**PLASMA DONOR APPLICATION**

**PROJECT REPORT**

*Submitted by*

HARIPRASAD S - 610819104012

GOWTHAM S - 610819104011

CHRIST N - 610819104006

SUNDARAM S - 610819104053

*In partial fulfilment of the award of the degree*

*of*

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

Er. PERUMAL MANIMEKALAI COLLEGE OF ENGINEERING

KONERIPALLI, HOSUR-635117

ANNA UNIVERSITY: CHENNAI 600025

NOVEMBER 2022

**ANNA UNIVERSITY: CHENNAI 600 025**

**BONAFIDE CERTIFICATE**

Certified that this mini project report “**PLASMA DONOR APPLICATION ”** is the bonafide work of **“HARIPRASAD S (610819104012), GOWTHAM S (610819104011), CHRIST N (610819104006) SUNDARAM S (610819104053)”** who carried out the project under my supervision.

**SIGNATURE SIGNATURE**

**Dr. B. MADHUSUDHANAN , M.E., Ph.D, Dr. SINDHUJA. M.E.,**

**HEAD OF THE DEPARTMENT SUPERVISOR**

**PROFESSOR, PROFESSOR,**

Department of CSE, Department of CSE

Er.Perumal Manimekalai College of Er.Perumal Manimekalai College of Engineering

Engineering , Koneripalli, koneripalli ,

Hosur – 635 117. Hosur – 635 117.

Submitted for the IBM project at Examination held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ at

Er.Perumal Manimekalai College of Engineering , Koneripalli , Hosur 635 117.

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**ACKNOWLEDGEMENT**

It is one of the most efficient tasks in life to choose the appropriate words to express one’s gratitude to the beneficiaries. We are very much grateful to God who helped us all the way through the project and how moulded us into what we are today.

We are grateful to our beloved Principal **Dr.S.CHITRA, M.E, Ph.D** College of , at Perumal Manimekalai College of Engineering ,Hosur for providing the opportunity to do this work in premises.

We acknowledge our heartful gratitude to **Dr B.Madhusudhanan M.E, Ph.D** Professor and Head of the Department, Department of Computer Science and Engineering, at Perumal Manimekalai College of Engineering ,Hosur, for her guidance and valuable suggestions and encouragement throughout this project.

We are highly indebted to **Mrs. S Sindhuja, M.E,**Professor, Department of Computer Science and Engineering, at Perumal Manimekalai College of Engineering ,Hosur, whose immense support, encouragement and valuable guidance made us to complete this project successfully.

We also extent our thanks to Project Coordinator and all Staff Members for their support in complete this project successfully.

Finally, we would like to thank to our parents, without their motivational and support would not have been possible for us to complete this project successfully.

**PLASMA DONOR APPLICATION**

**CONTENTS**

**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**

**4.2 Non-Functional requirements**

**5. PROJECT DESIGN**

**5.1 Data Flow Diagrams**

**5.2 Technical & Solution Architecture**

**5.3 User Stories**

**6. PROJECT PLANNING & SCHEDULING**

**6.1 Sprint Planning & Estimation**

**6.2 Sprint Delivery Schedule**

**6.3 Burn down chart**

**7. CODING & SOLUTIONING**

**7.1 Source code**

**7.2 Python code**

**8. TESTING**

**8.1 Test Cases**

**8.2 User Acceptance Testing**

**9. RESULTS**

**9.1 Performance Metrics**

**10. ADVANTAGES & DISADVANTAGES**

**11. CONCLUSION**

**12. FUTURE SCOPE**

**13. APPENDIX**

**Source Code**

**GitHub & Project Demo Link**

**CHAPTER 1**

**INTRODUCTION**

**Project Overview:**

In online **PLASMA DONOR APPLICATION** system is great project. this project is designed successful completion project on online plasma donation management system. The basic building aim is to provide plasma donation service to the villages, cities recently. Online plasma donation management system (PD) is a browser-based system that is designed to store process, retrieve and analyse information concerned with the administrative and inventory management within a plasma donation. This project aims at maintaining all the information pertaining to the plasma donors, different blood groups are available in each blood bank and help them manage in better way. Aim to provide transparency in this field, make the process obtaining a blood from a blood bank hassle free and corruption free and make the system of blood bank management effective.

The plasma donor system project report contains information related to blood

* + - Blood type
    - Date of Donation of blood
    - Validity of Bloods
    - Available Blood Group

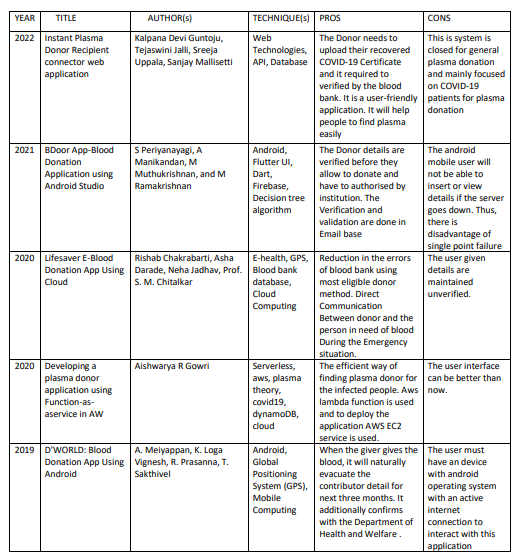
**PURPOSE :**

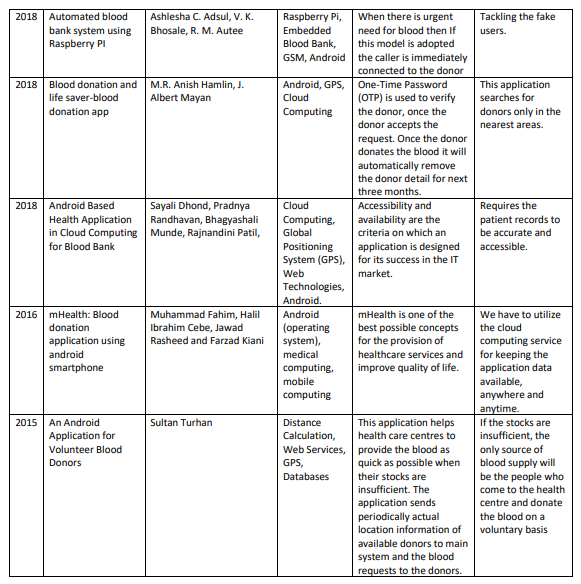
Blood plasma donations are used for slightly more specific purposes than a general blood donation. The most common uses of plasma donations include **individuals who have experienced a severe trauma, burn or shock, adults or children with cancer, and people with liver or clotting factor disorders**.

