PLASMA DONOR APPLICATION

## TeamMembers: S.Gokul ,M.Kishore Kumar,M.Hariprakash,G.Arun Prasad

Department of Computer Science Engineering,

University College of Engineering Kancheepuram,

BATCH ID: B2-2M4E TEAM ID: PNT2022TMID40392

**ABSTRACT:** During the COVID 19 crisis, the requirement of plasma became a high priority and the donor count has become low. Saving the donor information and helpingthe needy by notifying the current donors list, would be a helping hand. In regard to the problem faced, an application is to be built which would take the donor details, store them and inform them upon a request. This plasma therapy is considered to be safe & promising. This system proposed here aims at connecting the donors & the patients byan online application. By using this application, the users can either raise a request for plasma donation or requirement. This system is used if anyone needs a Plasma Donor.

# INTRODUCTION

### PROJECT OVERVIEW

Recently concern grows about the plasma donation for COVID-19 during the pandemic situation. This convalescent plasma was used to recover patients who are critically ill as it helps to grow antibodies on their body. Recent researches show that many people are willing to help someone in need through money, blood and plasma donation etc. but they ﬁnd it diﬃcult to identify and approach the needy people who arenot aware of technological innovations, including the use of social media. Plasma is used to various infectious diseases and it is one of the oldest methods known as plasma therapy. Plasma therapy is a Process where blood is donated by recovered patients in order to establish anti bodies that ﬁghts the infection. This system comprises of Admin, user and donor where both can request for Plasma. The proposedmethod helps the users to check the availability of donors. A donor has to register to thewebsite providing their details. The registered users can get the information about the donor count of each blood group. The database will have all the details such as name,

email, phone number, infected status. Whenever a user requests for a particular blood group then the concerned blood group donors will receive the notiﬁcation regarding therequirement. For instance, during COVID 19 crisis the requirement for plasma increaseddrastically as there were no vaccination found in order to treat the infected patients, with plasma therapy the recovery rates where high but the donor count was very low and in such situations it was very important to get the information about the plasma donors. Saving the donor information and notifying about the current donors would be ahelping hand as it can save time and help the users to track down the necessary information about the donors.

### PURPOSE

The main aim of developing this system is to provide blood to the people who are inneed of plasma. The numbers of persons who are in need of plasma are increasing in large number day by day. Using this system user can search blood group available in thecity and he can also get contact number of the donor who has the same blood group he/she needs for plasma. In order to help people who are in need of plasma, this plasma donor application can be used effectively for getting the details of available plasma and user can also get contact number of the plasma donors having the same blood group and within the same city.

# LITERATURE REVIEW

### EXISTING PROBLEM:

* + 1. **TITLE:** Instant Plasma Donor Recipient Connector web application.

**AUTHOR:** Kalpana Devi Guntoju1, Swinish Jalli2

The world is suffering from the COVID 19 crisis and no vaccine has been found yet.. But there is another scientiﬁc way in which we can help reduce mortality or help people affected by COVID19 by donating plasma from recovered patients. In the absence of an approved antiviral treatment plan for a fatal COVID19 infection,

plasma therapy is an experimental approach to treat COVID19-positive patients and help them faster recovery. Therapy is considered competent. In the recommendation system, the donor who wants to donate plasma can donate by uploading their COVID19 certiﬁcate and the blood bank can see the donors who have uploaded the certiﬁcate and they can make a request to the donor and the hospital can register/login and search for the necessary things. plasma from a blood bank and they can request a blood bank and obtain plasma from the blood bank. The main goal of our project is to design a user-friendly web application thatis like a scientiﬁc vehicle from which we can help reduce mortality or help those affected by COVID19 by donating plasma from patients who have recovered without approved anti- retro-viral therapy planning for a deadly COVID19 infection, plasma therapy is an experimental approach to treat those COVID-positive patientsand help them recover faster. Therapy, which is considered reliable and safe. If a particular person has fully recovered from COVID19, they are eligible to donate their plasma. As we all know, the traditional methods of ﬁnding plasma, one has toﬁnd out for oneself by looking at hospital records and contacting donors have beenrecovered, sometimes may not be available at home and move to other places. In this type of scenario, the health of those who are sick becomes disastrous. Therefore, it is not considered a rapid process to ﬁnd plasma.

IBM-48000-1662641884

* + 1. **TITLE :** A Web Application to Manage All Blood Donation and Transfusion Processes.

**AUTHOR:** Rehab S. Ali1, Tamer F. Hafez2, Ali Badawey Ali3.

Many lives could be lost due to the diﬃculty in obtaining a proper blood bag, Therefore, this work aims to help citizens fulﬁll their needs for a safe and reliable blood group by searching for and locating a speciﬁc blood group. In this paper, weillustrate the problem of the blood bags shortage which is represented in the uncontrolled blood banks and parallel markets, lack of awareness and conﬁdence,disappearance of the rare blood groups, and the diﬃculty in ﬁnding a speciﬁc

blood group. Hence, we proposed the Blood Bag web-based application that is connected to a centralized database to gather and organize the data from all bloodbanks and blood donation campaigns. The proposed application organizes and controls the whole critical processes related to blood donation, testing and storageof blood bags, and delivering it to the patient. One blood bag can save a life duringsurgeries or road accidents, etc. Usually patients or their families look for a speciﬁc blood group they indeed need in the blood banks but they normally cannotﬁnd it due to the shortage of blood bags. This is because of the fear of donating blood and the misconception that donating blood is harmful and transmits diseases. This is one of the obstacles to provide the blood bags. The availability ofthe blood bags is critical because of the high proportion of patients with renal failure, some cases of birth, surgeries processes and incidents that need to get theblood as soon as possible to save these cases’ lives. The blood bank is the pool ofdifferent blood groups where keeping a stockpile of blood to be distributed in case.The matched blood groups for a safe transfusion . Accidents (or any medical emergency and compensation of blood missing from the body), and keeping the blood in the freezer temperature.

IBM-48000-1662641884

* + 1. **TITLE :** Developing a plasma donor application using Function-as-a-service in AWS.

**AUTHOR:** Aishwarya R Gowri1

Plasma is a liquid portion of the blood, over 55% of human blood is plasma. Plasma is used to treat various infectious diseases and it is one of the oldest methods known as plasma therapy. Plasma therapy is a process where blood is donated by recovered patients in order to establish an antibody that ﬁghts the infection. In this project plasma donor application is being developed by using AWS services. The services used are AWS Lambda, API gateway, Dynamo DB, AWSElastic Compute Cloud with the help of these AWS services, it eliminates the need of conﬁguring the servers and reduces the infrastructural costs associated with it and helps to achieve serverless computing. For instance, during COVID 19 crisis

4

IBM-48000-166264188

the requirement for plasma increased drastically as there were no vaccination found in order to treat the infected patients, with plasma therapy the recovery rateswhere high but the donor count was very low and in such situations it was very important to get the information about the plasma donors. Saving the donor information and notifying about the current donors would be a helping hand as it can save time and help the users to track down the necessary information about the donors. The proposed method helps the users to check the availability of donors. A donor has to register to the website providing their details. The registered users can get the information about the donor count of each blood group. The database will have all the details such as name, email, phone number, infected status. Whenever a user requests for a particular blood group then the concerned blood group donors will receive the notiﬁcation regarding the requirement. A Json code is written to store the information, to fetch the requestedinformation in lambda.

* + 1. **TITLE :** Nearest Blood & Plasma Donor Finding: A Machine Learn in Approach.

**AUTHOR:** Nayan Das1, Asif Iqbal2.

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, Hemophilia, and acute viral infections like Dengue, etc. Approximately 85 million people require single or multiple blood transfusions for treatment. Voluntary blooddonors per 1,000 population of some countries are quite promising, such as Switzerland (113/1,000), Japan (70/1,000), while others have an unsatisfying resultlike India has 4/1,000, and Bangladesh has 5/1000. Recently a life-threatening virus, COVID-19, spreading throughout the globe, which is more vulnerable for older

people and those with pre-existing medical conditions. For them, plasma is neededto recover their illness. Our Purpose is to build a platform with clustering algorithms which will jointly help to provide the quickest solution to ﬁnd blood or plasma donor. Closest blood or plasma donors of the same group in a particular area can be explored within less time and more eﬃciently. Keywords—Blood donation, Plasma donation, K means clustering, Labeled Agglomerate clustering. Different methods have been used to solve this problem. This time, we have tried another way, a clustering approach, to solve the problem by grouping every user into small groups. This unsupervised machine learning approach is much faster and effective. In section II, we will discuss related work done previously to solve this problem. In section III, clustering algorithms relating to our project will explicitly be discussed. In section IV, our proposed method will be presented. In section V, we will analyze our experiment result.

IBM-48000-1662641884

* + 1. **TITLE :** Securing Information on a Web Application System to Facilitate Online Blood Donation Booking.

