PANACEA
(A Blood Donation App**)**

A PROJECT REPORT

Submitted in partial fulfilment of the Requirements for the Degree of

MASTER OF COMPUTER APPLICATION

by

SURUCHI SINHA
&
APOORVA SRIVASTAVA

(Univ. Roll No.: 1900290140040 & 1900290140008)

Under the Supervision of

Dr. Vipin Kumar
Professor, KIET Group of Institutions, Ghaziabad

Submitted to

DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions,
Ghaziabad Uttar Pradesh-201206
DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY, LUCKNOW (AUGUST ,2021)

DECLARATION

We hereby declare that the work presented in this report entitled “PANACEA", was carried out by us. We have not submitted the matter embodied in this report for the award of any other degree or diploma of any other University or Institute.

We have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, results, that are not our original contribution. We have used quotation marks to identify verbatim sentences and given credit to the originalauthors/sources.

We affirm that no portion of our work is plagiarized, and the experiments and results reportedin the report are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, we shall be fully responsible and answerable.

Name: Suruchi Sinha	Name: Apoorva Srivastava
Roll No.: 1900290140040	Roll No.: 1900290140008
Branch: MCA	Branch: MCA
(Candidate Signature)	**(Candidate Signature)**

CERTIFICATE

Certified that **Suruchi Sinha** **(**Roll No: 1900290140040) and **Apoorva Srivastava** **(**Roll No: 1900290140008) has carried out the project work presented in this thesis entitled **"PANACEA"** for the award of **Master of Computer Application** from Dr. APJ Abdul Kalam Technical University, Lucknow under my supervision. The thesis embodies results of original work, and studies are carried out by the students themselves and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University.

Dr. Vipin Kumar **External Examiner**

(Internal Examiner)

Professor

Dept. of Computer Applications

KIET Group of Institutions, Ghaziabad

Date:

ACKNOWLEDGEMENT

Success in life is never attained single handedly. First of all, we would like to thank the almighty, who has always guided us in the right way to make this project a great success.

The satisfaction that accompanies that the successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success. Our deepest gratitude goes to our thesis supervisor, **Dr. Vipin Kumar** for his guidance, help and encouragement throughout our Project with his enlightening ideas, comments, and suggestions.

Words are not enough to express our gratitude to **Dr. Ajay Kumar Shrivastava**, Professor and Head, Department of Computer Applications, for his insightful comments and administrative help at various occasions.

Fortunately, we have many understanding friends, who have helped us a lot in many critical conditions.

Finally, our sincere thank goes to our family members and all those who have directly and indirectly provided us moral support and other kinds of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

Suruchi Sinha & Apoorva Srivastava
MCA 4th Sem

TABLE OF CONTENTS

LIST OF ABBREVATIONS

IDE- Integrated Development

EnvironmentJSON- Java Script Object

Notation

OTP- One Time Password

XML- Extensible Markup Language

UI- User Interface

API- Application Programming Interface

REST- Restfull API

GPS- Global Positioning System

DOM- Document Object Model

SDK- Software Development Kit

LIST OF GRAPHS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION TO PROJECT

The blood is a lifesaver if there ever emerge any events of the emergency needs. The errand of the blood bank is to get the blood from the different types of people caning to donate the blood, to manage blood bundles' database and to provide the needed blood in between of the need to the mending donation if there emerge any events of emergencies. The issue here isn't the lacking number of the people caning to donate the blood, but finding any enthusiastic supporter/donor at the advantageous time. We have to make an arrangement of people who may help each other in between of an emergency. The android application in this project prompts the updates of the information for the supporters/donors where the chief gets the entire information about the blood bank system. Give away/donor can then be incited into entering a man's purposes of information, like name, email, phone number, and the blood group. At the usual time of any blood need, one can quickly check the red blood blank android application database or recuperating database planning related or explicit the blood gathering and the interface with them through android application. A noteworthy number of people caning to donate the blood can be pulled into using this android application. Since about every one now carries mobile phones with them, to ensure minute region surveillance and correspondent changes. Only an enrolled individual self, with the capacity to give away the blood, can have the ability to get to the society. This report gives an android application a system which is planned to give away most information needed for the blood social circle or affair which is reliably asked for an advancing reason. The system depicts the convenience and the ease to contact with different suppliers and dejects for different blood social events. The android application acquaints on the insightful contraptions with the assurance of the arrival of a greatest possible no. of the red blood benefactors within the country.

1.1.1 OBJECTIVE OF PROJECT

This task is defined for people who are eager to give away their portions of blood to the patients in need. With the help of this frame work we can discover a contributor for the correct blood classification and it becomes simpler to make the association among give away/donor and also the blood bank specialists without much of a stretch. The central target of making this product is a formal technique of an online blood bank and the arousing benefactors with an end goal to bank the blood. Here, we are endeavoring to keep up every one of the information of the contributor which is effortlessly defensible to the specialists who make it simple to identify the gives away/donor.

STATEMENT OF THE PROBLEM:

- Scarcity of rare blood group.
- Unavailability of blood during emergency.
- Less awareness among people about blood donation.
- Deaths due to lack of blood during operations.

1.2 PROJECT DESCRIPTION

This Project PANACEA is basically to reduce the time lag between searching for and contacting different blood donors across the country in just few clicks. Donor will be prompted to enter an individual's details, like name, phone number, and blood type. Thereafter your contact details will be saved in the database. At the urgent time of a blood requirement, one does not need to register but he can quickly check for contacts matching a particular or related blood group and reach out to them via phone call through the app. Using this app in case of emergency, a large number of blood donors are attracted using an android application. Scope of this app is to reduce the time lag between the searching and contacting for blood donors to a few seconds which can be of utter use to the people of this country in saving more and more lives. Since almost everyone carries a mobile phone with him, it ensures instant location tracking and communication. Not only a registered person, with willingness to donate blood, will be able to access the service but anyone who installs it on his mobile phone can search for a blood donor or a blood bank without even registering on the app. The sole purpose of this project is to develop a computer system that will link all donors and blood banks. the system will help control a blood transfusion service and create a database to hold data on donors and blood banks in each area or city. Furthermore, people will be able to register as donors and thus receive a call from their local clients who needs blood to donate blood in cases of need. the app will help develop public awareness amongst its visitors of the hospitals' need for blood in order to supply the appropriate donors.

1.2.1 Advantages of Project

- Provides a paperless donor room.
- Real time information form collection to testing and use of blood and blood products.
- Availability of blood in emergency.

1.3 PROBLEM DEFINITION

Disregarding the potential accessibility of the blood benefactors not over 5% of the aggregate Indian populace give away the blood advancement & medical procedures in science has expanded the blood request. Also, the blood-contributors more often than not don't come to think about the beneficiaries needing the blood. These reasons arouse us to make up a more proficient framework which can be help with investigating data to the individuals about the present the red blood bank frameworks. As the current and the android application comes up shortly on the dead of broadcasting posts over the internet based on life, this android application defeats all the disadvantages by presenting the dead of sharing posts & messages on online networking for a social cause.

1.4 SCOPE

The android application concentrated more on the acquisition and convey the investigation extraordinarily accentuated the creation & execution of an electronic administration data 15 framework which mechanized the red blood contributor information procurement a dispersal of results. This thus can straight forwardness & accelerates the arranging, basic leadership procedure of the opportune, secure, classified & dependable reports.

1.5 METHODOLOGIES

Android Studio: Android Studio is an authority incorporated advancement condition (IDE) for an android application improvement, in the light of the android Studio is planned explicitly for the android improvement accessibility for the download on Windows & Linux, android application being Google's essential development environment for the local android app advancement. Android Studio also offers adaptable Grade-based form framework, the code formats to enable the user to fabricate the regular android app highlights & the rich format editorial manager with the help for the intuitive topic altering & also worked help for the Google Cloud Platform, making it simple to coordinate with the Google Cloud Messaging android app Engine & considerably more android Studio as good as ever interface structure point of view where you can see the interface you are taking a shot at and its related segments. android Studio give away different UI instruments to help you with making designs, executing style subjects, & making realistic or content assets for your android application. The android manufacture framework the toolbox you use to assemble, test, run & bundle your android applications. The construct framework can keep running as a coordinated android application from the android studio menu & freely from the direction line.

JSON: JSON sets for the JavaScript Object Notation. It is a self-governing data exchange format & is one of the best alternatives for XML Android provides 4 different classes to manipulate the JSON data. These classes are JSON Array, JSON Object JSON Stringer & JSON Tokenizer. The first step is to recognize the fields in the JSON data in which you are concerned with is an important data exchange format. It stores the data in the key & value pair. Compared to XML, JSON is comparatively simple & easier to read.

