

# PROJECT REPORT

## **PLASMA DONAR APPLICATION**

submitted by

TEAM ID : PNT2022TMID16157

DHIVYA G - 927619BIT4017

LUCKSHANA K - 927619BIT4053

MIDUNA M - 927619BIT4055

NANDHINI K - 927619BIT4063

# TABLE OF CONTENTS

<b>1. INTRODUCTION</b>	
1.1 Project Overview	
1.2 Purpose	
<b>2. LITERATURE SURVEY</b>	
2.1 Existing problem	
2.2 References	
2.3 Problem Statement Definition	
<b>3. IDEATION &amp; PROPOSED SOLUTION</b>	
3.1 Empathy Map Canvas	
3.2 Ideation & Brainstorming	
3.3 Proposed Solution	
3.4 Problem Solution fit	
<b>4. REQUIREMENT ANALYSIS</b>	
4.1 Functional requirement	
4.2 Non-Functional requirements	
<b>5. PROJECT DESIGN</b>	
5.1 Data Flow Diagrams	
5.2 Solution & Technical Architecture	
5.3 User Stories	
<b>6. PROJECT PLANNING &amp; SCHEDULING</b>	
6.1 Sprint Planning & Estimation	
6.2 Sprint Delivery Schedule	
6.3 Reports from JIRA	
<b>7. CODING &amp; SOLUTIONING</b> (Explain the features added in the project along with code)	
7.1 Feature 1	
7.2 Feature 2	
7.3 Database Schema (if Applicable)	
<b>8. TESTING</b>	
8.1 Test Cases	
8.2 User Acceptance Testing	
<b>9. RESULTS</b>	
9.1 Performance Metrics	
<b>10. ADVANTAGES &amp; DISADVANTAGES</b>	
<b>11. CONCLUSION</b>	
<b>12. FUTURE SCOPE</b>	
<b>13. APPENDIX</b>	
<i>Source Code</i>	
<i>GitHub &amp; Project Demo Link</i>	

## **PLASMA DONOR APPLICATION**

### **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 helping the 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 by an 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.

# CHAPTER 1

## 1. INTRODUCTION

### 1.1 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 are not 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 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. For instance, during COVID 19 crisis the requirement for plasma increased drastically as there were no vaccination found in order to treat the infected patients, with plasma therapy the recovery rates were 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.

### 1.2 PURPOSE

The main aim of developing this system is to provide blood to the people who are in need 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 the city 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.

# CHAPTER 2

## 2. LITERATURE REVIEW

### 2.1 EXISTING PROBLEM

#### 2.1.1 TITLE : Instant Plasma Donor Recipient Connector web application

**AUTHOR: Kalpana Devi Guntoju\*1, Tejaswini Jalli\*2**

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 that is 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 antiretroviral therapy planning for a deadly COVID19 infection, plasma therapy is an experimental approach to treat those COVID-positive patients and 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 to find out for oneself by looking at hospital records and contacting donors have been recovered, 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.

## **2.1.2 TITLE : A Web Application to Manage All Blood Donation and Transfusion Processes**

**AUTHOR: Rehab S. Ali,<sup>1</sup> Tamer F. Hafez,<sup>2</sup> Ali Badawey Ali**

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, we illustrate 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 blood banks and blood donation campaigns. The proposed application organizes and controls the whole critical processes related to blood donation, testing and storage of blood bags, and delivering it to the patient. One blood bag can save a life during surgeries 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 cannot find 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 of the 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 the blood as soon as possible to save these cases' lives. The blood bank is the pool of different 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.

### **2.1.3 TITLE :** Developing a plasma donor application using Function-as-a-service in AWS

**AUTHOR:** Aishwarya R Gowri

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, AWS Elastic 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 the requirement for plasma increased drastically as there were no vaccination found in order to treat the infected patients, with plasma therapy the recovery rates were 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 requested information in lambda.

## **2.1.4 TITLE :** Nearest Blood & Plasma Donor Finding: A Machine Learning Approach

**AUTHOR:** Nayan Das, Asif Iqbal.

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 blood donors 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 result like India has 4/1,000, and Bangladesh has 5/1000. Recently a lifethreatening 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 needed to 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, Kmeans clustering, Labeled Agglomerative 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.



## **2.1.5 TITLE :** Securing Information on a Web Application System to Facilitate Online Blood Donation Booking

**AUTHOR:** Hrishitva Patel

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. Many benefits arise from blood donation to both the donor and the blood recipients. With blood donation, cancer patients, people involved in accidents, or those battling diseases that require blood donation have access to enough blood to sustain their survival. 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 World Health Organization describes blood as the most precious gift a person can give 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 for hospitals to know which blood component they need most and thus inform the system administrator to prompt for more blood donors.

## 2.2 REFERENCES

1. Kalpana Devi Guntoju\*1, Tejaswini Jalli\*2, Instant Plasma Donor Recipient Connector web application, 2022.
2. Rehab S. Ali,<sup>1</sup> Tamer F. Hafez,<sup>2</sup> Ali Badawey Ali, 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 Facilitate Online Blood Donation Booking, 2022.

