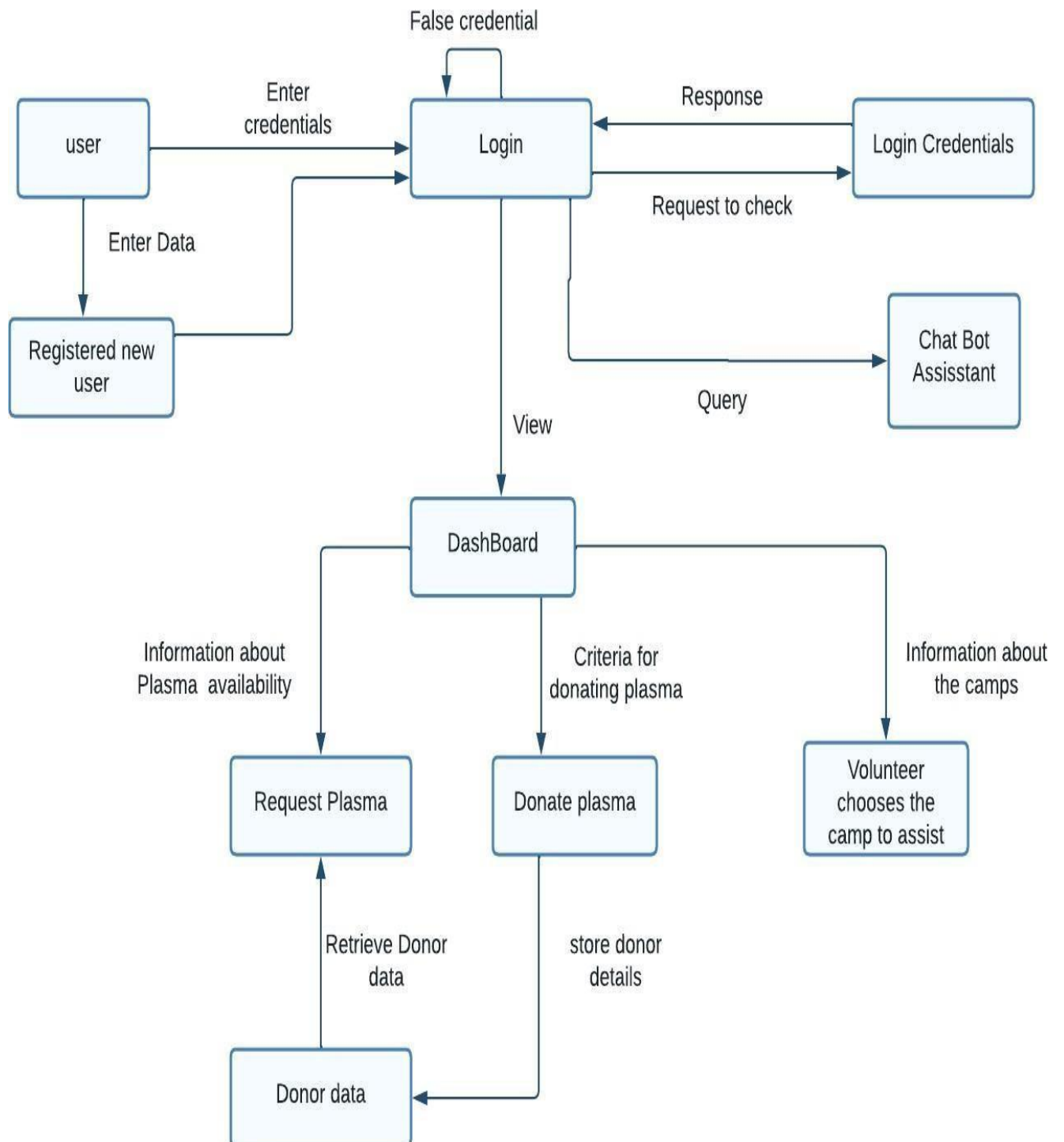
REQUIREMENTS

<u>S.No</u>	<u>Non-functional requirements</u>	<u>Description</u>
NFR-1	Usability	The user interface must be well designed and should be user-friendly.
NFR-2	Security	Every user must have their own password to secure their accounts.A database has to be maintained to store all user info.It shouldcontain different tables which store receiverinfo,donor info,camps,blood bank info etc..
NFR-3	Reliability	The system must be reliable in every means.Every user's information will be maintained in a secure manner.The donor info won't be published in the app without their consent.
NFR-4	Performance	Performance of the system should satisfy the user.Every info published in the websiteshould be verified.The system should respond to every user anytime apart from system crash.
NFR-5	Availability	The system should be available to users anytime. (i.e) Both online and offline components of the system should be available 24x7.