**CHAPTER 2**

**2. LITERATURE SURVEY**

**2.1 Existing problem:**





**2.2 Reference**

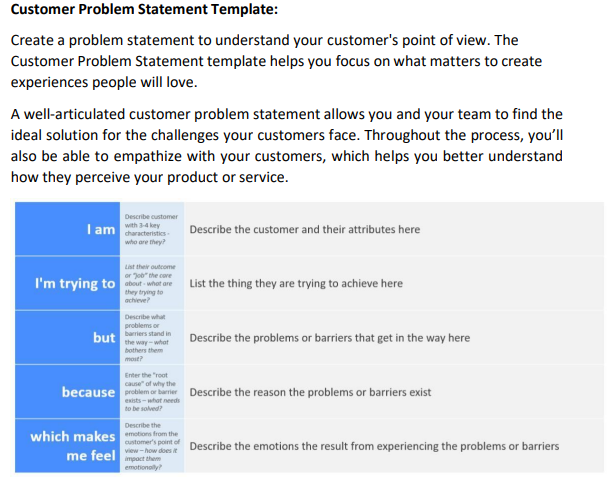
<https://www.researchgate.net/publication/273067813_Free_Blood_Donation_Mobile_Applications>

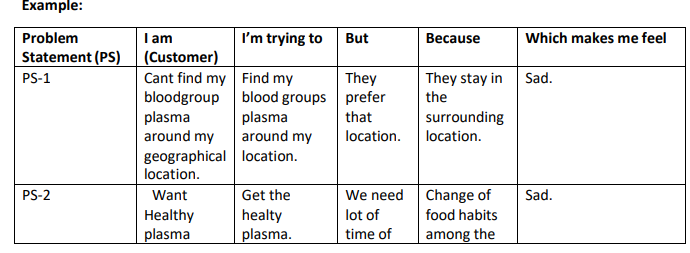
Android application is available on all smart phones but internet is not available and signal is level is low but in our workspace a web application is required for plasma donor application

<https://nevonprojects.com/instant-plasma-donor-recipient-connector-android-app/>

Android application is available on all smart phones but internet is not available and signal is level is low but in our workspace a web application is required for plasma donor application

**2.3 PROBLEM Statement :**

****

****

**CHAPTER 3**

**IDEATION & PROPOSED SOLUTION**

**3.1 Empathy Map Canvas:**

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 helps teams better understand their users.

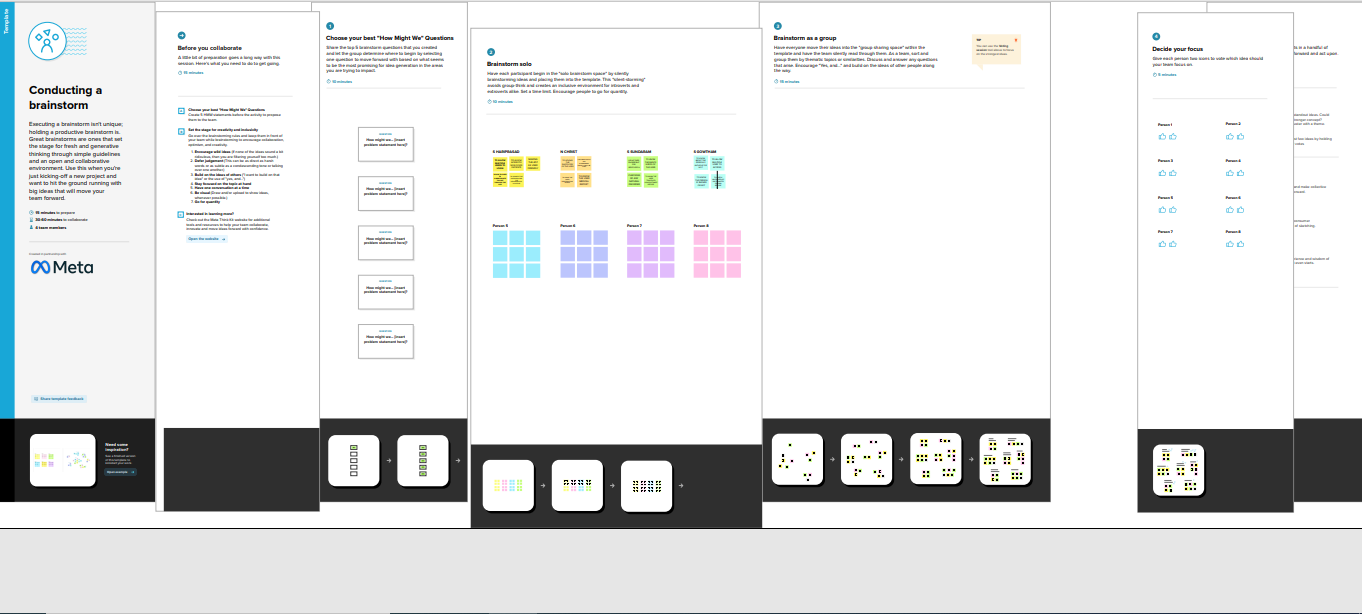
Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user’s perspective along with his or her goals and challenge

****

**3.2 Ideation & 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 number of creative solutions.

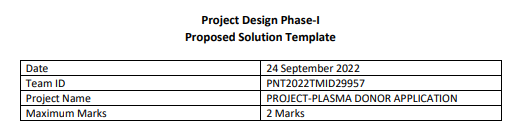
Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

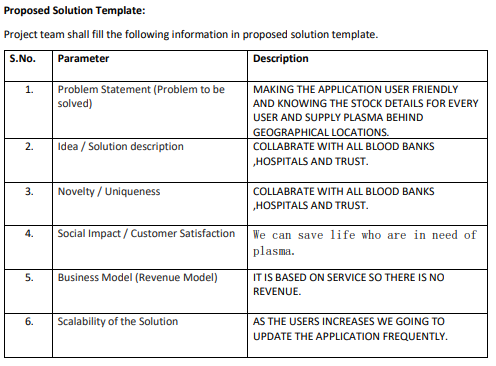
****

**3.3 Proposed Solution:**

**Proposed Solution Template:**

Project team shall fill the following information in proposed solution template

****

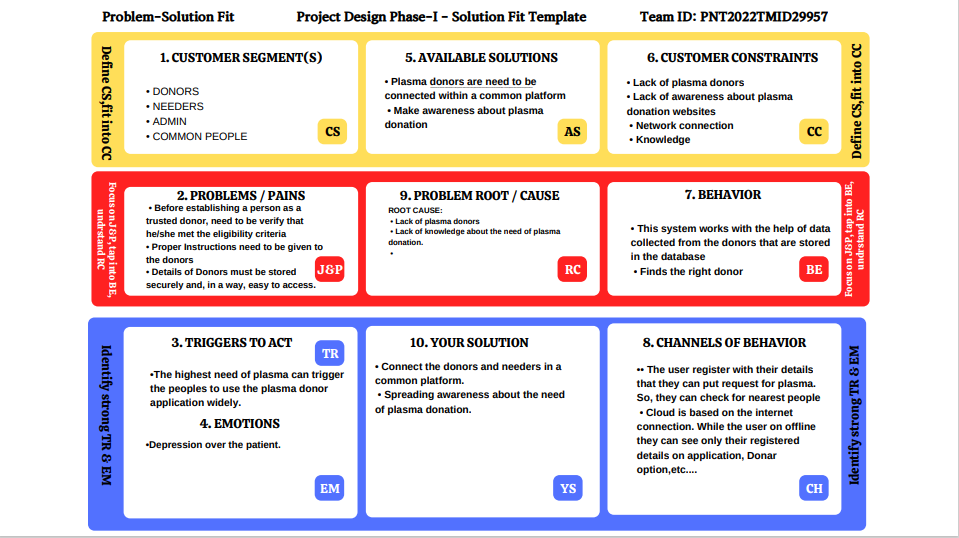
****

**3.4 Problem Solution fit:**

**Problem – Solution Fit Template:** 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. It helps entrepreneurs, marketers and corporate innovators identify behavioral patterns and recognize what would work and why

**Purpose:**

* Solve complex problems in a way that fits the state of your customers.
* Succeed faster and increase your solution adoption by tapping into existing mediums and channels of behaviour.
* Sharpen your communication and marketing strategy with the right triggers and messaging.
* Increase touch-points with your company by finding the right problem behaviour fit and building trust by solving frequent annoyances, or urgent or costly problems.
* Understand the existing situation in order to improve it for your target group.