JSON – Elements: A JSON file consist of many components. Here the table defines the components of a JSON file & their corresponding description –

Table no. 1.6.1 Elements of JSON

S no.	**Component & description**
1	Array ([) In a JSON file, square bracket ([) represents a JSON array.
2	Objects ({) In a JSON file, curly bracket ({) represents a JSON object.
3	Key A JSON object contains a key which is just a string. Pairs of key/value make up a JSON object.
4	Value Each key has a value which could be string, integer or double etc.

CHAPTER 2

2.1 LITERATURE REVIEW

P Priya, V Saranya, Shabana, Kavitha Subramani [1], has suggested an all-encompassing web android application to opportune & refresh the data with respect to all the contributors, the acceptor & the patients amongst which the manager gets the entire data about the red blood donation center administration framework. Also, the proposed work has enough security, to ensure the contact as the subtle elements of the gives away/donors for the web android application where it tends to be abused by the outsiders. Likewise, it keeps up the measure of each accessible blood group around, the load of a specific blood, amass lower than the needed sum and then the suggested technique advises the benefactors to give away the blood, not with string web application, an android versatile android app proposed to look through the givers/donors who are accessibly close-by in the amid of the crisis cases, for example, mischance's. The electronic android application is promptly adaptable, effective & versatile in order to meet the intricate need of the blood donation center which scratches the facilitators for the social insurance area. A Survey Paper on E-blood Bank and the idea to use it on the Smartphone. The blood is a critical angle for each one of the single living things ends up being to be the life-saving segment event of the crisis necessity. None of the online the blood donation center still offers the immediate contact info among the contributors & the red blood donation center i.e. the givers. This is the real downside of the current framework. Existing frameworks are now tedious; and require more inflated. The optimization of the blood donor information & management system by Technopedia talks about the blood saver of every current life who should arise an occurrence of crisis needs.

Tushar Pandit, Satish Niloor &A.S. Shinde, [2] has presented examination between the existing framework & the enhanced framework. The new thought can also enhance the current framework & can move from the ordinary work area framework to the portable framework. E-blood donation centers incorporate the blood 18 donation center computerization framework. The fundamental motivation behind E-donation center is to interconnect all the red blood donation centers of the region into a solitary system, i.e., an android application oval, stockpiling & flow of different live information & the data by utilizing the calculation innovation. The information which puts away on the registering gadgets can then help the general society for a simple access to the blood accessibility status of the red blood donation centers on finger tips so that he can put out a dam or tell specifically there's a blood aggregate adjacent to the blood donation center spare is a profitable life.

Vikas Kulshreshtha Research Scholar, Dr Sharad Maheshwari [3] has presented an audit of the primary highlights, the benefits & the negative marks given away by the current Web-Based information System for the blood Banks. The blood is all around perceived as one of the most valuable components which continues to life spares on endless lives over the world on an assortment of conditions. The blood donation centers place is structured particularly for a capacity of the blood & the blood items. The term "blood bank" regularly mentions to a partition of a healing center lab wherever the capacity of the blood android application & where the android application testing is performed to lessen the danger of the transfusion related occasions. Large coolers hold these samples at a steady temperature & they are also accessible at a moment's see. The blood donation center administration data framework offers functionalities to android application to access the gives away/donor records gathered from the different parts of the nation. This empowers observing of the outcomes and the execution of the red blood bank action to such an extent which is important & also quantifiable destinations of association which can be checked. They are giving a productive pursuit of who needs the blood based on their very own city as quickly as could be expected under the respective circumstances.

2.2 EXISTING SYSTEMS

Blood Connect:

The blood can't be produced production lines can just originate from liberal contributors. To take into account this interest, The blood connection had been propelled by1st April, 2010 (as an undertaking under NSS IT, Delhi) with an unparallel goal of taking care of 19 the corresponding issue of the red blood lacking the nation. According to WHO information, India faces a deficiency of 3 million the red blood units. This deficiency can without much of a stretch be killed just an extra 2% of India's childhood give away the blood. Blood Connect goes about the blood giveaway/donors with the individuals who require the blood adolescent run association & give away free help & exceptionally attempts to focus on poor people & the penniless.

Since commencement, the association has grown a great deal as far as working for this reason. The blood Connection has built up a 360degree answer for the issue of the red blood lack show spins around 4 central focuses:

• Ensuring consistent, adequate the red blood supply to the blood donation centers.

• Improving mindfulness.

• Helping those in need.

• Establishing a system of youth E-blood bank:

This android application encourages you discover individuals giving the blood your general vicinity. You can get touch with them through telephone number or email address. You can see the area of client guide & enlist yourself with android application, you can get push notice the event which you’re the blood aggregate matches with the need of the blood. You can discover adjacent doctor's facilities & access them Features:

• Find give away/donor with the particular blood group & with the separate states & urban areas.

• Provide notice: This can enable you to know all the identities having a similar the red blood amassing your neighbourhood.

• Find the close-by healing centers maps. • Provided helpline numbers there should be able to arise an occurrence of crisis.

CHAPTER 3

SYSTEM DEVELOPMENT

3.1 DESIGN & IMPLEMENTATION CONSTRAINTS

Making a UI which is both effortlessly powerful & traversable can be troublesome test. The essential requirement may be which we might be making an android application for the portable stage. The real limitation can be the goals & can be restricted manage estimate as the android application is for the versatile environment. The other imperative regards to the versatile can be process power & restricted memory. Our undertaking is just intended to be the responsive administration of the capacities which manages the colossal data regards to the healing centers, the red blood donation centers, contributors, patients, stock administration & can be created with proficiency.

3.2 PREREQUISITES

• Android Studio
• Google-services.json record produced for the android application.
• Android SDK (This rendition composed for the SDK 25 as a target)
NB: A steady web associationis required first.

3.3 ASSUMPTIONS & DEPENDENCIES

3.3.1 ASSUMPTIONS:

• GPS can follow the present area of the client precisely.
• The proper administration of the blood over the portable servers.
• The database can work for an android application and then store the records of the respective user.

3.3.2 DEPENDENCIES

• The login page comprises of Username & Password which is stored in the database.

• This venture also relies on the REST web administrations.

• It is suitable for any sort of individual.

3.4 SYSTEM FEATURES

3.4.1 MOBILE ANDROID APP OVER WEB BASED ANDROID APPLICATION

The electronic framework's in India for the blood bank is not accessible as indicated by the client's choice as they are conveyed on web in which are convenient to use as they can be gotten to the account of crisis or any injury situations. The mobility gives away by the android put framework together which is open with respect to the portable through android application are accessible in a hurry

3.4.2 HIGHLY TRAINED DATASET

Dataset Training information stockpiling, control, benefit and so on., give always an incredible supporting model to the android application. The separated dataset which is consistently organized encourages to the end client to get to this data and make natural product from. The key highlights & data are date of birth of the gives away/donor & additionally persistent, the blood group of the patient/benefactor, date of last the red blood gave, portable number, address with city and state, email give away the criteria agreeing doctor's facility, the red blood donation center astute development scan for the rundown & recovery of information.

3.4.3 INVENTORY MANAGEMENT

There are many android applications which stores & the gives away of the data of both the client resembles benefactor of the blood & the blood pack. There is blood donation center of the society are moderately expansive with regards to their physical and one of the angles. So, the stock administration of the blood bank systems exceptionally important the data to be taken care of extremely huge. Generation of report for the inventories utilized the android application bought to be the legitimately done, inventory corporate the patient, give away/donor, clinics, the red blood donation centers, stock & the searcher stock. Secondly there is the blood sacks stock bought to be disposed of naturally once has lapsed & the expiry android application approach are just 21 days from the day of stock included, so android application record bought to be kept for the terminated the blood can spare manual work of the client who embeddings the record into database.

3.4.4 SYSTEM SECURITY

The blood Bank's executive frameworks encourage the android application with cutting edge of the security highlight like android application oval & confirmation for web base android application. This security benefit gives away by the android application were the clients already enrolled to the framework and profile related data put away and kept up for the further android application oval & checks of the client.

3.4.5 ALERT SYSTEM

If there should be an occurrence of crisis were the accessibility of the bloods' known & the time the significant worry for client as they can't physically look for required the red blood from every healing center & the blood donation centers. So, the android application gives away an alarm catch which when the android application of the data as indicated by the best pursuit credit showed to the client profile with the closest clinic & the blood donation center with course to the goal.

3.5 MODEL DEVELOPMENT

3.5.1 CLASS DIAGRAM

The class diagram here is used to show the different objects in the red blood bank system, their attributes, their operations & the relationships among them.