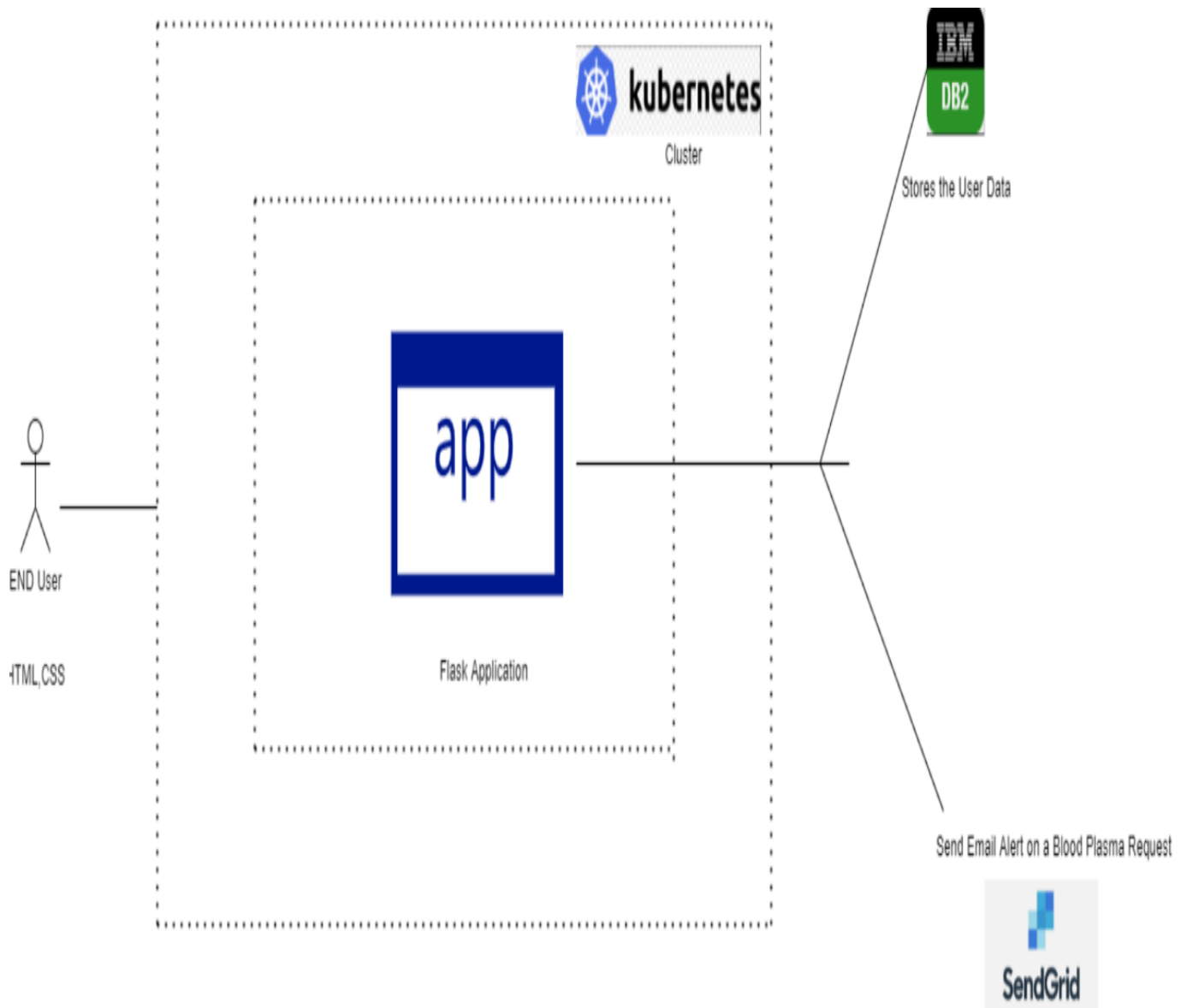
CHAPTER 5

PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS



5.2 SOLUTION AND TECHNICAL ARCHITECTURE



5.3 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account /dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through 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 in dashboard	High	Sprint-1
	Dashboard	USN-5	As a user, I can send the proper requests to donate 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 to view the profile	I can access into my User profile and view details in dashboard	High	Sprint-1
	Dashboard	USN-7	As a user, I can send the proper requests to donate 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 with the 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 to 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