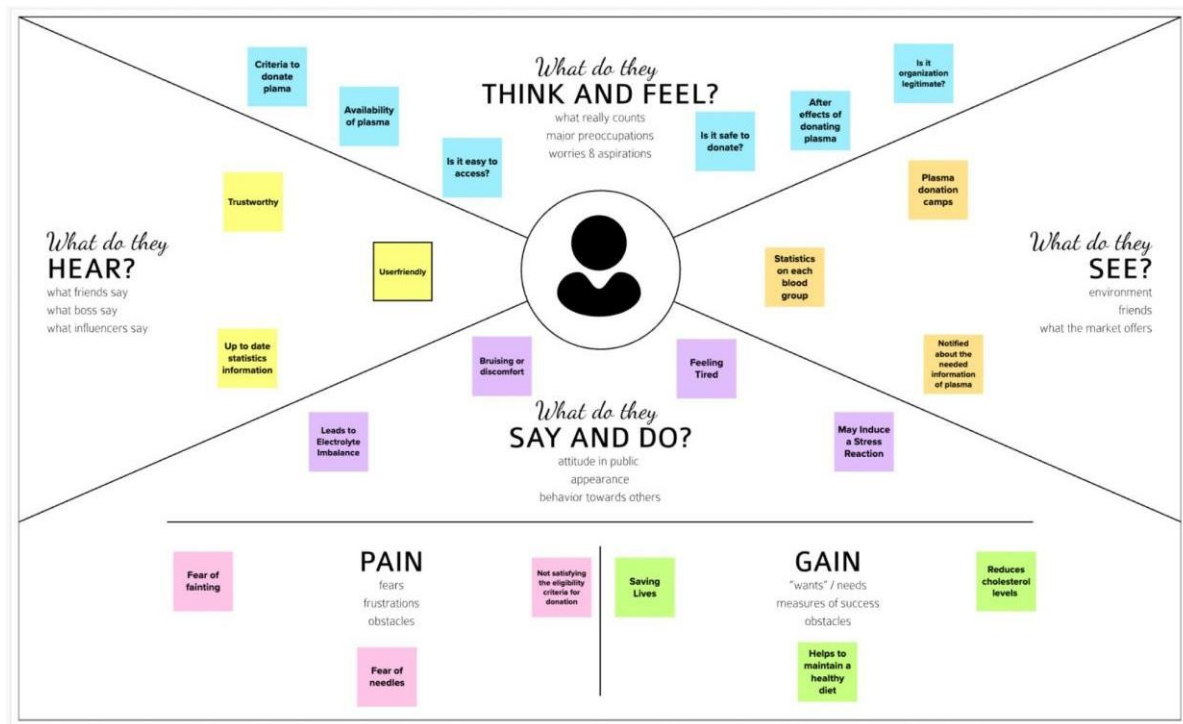
## **2.3 PROBLEM STATEMENT DEFINITION**

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 lot of 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.

# CHAPTER 3

## 3. IDEATION & PROPOSED SOLUTION

### 3.1 EMPATHY MAP CANVAS



## 3.2 IDEATION & BRAINSTORMING

### Brainstorm & idea prioritization

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.

- 10 minutes to prepare
- 1 hour to collaborate
- 2-3 prompts recommended

**Before you collaborate**

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

- 10 minutes

**Define your problem statement**

Please gather people with various, rare disorders, immune system problems, and genetic encoders. During the COVID-19 crisis, the need for plasma increased, while the number of donors significantly reduced. It would be beneficial to have the donor information so that the list of current donors might be returned in order to help the less fortunate.

**PROMPT 1**

In order to address the issue, a new application can be developed and made it available open platform.

**Key rules of brainstorming**

To run an smooth and productive session

- Stay in topic
- Encourage wild ideas
- Defer judgement
- Listen to others
- Go for volume
- If possible, be visual

**Brainstorm**

Write down any ideas that come to mind that address your problem statement.

- 10 minutes

**Group ideas**

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, one each cluster a sentence-like idea. 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**

**PROCESS**

1. Find a plasma center  
2. Make an appointment  
3. Get a blood test  
4. Donate plasma  
5. Get paid

**METHODS**

1. In-person  
2. Online  
3. Mobile app  
4. Text message  
5. Email

**BACKEND/ADMIN WORKS**

1. Data collection  
2. Data analysis  
3. Data visualization  
4. Data storage  
5. Data security

**SUPPORT**

1. Financial support  
2. Technical support  
3. Legal support  
4. Medical support  
5. Psychological support

**AWARENESS USING DIFFERENT FORUMS**

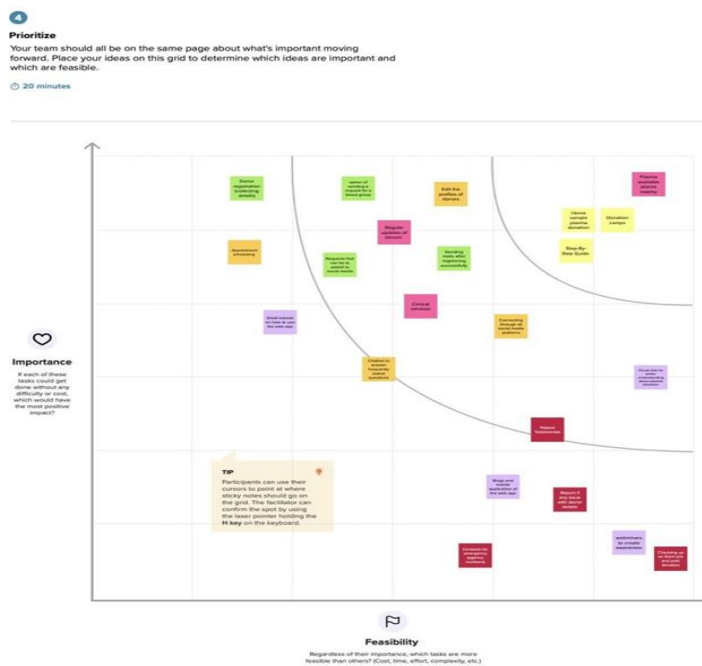
1. Social media  
2. News outlets  
3. Podcasts  
4. Webinars  
5. Conferences

**STATISTICS AND UPDATES**

1. Blood donation statistics  
2. Plasma donation statistics  
3. COVID-19 statistics  
4. Healthcare statistics  
5. Government statistics

**AGENCIES AND ORGANISATION**

1. Red Cross  
2. American Red Cross  
3. World Health Organization  
4. Centers for Disease Control and Prevention  
5. National Institutes of Health



### 3.3 PROPOSED SOLUTION

The new idea will improve the existing system and it will move from conventional desktop 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 application allows 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 for the Healthcare Sector, it also supports all the functionalities of plasma donor application.

### 3.4 PROBLEM SOLUTION FIT

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 providing each 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.

<b>1. CUSTOMER SEGMENTS</b>  Donor has to upload their blood group details for plasma donation.	<b>5. AVAILABLE SOLUTIONS</b>  Get information about all the blood campaigns.	<b>8. CHANNELS OF BEHAVIOUR</b>  It connects plasma donors and recipients through a Single and scalable platform.
<b>2. JOBS TO BE DONE / PROBLEM</b>  Ineffectual to get the details systematically	<b>6 CUSTOMER CONSTRAINTS</b>  Takes more time to get the information	<b>9. PROBLEM ROOT CAUSE</b>  There isn't a systematic approach to gather donor information rapidly.
<b>3. TRIGGERS</b>  It saves time as he can search donors online without going anywhere.	<b>7 BEHAVIOUR</b>  Search donors of suitable blood groups and contact them if needed.	<b>10.YOUR SOLUTION</b>  This system proposed here aims at connecting the donors & the patients by an 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.
<b>4. EMOTIONS: BEFORE / AFTER</b>  Before, searching for donors took a lot of time.  After, The people in need of plasma can search for the donors by giving their blood group and city name.		