**AUTHOR:** Hrishitva Patel1.

Blood donation has saved many lives in the past. According to the American Red Cross statistics, a patient needs a blood transfusion every two seconds. Manybeneﬁts arise from blood donation to both the donor and the blood recipients. Withblood donation, cancer patients, people involved in accidents, or those battling diseases that require blood donation have access to enough blood to sustain theirsurvival. There is a need to digitize the blood donation booking to facilitate blood donation across the United States, and ensure patients in need of blood, receive their donation from eligible donors on time. This report demonstrates the security measures implemented to secure patient and blood donor data on a blood donation booking web application. Blood is donated for different reasons in hospitals and other blood banks. It is essential to help blood recipients survive surgeries, cancer treatment, and chronic illnesses, among other illnesses. The

4

IBM-48000-166264188

World Health Organization describes blood as the most precious gift a person cangive to a person in need of it. Blood donated comprises four components: platelets, plasma, white blood cells, and red blood cells. Cancer patients require a blood transfusion to enhance platelets back into the body after radiation therapy. With a developed web application to book a blood donation session, it is easier forhospitals to know which blood component they need most and thus inform the system administrator to prompt for more blood donors.

### REFERENCES :

1. Kalpana Devi Guntoju1, Tejaswini Jalli2, Instant Plasma Donor Recipient Connector web application, 2022.
2. Rehab S. Ali1, Tamer F. Hafez2, Ali Badawey Ali3, A Web Application to Manage All Blood Donation and Transfusion Processes, 2017.
3. Aishwarya R Gowri, Developing a plasma donor application using Function-as-a-service in AWS, 2020.
4. Nearest Blood & Plasma Donor Finding: A Machine Learning Approach, 2021
5. Hrishitva Patel, Securing Information on a Web Application System to FacilitateOnline Blood Donation Booking, 2022.

### PROBLEM STATEMENT DEFINITION :

In critical or emergency situations where accident occurs or during on-going treatments and surgeries etc there is urgent need for speciﬁc blood group. It requires lotof time to make the blood available and it is inconvenient during emergency situation, some rare blood groups are time consuming and diﬃcult to arrange which are O- , AB- etc. In our country there is less awareness of blood donation, near about 20% of Indian population donates blood. In existing system the blood bank management system exhibited at a lot of ineffectiveness and ineﬃciency that had fetched impact taken by management. The system which was manual that is based on paper card to collect

blood donor data, keep record of blood donors and disseminate results to blood donors,had weakness that needed IT based solutions.

IBM-48000-1662641884

### PROBLEM STATEMENT:

In the emergency condition, sometimes it becomes very much diﬃcult to look for the exact match of blood group of donor and acceptor. It may lead to delay in transaction of plasma within the speciﬁed amount of time. This application is providingeach entity the facility to approach nearby blood donors so that it will become much easier to search rare blood groups in the hour of need.

### I am (USER)

Donor has to upload their blood group details for plasma donation**.**

### I am Trying To

The project blood bank management system is known to be a pilot project that is designed for the blood bank to gather blood from various sources and distribute it to theneedy people who have high requirements for it.

### But

This might be the result of a human error, It leads to error prone results, It consumeslot of manpower to better results.

### Because

It is hard, and there is a delay to know about the donor details.

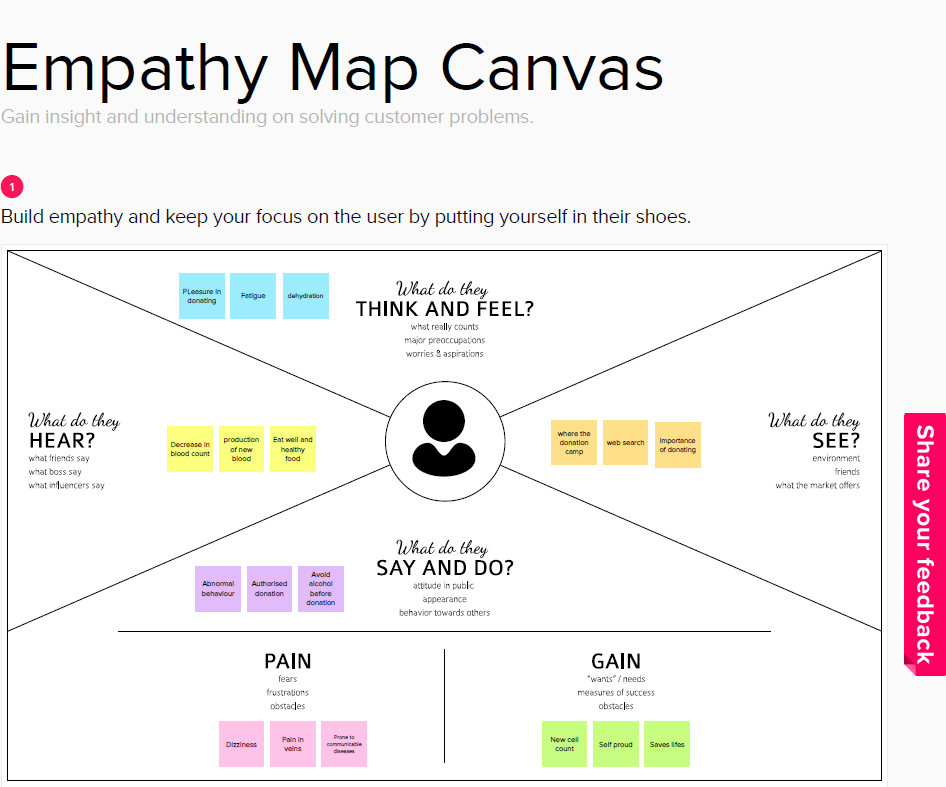
### Which makes me feel?

If the blood group is not available in the blood bank user can request the donor to donate the plasma to him and save someone life. Using this system people can registerhimself or herself who want to donate plasma. To register in the system they have to enter their contact information like address mobile number etc.