6.1 SPRINT PLANNING AND ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Simulation creation	USN-1	Connect with pythoncode	2	High	Priya M Venkateshwari M Glara Pushpam G Subbiah@Suresh S
Sprint-2	Software	USN-2	Creating an IBM WatsoninCloud platform	2	High	Priya M Venkateshwari M Glara Pushpam G Subbiah@Suresh S
Sprint-3	MIT App Inventor	USN-3	Develop an Plasmadonor application	2	High	Priya M Venkateshwari M Glara Pushpam G Subbiah@Suresh S
Sprint-4	Dashboard	USN-4	Design the Modules andtestthe app	2	High	Priya M Venkateshwari M Glara Pushpam G Subbiah@Suresh S

6.2SPRINT DELIVERY SCHEDULE

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

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")
```

```

@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

8.1 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

8.2 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 volunteersdetails 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

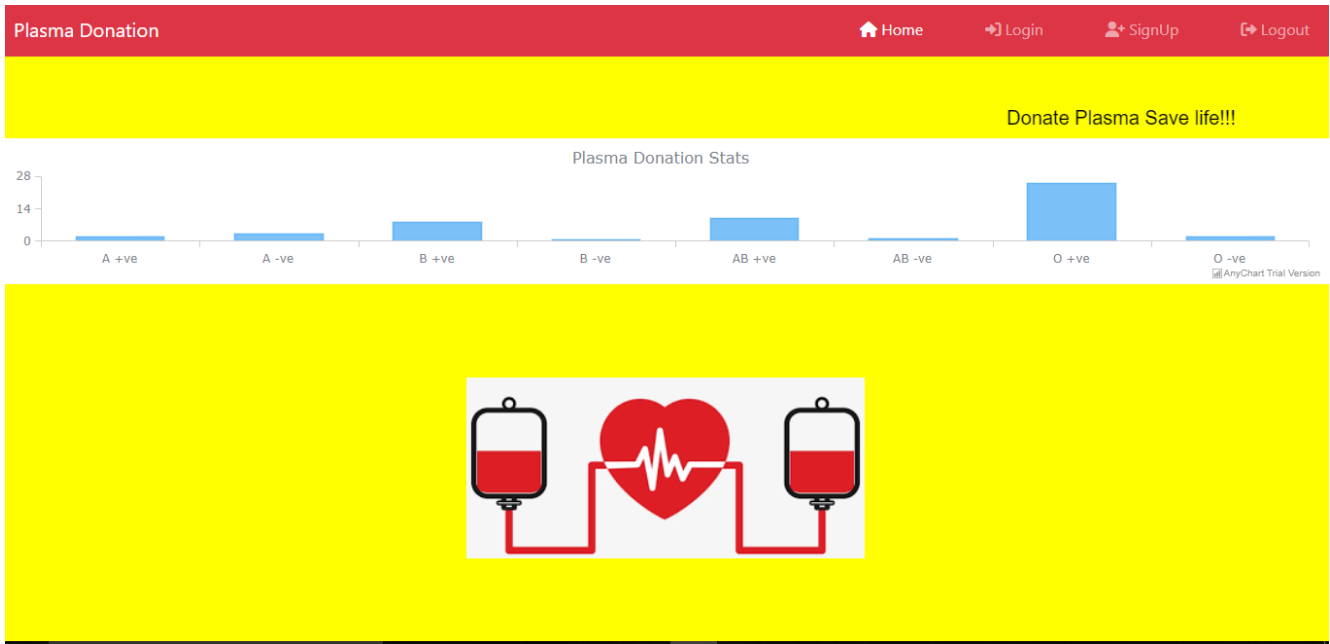


FIG.HOME PAGE

Plasma Donation

Home

SIGN UP

USERNAME:

PASSWORD:


Create account


FIG.SIGIN PAGE

FIG.LOGIN PAGE

Plasma Donation

HomeLogout



 **Logo**

Sign into your account

Userid

Password

LOGIN

[Forgot password?](#)

Don't have an account? Register here

[Terms of use](#), [Privacy policy](#)

Plasma Donation Home Logout

DONAR REGISTRATION FORM

Username	Name
<input type="text" value="usqr4"/>	<input type="text" value="Vicky"/>
Father's name	Age
<input type="text" value="Vinoth"/>	<input type="text" value="34"/>