# CHAPTER 4

## 4. REQUIREMENT ANALYSIS

### 4.1 FUNCTIONAL REQUIREMENT

➤ **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 module user can search donor for blood.

➤ **Donors Registration**

In this module, people who are interested in donating blood get registered in this site 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 details of donors having the same blood group and within the same city.

➤ **Notification**

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

## **4.2 NON FUNCTIONAL REQUIREMENTS**

### **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 processing demands.

### **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 7 days a week. 24 hours a day.




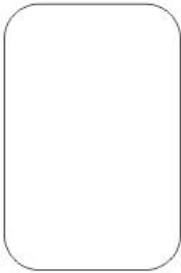


# CHAPTER 5

## 5. PROJECT DESIGN

### 5.1 DATA FLOW DIAGRAMS

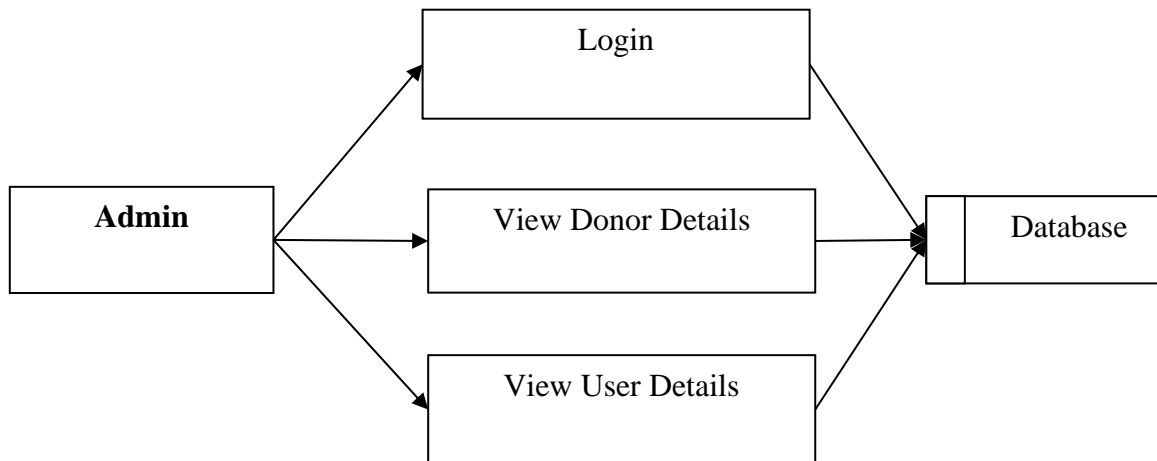
A data-flow diagram is a visual representation of how data moves through a system or a process (usually an information system). The DFD additionally gives details about each entity's inputs and outputs as well as the process itself. A data-flow diagram lacks control flow, loops, and decision-making processes.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

#### Data flow Symbols:

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

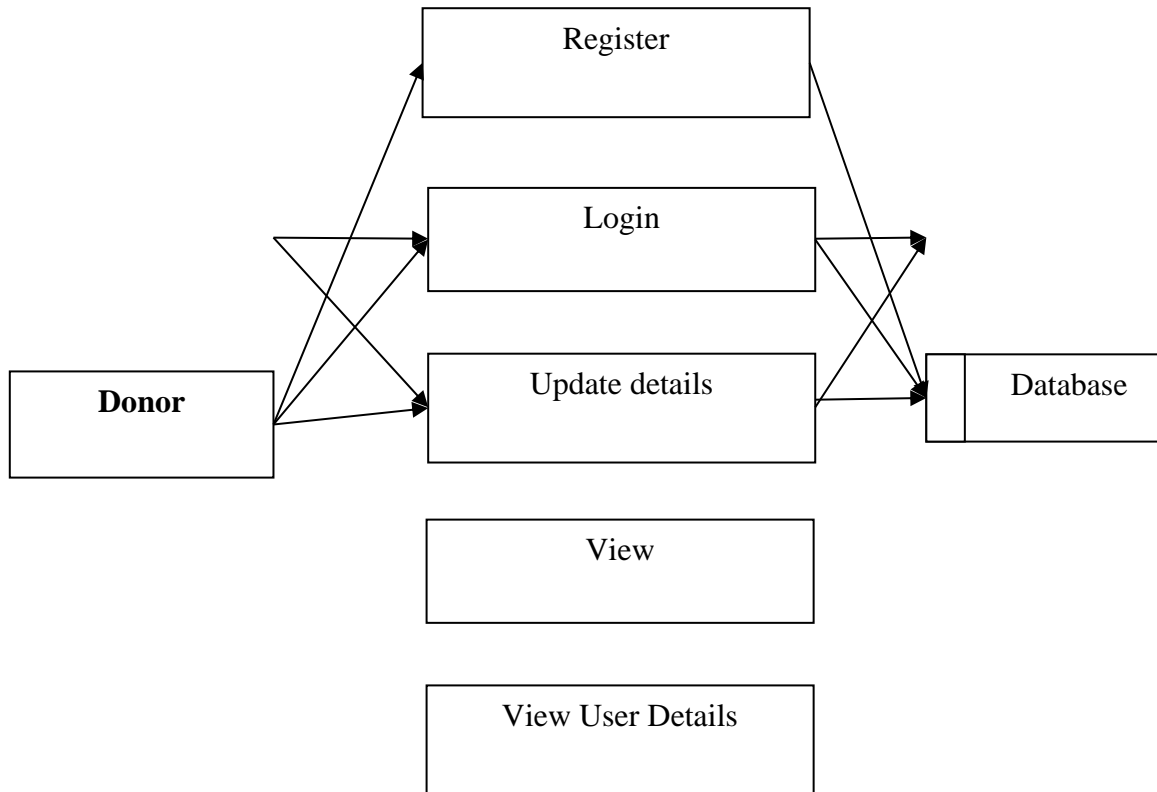
## LEVEL 0

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, and which together provide all of the functionality of the system as a whole. It also identifies internal 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.