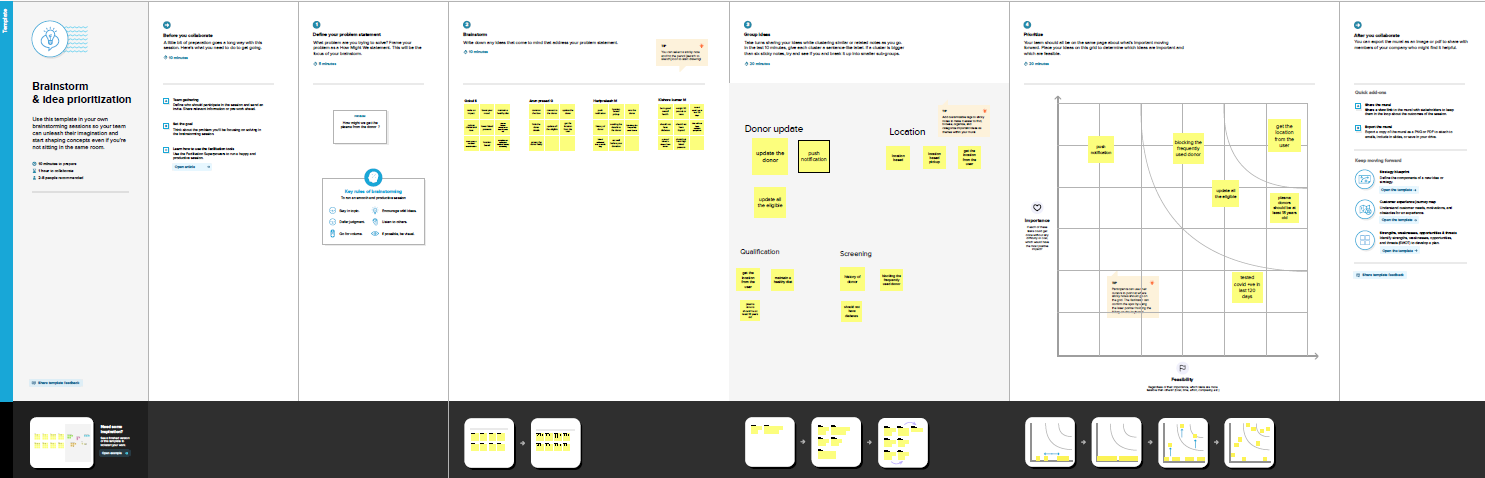
# IDEATION & PROPOSED SOLUTION

IBM-48000-1662641884

### EMPATHY MAP CANVAS:

****

* 1. **IDEATION & BRAINSTROMING:**

****

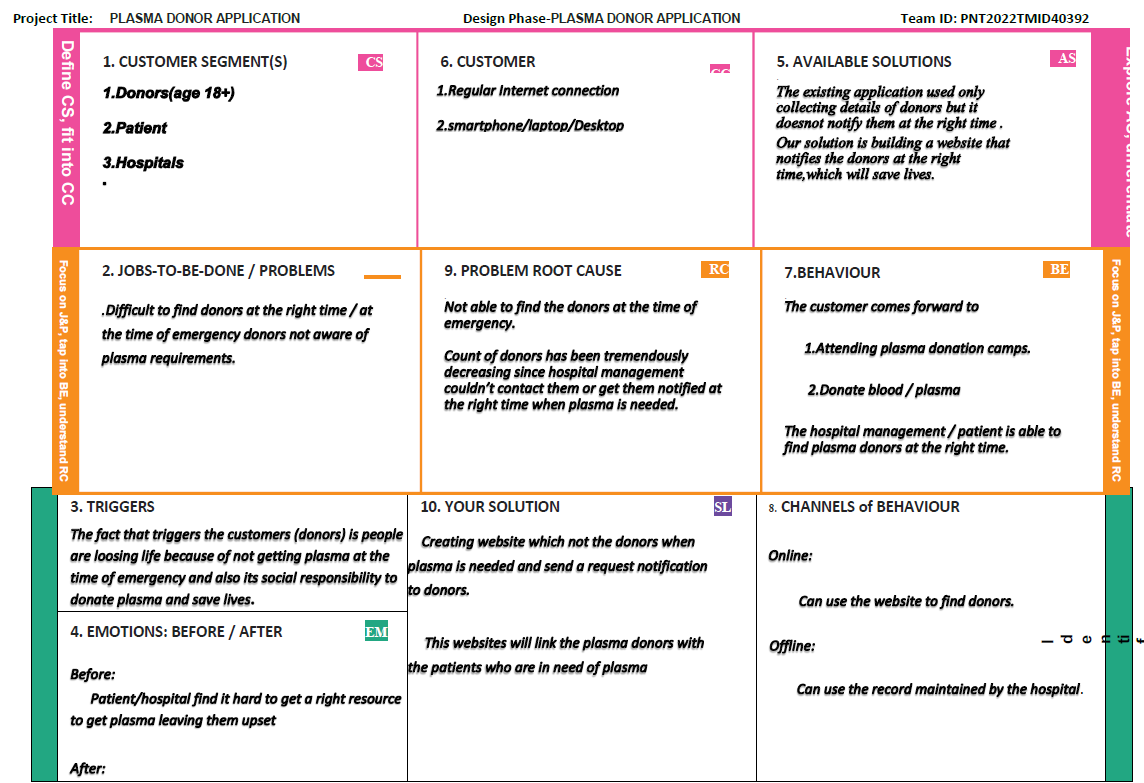
### PROPOSED SOLUTION:

The new idea will improve the existing system and it will move from conventionaldesktop system to mobile system. This paper introduces new features of improved system over existing system in many aspects. The proposed plasma donor application helps the people who are in need of plasma by giving them all details of plasma availability or regarding the donors with the same blood group. This is a web applicationallows you to access the whole information about plasma donor application, readily scalable and adaptable to meet the complex need of plasma Who are Key Facilitator forthe Healthcare Sector, it also supports all the functionalities of plasma donor application.

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Parameter** | **Description** |
| 1. | Problem Statement (Problemto be solved) | In existing system the blood bank management system exhibited at a lot of ineffectiveness and ineﬃciency that had fetched impact taken by management. |
| 2. | Idea / Solution description | It saves time as he can search donorsonline without going anywhere. |
| 3. | Novelty / Uniqueness | It is a user-friendly application. |
| 4. | Social Impact / Customer Satisfaction | Accurate ﬁndings increase people's satisfaction. |
| 5. | Business Model (Revenue Model) | - |
| 6. | Scalability of the Solution | This system proposed here aims at connecting the donors & the patients by an online application. |

* 1. **PROBLEM SOLUTION FIT :**

|  |
| --- |
|  |
|



# REQUIREMENT ANALYSIS

### FUNCTIONAL REQUIREMENT

* + 1. **Admin:**

Admin can manage both donors and users. Admin has the only responsibility maintain and stored the record.

### Users:

From this module user can create their account, when user create his account the user get a user id and password which identiﬁes him uniquely. From this moduleuser can search donor for blood.

### Donors Registration:

In this module, people who are interested in donating blood get registered in thissite and give his overall details related to donor. User details contain name, address,city, gender, blood group, location, contact number etc.

### Donor Search:

The people who are in need of blood can search in our site for getting the detailsof donors having the same blood group and within the same city.

### Notification:

In this module, notiﬁcation sends to donors for emergency. SMS send toregistered donors phone number.

### NON FUNCTIONAL REQUIREMENTS:

* + 1. **Usability:**

The system shall allow the users to access the system with pc using web application.

The system uses a web application as an interface. The system is user

friendly which makes the system easy

### Availability:

The system is available 100% for the user and is used 24 hrs a day and 365 days a year. The system shall be operational 24 hours a day and 7 days a week.

### Scalability :

Scalability is the measure of a system's ability to increase or decrease in performance and cost in response to changes in application and system processingdemands.

### Security:

A security requirement is a statement of needed security functionality that ensures one of many different security properties of software is being satisﬁed.

### Performance:

The information is refreshed depending upon whether some updates have occurred or not in the application. The system shall respond to the member in not less than two seconds from the time of the request submittal. The system shall be allowed to take more time when doing large processing jobs. Responses to view information shall take no longer than 5 seconds to appear on the screen.

### Reliability:

The system has to be 100% reliable due to the importance of data and the damages that can be caused by incorrect or incomplete data. The system will run 7days a week. 24 hours a day.

# PROJECT DESIGN

### DATA FLOW DIAGRAMS :

A data-ﬂow diagram is a visual representation of how data moves through a systemor a process (usually an information system). The DFD additionally gives details about each entity's inputs and outputs as well as the process itself. A data-ﬂow diagram lackscontrol ﬂow, loops, and decision- making processes. Using a ﬂowchart, certain

operations depending on the data may be depicted.

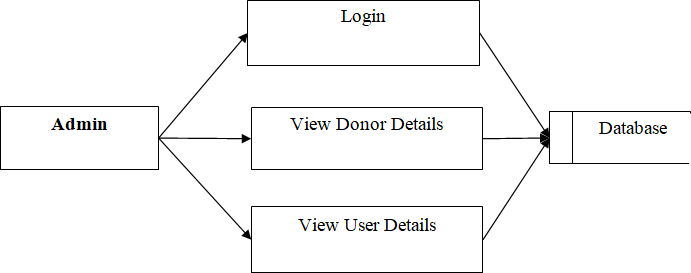
### Data flow Symbols:

|  |  |
| --- | --- |
| **Symbol** | **Description** |
|  | An **entity**. A source of data or a destination for data. |
|  | A **process** or task that is performed by the system. |
|  | A **data store**, a place where data is held between processes. |
|  | A **data flow**. |

**LEVEL 0:**

IBM-48000-1662641884

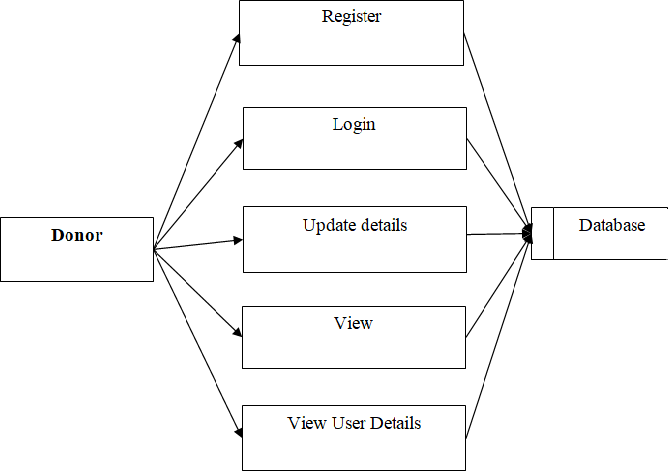
The Level 0 DFD shows how the system is divided into 'sub-systems' (processes),each of which deals with one or more of the data ﬂows to or from an external agent, andwhich together provide all of the functionality of the system as a whole. It also identiﬁesinternal data stores that must be present in order for the system to do its job, and shows the ﬂow of data between the various parts of the system.



### LEVEL 1:

The next stage is to create the Level 1 Data Flow Diagram. This highlights the main functions carried out by the system. As a rule, to describe the system was using between two and seven functions - two being a simple system and seven being a complicated system. This enables us to keep the model manageable on screen or paper.

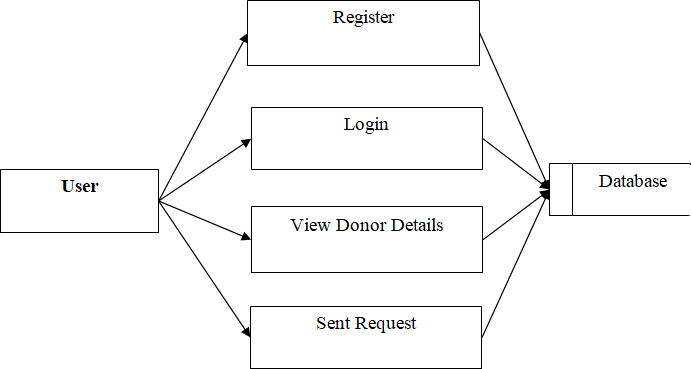
IBM-48000-1662641884



### LEVEL 2:

A Data Flow Diagram (DFD) tracks processes and their data paths within the business or system boundary under investigation. A DFD deﬁnes each domain boundaryand illustrates the logical movement and transformation of data within the deﬁned boundary. The diagram shows 'what' input data enters the domain, 'what' logical processes the domain applies to that data, and 'what' output data leaves the domain. Essentially, a DFD is a tool for process modeling and one of the oldest.