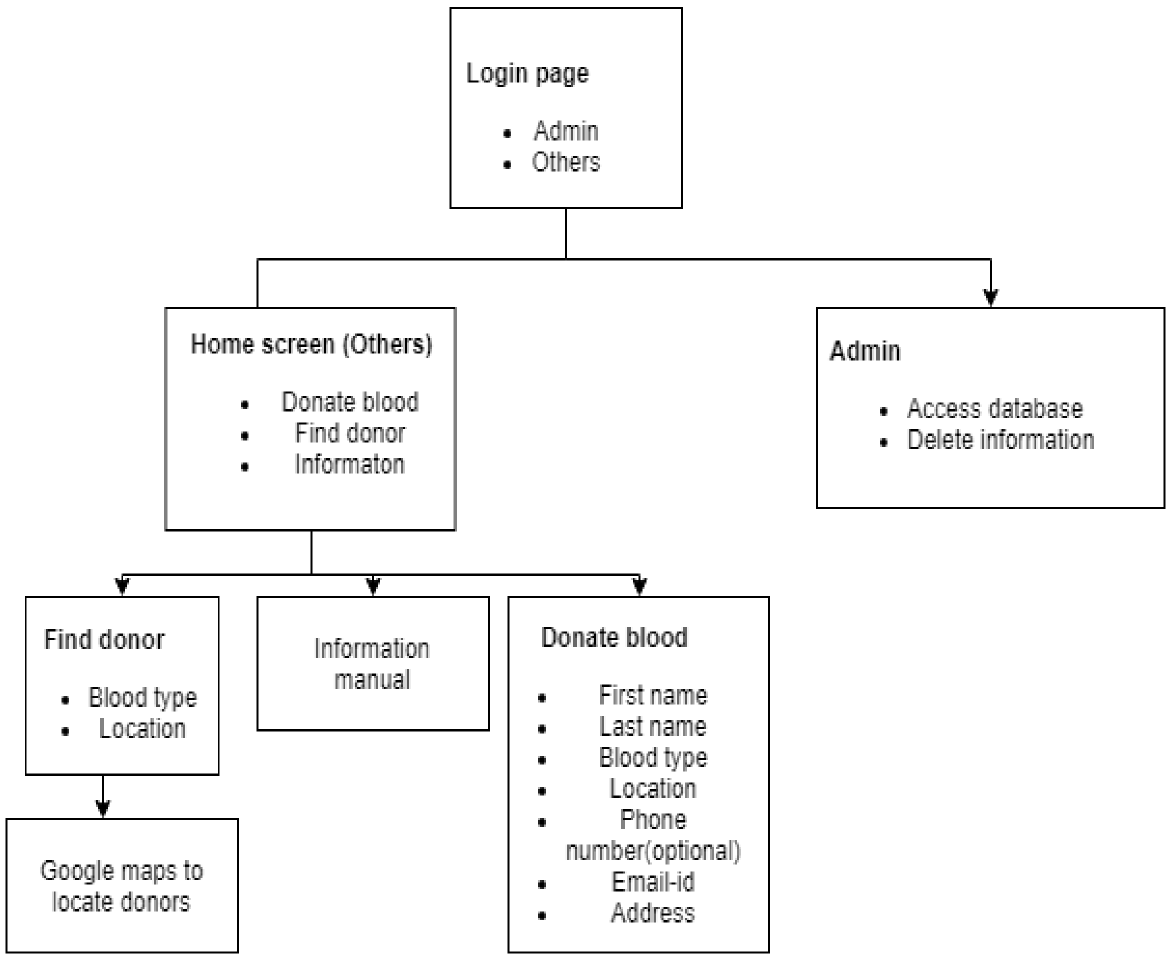


Fig 3.5.1. Class diagram for PANACEA (Android app)

CLASSES:

• LOGIN PAGE:

Attributes-:

Admin

Others

• HOME SCREEN:

Attributes -:

Donate Blood

Find Donor

Information

- ADMIN:
  - Operators -:
    - Access database ()
    - Delete information ()
- FIND DONOR:
  - Attributes -:
    - Red blood type
    - Location
- DONATE BLOOD:
  - Attributes -:
    - First name
    - Last name
    - Red blood type
    - Phone

3.5.2 FLOWCHART

The flow chart here, depicts the overall flow of the red blood bank android application, i.e., all the switches & linkages between the activities in the procedural format.

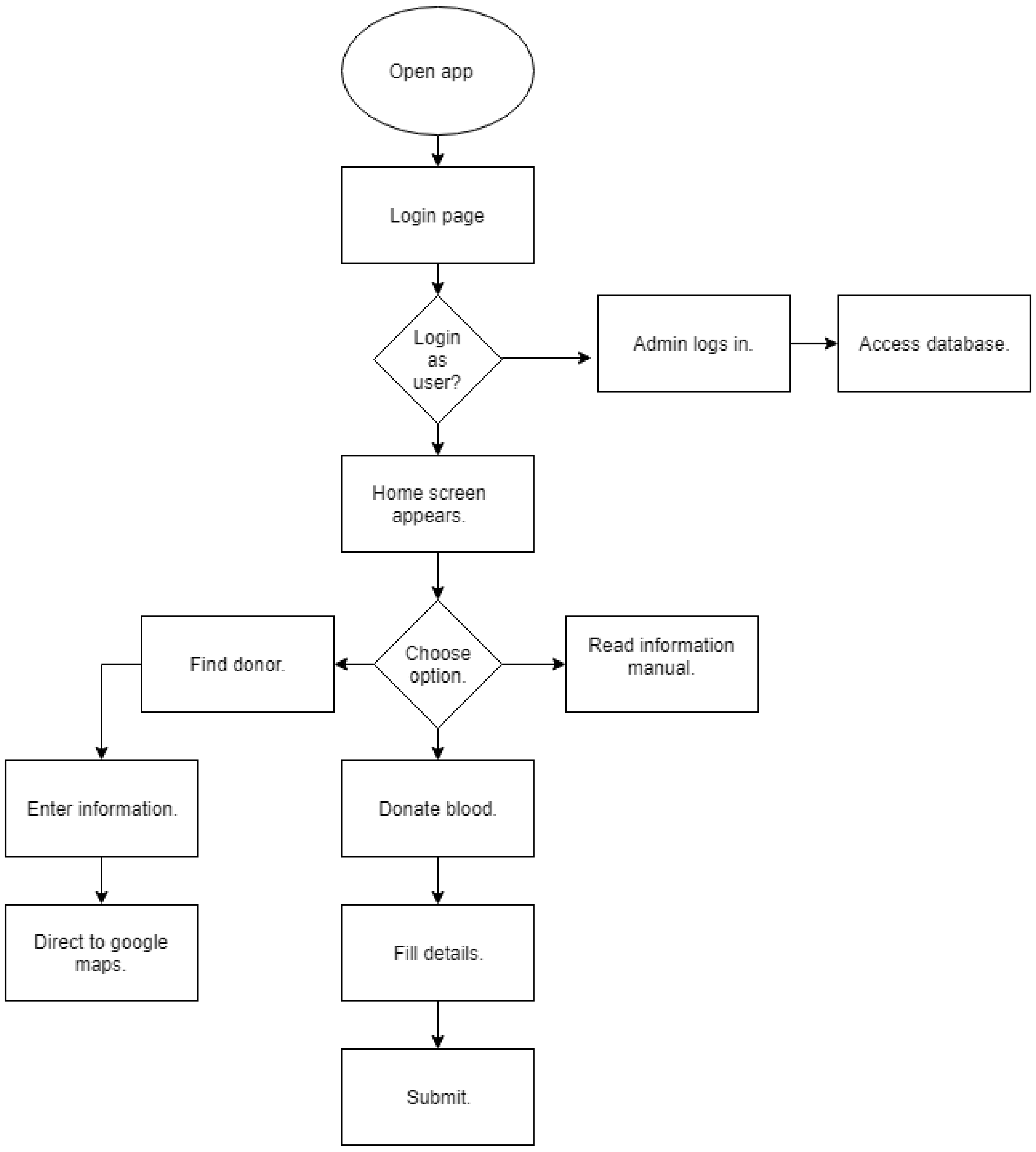


Fig 3.5.2. Flow diagram for Blood Bank (Android app)

3.5.3 ACTORS-

1. DONOR:

   The donor is the person willing to donate red blood to those in need, therefore, he/she accesses the login & then fills the donor form consisting of vital information about the donor including, name, phone no. etc. Later on the donor is connected to the group message server where he can always tell about availability.

2. PATIENT:

   The patient is the person who needs blood. This person logs in using OTP & then shares the requirement by going into the need red blood activities. He then gets access to the 28 donors' list from the database as per the requirements from which he can further access Google maps location.

3. ADMIN:

   The admin is the person who overall manages the application i.e. makes & deletes entries to the database etc.