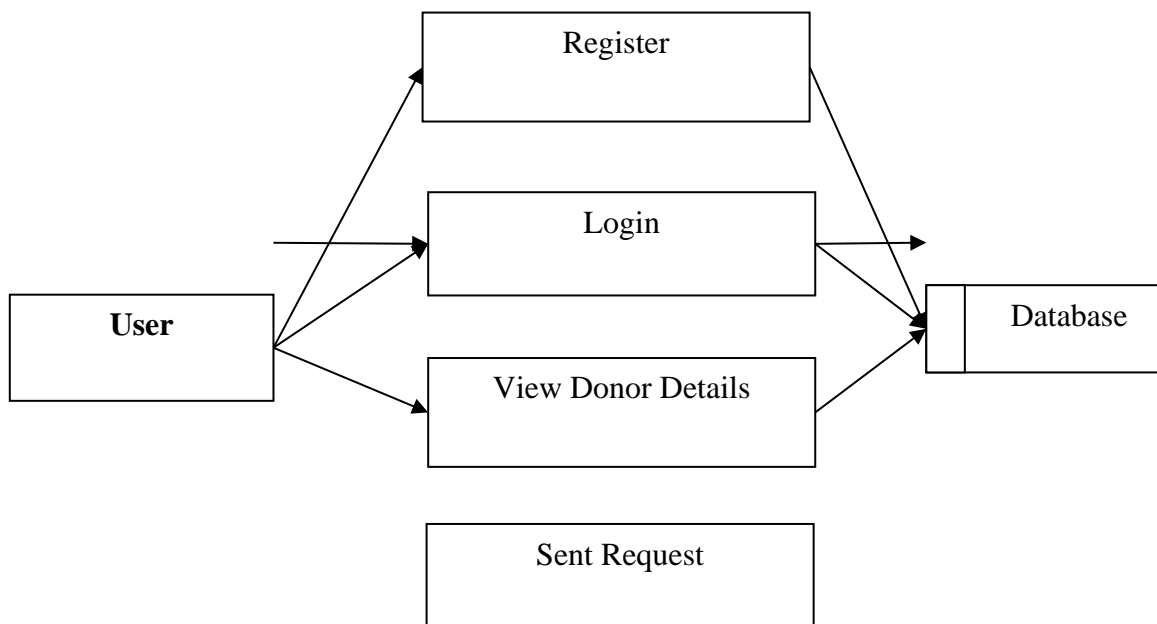
## LEVEL 1

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.

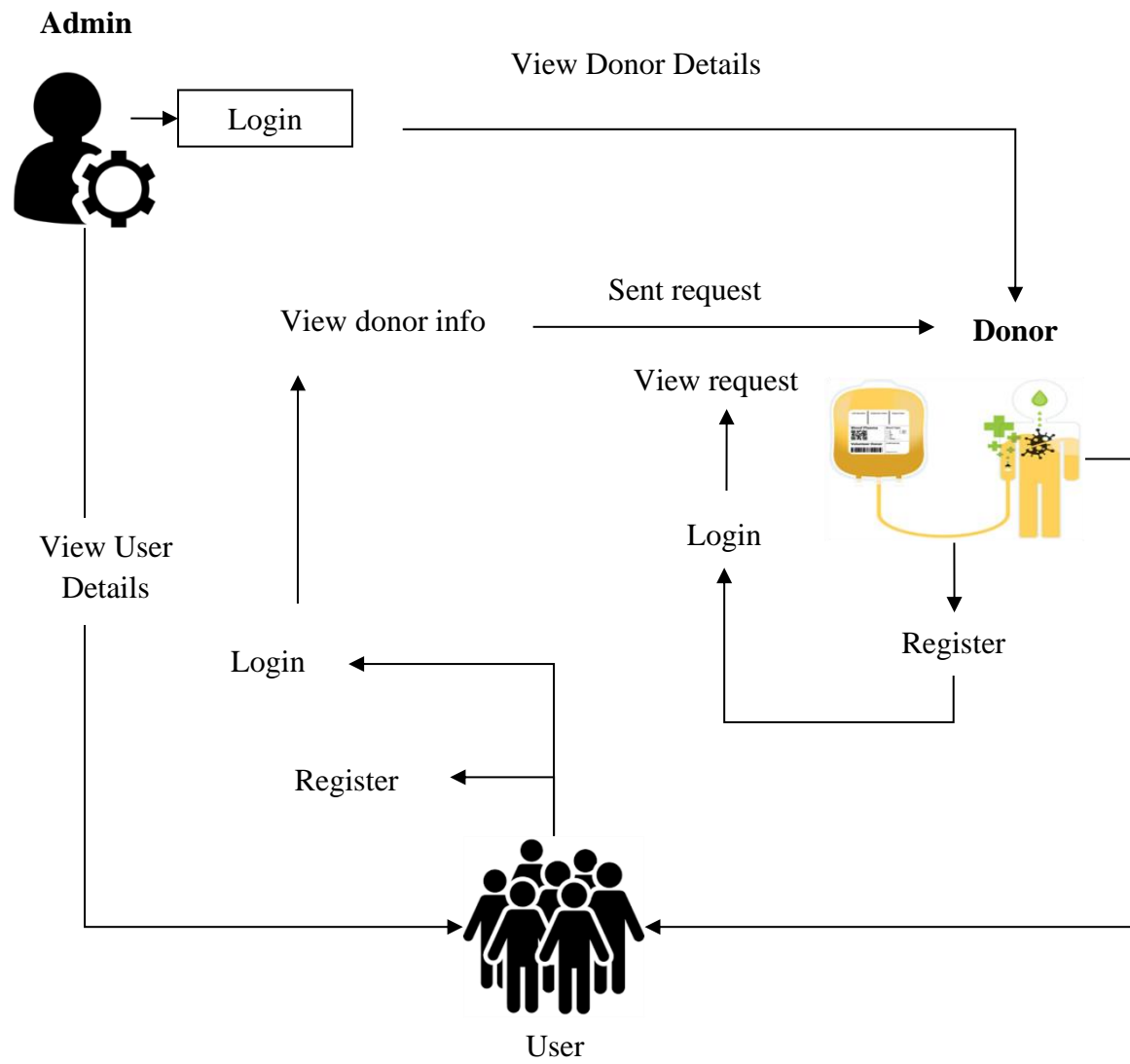


## LEVEL 2

A Data Flow Diagram (DFD) tracks processes and their data paths within the business or system boundary under investigation. A DFD defines each domain boundary and 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.