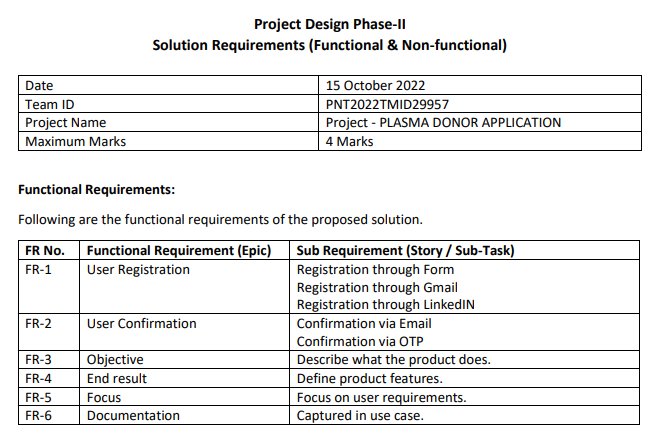


**CHAPTER 4**

**REQUIREMENT ANALYSIS**

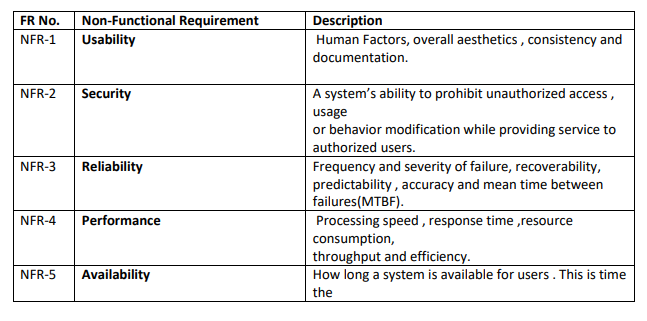
**4.1 Functional requirement:**

Following are the functional requirements of the proposed solution.



**4.2 Non-functional Requirements:**

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

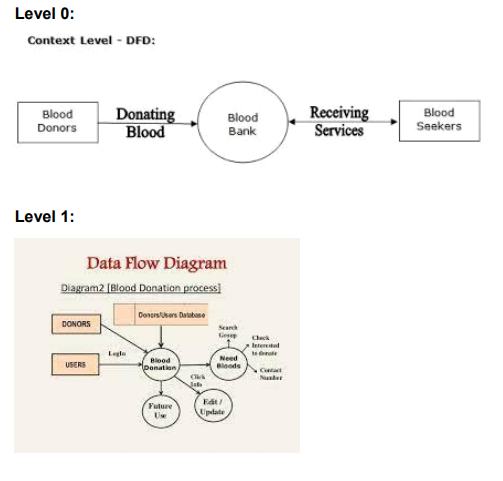


**CHAPTER 5**

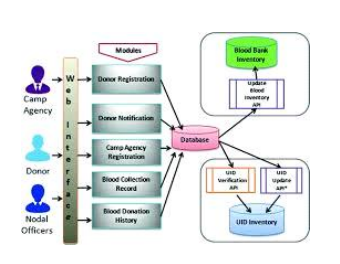
**PROJECT DESIGN**

**5.1 Data Flow Diagrams:**

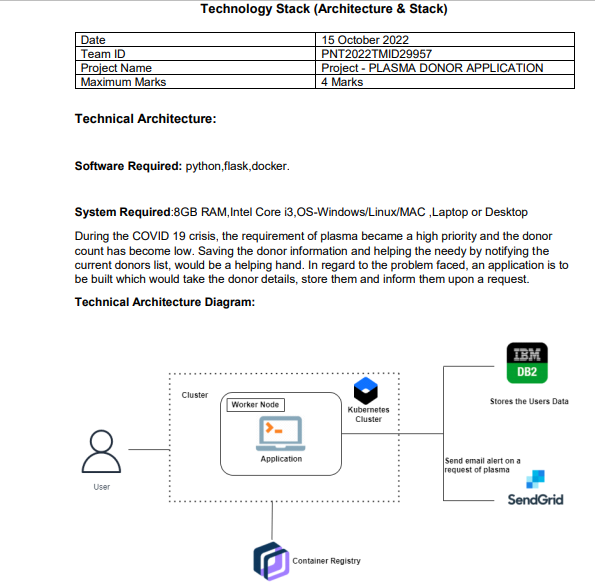
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



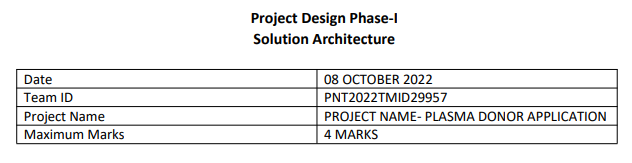
**Level 2**

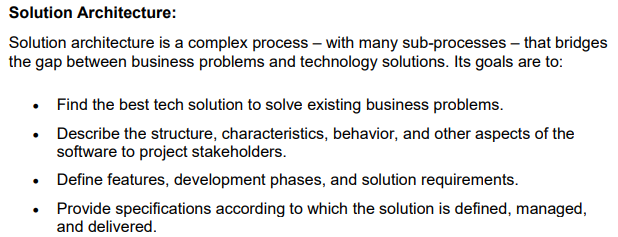


**5.2 Technical Architecture:**

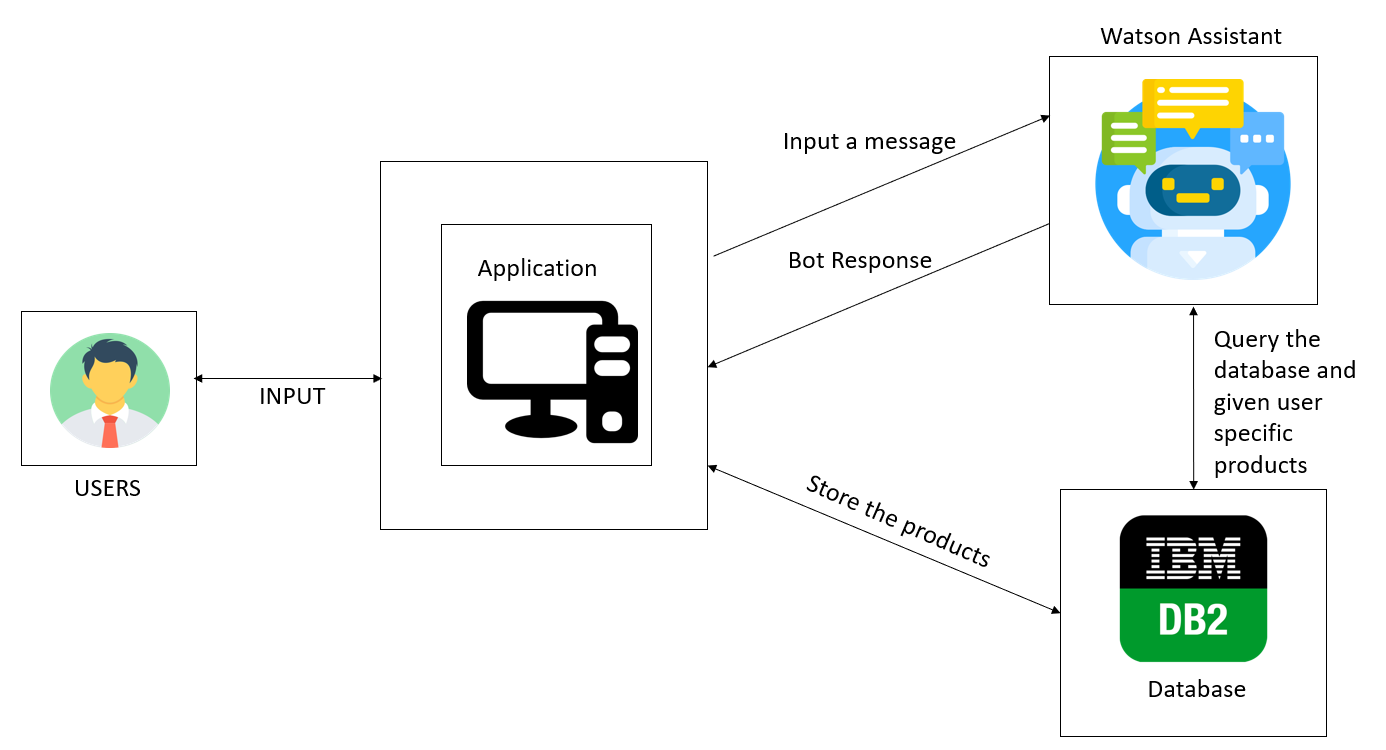
****

**5.3 Solution Architecture:**

****



Example: solution architecture Diagram



**5.4 User Stories:**

Use the below template to list all the user stories for the product.