IBM-48000-1662641884

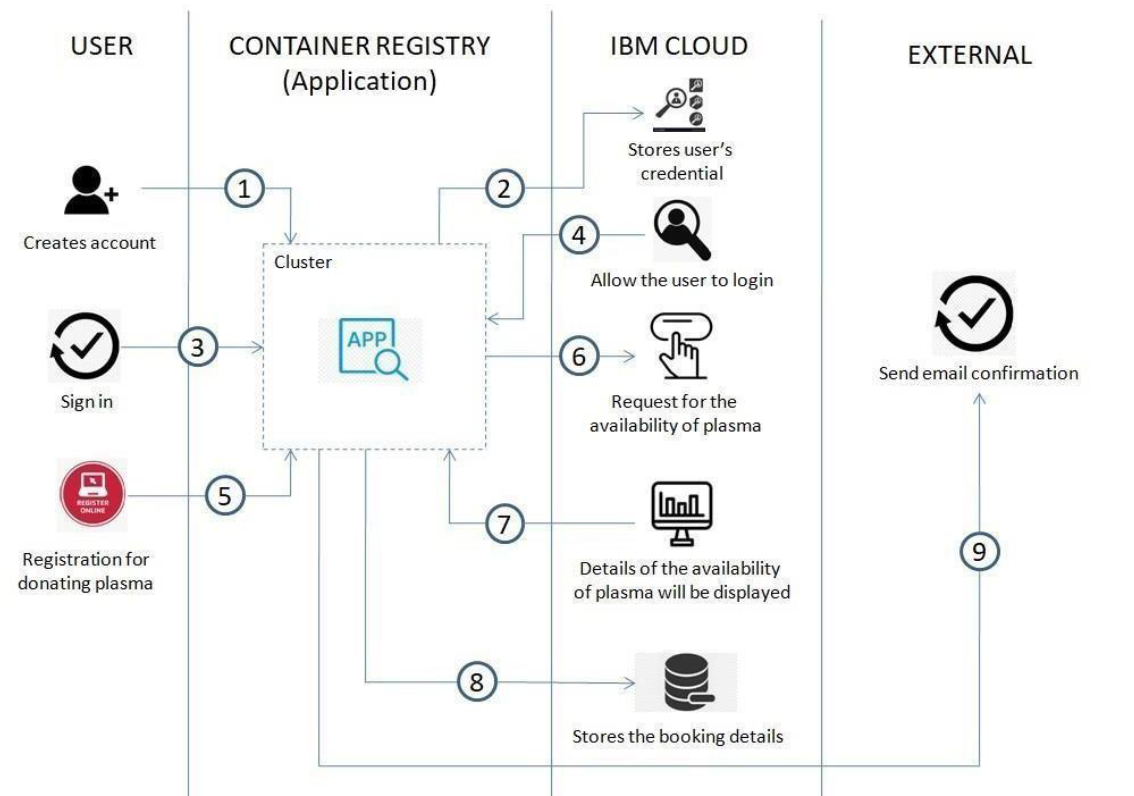


### SOLUTION & TECHNICAL ARCHITETURE:

* + 1. **SOLUTION ARCHITETURE:**

System architecture, also known as systems architecture, is the conceptual model that describes a system's structure, behavior, and additional viewpoints. An architecturedescription is a formal description and representation of a system that is organized in away that allows for reasoning about the system's structures and behaviors. System architecture might include system components, their outwardly evident attributes, and the connections (e.g., behavior) between them. It can give a strategy for acquiring itemsand developing systems that will operate together to accomplish the entire system. There have been initiatives to codify languages for describing system architecture; theyare referred to collectively as architecture description languages (ADLs).

### TECHNICAL ARCHITETURE:

****

**COMPONENTS & TECHNOLOGIES:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1. | User Interface | Web UI | HTML, CSS, JavaScript |
| 2. | Application Logic-1 | Logic for a process in the application | Python |

|  |  |  |  |
| --- | --- | --- | --- |
| 3. | Application Logic-2 | Logic for a process in the application | SMTP |
| 4. | Application Logic-3 | Logic for a process in the application | SMTP |
| 5. | Database | Data Type, Conﬁgurations etc. | MySQL |
| 6. | Cloud Database | Database Service on Cloud | Local host |
| 7. | File Storage | File storage requirements | Local host |
| 8. | External API-1 | Purpose of External API used in the application | - |
| 9. | External API-2 | Purpose of External API used in the application | - |
| 10. | Machine Learning Model | Purpose of Deep Learning  Model | Classiﬁes and detect the images with high accuracy |
| 11. | Infrastructure (Server /  Cloud) | Application Deployment on Local System / Cloud Local Server Conﬁguration:  Cloud Server Conﬁguration : | Local Server Conﬁguration |

### APPLICATION CHARACTERISTICS:

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 1. | Open-Source Frameworks | List the open-source frameworks used | PYCHARM |
| 2. | Security Implementations | List all the security / access controls implemented, use of ﬁrewalls etc. | - |
| 3. | Scalable Architecture | Justify the scalability of architecture (3 – tier, Micro- services) | Able to respond the changes in an application |
| 4. | Availability | Justify the availability of application (e.g. use of load balancers, distributed servers etc.) | The system must alwaysbe functional |

|  |  |  |  |
| --- | --- | --- | --- |
| 5. | Performance | Design consideration for the performance of theapplication (number of requests per sec, use of  Cache, use of CDN’s) etc. | Takes no longer time to response |

* 1. **USER STORIES:**

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

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **User Type** | **Functional Requireme nt (Epic)** | **User Story Numb er** | **User Story/ Task** | **Acceptancecrite ria** | **Prior ity** | **Relea se** |
| Custo mer (Mobi le 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  conﬁrmati on email | I can receive conﬁrmation email &click conﬁrm | High | Sprint- 1 |
|  | once I have |  |  |  |
|  | registered |  |  |  |
|  | for the |  |  |  |
|  | application |  |  |  |
|  |  | USN-3 | As a user, I can  register for the application through Gmail | I can receive conﬁrmation notiﬁcations through Gmail | Medi um | Sprint- 1 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Login | USN-4 | As a user, I can  login to the | I can access  into my User | High | Sprint-1 |
|  |  | application by | proﬁle and |  |  |
|  |  | entering email | view details |  |  |
|  |  | &password | in |  |  |
|  |  |  | dashboard |  |  |
|  | Dashboard | USN-5 | As a user,I can send the | I can receive appropriate | High | Sprint-1 |
|  |  | proper | notiﬁcations |  |  |
|  |  | requests to | through |  |  |
|  |  | donate and  obtain | email |  |  |
|  |  | plasma. |  |  |  |
| Cust | Login | USN-6 | As a user,Ican register and log into the application by entering email& password to view the  proﬁle | I can access into my User proﬁle and view details in dashboard | High | Sprint-1 |
| om |  |  |  |  |
| er |  |  |  |  |
| (W |  |  |  |  |
| eb |  |  |  |  |
| use |  |  |  |  |
| r) |  |  |  |  |
|  | 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 | Applicati on | 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 | Medi um | Sprint-2 |
| Administra tor | Applicati on | USN-9 | As an administrator I can help with user-facing aspects of a | I can change the  appearance and  navigation in a  userfriendly manner | Medi um | Sprint-3 |
|  |  |  | website, like |  |  |
|  |  |  | its |  |  |
|  |  |  | appearance, |  |  |
|  |  |  | navigation |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  | and use of media. |  |  |  |
|  |  | USN- 10 | As an administrator, I can involve working withthe technical side of websites. | I can help with such as troubleshooti  ng issues,  setting up web hosts,ensuri ng  users have-access and programmi ngservers | Medi um | Sprint-1 |

# PROJECT PLANNING & SCHEDULING

### SPRINT PLANNING & ESTIMATION:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sprint** | **Functional Requireme nt(Epic)** | **User**  **StoryNumb er** | **User Story / Task** | **Priority** |
| Sprint- 1 | **Registration andLogin** | USN-1 | Create UI to interact with pages. To createthe user and admin login functionality | High |
| Sprint- 2 | **Cloud and Database** | USN-2 | Connecting ﬂask app with database [IBMDB2] Implementation of IBM chatbot | High |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sprint- 3 | **Deployme ntin Develops phase** | USN-3 | Creating images with docker, Deploying Kubernetes and add the mailing service. | High |
| Sprint- 4 | **Testing and Deployment to user** | USN-4 | To make sure that the software is handy to users. | High |