ACTORS VS. USE CASES

1. DONOR:
   - Open application
   - Login
   - Donor Form
   - Group message

2. PATIENT:
   - Open application
   - Login
   - Need blood
   - Group message
   - Access donors' list

3. ADMIN:
   - Open application
   - Login
   - Access donors' list
   - Manage database

3.5.6 ANALYTICAL THEORY

In framework structure of the red blood donation center administrations, the exercises & stream of operation ought to be considered for satisfactory usage of room. The practical arrangement of the blood donation center administrations consequently based on the ways taken by the give await/donors, the red blood unit, the red blood tests & material. This likewise required for accommodation to the hospital and any resulting adjustments should be endorsed once more.

DONOR COMPLEX: This gives away/donor complex of a contributor holding up territory, benefactor enlistment, & medicinal examination live with fundamental testing, the red blood accumulation zone, aphesis zone, contributor rest room & kitchen/wash room. The stream of gives away/donors should uniform& plainly characterized to characterized to maintain a strategic distance from superfluous movement the passage ways.

THE RED BLOODSTORAGE: Starting stockpiling of the red blood ought to be the region of where give away/donor bloods gathered, this called as the quarantine stockpiling. After every one of the tests are performed, the red bloods gathered & put away region of the issue zone.

COMPONENT PREPARATION: The territory for the red blood segment ought to be close to solute capacity. The segment research facility ought to be perfect, sans dust & sufficiently bright. Administration of the red blood part planning need an extraordinary permit from the DCI.

CONTACT MAINTAINENCE: In order to donate the blood, the donor must be caning to go to the nearby collection dispensary regularly after 4 months but through this model the donor knows about the exact need & can be available then.

CHAPTER 4

ALGORITHMS

4.1 INTRODUCTION

With the incrementing usage of social media across the entire world, there's also an increased probability forth success in the usage of the red blood bank android application. This android based android application can play a vital role in saving the lives of human beings provides a means of communication in between the blood seekers, the blood people willing to donate thered blood with our app PANACEA.

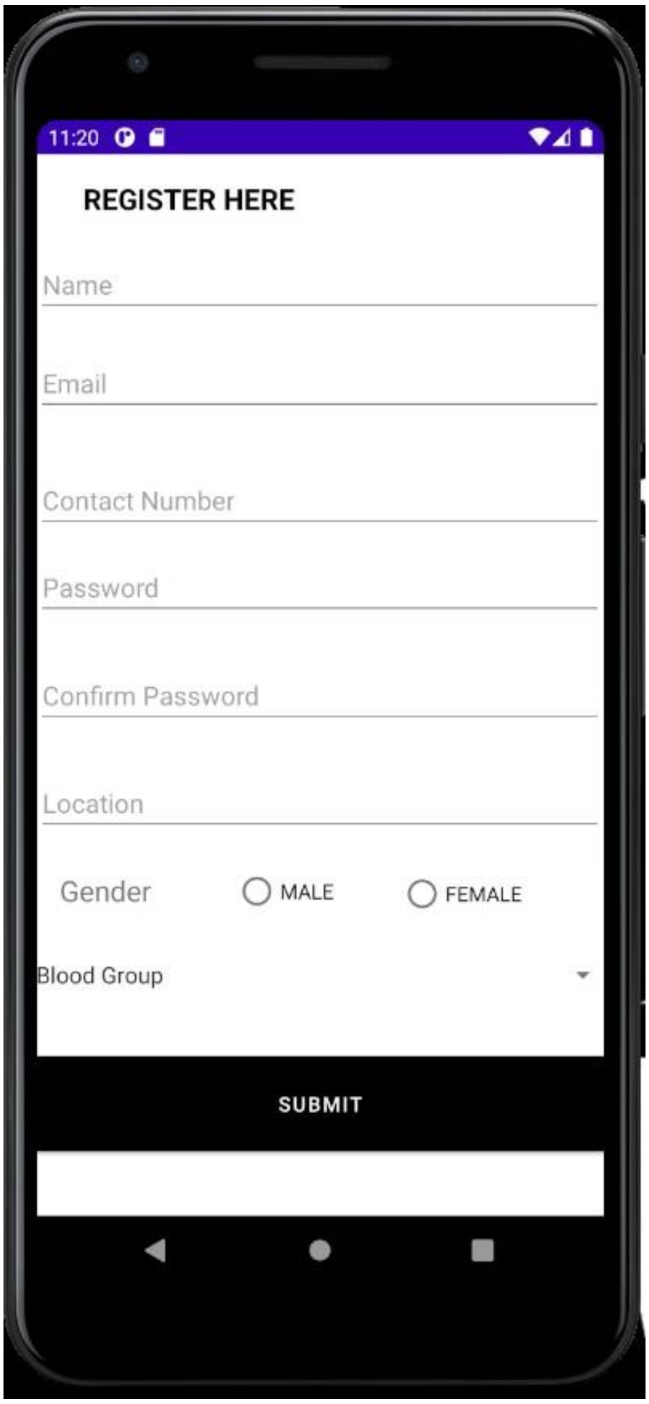
PANACEA is android application which utilizes the Firebase Real-time Database in order to gather & sort out the data of each of the red blood benefactors with a quick & efficient seek. (A google-services. json record should be produced for the android application utilizing the Firebase Console). This venture then goes about as very important job in order to spare the existence of individuals & which is also likewise to its central point. The undertaking of the android (The red blood Bank) framework was created so that the clients see the data about the enrolled blood contributors, for example, name, address, & some other such close to the home data alongside their subtle element of the blood groups & the other beneficial data of gives away/donor. This task additionally has one login page where the client's required to enlist the credentials & at exactly note at which point can he see the accessibility of required blood & similarly may likewise then enlist himself to give away the needed blood in an event which he wishes to. Then this venture requires an internet access permission granted & then in this way there occurs a burden of the huge internet disappointment. The android application selects the correct benefactor online then & flash utilizing & restorative subtle elements which are alongside the available red blood bunch. It is the primary point of making this android application to decrease opportunity, as it was, which was spent forbidding the correct giver/donor & also the accessibility of the corrected blood required. Thus, this android application gives away the exact required data no matter of moments & then furthermore helps for a faster basic leadership.

The android application acquaints on the insightful contraptions with the assurance of the arrival of a greatest possible no. of the red blood benefactors within the country. This tackles PDAs with the android system laid by the app. This associates the blood bank android application with its suppliers by providing the messages to the advocate who gives away the blood arrangements simply & which also enables patient to abuse the same.

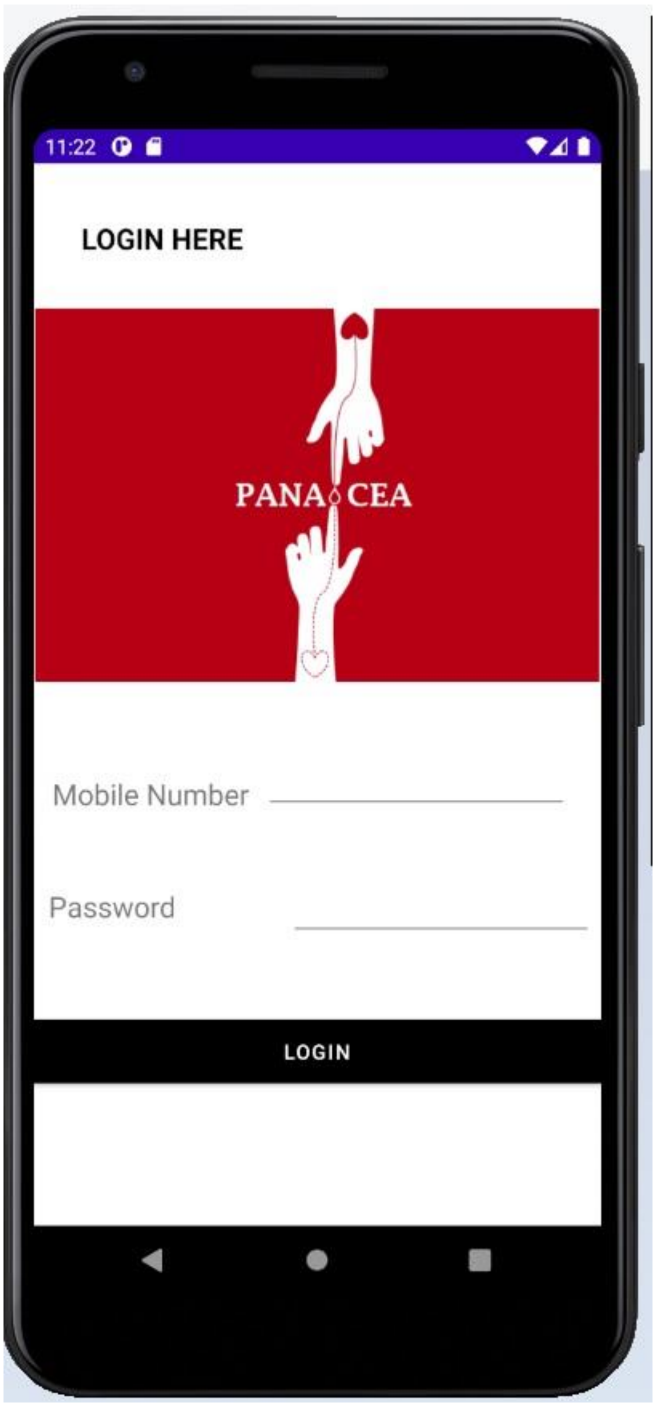
4.2 WORKING OF THE APPLICATION

The android application supports android operating system & therefore, when put live can be downloaded from the play store. It has an icon with two joining hands on it & the name of the android applications is PANAEA.