## 5.2 SOLUTION & 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.	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 successful message	High	Sprint-1
	Login	USN-3	As a user, I can log into the application by entering email & password	I can access into my Profile and view my dashboard	High	Sprint-1
	Dashboard	USN-4	As a user, I can login using my credentials and it will direct it to my dashboard	I can view and access what are the features are provided in dashboard	High	Sprint-1
Cusomr (Webuser)		USN-5	As a user, I can login using my credentials and it will direct it to my dashboard	I can view and access what are the features are provided in dashboard	High	Sprint -1
Customer Care Executive	Query	USN-6	As a user had an any query about the given requirements	I can view a query and rectify the given query	medium	Sprint-2
Administrator	Login	USN-7	As a admin ,have credentials using that they can login	They can view and modify the data in database	medium	Sprint-2
	View	USN-8	As a admin I can view plasma information	View and modify	High	Sprint-1
	Modify	USN-9	As a admin I can modify the plasma information.	Modify only if there is a false information/	Medium	Sprint-1

# CHAPTER 6

## 6. PROJECT PLANNING & SCHEDULING

### 6.1 SPRINT PLANNING & ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Donor Registration	USN-1	As a user, I can register in the donor application by entering my name, phone_no, Email id, blood group ,aadhar no	9	High	Team Lead (Dhivya G)
Sprint-1	Login	USN-2	As a admin, I can log into the application by entering email & password	9	High	Team Lead
Sprint-1	Chatbot	USN-3	As a user I can ask query in chatbot.	2	Medium	Team Lead
Sprint - 2	Confirmation	USN-4	As a user, I can receive confirmation mail.	4	Medium	Team Lead
Sprint - 2	Dashboard	USN-5	As a user, I can view dashboard and select	5	Medium	Team Member 1
Sprint-2	View Donor List	USN-6	As a user, I can view all the donor list and contact them directly	9	High	Team Lead
Sprint-2	Search Donor	USN-7	As a user, I can search for the donor	9	Medium	Team Lead
Sprint-3	About us	USN-8	As a User, I can view the about us page which contains all contact information	5	Medium	Team Member 2

### 6.2 SPRINT DELIVERY SCHEDULE

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	5 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

# CHAPTER 7

## 7. CODING & SOLUTIONING

### 7.1 FEATURE 1

```
100 <td><div align="center">{{item[5]}}</div></td>
101 <td><div align="center">{{item[6]}}</div></td>
102 <td><div align="center">{{item[7]}}</div></td>
103 <td><div align="center">{{item[8]}}</div></td>
104
105 </tr>
106 {% endfor %}
107
108 </table>
109
110 </form>
111
112 </div>
113 </div>
114
115 <!-- contact -->
116 <br> <br>
117 <div style="padding:20px;background-color:black;">
118 <p style="text-align: center;color:white;">Copyright to plasma donation hub</p>
119 </div>
120 </body>
121 </html>
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2
```



## 7.2 FEATURE 2

```
3
4
5 import ibm_db
6 import pandas
7 import ibm_db_dbi
8 from sqlalchemy import create_engine
9
10 engine = create_engine('sqlite://',
11                        echo = False)
12
13 dsn_hostname = "2f3279a5-73d1-4859-88f8-a6c3e6b4907.c3b4icw8mqrk39f8g.databases.appdomain.cloud"
14 dsn_uid = "id76283"
15 dsn_pwd = "ITOpduhup08Q8e"
16
17 dsn_driver = "{IBM DB2 ODBC DRIVER}"
18 dsn_database = "BLUDB"
19 dsn_port = "30055"
20 dsn_protocol = "TCPIP"
21 dsn_security = "SSL"
22
23 dsn = {
24     "DRIVER={0}:"
25     "DATABASE={1}:"
26     "HOSTNAME={2}:"
27     "PORT={3}:"
28     "PROTOCOL={4}:"
29     "UID={5}:"
30     "PWD={6}:"
31     "SECURITY={7};".format(dsn_driver, dsn_database, dsn_hostname, dsn_port, dsn_protocol, dsn_uid, dsn_pwd,dsn_security)
32 }
33
34
35 try:
36     conn = ibm_db.connect(dsn, "", "")
37     print ("Connected to database: ", dsn_database, "as user: ", dsn_uid, "on host: ", dsn_hostname)
38
39 except:
```

```
38
39 except:
40     print ("Unable to connect: ", ibm_db.conn_errormsg() )
41
42 app = Flask(__name__)
43 app.config['DEBUG'] = True
44 app.config['SECRET_KEY'] = "7d441f276441f27567d441f2b6176a"
45
46 @app.route("/")
47 def homepage():
48     return render_template("index.html")
49
50
51 @app.route("/Adminlogin")
52 def Adminlogin():
53     return render_template("Adminlogin.html")
54
55
56
57 @app.route("/Donorlogin")
58 def Donorlogin():
59     return render_template("Donorlogin.html")
60
61
62 @app.route("/NewDonor")
63 def NewDonor():
64     return render_template("NewDonor.html")
65
66
67 @app.route("/Userlogin")
68 def Userlogin():
69     return render_template("Userlogin.html")
70
71
72 @app.route("/PersonalInfo")
73 def PersonalInfo():
74     return render_template("DonorPersonal.html")
75
76
77
78
```

```
75 @app.route("/NewUser")
76 def NewUser():
77     return render_template("NewUser.html")
78
79
80
81 @app.route("/AdminHome")
82 def AdminHome():
83
84     conn = ibm_db.connect(dsn, "", "")
85     pd_conn = ibm_db_dbi.Connection(conn)
86     selectQuery = "SELECT * from regtb "
87     dataframe = pandas.read_sql(selectQuery, pd_conn)
88
89     dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
90     data = engine.execute("SELECT * FROM Employee_Data").fetchall()
91     return render_template("AdminHome.html",data=data)
92
93
94
95
96 @app.route("/AdminDonorInfo")
97 def AdminDonorInfo():
98
99     conn = ibm_db.connect(dsn, "", "")
100     pd_conn = ibm_db_dbi.Connection(conn)
101     selectQuery = "SELECT * from personlth "
102     dataframe = pandas.read_sql(selectQuery, pd_conn)
103
104     dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
105     data = engine.execute("SELECT * FROM Employee_Data").fetchall()
106
107
108
109     return render_template("AdminDonorInfo.html", data=data)
110
111
112
113
```

```

118
119
120 @app.route("/UserHome")
121 def UserHome():
122     user = session['uname']
123
124
125
126     conn = ibm_db.connect(dsn, "", "")
127     pd_conn = ibm_db_dbi.Connection(conn)
128     selectQuery = "SELECT * FROM regtb where username='" + user + "'"
129     dataframe = pandas.read_sql(selectQuery, pd_conn)
130
131     dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
132     data = engine.execute("SELECT * FROM Employee_Data").fetchall()
133     return render_template("UserHome.html", data=data)
134
135
136 @app.route("/DonorHome")
137 def DonorHome():
138     cuname = session['cname']
139
140
141
142     conn = ibm_db.connect(dsn, "", "")
143     pd_conn = ibm_db_dbi.Connection(conn)
144     selectQuery = "SELECT * FROM donorth where username='" + cuname + "'"
145     dataframe = pandas.read_sql(selectQuery, pd_conn)
146
147     dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
148     data = engine.execute("SELECT * FROM Employee_Data").fetchall()
149
150     return render_template("DonorHome.html", data=data)
151
152
153