* 1. **SPRINT DELIVERY SCHEDULE:**

### PROJECT TRACKER:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Total Story Poin ts** | **Durati on** | **Sprint Start Date** | **Sprint EndDa te (Planne d)** | **Story Points Complet ed(as on Planned End Date)** | **Sprint Relea se Date (Actua l)** |
| 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  Nov20 22 | 12 Nov 2022 | 20 | 12  Nov2022 |
| Sprint- 4 | 20 | 6 Days | 14  Nov20 22 | 19 Nov 2022 | 20 | 19  Nov2022 |

* + 1. **VELOCITY:**

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

Sprint duration = 6 days Velocity of the team = 20 points

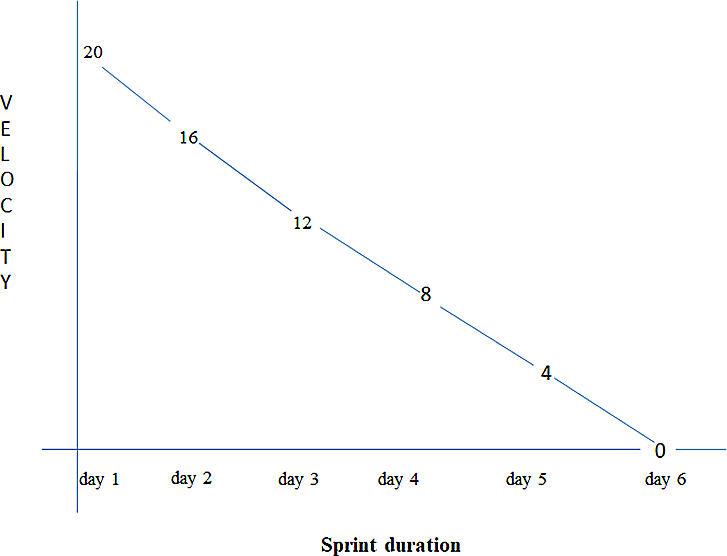


AV = 20/6 = 3.34

Average Velocity = 3.34

### Burndown Chart:

A burn down chart is a graphical representation of work left to do versustime. It is often used in agile [software development](https://www.visual-paradigm.com/scrum/what-is-agile-software-development/) methodologies such as[Scrum.](https://www.visual-paradigm.com/scrum/scrum-in-3-minutes/) However, burn down charts can be applied to any project containing measurable progress over time.



# CODING AND SOLUTIONING

IBM-48000-1662641884

### FEATURE -1:

#main.py

# importing libraries from ﬂask import Flask

from ﬂask\_mail import Mail, Messageapp = Flask( name )

mail = Mail(app) # instantiate the mail class # conﬁguration of mail app.conﬁg['MAIL\_SERVER'] = 'smtp.gmail.com' app.conﬁg['MAIL\_PORT'] = 465

app.conﬁg['MAIL\_USERNAME'] = 'sampletest685@gmail.com' app.conﬁg['MAIL\_PASSWORD'] = 'hneucvnontsuwgpj' app.conﬁg['MAIL\_USE\_TLS'] = False app.conﬁg['MAIL\_USE\_SSL'] = True

mail = Mail(app)

# message object mapped to a particular URL ‘/’@app.route("/") def index():

msg = Message( 'Hello',

sender='sampletest685@gmail.com', recipients=['sangeeth5535@gmail.com']

)

msg.body = 'Hello Flask message sent from Flask-Mail' mail.send(msg)

return 'Sent'

if name == ' main ': app.run(debug=True)

### FEATURE-2:

-- phpMyAdmin SQL Dump

-- version 2.11.6

-- [http://www.phpmyadmin.net](http://www.phpmyadmin.net/)

--

-- Host: localhost

-- Generation Time: Nov 05, 2022 at 04:58 AM

-- Server version: 5.0.51

-- PHP Version: 5.2.6

SET SQL\_MODE="NO\_AUTO\_VALUE\_ON\_ZERO";

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

/\*!40101 SET NAMES utf8 \*/;

--

-- Database: `1plasmadb`

--

--

-- Table structure for table `admintb`

--

CREATE TABLE `admintb` (

`UserName` varchar(250) NOT NULL,

`Password` varchar(250) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--

-- Dumping data for table `admintb`

--

INSERT INTO `admintb` (`UserName`, `Password`) VALUES ('admin', 'admin');

--

-- Table structure for table `donortb`

--

CREATE TABLE `donortb` (

`id` bigint(10) NOT NULL auto\_increment,

`Name` varchar(250) NOT NULL,

`Mobile` varchar(250) NOT NULL,

`Email` varchar(250) NOT NULL,

`UserName` varchar(250) NOT NULL,

`Password` varchar(250) NOT NULL, PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=2 ;

* 1. **TEST CASES:**

# TESTING

A test case has components that describe input, action and an expected response, in order to determine if a feature of an application is working correctly.A test case is a set of instructions on “HOW” to validate a particular test objective/target, which when followed will tell us if the expected behavior of the system is satisﬁed or not.

Characteristics of a good test case:

* + - * 1. Accurate: Exacts the purpose.
        2. Economical: No unnecessary steps or words.
        3. Traceable: Capable of being traced to requirements.
        4. Repeatable: Can be used to perform the test over and over.
        5. Reusable: Can be reused if necessary.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.NO** | **Scenario** | **Input** | **Excepted output** | **Actual output** |
| 1 | Admin Login Form | User name and password | Login | Login success. |
| 2 | Donor Registration Form | Donor basic details | Registration | Donor registration details stored in  database. |
| 3 | User Registration Form | User basic details | Registration | User registration  details stored in database. |
| 4 | User Login Form | User name and  password | Login | Login success. |

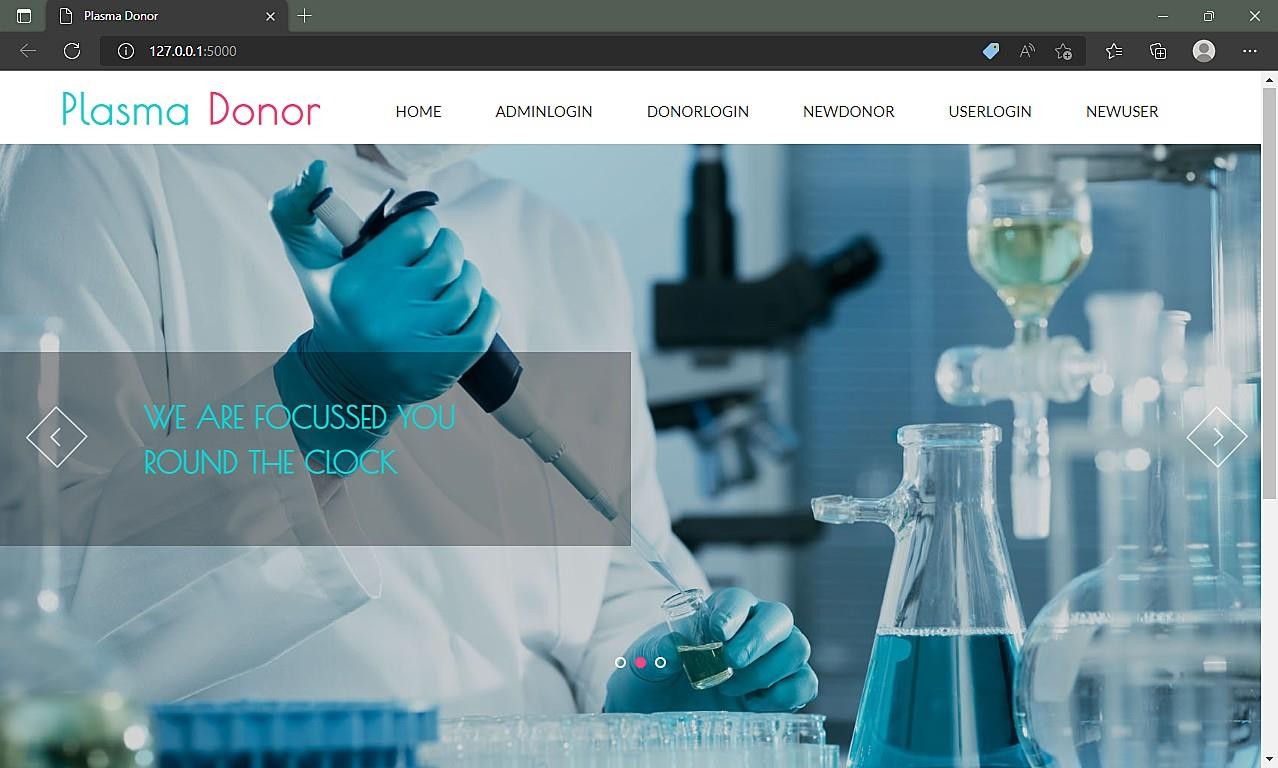
### USER ACCEPTANCE TESTING :

This is a type of testing done by users, customers, or other authorized entities to determine application/software needs and business processes. Acceptance testing is the most important phase of testing as this decides whether the client approves the application/software or not. It may involve functionality, usability, performance, and U.Iof the application. It is also known as user acceptance testing (UAT), operational acceptance testing (OAT), and end-user testing.