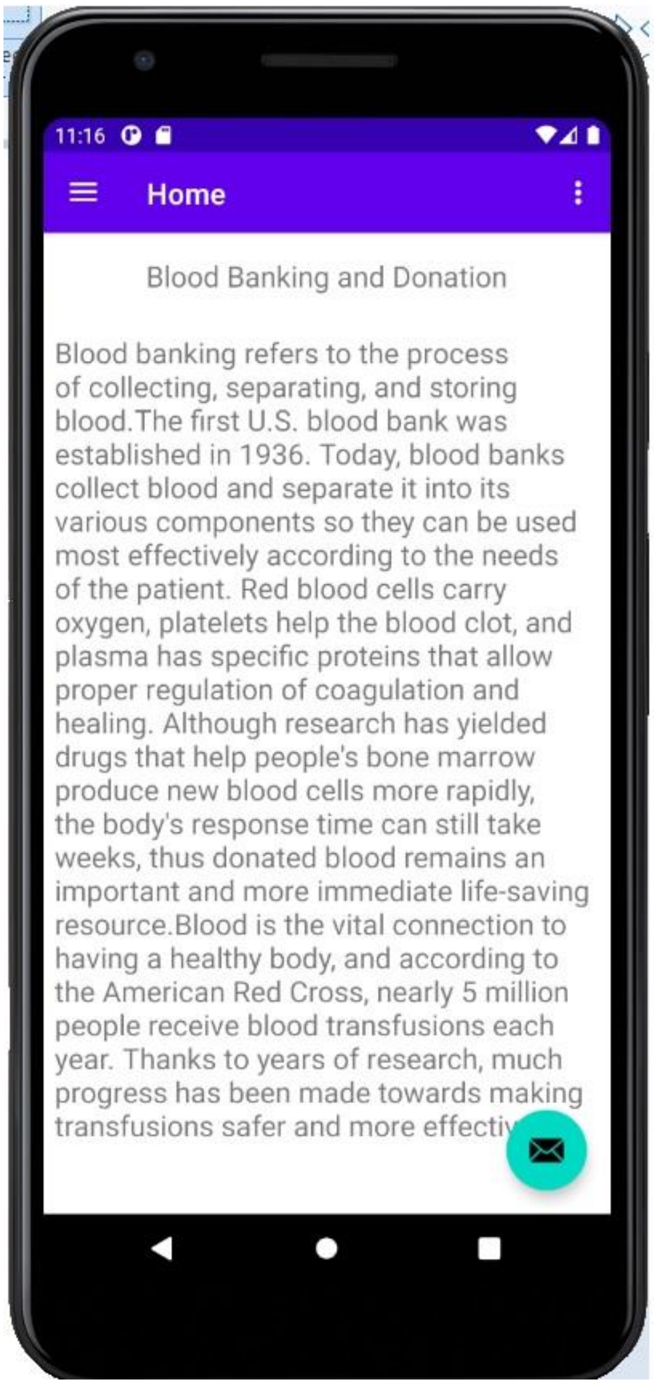
REGISTER HERE: When one opens this android application, All the new users can register themselves here by simply entering their personal details such as Name, Email id, Contact Number, Blood Group, Gender, Location and Generating Passwords. This page will appear as shown below:



<u>LOGIN HERE:</u> As soon as you are done with Register, you will be moved to the Login page. Then after entering the credentials that you generated while registering, you have to click on the LOGIN button, in case, the credentials are wrong a message displayed saying ‘wrong credentials’ otherwise you are moved to the next page i.e. the home page.



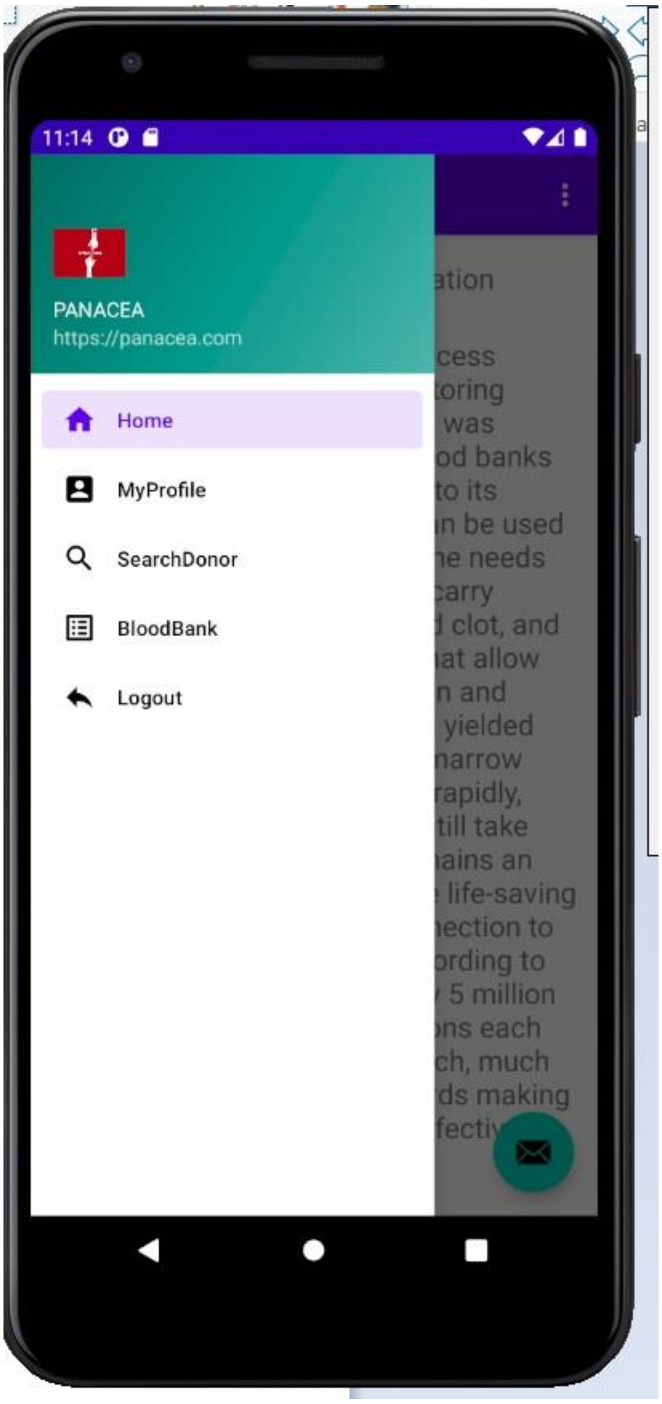
HOME PAGE: After logging in, Home Page appears where you read the information regarding Blood Donation and facts related to it. And on this top left corner, there's a Menu Button through which you navigate other pages as well.