```

## 7.3 DATABASE SCHEMA

```

1 Plugins supporting *.sql files found.
2
3 -- phpMyAdmin SQL Dump
4 -- version 2.11.6
5 -- http://www.phpmyadmin.net
6
7 -- Host: localhost
8 -- Generation Time: Nov 05, 2022 at 04:58 AM
9 -- Server version: 5.0.51
10 -- PHP Version: 5.2.6
11
12 SET SQL_MODE="NO_AUTO_VALUE_ON_ZERO";
13
14 /*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
15 /*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
16 /*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
17 /*!40101 SET NAMES utf8 */;
18
19 -- Database: `iplasmadb`
20
21
22
23
24
25 -- Table structure for table `admith`
26
27
28 CREATE TABLE `admith` (
29   `UserName` varchar(250) NOT NULL,

```

```

34 ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=2 ;
35
36
37 --
38 -- Dumping data for table `donorth`
39 --
40
41 INSERT INTO `donorth` (`id`, `Name`, `Mobile`, `Email`, `UserName`, `Password`) VALUES
42 (1, 'Sangeeth kumar', '9887556035', 'san@gmail.com', 'san', 'san');
43
44
45
46
47 --
48 -- Table structure for table `personlth`
49 --
50
51 CREATE TABLE `personlth` (
52   `id` bigint(10) NOT NULL auto_increment,
53   `Name` varchar(250) NOT NULL,
54   `Gender` varchar(250) NOT NULL,
55   `Age` varchar(250) NOT NULL,
56   `Email` varchar(250) NOT NULL,
57   `Phone` varchar(250) NOT NULL,
58   `Address` varchar(500) NOT NULL,
59   `blood` varchar(250) NOT NULL,
60   `Health` varchar(250) NOT NULL,
61   `UserName` varchar(50) NOT NULL,
62   PRIMARY KEY (`id`)
63 ) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=2 ;

```

```

83 --
84 -- Dumping data for table 'personlth'
85 --
86
87 INSERT INTO `personlth` (`id`, `Name`, `Gender`, `Age`, `Email`, `Phone`, `Address`, `blood`, `Health`, `UserName`) VALUES
88 (1, 'san', 'male', '20', 'sangeeth5535@gmail.com', '9486365535', 'No 16, Sammath piazza, Melapudur', 'A+', 'null', 'san');
89
90 -----
91
92 --
93 -- Table structure for table 'regtb'
94 --
95
96 CREATE TABLE `regtb` (
97   `Name` varchar(250) NOT NULL,
98   `Gender` varchar(250) NOT NULL,
99   `Age` varchar(250) NOT NULL,
100   `Email` varchar(250) NOT NULL,
101   `Mobile` varchar(250) NOT NULL,
102   `Address` varchar(250) NOT NULL,
103   `UserName` varchar(250) NOT NULL,
104   `Password` varchar(250) NOT NULL
105 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;
106
107 --
108 -- Dumping data for table 'regtb'
109 --
110

```

```

92 --
93 -- Table structure for table 'regtb'
94 --
95
96 CREATE TABLE `regtb` (
97   `Name` varchar(250) NOT NULL,
98   `Gender` varchar(250) NOT NULL,
99   `Age` varchar(250) NOT NULL,
100   `Email` varchar(250) NOT NULL,
101   `Mobile` varchar(250) NOT NULL,
102   `Address` varchar(250) NOT NULL,
103   `UserName` varchar(250) NOT NULL,
104   `Password` varchar(250) NOT NULL
105 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;
106
107 --
108 -- Dumping data for table 'regtb'
109 --
110
111 INSERT INTO `regtb` (`Name`, `Gender`, `Age`, `Email`, `Mobile`, `Address`, `UserName`, `Password`) VALUES
112 ('san', 'male', '20', 'sangeeth5535@gmail.com', '9486365535', 'no 6 trichy', 'san', 'san'),
113 ('sanNew', 'male', '20', 'sangeeth5535@gmail.com', '9486365535', 'no ', 'sanNew', 'sanNew'),
114 ('mani', 'male', '33', 'ishu@gmail.com', '9486365535', 'dgh', 'mani', 'mani');
115

```

# CHAPTER 8

## 8. TESTING

### 8.1 TEST CASES

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:

- Accurate: Exacts the purpose.
- Economical: No unnecessary steps or words.
- Traceable: Capable of being traced to requirements.
- Repeatable: Can be used to perform the test over and over.
- 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.