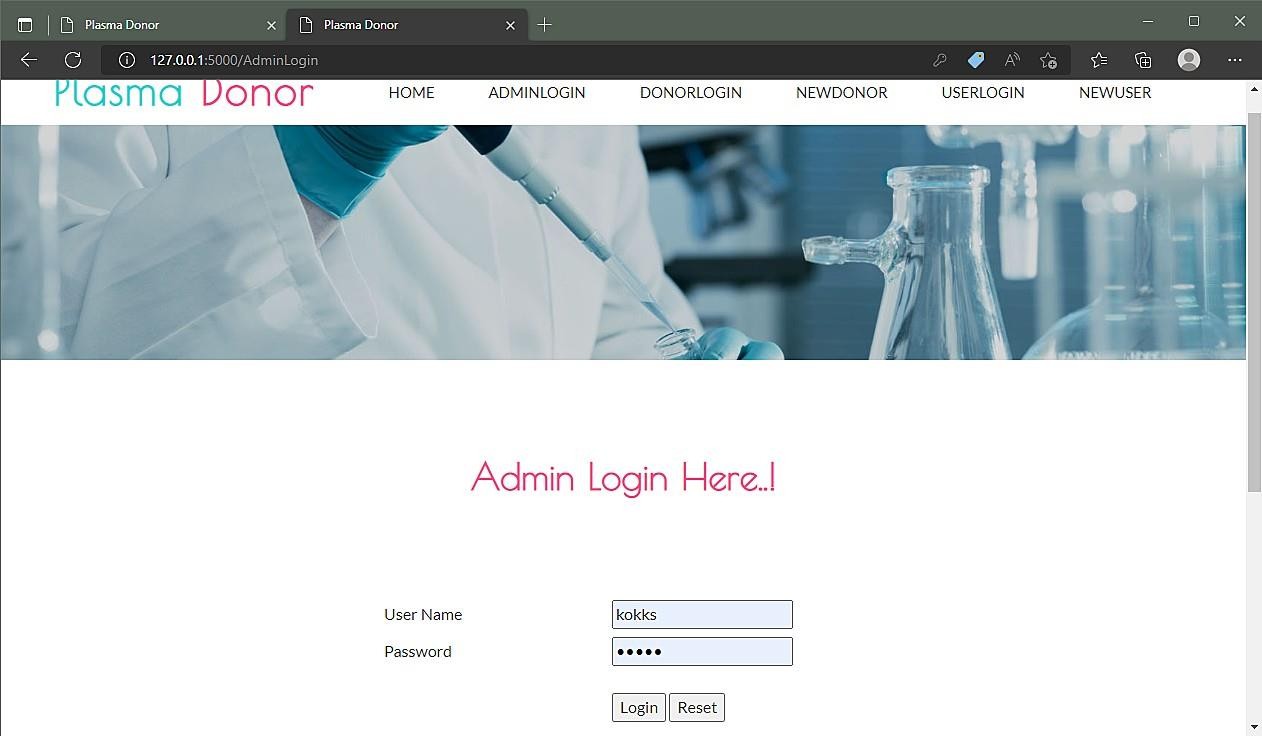
# RESULTS

### PERFORMANCE METRICS:

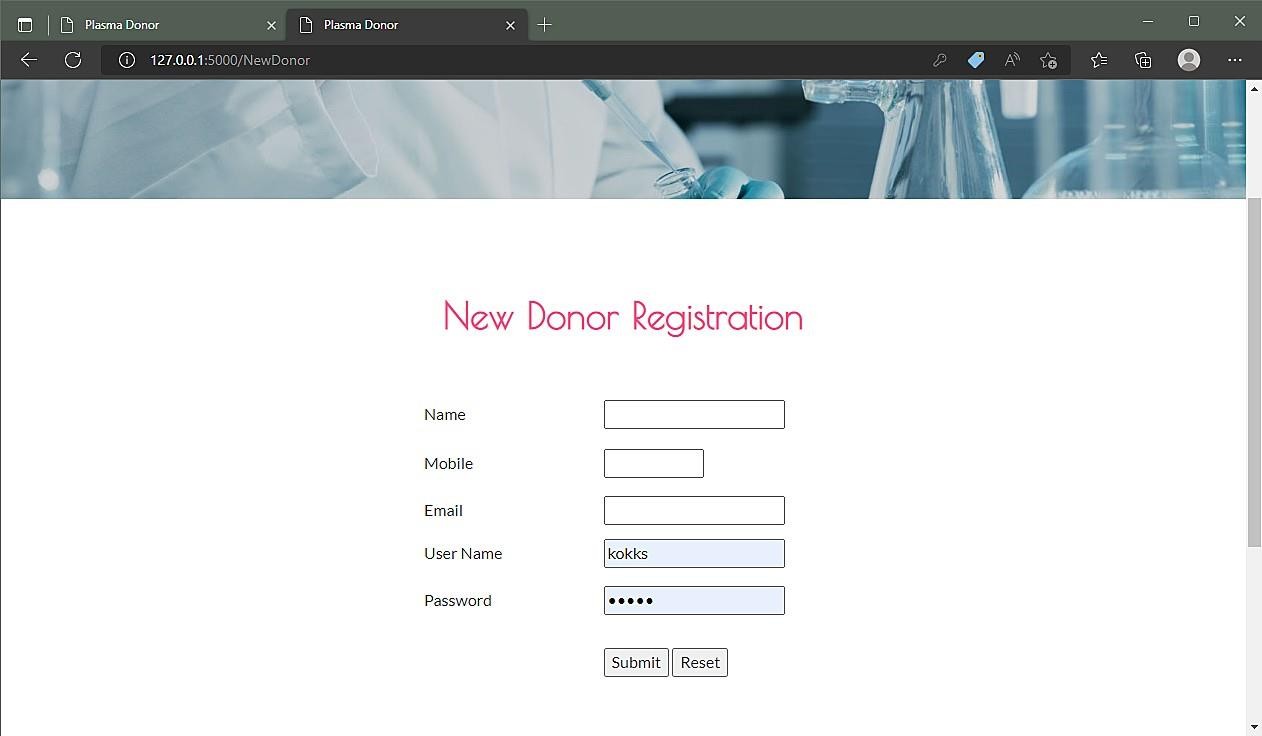
* + 1. **HOME PAGE:**



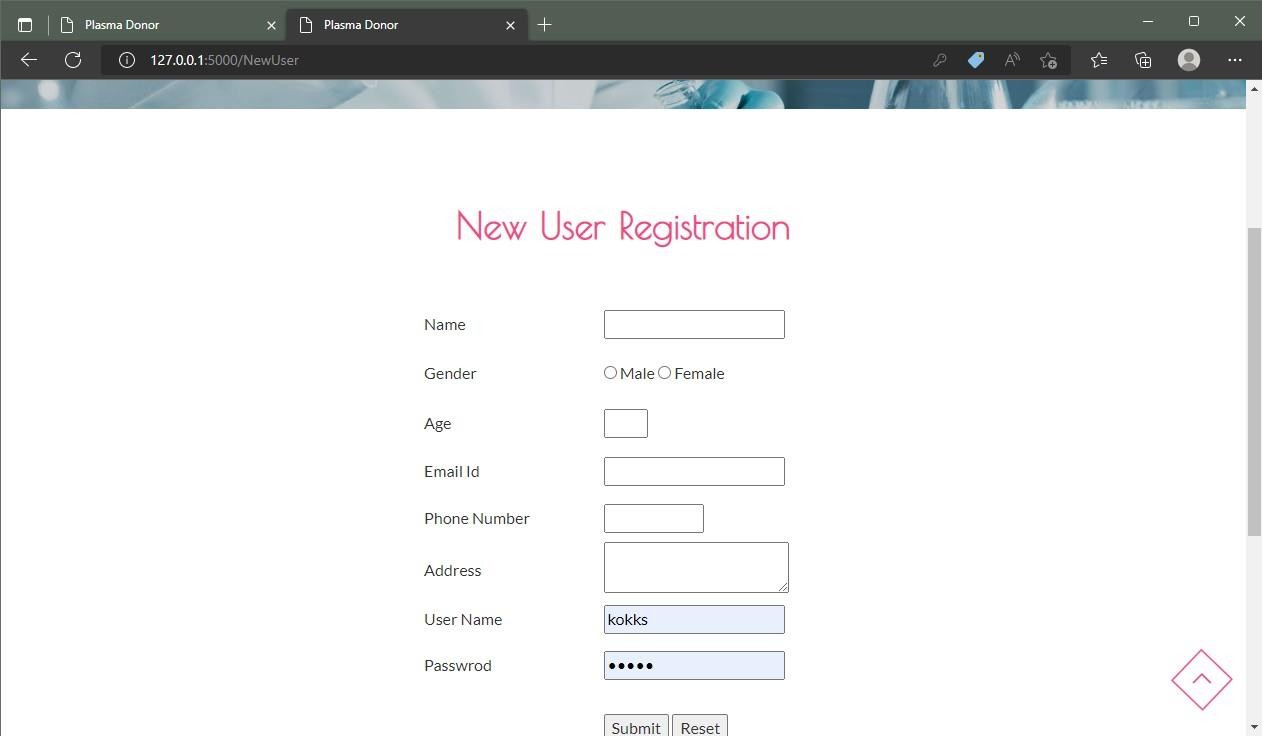
### ADMIN PAGE:



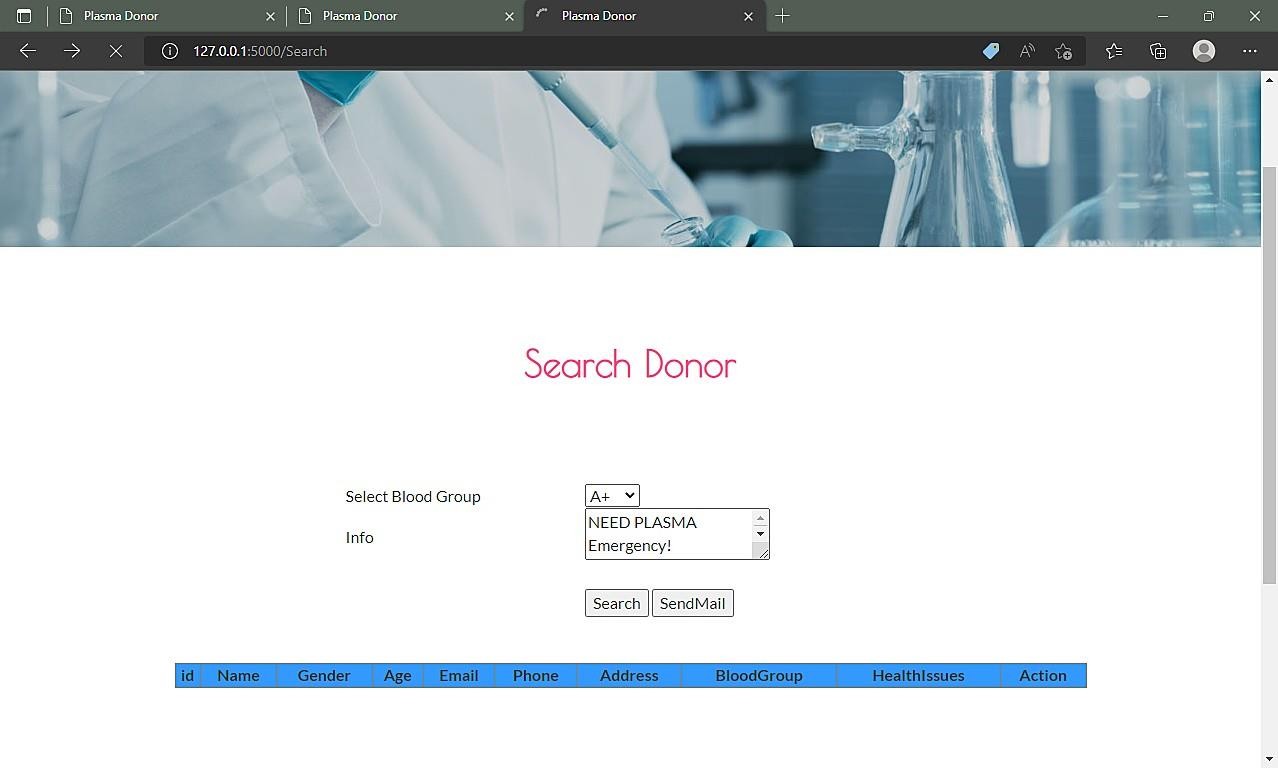
* + 1. **DONOR PAGE:**



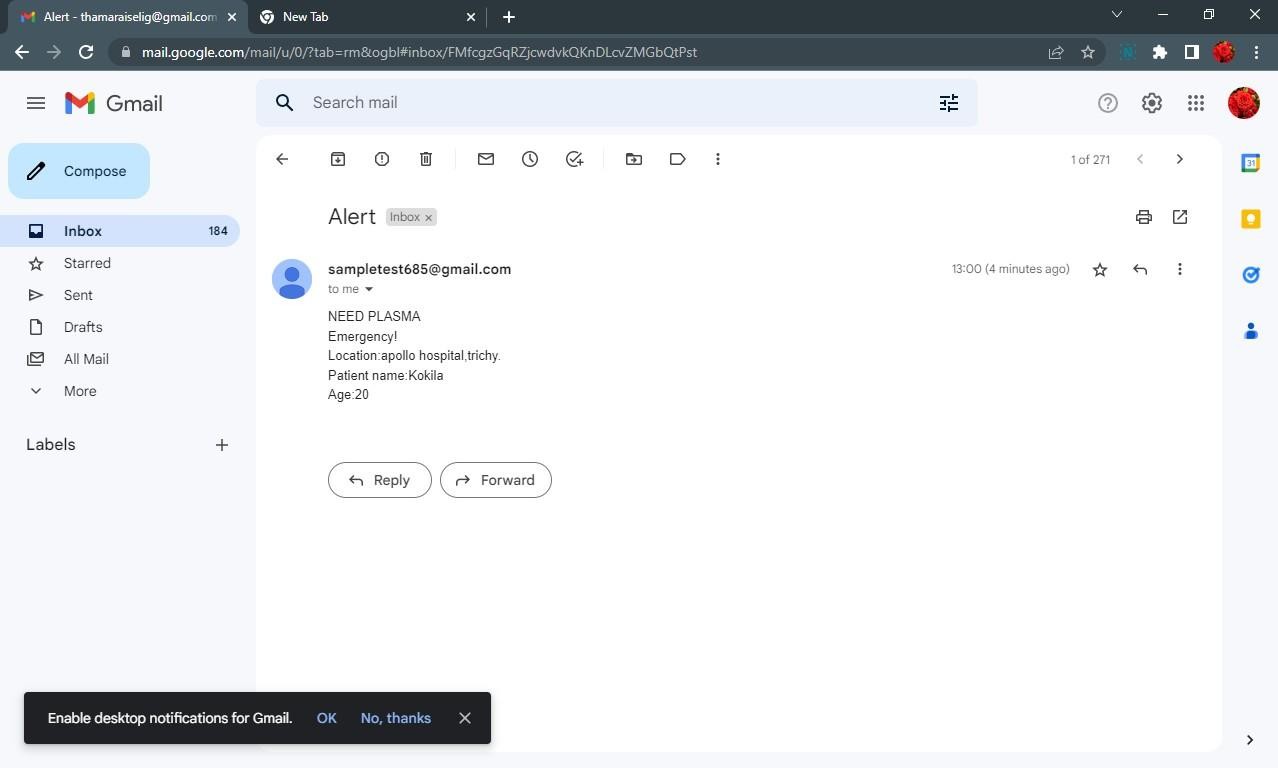
IBM-48000-1662641884



* 1. **INPUT:**



IBM-48000-1662641884



# ADVANTAGES & DISADVANTAGES

### ADVANTAGES :

* + 1. It is a user-friendly application.
    2. The people in need of plasma can search for the donors by giving their blood group and city name.
    3. It saves time as he can search donors online without going anywhere.
    4. Using this system user can get plasma in time and can save and here our system work, whenever a person needs plasma user get information of the person who has the same blood group needs.

### DISADVANTAGES:

IBM-48000-1662641884

* + 1. It is time consuming.
    2. It leads to error prone results.
    3. It consumes lot of manpower to better results.
    4. It lacks of data security.
    5. Retrieval of data takes lot of time.

# CONCLUSION

This project is designed for successful completion of project on Plasma Donor Application system. The basic building aim is to provide plasma donation service to thecity recently.Plasma Donor Application System is a Web based application that is designed to store, process, retrieve and analyze information concerned with the administrative and inventory management within a plasma. This project aims at maintaining all the information pertaining to plasma donors, different blood groups available in each plasma bank and helps them manage in a better wayplasma donation system can collect plasma from many donors in short from various sources and distribute that plasma to needy people who require plasma. To do all this we require high quality Web Application to manage those jobs. Plasma application provides a reliable platform to connect local plasma donors with patients.

# FUTURE SCOPE

This system is developed such a way that additional enhancement can be done without much diﬃculty. The renovation of the project would increase the ﬂexibility of thesystem. In future, we can develop this project in android platform. We will add extra features like donor location tracking system (GPS), Feedback form, and enable call option etc.