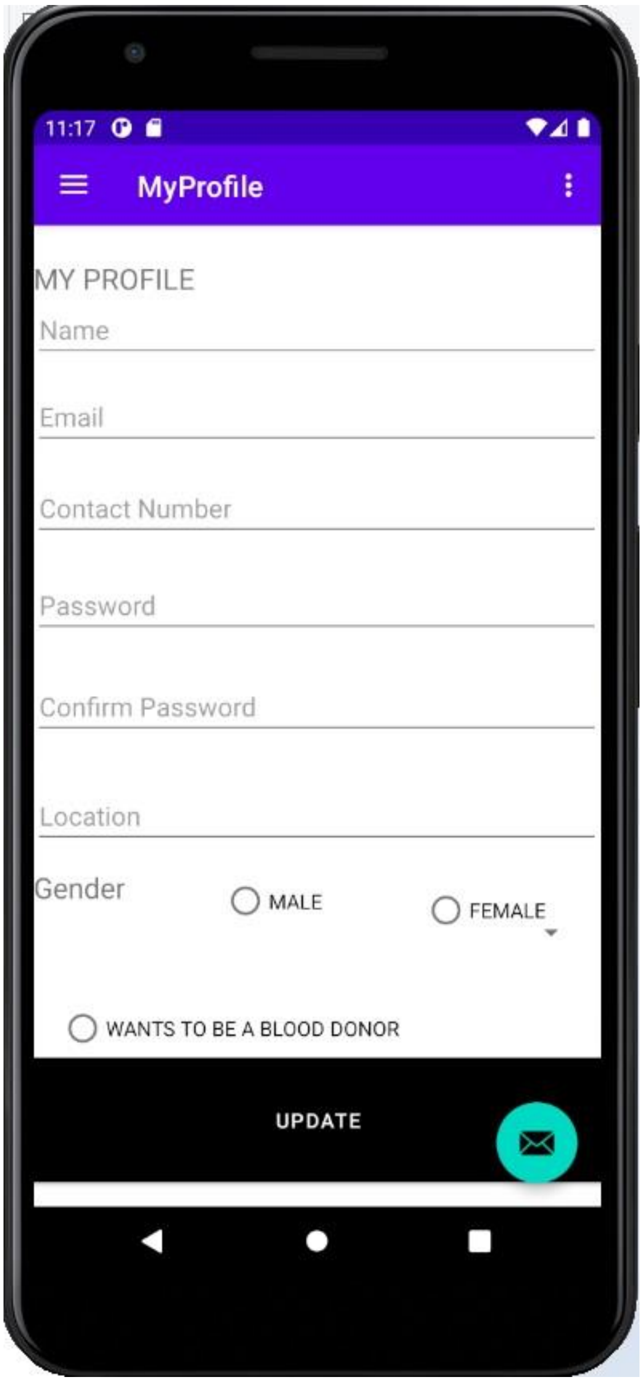
Blood Banking and Donation

Blood banking refers to the process of collecting, separating, and storing blood.The first U.S. blood bank was established in 1936. Today, blood banks collect blood and separate it into its various components so they can be used most effectively according to the needs of the patient. Red blood cells carry oxygen, platelets help the blood clot, and plasma has specific proteins that allow proper regulation of coagulation and healing. Although research has yielded drugs that help people's bone marrow produce new blood cells more rapidly, the body's response time can still take weeks, thus donated blood remains an important and more immediate life-saving resource.Blood is the vital connection to having a healthy body, and according to the American Red Cross, nearly 5 million people receive blood transfusions each year. Thanks to years of research, much progress has been made towards making transfusions safer and more effecti

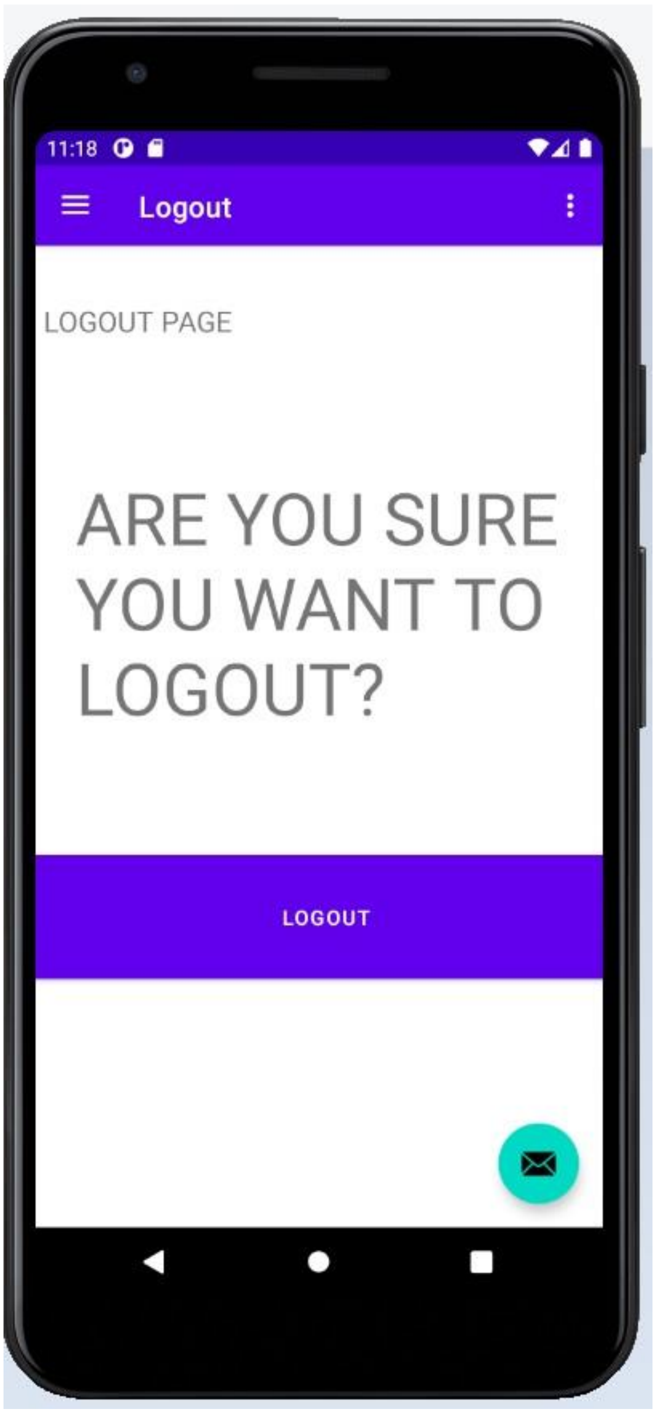
MENU: As shown above, after clicking on the Menu Button, various buttons appears such as, Home, My Profile, Search Donors, Blood Bank and Logout. On the top of these button, the logo with the name of the application appears.



MyProfile: Under this, you can view their Profile details provided by you, only when you are logged in. You can also update your details anytime by clicking the UPDATE button at the bottom of this page.



LOGOUT: One can use LOGOUT button when done. This is simply let you out temporarily. You can anytime use the application again by just logging in.


11:18
Logout
LOGOUT PAGE
ARE YOU SURE YOU WANT TO LOGOUT?
LOGOUT

CHAPTER 5

PERFORMANCE ANALYSIS

User interface routine testing certifies that the application not only congregates its functional requirements, but that the interface of the user interactions with the app is very smooth, running consistently at a 60 frames per second speed, without any number of dropped or delayed frames, or junk. This chapter explains the tools offered to measure the UI performance, and also provides an approach to assimilate the UI performance extents into the testing practices.

5.1 DATA SYNCHRONIZATION

5.1.1 MEASURING UI PERFORMANCE

In order to improvise the performance first needed is the ability to compute performance of the system, and then to diagnose and to identify problems which may arise from the various parts of the pipeline. dumpsys is an android tool which runs on a device and then dumps some interesting info about the status of the system services. After passing the 'gfxinfo' command to dumpsys one gets an o/p in thelog cat with the performance information relating to the frames of the animation which occurs during the recording phase. >adb shell dumpsys gfxinfo This command produces multiple distinguished variants of the frame timing data.

5.1.2 AGGREGATE FRAME STATS

With Android 6.0 (API level 25) this command prints the aggregated analysis of the frame data to the logcat, which is collected across entire lifetime of the particular process.

```
39 Stats since: 754958278148ns

Jankyframes: 36335 (42.99%)

Total frames rendered: 82179

90th percentile: 35 ms

95th percentile: 43 ms

99th percentile: 68 ms
```

Number High input latency: 143

Number Missed Vsync: 4705

Number Slow UI thread: 17260

Number Slow draw: 23342

Number Slow bitmap uploads: 1542

These high-level statistics convey at a high level the rendering performance of the app, as well as its stability across many frames.

5.1.3 PRECISE FRAME TIMING INFO

With the android 6.0 there comes this new command for 'gfxinfo', and that's '*framestats'* that provides very detailed frame timing info from recent frames, so that you can easily track down and debug the problems more accurately.

```
>adb shell dumpsysgfxinfo<PACKAGE_NAME>framestats
```

The above command prints the frame timing info, with the nanosecond timestamps, from last 120 frames formed by the application. Below is the raw output from

Adb dumpsys gfxinfo<PACKAGE_NAME>framestats:

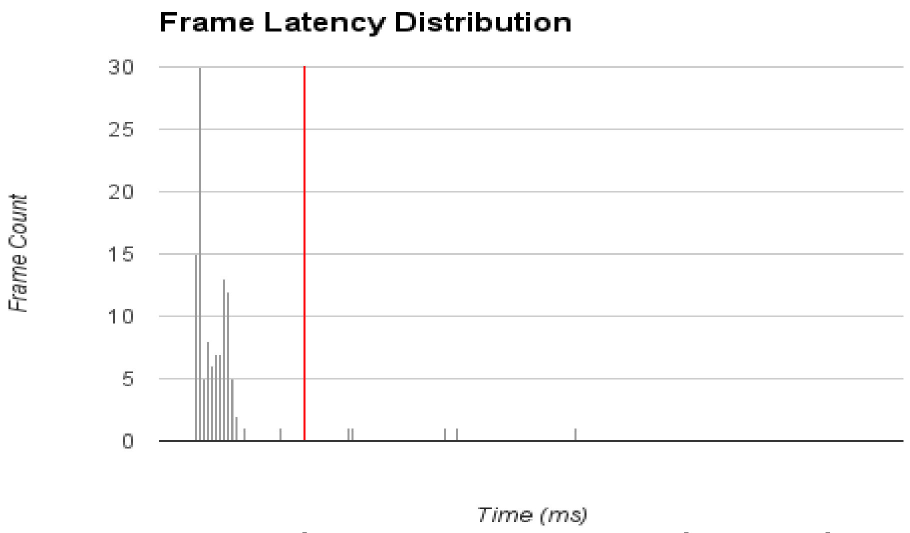
Table no. 5.1.1. Frame stats

Frame seq. no	Time spent during each stage of frame-producing pipeline. (ms)
1.	27964466202353
2.	27964466202353
3.	27964461202353
4.	27964467153286
5.	27964489520682
6.	27964503736099
7.	27964516575320

8.	27964497155000
9.	27964524884536
10.	27964531371203

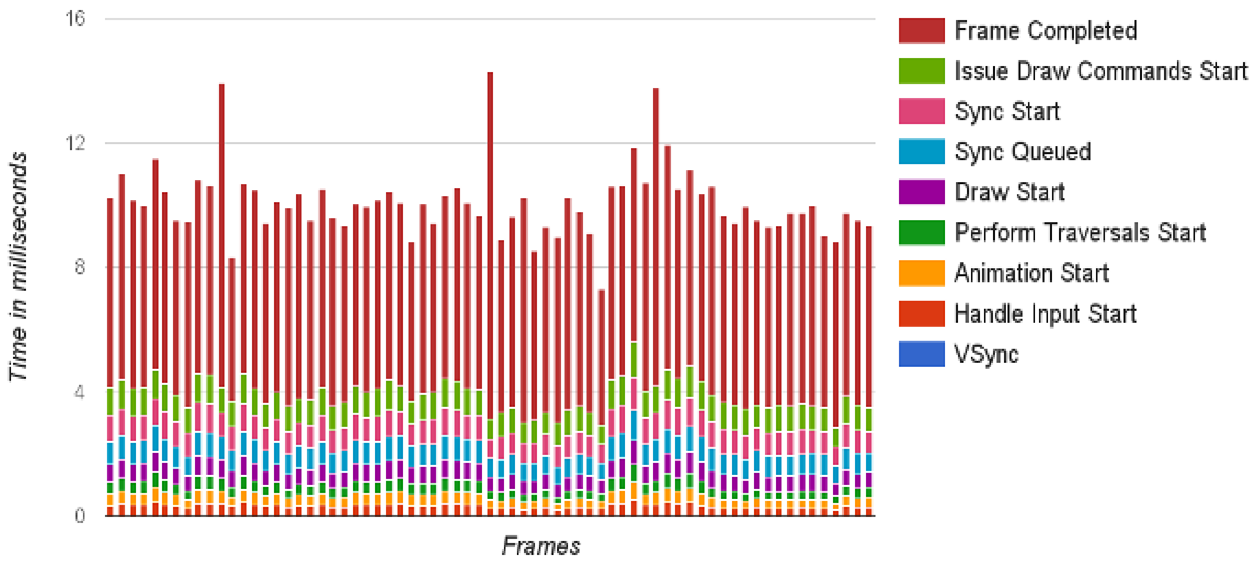
Each line of the above output is representing a frame which is produced by the app. Each line is having a fixed no. of columns describing the time spent during each stage of frameproducing pipeline.

One simple but valuable visualization is histogram showing distribution of the frames times (FRAME_COMPLETED - INTENDED_VSYNC) in the different latency buckets. This graph depicts at a glance that most of the frames were good –i.e. below the 16 ms deadline (depicted in red), but few frames were pointedly over the deadline. One can look at changes in the histogram over the time to see the wholesale shifts or the new outliers being created. One can also graph input time spent in layout, latency, or other similar interesting metrices based on many timestamps in the data.



Graph no.5.1.1. Frame Latency Distribution Graph

5.1.4 SIMPLE FRAME TIMING DUMP



Graph no.5.1.2. Simple Frame timing Graph

The result of the running 'gfxinfo', copying output, pasting the same into the spreadsheet application, and graphing data as stacked bars.

5.2 RESPONSE TIME

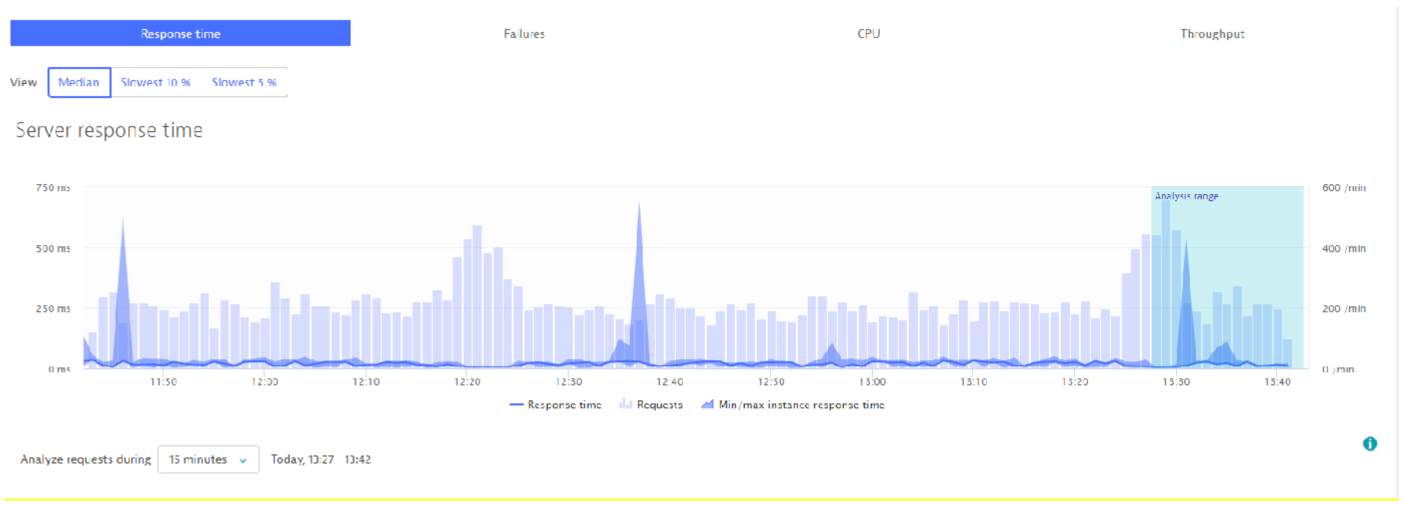
The Response time chart shows how response times of requests triggered by the app service have been distributed during selected timeframes. The chart also shows the average no. of requests over the time frame along with the minimum/maximum response times of each service instance. For response time analysis, we have an option of viewing Median, Slowest 10%, or Slowest 5% percentiles.

On the x-axis,

Response time requests at intervals of 10 mins starting from 1150.

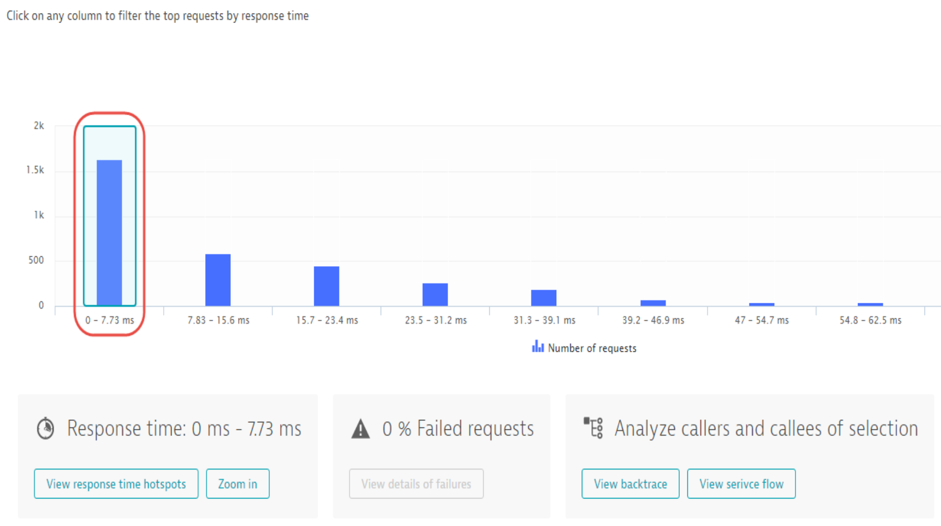
On the y-axis,

Response time at intervals of 250ms.