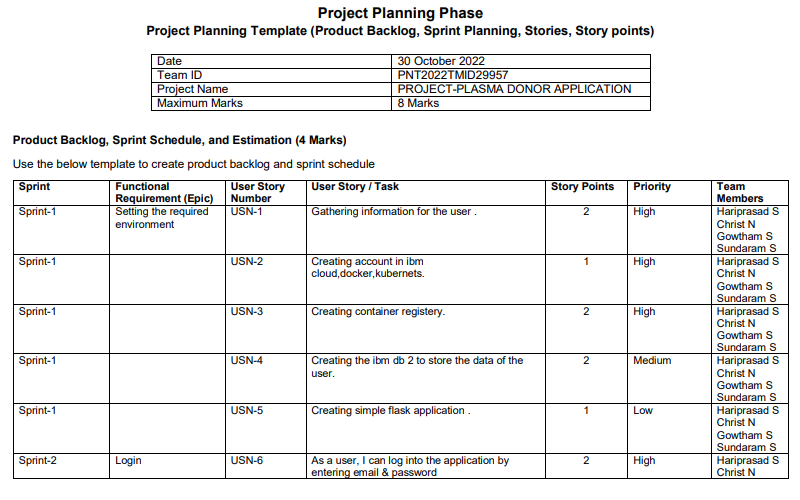
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **User Type** | **Functional**  **Requirement (Epic)** | **User Story**  **Number** | **User Story / Task** | **Acceptance criteria** | **Priority** | **Release** |
| Customer  (Mobile user) | Registration | USN-1 | User can register for the application by  entering my email, password, and confirming  my password | User can access my account /  dashboard | High | Sprint-1 |
|  |  | USN-2 | User will receive confirmation email  once the user have registered for the application | User can receive confirmation  email & click confirm | High | Sprint-1 |
|  |  | USN-3 | User can register for the application through Facebook | User can register & access the  dashboard with Facebook  Login | Low | Sprint-2 |
|  |  | USN-4 | User can register for the application through G mail |  | Medium | Sprint-1 |
|  | Login | USN-5 | User can log into the application by entering email & password | User can access my data by Login | High | Sprint-1 |
|  | Dashboard | USN-6 | User can view the dashboard and by  products |  | High | Sprint-2 |
| Customer (Web  user) | Registration /  Login | USN-7 | User can register for the application by  entering my email, password, and confirming  my password | User can access my account /  dashboard |  | Sprint-1 |
| Customer Care  Executive | Contact with  Customers | USN-8 | Customer care executive solve the Customer Requirements and  feedback | Can receive calls from customers | High | Sprint-1 |
| Administrator | Check stock and  Price,orders | USN-9 | Administrator can check the database and stock details and buying and selling  prices | I am the administrator of  the company | High | Sprint-2 |

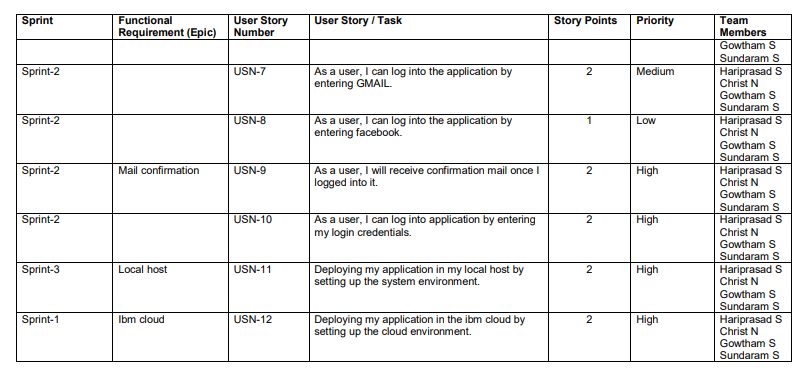
**CHAPTER 6**

**PROJECT PLANNING & SCHEDULING**

**6.1 Sprint Planning & Estimation:**

Use the below template to create product application on and sprint schedule





**6.2 Sprint Delivery Schedule:**

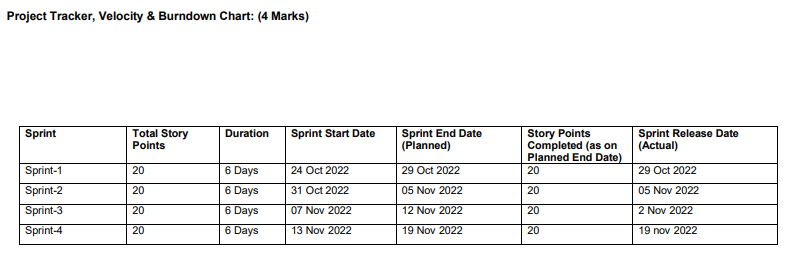
**Project Tracker, Velocity:**

**Velocity:**

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let’s calculate the team’s average velocity (AV) per iteration unit (story points per day)

**Velocity:**

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let’s calculate the team’s average velocity (AV) per iteration unit (story points per day)



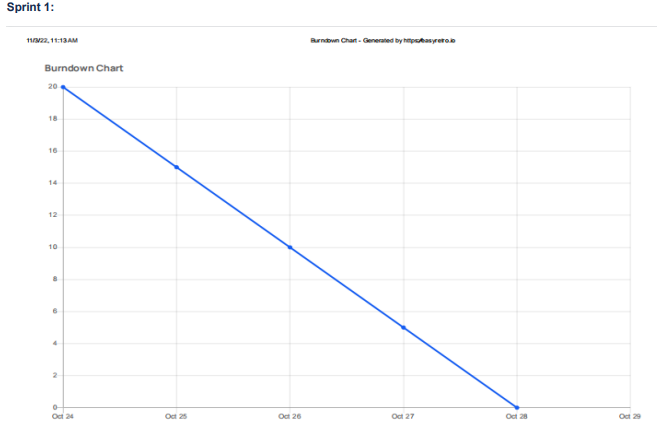
**Velocity:**

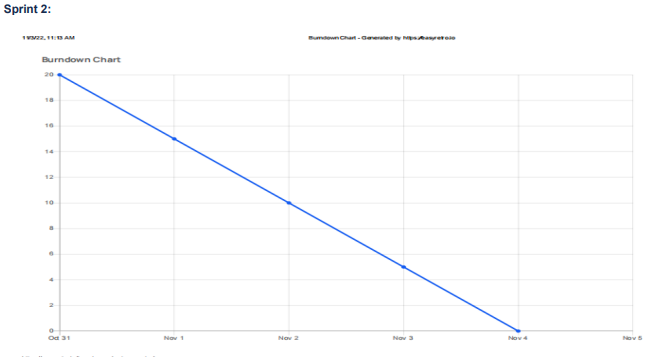
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let’s calculate the team’s average velocity (AV) per iteration unit (story points per day)