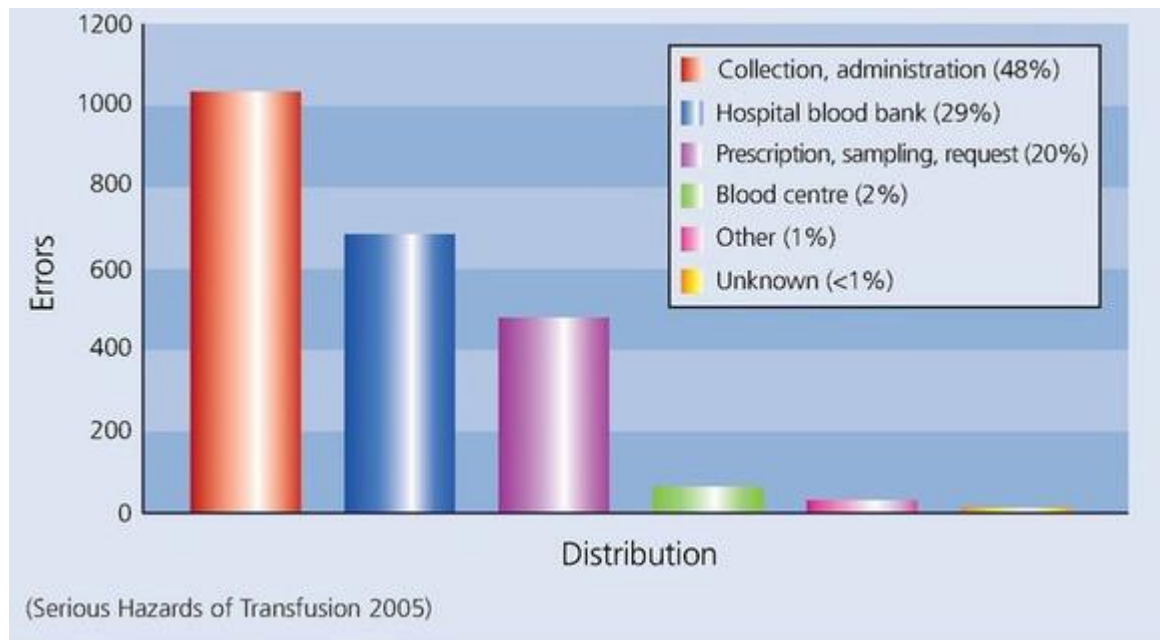
## **8.2 USER ACCEPTANCE TESTING**

This is a type of testing done by users, customers, or other authorised 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.I of the application. It is also known as user acceptance testing (UAT), operational acceptance testing (OAT), and end-user testing.

# CHAPTER 9

## 9. RESULTS

### 9.1 PERFORMANCE METRICS



# CHAPTER 10

## 10. ADVANTAGES & DISADVANTAGES

### ADVANTAGES

- It is a user-friendly application.
- The people in need of plasma can search for the donors by giving their blood group and city name.
- It saves time as he can search donors online without going anywhere.
- 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

- It cannot auto verify user genuineness.
- It is time consuming
- It leads to error prone results
- It consumes lot of manpower to better results
- It lacks of data security
- Retrieval of data takes lot of time

# CHAPTER 11

## 11. 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 the city 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 way plasma donation system can collect plasma from many donators 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.



# CHAPTER 12

## 12. 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 the system. 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.

# APPENDIX

## SOURCE CODE

```
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():

    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']
        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 personlth 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"] == "Search":
        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 personlth 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"
    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")

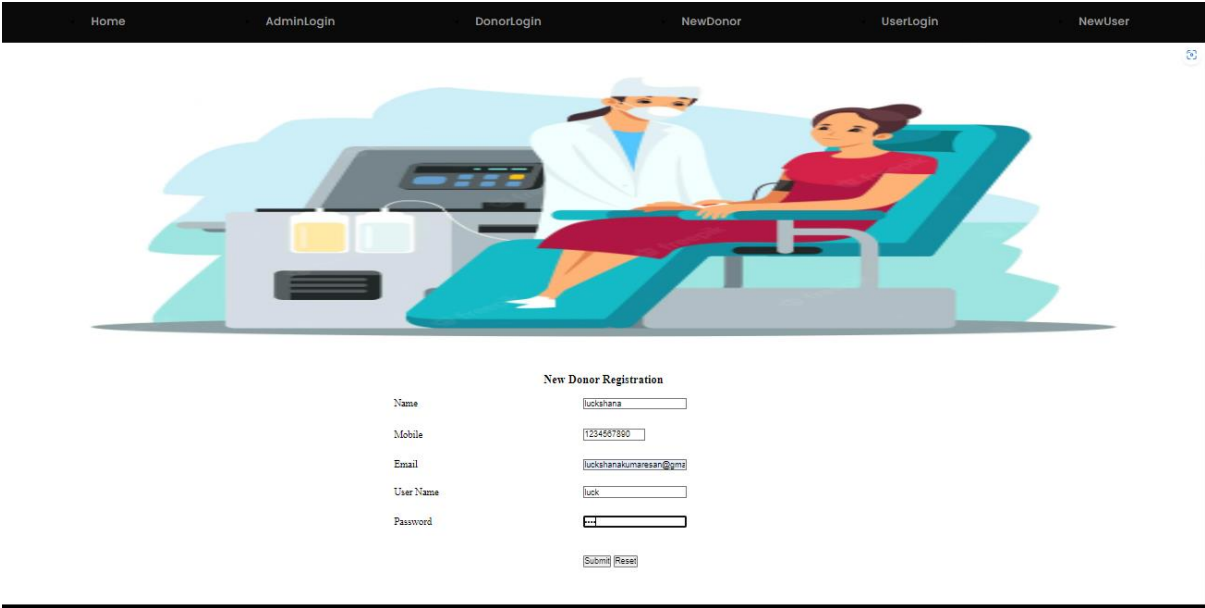
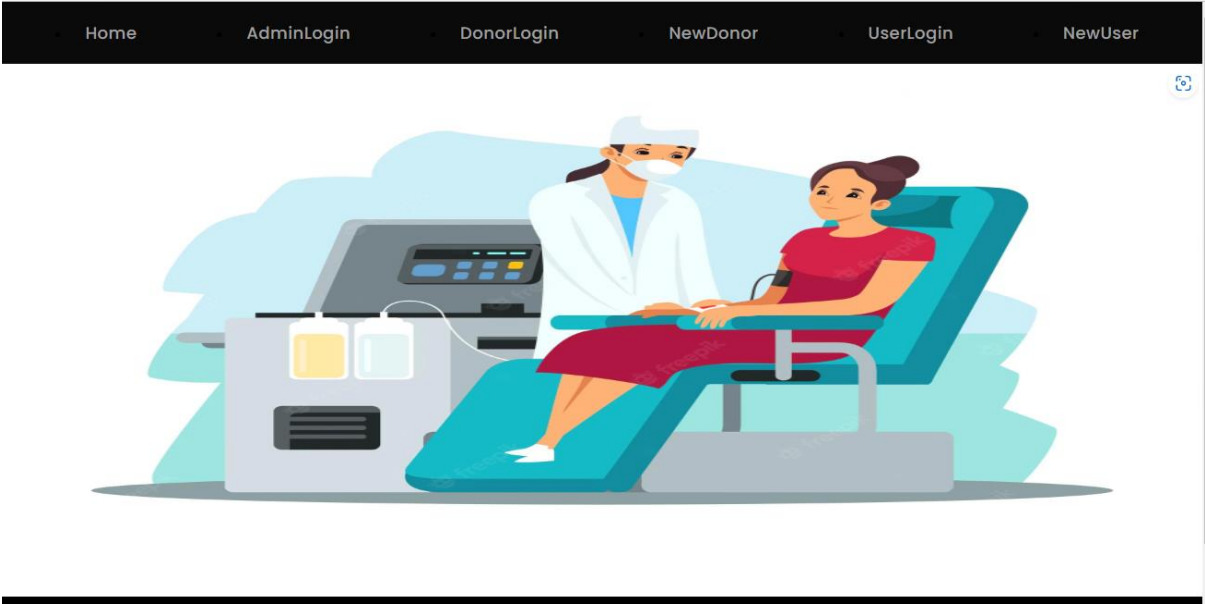
# Converts the Multipart msg into a string
text = msg.as_string()

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


# terminating the session
s.quit()

if __name__ == '__main__':
    app.run(debug=True, use_reloader=True)
```