Gender: ☒ Male ☐ Female ☐ Other

Blood Group: ☐ A+ ☐ A- ☐ A1+ ☒ A1- ☐ B+ ☐ B- ☐ AB+ ☐ AB- ☐ O+ ☐ O-

Phone number

Email ID

Address

City

State

Pincode

FIG.ACCOUNT CREATION

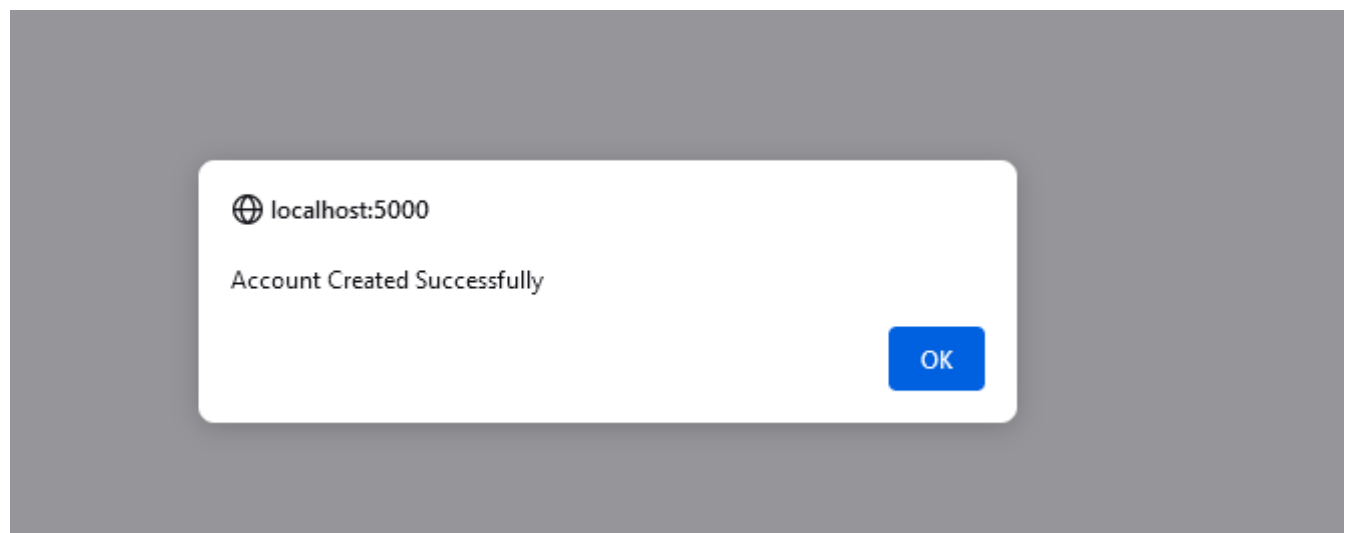


FIG.ACCOUNT CREATION MESSAGE



SAVE LIFE WITH YOUR HANDS 



Lokesh

FATHER :Vijay

AGE :26

GENDER :Male

BLOOD :O-

PHONE :324241234

MAIL :dsfghsdg@gmail.com

ADDRESS :32/23
chennai
Tamilnadu-4321

[Request for Plasma](#)

FIG. SEARCH & REQUEST PLASMA FOR DONOR PAGE

**Your Requested is noted ... soon we will contact
contact:GVP**

mobile number :9876528921

FIG .FEEDBACK PAGE



FIG.MESSAGE DELIVERABLE PAGE

ADMIN SIDE ACTIVITIES

DWS78237.ADMIN		Back
		<div><div></div>Export to CSV</div>
USERNAME	PASSWORD	

FIG.SIGNUP DB

DWS78237.DONORS											Back
											<div><div></div>Export to CSV</div>
USERNAME	NAME	FATHER	AGE	GENDER	BLOOD	PHONE	MAIL	ADDRESS	CITY	STATE	PIN

FIG.DONOR DB

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 .
- It provides high level of security with different level of authentication.

DISADVANTAGES

- Cannot upload and download the latest updates
- .Mostly the details of donations and donors were managed and maintained manually.
- 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.
- There is less connection between the plasmaauthority 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 plasma donors. While making this project, we gained a lot of experience of working as a team. We discovered Plasma Donor Application

IBM-Project-31592-1660203275 various predicted and unpredicted problems and we enjoyed a lot 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 in our application

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-31592-1660203275

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-31592-1660203275

<https://github.com/IBM-EPBL/IBM-Project-31592-1660203275>

Project Demo Link :

<https://vimeo.com/771580892/a5eed15821>