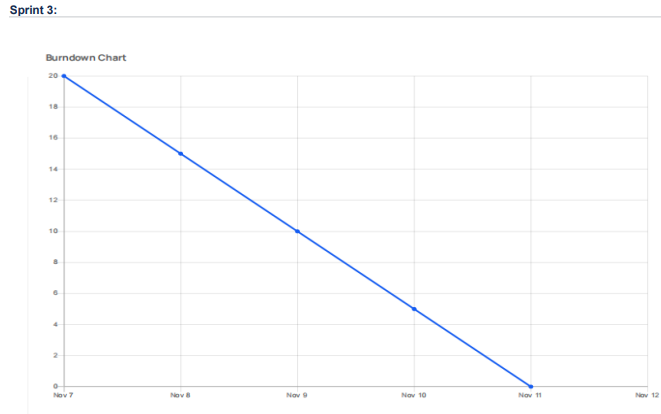


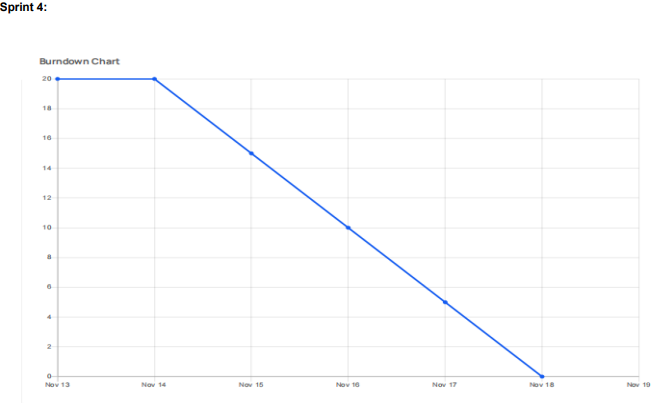
**6.3 Burndown chart :**

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time

****

****

****



**CHAPTER 7**

**CODING & SOLUTIONING**

**7.1 Feature 1(Source code)**

**Register :**

<!DOCTYPE html>

<html >

<!--From https://codepen.io/frytyler/pen/EGdtg-->

<head>

  <meta charset="UTF-8">

  <title>Plasma Donor App</title>

    <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>

    <link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>

    <link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>

    <link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet' type='text/css'>

    <link rel="stylesheet" href="{{ url\_for('static', filename='style1.css') }}">

    <link rel="stylesheet" href="..//static/style.css">

<style>

.login{

top: 20%;

}

</style>

</head>

<body>

<div class="header">

<div>Plasma Donor App</div>

    <ul>

        <li><a href="stats.html">Stats</a></li>

        <li><a class="active" href="login.html">Home</a></li>

    </ul>

</div>

 <div class="login">

     <!-- Main Input For Receiving Query to our ML -->

    <form action="{{ url\_for('register')}}"method="post">

        <input type="text" name="name" placeholder="Enter Your Name" required="required" style="color:black"/>

        <input type="email" name="email" placeholder="Enter Email" required="required" style="color:black"/>

        <input type="text" name="phone" placeholder="Enter 10-digit mobile number" required="required" style="color:black"/>

        <input type="city" name="city" placeholder="Enter Your City Name" required="required" style="color:black"/>

        <select name="infect">

                      <option value="select" selected>Select COVID infection status</option>

                      <option value="infected">Infected</option>

                      <option value="uninfected">Uninfected</option>

        </select>

        <select name="blood">

                      <option value="select" selected>Choose your blood group</option>

                      <option value="O Positive">O Positive</option>

                      <option value="A Positive">A Positive</option>

                      <option value="B Positive">B Positive</option>

                      <option value="AB Positive">AB Positive</option>

                      <option value="O Negative">O Negative</option>

                      <option value="A Negative">A Negative</option>

                      <option value="B Negative">B Negative</option>

                      <option value="AB Negative">AB Negative</option>

        </select>

        <input type="password" name="passw" placeholder="Enter Password" required="required" style="color:black"/>

        <button type="submit" class="btn btn-primary btn-block btn-large">Register</button>

    </form>

 <br><br>

<div style="color:black">

 </div>

 </div>

</body>

</html>

**Login :**

<!DOCTYPE html>

<html >

<head>

  <meta charset="UTF-8">

  <title>Plasma Donor App</title>

    <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>

    <link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>

    <link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>

    <link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet' type='text/css'>

    <link rel="stylesheet" href="{{ url\_for('static', filename='style1.css') }}">

    <link rel="stylesheet" href="..//static/style.css">

<style>

.login{

top: 20%;

}

</style>

</head>

<body>

<div class="header">

<div>Plasma Donor App</div>

    <ul>

        <li><a href="stats.html">Stats</a></li>

        <li><a href="request.html">Request</a></li>

        <li><a href="register.html">Register</a></li>

        <li><a class="active" href="login.html">Home</a></li>

    </ul>

</div>

 <div class="login" >

        <div>

        </div>

     <!-- Main Input For Receiving Query to our ML -->

    <form action="{{ url\_for('loginpage')}}"method="post">

        <input type="text" name="user" placeholder="Enter UserName" required="required" style="color:black" />

        <input type="password" name="passw" placeholder="Enter Password" required="required" style="color:black" />

        <button type="submit" class="btn btn-primary btn-block btn-large">Login</button>

    </form>

<br><br>

<div style="color:black">

 </div>

 </div>

</body>

</html>

**Request:**

<!DOCTYPE html>

<html >

<!--From https://codepen.io/frytyler/pen/EGdtg-->

<head>

  <meta charset="UTF-8">

  <title>Plasma Donor App</title>

    <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>

    <link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>

    <link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>

    <link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet' type='text/css'>

    <link rel="stylesheet" href="{{ url\_for('static', filename='style1.css') }}">

    <link rel="stylesheet" href="..//static/style.css">

<style>

.login{

top: 20%;

}

</style>

</head>

<body>

<div class="header">

<div>Plasma Donor App</div>

    <ul>

        <li><a href="request.html">Request</a></li>

        <li><a href="register.html">Register</a></li>

        <li><a href="stats.html">Stats</a></li>

        <li><a class="active" href="login.html">Home</a></li>

    </ul>

</div>

 <div class="login">

        <div>

        </div>

     <!-- Main Input For Receiving Query to our ML -->

    <form action="{{ url\_for('requested')}}"method="post">

        <select name="bloodgrp">

                      <option value="select" selected>Choose your blood group</option>

                      <option value="O Positive">O Positive</option>

                      <option value="A Positive">A Positive</option>

                      <option value="B Positive">B Positive</option>

                      <option value="AB Positive">AB Positive</option>

                      <option value="O Negative">O Negative</option>

                      <option value="A Negative">A Negative</option>

                      <option value="B Negative">B Negative</option>

                      <option value="AB Negative">AB Negative</option>

        </select>

        <textarea rows="4" placeholder="Enter the address" required="required" style="color:black" name="address"></textarea>

        <!input type="textarea" name="address" rows="4" placeholder="Enter the address" required="required" style="color:black" />

        <button type="submit" class="btn btn-primary btn-block btn-large">Submit the request</button>

    </form>

 <br><br>

<div style="color:black">

 </div>

 </div>

</body>

</html>

**Stats :**

<!DOCTYPE html>

<html lang="en">

<head>

  <title>Plasma Donar App</title>

  <meta charset="utf-8">

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>

  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

  <link rel="stylesheet" href="{{ url\_for('static', filename='style1.css') }}">

  <link rel="stylesheet" href="..//static/style.css">

</head>

<style>

        .big{

        top:70;

        background-color:white;

        margin-top:80px;

        margin-left:550px;

        margin-right:550px;

        height:200px;

        border-radius: 25px;

        border: 3px solid #4a77d4;

        box-shadow: 6px 8px 4px grey;

        text-align:center;

        }

        .row{

        height:150px;

        }

        .col{

            margin:10px;

            margin-left:50px;

            margin-right:50px;

            border-radius: 25px;

            border: 1px solid #4a77d4;

            box-shadow: 0px 8px 4px grey;

            text-align:center;

        }

        .ext{

        margin-top:25px;

        line-height:40px;

        }

        .ext1{

        margin-top:40px;

        line-height:50px;

        font-size:25px;

        color:#f95450;