**13.1 SOURCE CODE:**

IBM-48000-1662641884

# APPENDIX

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

#from wtforms import Form, TextField, TextAreaField, validators, StringField, SubmitField from werkzeug.utils import secure\_filename

import datetime

from flask\_mail import Mail, Message import mysql.connector

import sys

app = Flask( name )

app.config['DEBUG']

app.config['SECRET\_KEY'] = '7d441f27d441f27567d441f2b6176a' @app.route("/")

def homepage():

return render\_template('index.html') @app.route("/AdminLogin")

def AdminLogin():

return render\_template('AdminLogin.html') @app.route("/DonorLogin")

def DonorLogin():

return render\_template('DonorLogin.html')

@app.route("/NewDonor") def NewDonor():

IBM-48000-1662641884

return render\_template('NewDonor.html') @app.route("/UserLogin")

def UserLogin():

return render\_template('UserLogin.html') @app.route("/PersonalInfo")

def PersonalInfo():

return render\_template('DonorPersonal.html') @app.route("/NewUser")

def NewUser():

return render\_template('NewUser.html') @app.route("/AdminHome")

def AdminHome():

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') cur = conn.cursor()

cur.execute("SELECT \* FROM regtb ") data = cur.fetchall()

return render\_template('AdminHome.html',data=data) @app.route("/AdminDonorInfo")

def AdminDonorInfo():

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb')

cur = conn.cursor()

cur.execute("SELECT \* FROM personltb ") data = cur.fetchall()

return render\_template('AdminDonorInfo.html', data=data) @app.route("/UserHome")

def UserHome():

user = session['uname']

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') # cursor = conn.cursor()

cur = conn.cursor()

cur.execute("SELECT \* FROM regtb where username='" + user + "'") data = cur.fetchall()

return render\_template('UserHome.html',data=data) @app.route("/DonorHome")

def DonorHome():

cuname = session['cname']

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') # cursor = conn.cursor()

cur = conn.cursor()

cur.execute("SELECT \* FROM companytb where username='" + cuname + "'") data = cur.fetchall()

return render\_template('DonorHome.html', data=data)

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

error = None

if request.method == 'POST':

if request.form['uname'] == 'admin' or request.form['password'] == 'admin':

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') # cursor = conn.cursor()

cur = conn.cursor() cur.execute("SELECT \* FROM regtb ") data = cur.fetchall()

return render\_template('AdminHome.html' , data=data) else:

return render\_template('index.html', error=error) @app.route("/donorlogin", methods=['GET', 'POST']) def donorlogin():

error = None

if request.method == 'POST': username = request.form['uname'] password = request.form['password']

session['dname'] = request.form['uname']

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') cursor = conn.cursor()

cursor.execute("SELECT \* from donortb where username='" + username + "' and Password='" + password + "'")

data = cursor.fetchone() if data is None:

alert = 'Username or Password is wrong'

return render\_template('goback.html', data=alert) else:

print(data[0]) session['uid'] = data[0]

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') # cursor = conn.cursor()

cur = conn.cursor()

cur.execute("SELECT \* FROM donortb where username='" + username + "' and Password='" + password + "'")

data = cur.fetchall()

return render\_template('DonorHome.html', data=data) @app.route("/userlogin", methods=['GET', 'POST'])

def userlogin():

if request.method == 'POST': username = request.form['uname'] password = request.form['password']

session['uname'] = request.form['uname']

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') cursor = conn.cursor()

cursor.execute("SELECT \* from regtb where username='" + username + "' and Password='" + password + "'")

data = cursor.fetchone() if data is None:

alert = 'Username or Password is wrong'

return render\_template('goback.html', data=alert) else:

print(data[0]) session['uid'] = data[0]

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') # cursor = conn.cursor()

cur = conn.cursor()

cur.execute("SELECT \* FROM regtb where username='" + username + "' and Password='" + password + "'")

data = cur.fetchall()

return render\_template('UserHome.html', data=data ) @app.route("/newuser", methods=['GET', 'POST'])

def newuser():

if request.method == 'POST': name1 = request.form['name']

gender1 = request.form['gender'] Age = request.form['age']

email = request.form['email'] pnumber = request.form['phone'] address = request.form['address']

uname = request.form['uname'] password = request.form['psw']

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') cursor = conn.cursor()

cursor.execute(

"INSERT INTO regtb VALUES ('" + name1 + "','" + gender1 + "','" + Age + "','" + email + "','" +

pnumber + "','" + address + "','" + uname + "','" + password + "')") conn.commit()

conn.close()

# return 'file register successfully' return render\_template('UserLogin.html')

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

if request.method == 'POST': name1 = request.form['name'] gender1 = request.form['gender']

Age = request.form['age'] email = request.form['email']

IBM-48000-1662641884

pnumber = request.form['phone'] address = request.form['address'] blood = request.form['blood'] health = request.form['health'] dname = session['dname']

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') cursor = conn.cursor()

cursor.execute(

"INSERT INTO personltb VALUES ('','" + name1 + "','" + gender1 + "','" + Age + "','" + email + "','" + pnumber + "','" + address + "','" + blood + "','" + health + "','"+ dname+"')")

conn.commit() conn.close()

alert = 'Record Saved'

return render\_template('goback.html', data=alert) @app.route("/appr")

def appr():

cid = request.args.get('cid') dname = session['dname']

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') cursor = conn.cursor()

cursor.execute(

"delete from personltb where id='" + str(cid) + "' ") conn.commit()

conn.close()

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') cur = conn.cursor()

cur.execute("SELECT \* FROM personltb where Username='"+ dname +"' ") data = cur.fetchall()

return render\_template('DonorPersonalInfo.html', data=data) @app.route("/DonorPersonalInfo")

def DonorPersonalInfo(): dname = session['dname']

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') cur = conn.cursor()

cur.execute("SELECT \* FROM personltb where Username='" + dname + "' ") data = cur.fetchall()

return render\_template('DonorPersonalInfo.html', data=data) @app.route("/newdonor", methods=['GET', 'POST'])

def newdonor():

if request.method == 'POST': name1 = request.form['name'] phone = request.form['phone']

email = request.form['email'] uname = request.form['uname'] password = request.form['psw']

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') cursor = conn.cursor()

cursor.execute(

"INSERT INTO donortb VALUES ('','" + name1 + "','" + phone + "','" + email + "','" + uname + "','" + password + "')")

conn.commit() conn.close()

return render\_template('DonorLogin.html') @app.route("/Search")

def Search():

return render\_template('Search.html') @app.route("/dsearch", methods=['GET', 'POST']) def dsearch():

if request.form["submit"] == "Submit": blood = request.form['blood']

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') cur = conn.cursor()

cur.execute("SELECT \* FROM personltb where blood ='" + blood + "'") data = cur.fetchall()

return render\_template('Search.html', data=data) elif request.form["submit"] == "SendMail":

blood = request.form['blood'] info = request.form['info']

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') cursor = conn.cursor()

cursor.execute("SELECT \* FROM personltb where Blood like '%" + blood + "%'") data = cursor.fetchall()

for item in data: sendmsg(item[4], info) print(item[4])

alert = 'Send Notication'

return render\_template('goback.html', data=alert) @app.route("/SendRequest")

def SendRequest():

session['cid'] = request.args.get('cid') return render\_template('Notification.html')

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

info = request.form['info'] did = session['cid'] print(did)

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') cursor = conn.cursor()

cursor.execute("SELECT \* FROM personltb where id='" + did + "'") data = cursor.fetchone()

if data:

bloo =data[7] print(bloo)

else:

return 'Incorrect username / password !'

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1Plasmadb') cursor = conn.cursor()

cursor.execute("SELECT \* FROM personltb where Blood like '%" + bloo + "%'") data = cursor.fetchall()

for item in data: sendmsg(item[4], info) print(item[4])

alert = 'Send Notication'

return render\_template('goback.html', data=alert) def sendmsg(Mailid,message):

import smtplib

from email.mime.multipart import MIMEMultipart from email.mime.text import MIMEText

from email.mime.base import MIMEBase from email import encoders

fromaddr = ["sampletest685@gmail.com"](mailto:sampletest685@gmail.com) toaddr = Mailid

# instance of MIMEMultipart msg = MIMEMultipart()

# storing the senders email address msg['From'] = fromaddr

# storing the receivers email address msg['To'] = toaddr

# storing the subject msg['Subject'] = "Alert"

# string to store the body of the mail body = message

# attach the body with the msg instance msg.attach(MIMEText(body, 'plain'))

# creates SMTP session

s = smtplib.SMTP('smtp.gmail.com', 587) # start TLS for security

s.starttls()

# Authentication

s.login(fromaddr, "hneucvnontsuwgpj")

# Converts the Multipart msg into a string text = msg.as\_string()

IBM-48000-1662641884

# sending the mail s.sendmail(fromaddr, toaddr, text) # terminating the session

s.quit()

if name == ' main ':

app.run(debug=True, use\_reloader=True)

**13.2 GITHUB & PROJECT DEMO LINK :**

**Github:** https://github.com/IBM-EPBL/IBM-Project-48000-1660803794

**Demolink:**