SCREENSHOTS



HomeAdminLoginDonorLoginNewDonorUserLoginNewUser



Donor Login Here..!

User Name

luck

Password


\*\*\*\*

Login

Reset

Copyright to plasma donation hub

HomePersonalInfoReportLogout



Personal Info!

id	Name	Mobile	LandLine	Username
0	luckshana	1234567890	luckshanakumaresan@gmail.com	luck

Home

PersonalInfo

Report

Logout

Donor Personal Information

Name

Gender

Age

Email Id

Phone Number

Blood Group

Address

Health Issues

☐ Male ☐ Female

Copyright to plasma donation hub

New User Registration

Name

Gender

Age

Email Id

Phone Number

Address

User Name

Password

luckan

☐ Male ☒ Female

20

luckshanakumaresan@gmail

1234567890

Bharat

luck


\*\*\*\*

Copyright to plasma donation hub

48



[Home](#)[AdminLogin](#)[DonorLogin](#)[NewDonor](#)[UserLogin](#)[NewUser](#)



User Login Here..!

User Name

luck


Password

\*\*\*\*

LoginReset

Copyright to plasma donation hub

[Home](#)[Search](#)[Logout](#)



Personal Info

Name	Gender	Age	EmailId	Phone	UserName
lucksh	female	20	luckshanakumaresan@gmail.com	1234567890	luck

Copyright to plasma donation hub

Record Saved

[Go Back](#)



#### Search Donor

Select Blood Group

[Info](#)

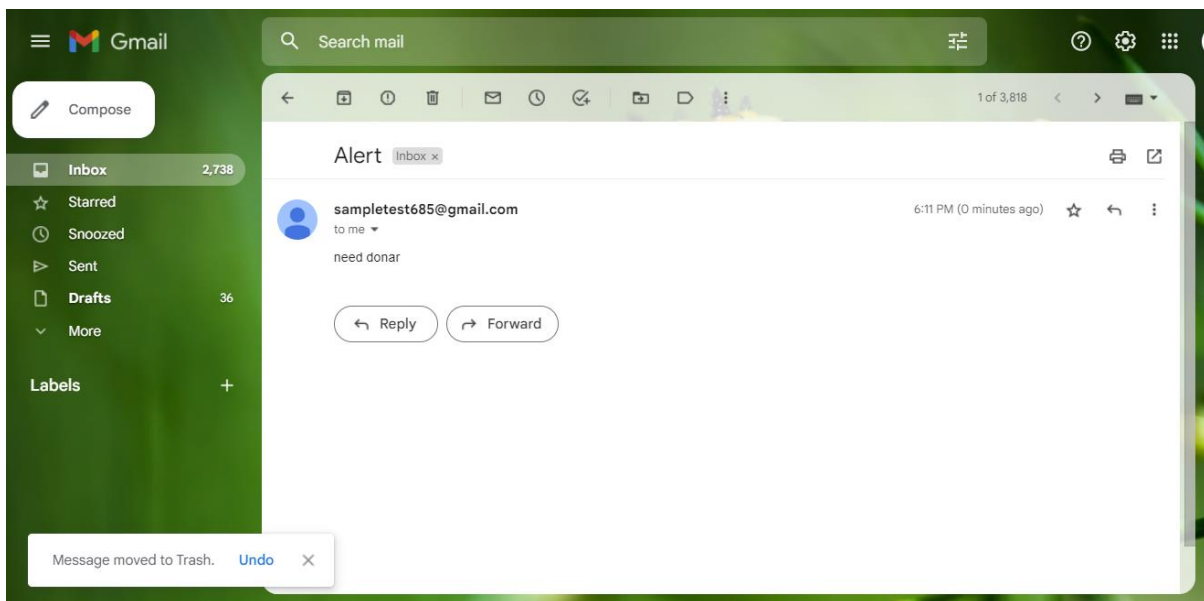
A+ ▼

[Search](#) [SendMail](#)

Name	Gender	Age	Email	Phone	Address	BloodGroup	HealthIssues	Action
luckshana	female	20	luckshanakumaresan@gmail.com	1234567890	karur	A+	nil	

Send Notification

[Go Back](#)





Admin Login Here..!

User Name

Password

User Information!

Name	Gender	Age	EmailId	Phone	UserName
suryaprakash	male	20	surya@gmail.com	6379585139	surya
luckah	female	20	luckshanakumaresan@gmail.com	1234567890	luck

**GITHUB**

<https://github.com/IBM-EPBL/IBM-Project-17445-1659670685>

**PROJECT DEMO LINK**

[https://drive.google.com/drive/folders/1gb7Nwl3FBDA7wzNOJk23h0\\_e0oEqMsGG](https://drive.google.com/drive/folders/1gb7Nwl3FBDA7wzNOJk23h0_e0oEqMsGG)