        }

</style>

<body>

<div class="container-fluid">

<div class="header">

<div><b>Plasma Donar App</b></div>

<ul>

        <li><a href="request.html">Request</a></li>

        <li><a class="active" href="login.html">Home</a></li>

    </ul>

</div>

  <br>

  <div class="big">

    <div class="box">

        <div class="ext1"><font-size="20px">{{b}}</font-size><br><b>Donors</b></div>

    </div>

  </div>

  <br>

  <div class="row">

    <div class="col" >

        <div class="ext">{{b1}}<br><b>O Positive</b></div>

    </div>

    <div class="col" >

        <div class="ext">{{b2}}<br><b>A Positive</b></div>

    </div>

    <div class="col" >

        <div class="ext">{{b3}}<br><b>B Positive</b></div>

    </div>

    <div class="col" >

        <div class="ext">{{b4}}<br><b>AB Positive</b></div>

    </div>

  </div>

  <br>

  <div class="row">

    <div class="col" >

        <div class="ext">{{b5}}<br><b>O Negative</b></div>

    </div>

    <div class="col" >

        <div class="ext">{{b6}}<br><b>A Negative</b></div>

    </div>

    <div class="col" >

        <div class="ext">{{b7}}<br><b>B Negative</b></div>

    </div>

    <div class="col" >

        <div class="ext">{{b8}}<br><b>AB Negative</b></div>

    </div>

  </div>

</div>

</body>

</html>

**7.2 Feature 2 (Python code )**

**app.py:**

from flask import Flask, render\_template, request, redirect, url\_for, session

import ibm\_db

import \_json

app = Flask(\_\_name\_\_)

conn = ibm\_db.connect("DATABASE=bludb;HOSTNAME=125f9f61-9715-46f9-9399-c8177b21803b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=30426;SECURITY=SSL;SSLServerCertificate=Certificate.crt;UID=phz1431;PWD=3gtJ6boYjUjQ1fna",'','')

@app.route('/registration')

def home():

    return render\_template('register.html')

@app.route('/register',methods=['POST'])

def register():

    x = [x for x in request.form.values()]

    print(x)

    name=x[0]

    email=x[1]

    phone=x[2]

    city=x[3]

    infect=x[4]

    blood=x[5]

    password=x[6]

    sql = "SELECT \* FROM user WHERE email =?"

    stmt = ibm\_db.prepare(conn, sql)

    ibm\_db.bind\_param(stmt,1,email)

    ibm\_db.execute(stmt)

    account = ibm\_db.fetch\_assoc(stmt)

    print(account)

    if account:

        return render\_template('register.html', pred="You are already a member, please login using your details")

    else:

        insert\_sql = "INSERT INTO  user VALUES (?, ?, ?, ?, ?, ?, ?)"

        prep\_stmt = ibm\_db.prepare(conn, insert\_sql)

        ibm\_db.bind\_param(prep\_stmt, 1, name)

        ibm\_db.bind\_param(prep\_stmt, 2, email)

        ibm\_db.bind\_param(prep\_stmt, 3, phone)

        ibm\_db.bind\_param(prep\_stmt, 4, city)

        ibm\_db.bind\_param(prep\_stmt, 5, infect)

        ibm\_db.bind\_param(prep\_stmt, 6, blood)

        ibm\_db.bind\_param(prep\_stmt, 7, password)

        ibm\_db.execute(prep\_stmt)

        return render\_template('register.html', pred="Registration Successful, please login using your details")

@app.route('/')

@app.route('/login')

def login():

    return render\_template('login.html')

@app.route('/loginpage',methods=['POST'])

def loginpage():

    user = request.form['user']

    passw = request.form['passw']

    sql = "SELECT \* FROM user WHERE email =? AND password=?"

    stmt = ibm\_db.prepare(conn, sql)

    ibm\_db.bind\_param(stmt,1,user)

    ibm\_db.bind\_param(stmt,2,passw)

    ibm\_db.execute(stmt)

    account = ibm\_db.fetch\_assoc(stmt)

    print (account)

    print(user,passw)

    if account:

            return redirect(url\_for('stats'))

    else:

        return render\_template('login.html', pred="Login unsuccessful. Incorrect username / password !")

@app.route('/stats')

def stats():

    '''sql = "SELECT blood FROM user group by blood"

    stmt = ibm\_db.prepare(conn, sql)

    ibm\_db.execute(stmt)

    count = ibm\_db.fetch\_assoc(stmt)

    print(count)'''

    return render\_template('stats.html',b=1,b1=0,b2=0,b3=1,b4=0,b5=0,b6=0,b7=0,b8=0)

@app.route('/requester')

def requester():

    return render\_template('request.html')

@app.route('/requested',methods=['POST'])

def requested():

    bloodgrp = request.form['bloodgrp']

    address = request.form['address']

    print(address)

    sql = "SELECT \* FROM user WHERE blood=?"

    stmt = ibm\_db.prepare(conn, sql)

    ibm\_db.bind\_param(stmt,1,bloodgrp)

    ibm\_db.execute(stmt)

    data = ibm\_db.fetch\_assoc(stmt)

    msg = "Need Plasma of your blood group for: "+address

    while data != False:

        print ("The Phone is : ", data["PHONE"])

        url="https://www.fast2sms.com/dev/bulk?authorization=xCXuwWTzyjOD2ARd1EngbH3a7tKIq5PklJ8YSf0Lh4FQZecs9iNI1dSvuqprxFwCKYJXA5amQkBE36Rl&sender\_id=FSTSMS&message="+msg+"&language=english&route=p&numbers="+str(data["PHONE"])

        result=request.request("GET",url)

        print(result)

        data = ibm\_db.fetch\_assoc(stmt)

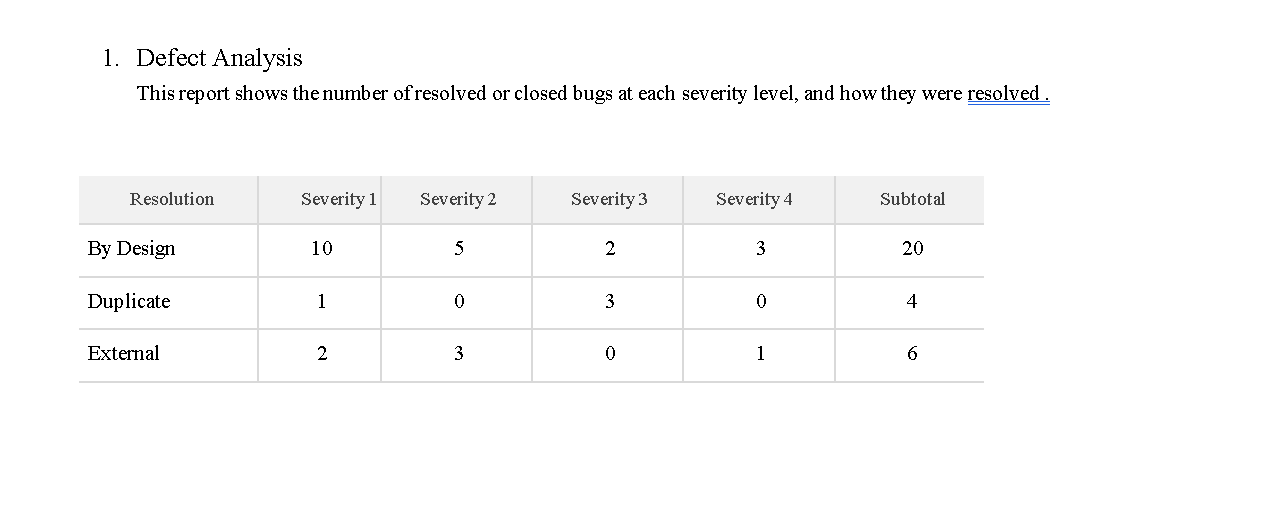
    return render\_template('request.html', pred="Your request is sent to the concerned people.")

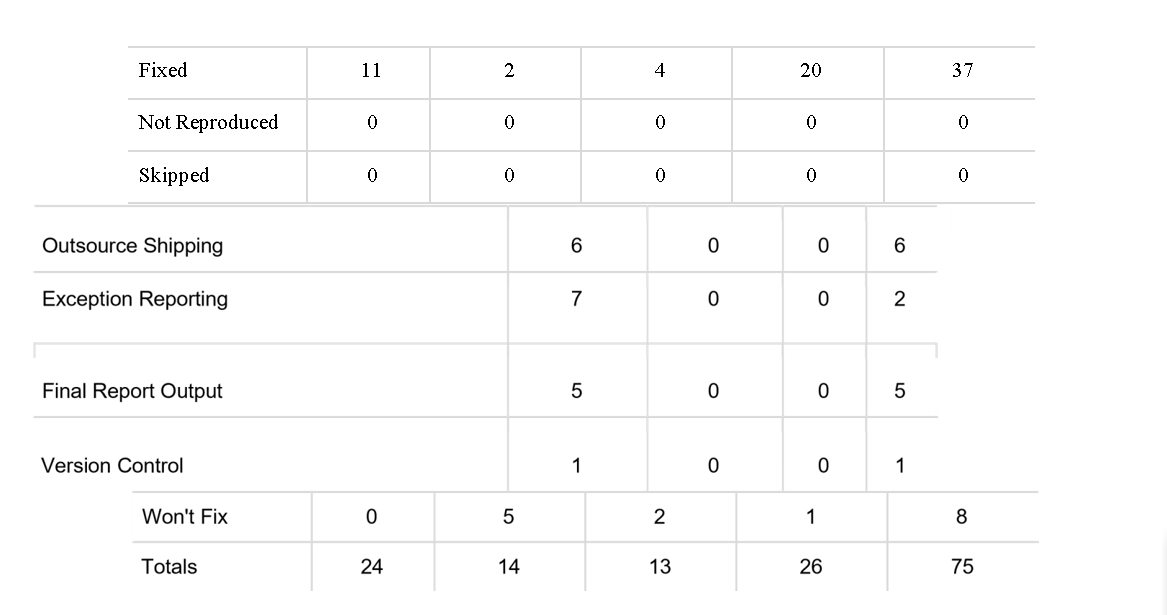
if \_\_name\_\_ == "\_\_main\_\_":

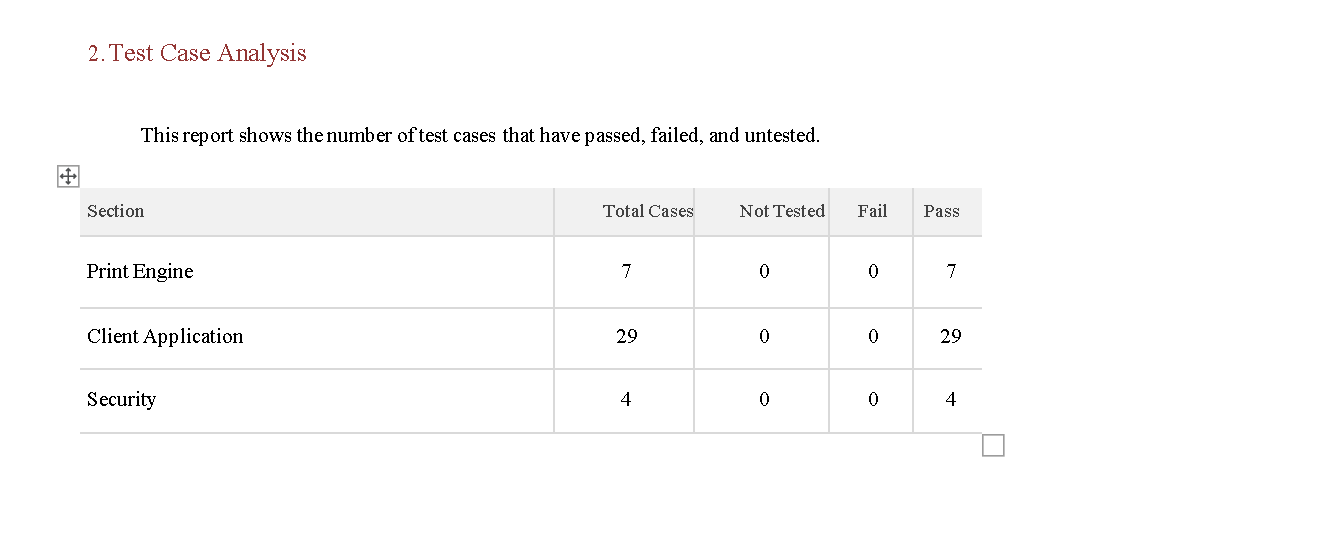
    app.run(host='0.0.0.0', port=8080)

**CHAPTER 8**

**User Acceptance Testing**

****

****

****

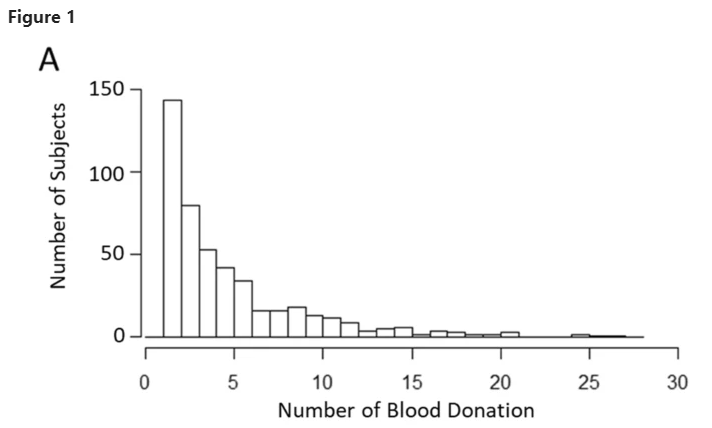
**CHAPTER 9**

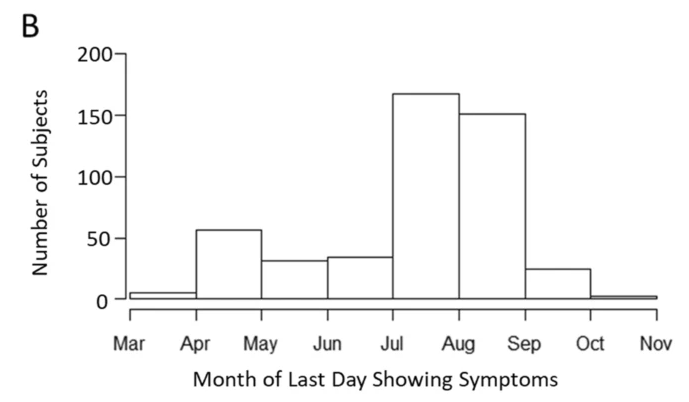
**9.RESULTS**

**9.1Performance Matrix:**

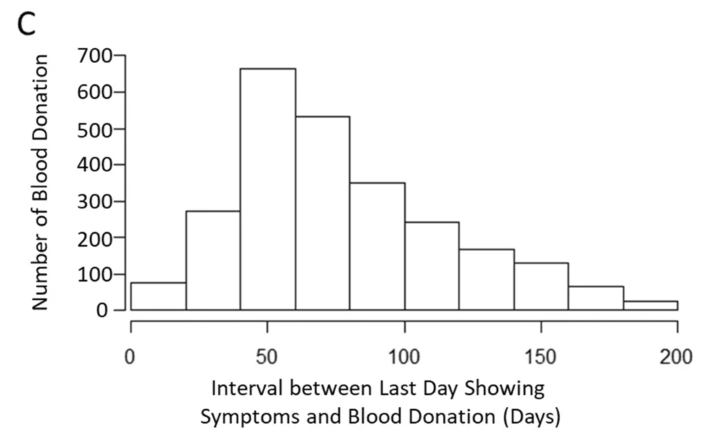
### **Collection of blood samples from COVID-19 patients**

We have analyzed anti-N IgG levels from 943 COVID-19 patients who tested positive for SARS-CoV-2 via RT-PCR testing. Among 472 subjects with multiple donations, most people donated blood less than five times, and the maximum number of times that one person donated blood was 27 times . The majority of patients exhibited their last days of symptoms in July and August . The average time between the last day showing symptoms and the blood donation was 78 ± 38 days, up to the longest period of 200 days, which reflects timing of our blood collection and antibody test





­­­­



Distribution of COVID-19 patients with multiple blood donation. (**A**) Distribution of the number of blood donations from COVID-19 patients. (**B**) Distribution of patients with their last days showing COVID-19 symptoms. (**C**) Distribution of the time interval between the last day showing symptoms and the latest day of blood donation.

**CHAPTER 10**

**10.ADVANTAGES & DISADVANTAGES**

**Advantages :**

* Make an Impact.
* Boost Your Mood.
* Maintain a Healthy Diet.
* Reduce Cholesterol Levels.
* Lower Blood Pressure.
* Prevents Hemochromatosis. Health benefits of blood donation include reduced risk of hemochromatosis.
* Anti-cancer Benefits. Blood donation helps in lowering the risk of cancer. ...
* Maintains Healthy Heart & Liver.
* Weight loss.
* Stimulates Blood Cell Production.
* Earn Up to $4,000 per Year. What attracts many people to plasma donation is the fact that you can earn a substantial amount of money every time you donate.

**Disadvantages :**

* Bruising. When you donate blood, you sit or lie on a reclining chair with your arm extended on an armrest. ...
* Continued bleeding. ...
* Dizziness, lightheadedness, and nausea. ...
* Pain. ...
* Physical weakness. ...
* Time-consuming. ...
* But donating blood can do a lot of good

**CHAPTER 11**

**11.CONCLUSION**

A majority of the donors were willing to be regular donors. The donors showed positive effects like a sense of satisfaction after the donation. Creating an opportunity for blood donation by conducting many blood donation camps may increase the voluntary blood donations.

**KEYWORDS:** Blood donation, Knowledge, Motivation, Voluntary donors

**CHAPTER 12**

**12.FUTURE SCOPE**

The most important practical applications of plasmas lie in the future, **largely in the field of power production**.

* Provide safe and quality blood and blood components collected from voluntary donors, round the clock, at affordable cost to the general public and free of cost to the poor.
* Ensure safety and quality of blood.
* Motivate and maintain a permanent well-indexed record of voluntary blood donors.
* Educating the community on the beneficial aspects of blood donation and harmful effect of collecting blood from paid donors.
* Actively encourage voluntary blood donation and gradually eliminate professional blood donors.
* Promote AIDS awareness and education to the general public.
* Assists the various Organizations, Clubs, Colleges, Public & Private Institutions and the Public to conduct voluntary blood donation drives and arrange for motivational talks to enable progressively increase the number of voluntary non-remunerated blood donors every year.

**CHAPTER 13**

**APPENDIX**

**13.1 Source Code**

from flask import Flask, render\_template, request, redirect, url\_for, session

import ibm\_db

import json

app = Flask(\_\_name\_\_)

conn = ibm\_db.connect("DATABASE=bludb;HOSTNAME=2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=32328;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=rzp60332;PWD=pQKhhRttcrGEyBKA",'','')

@app.route('/registration')

def home():

    return render\_template('register.html')

@app.route('/register',methods=['POST'])

def register():

    x = [x for x in request.form.values()]

    print(x)

    name=x[0]

    email=x[1]

    phone=x[2]

    city=x[3]

    infect=x[4]

    blood=x[5]

    password=x[6]

    sql = "SELECT \* FROM user WHERE email =?"

    stmt = ibm\_db.prepare(conn, sql)

    ibm\_db.bind\_param(stmt,1,email)

    ibm\_db.execute(stmt)

    account = ibm\_db.fetch\_assoc(stmt)

    print(account)

    if account:

        return render\_template('register.html', pred="You are already a member, please login using your details")

    else:

        insert\_sql = "INSERT INTO  user VALUES (?, ?, ?, ?, ?, ?, ?)"

        prep\_stmt = ibm\_db.prepare(conn, insert\_sql)

        ibm\_db.bind\_param(prep\_stmt, 1, name)

        ibm\_db.bind\_param(prep\_stmt, 2, email)

        ibm\_db.bind\_param(prep\_stmt, 3, phone)

        ibm\_db.bind\_param(prep\_stmt, 4, city)

        ibm\_db.bind\_param(prep\_stmt, 5, infect)

        ibm\_db.bind\_param(prep\_stmt, 6, blood)

        ibm\_db.bind\_param(prep\_stmt, 7, password)

        ibm\_db.execute(prep\_stmt)

        return render\_template('register.html', pred="Registration Successful, please login using your details")

@app.route('/')

@app.route('/login')

def login():

    return render\_template('login.html')

@app.route('/loginpage',methods=['POST'])

def loginpage():

    user = request.form['user']

    passw = request.form['passw']

    sql = "SELECT \* FROM user WHERE email =? AND password=?"

    stmt = ibm\_db.prepare(conn, sql)

    ibm\_db.bind\_param(stmt,1,user)

    ibm\_db.bind\_param(stmt,2,passw)

    ibm\_db.execute(stmt)

    account = ibm\_db.fetch\_assoc(stmt)

    print (account)

    print(user,passw)

    if account:

            return redirect(url\_for('stats'))

    else:

        return render\_template('login.html', pred="Login unsuccessful. Incorrect username / password !")

@app.route('/stats')

def stats():

    '''sql = "SELECT blood FROM user group by blood"

    stmt = ibm\_db.prepare(conn, sql)

    ibm\_db.execute(stmt)

    count = ibm\_db.fetch\_assoc(stmt)

    print(count)'''

    return render\_template('stats.html',b=5,b1=2,b2=3,b3=4,b4=2,b5=1,b6=2,b7=1,b8=1)

@app.route('/requester')

def requester():

    return render\_template('request.html')

@app.route('/requested',methods=['POST'])

def requested():

    bloodgrp = request.form['bloodgrp']

    address = request.form['address']

    print(address)

    sql = "SELECT \* FROM user WHERE blood=?"

    stmt = ibm\_db.prepare(conn, sql)

    ibm\_db.bind\_param(stmt,1,bloodgrp)

    ibm\_db.execute(stmt)

    data = ibm\_db.fetch\_assoc(stmt)

    msg = "Need Plasma of your blood group for: "+address

    while data != False:

        print ("The Phone is : ", data["PHONE"])

        url="https://www.fast2sms.com/dev/bulk?authorization=xCXuwWTzyjOD2ARd1EngbH3a7tKIq5PklJ8YSf0Lh4FQZecs9iNI1dSvuqprxFwCKYJXA5amQkBE36Rl&sender\_id=FSTSMS&message="+msg+"&language=english&route=p&numbers="+str(data["PHONE"])

        result=request.request("GET",url)

        print(result)

        data = ibm\_db.fetch\_assoc(stmt)

    return render\_template('request.html', pred="Your request is sent to the concerned people.")

if \_\_name\_\_ == "\_\_main\_\_":

    app.run(host='0.0.0.0', port=8080)

**13.2 Github Link**

<https://github.com/IBM-EPBL/IBM-Project-21349-1659778433>