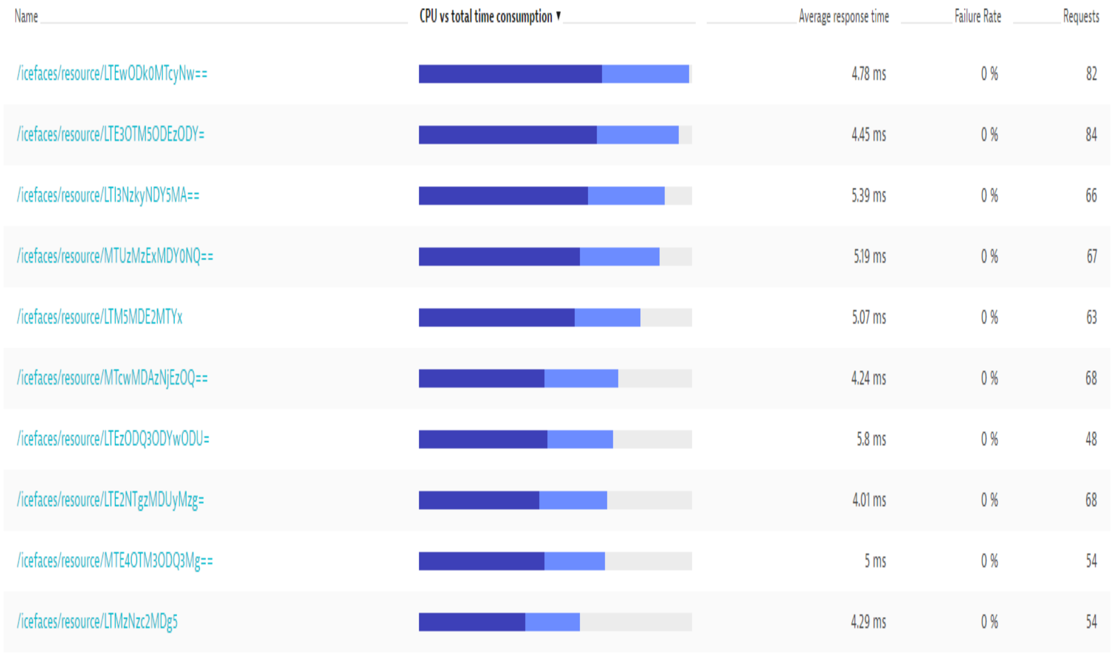


Graph no. 5.2.1. Response Time Chart

The diagram below shows the number of the requests which fall into the respective response time range. In the top requests section, the slowest requests (i.e., the longest response times) can take dramatically longer time to execute than fastest requests. These outliers can always have a huge influence on overall response time of the service.



Graph no. 5.2.2. Response time for number of requests



Name	CPU vs total time consumption ▾	Average response time	Failure Rate	Requests
/icefaces/resource/LTEwODk0MTcyNw==		4.78 ms	0 %	82
/icefaces/resource/LTE3OTM5ODEzODY=		4.45 ms	0 %	84
/icefaces/resource/LTI3NzkyNDY5MA==		5.39 ms	0 %	66
/icefaces/resource/MTUzMzExMDY0NQ==		5.19 ms	0 %	67
/icefaces/resource/LTM5MDE2MTYx		5.07 ms	0 %	63
/icefaces/resource/MTcwMDAzNjEzOQ==		4.24 ms	0 %	68
/icefaces/resource/LTEzODQ3ODYwODU=		5.8 ms	0 %	48
/icefaces/resource/LTE2NTgzMDUyMzg=		4.01 ms	0 %	68
/icefaces/resource/MTE4OTM3ODQ3Mg==		5 ms	0 %	54
/icefaces/resource/LTMzNzc2MDg5		4.29 ms	0 %	54

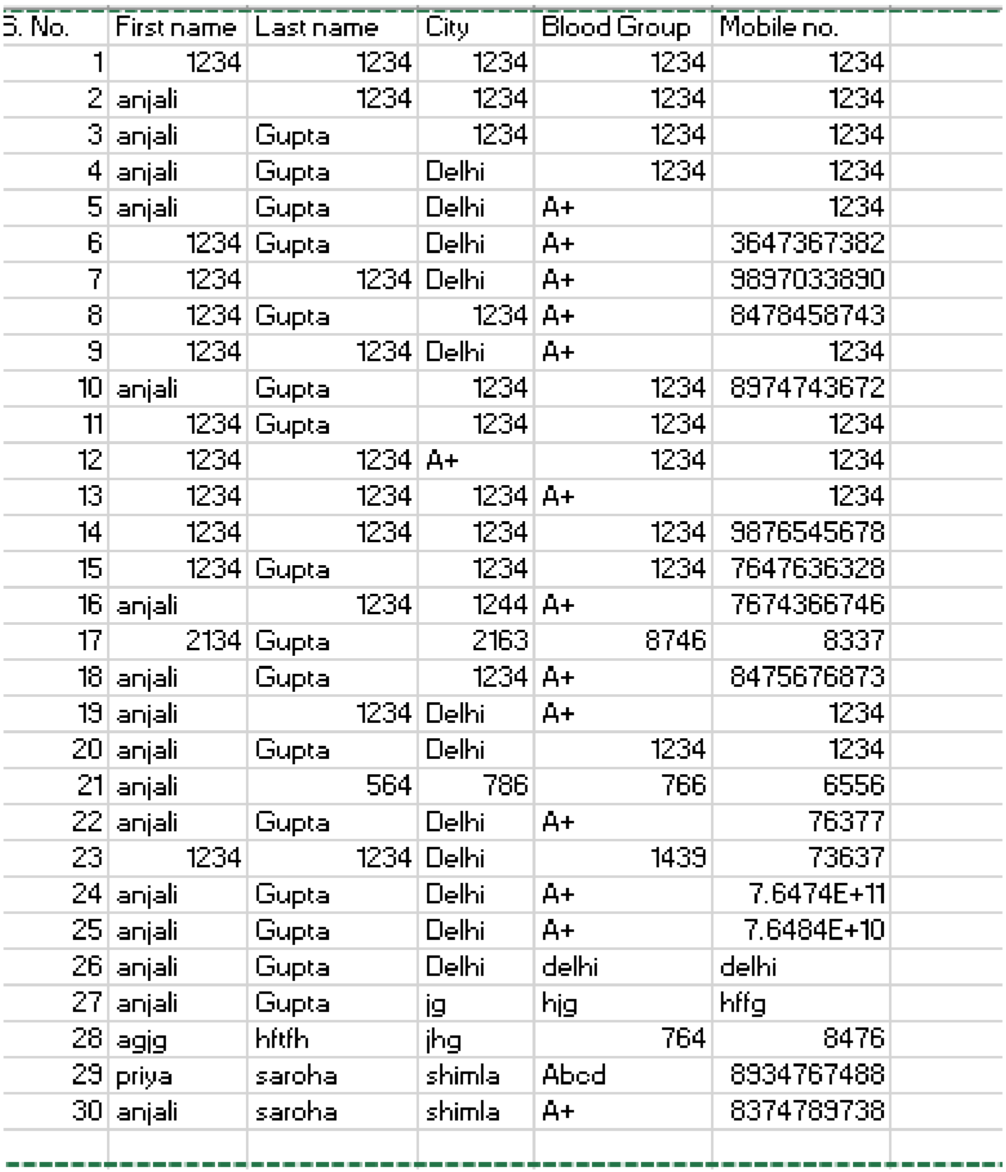
Table no 5.2.1. Response time for number of requests

5.3 UNIT TESTING

These test cases were devised for donor form fill up database fields. In the sample test cases, except test case 30 since testcase 30 is the accurate field fill up, all the test cases were worked upon & a new set was determined which constrained all the other 29 test cases & only test case 30 could be passed the new test.

Table no.5.3.1. Test cases

We have 5 fields for filling up.



S. No.	First name	Last name	City	Blood Group	Mobile no.
1	1234	1234	1234	1234	1234
2	anjali	1234	1234	1234	1234
3	anjali	Gupta	1234	1234	1234
4	anjali	Gupta	Delhi	1234	1234
5	anjali	Gupta	Delhi	A+	1234
6	1234	Gupta	Delhi	A+	3647367382
7	1234	1234	Delhi	A+	9897033890
8	1234	Gupta	1234	A+	8478458743
9	1234	1234	Delhi	A+	1234
10	anjali	Gupta	1234	1234	8974743672
11	1234	Gupta	1234	1234	1234
12	1234	1234	A+	1234	1234
13	1234	1234	1234	A+	1234
14	1234	1234	1234	1234	9876545678
15	1234	Gupta	1234	1234	7647636328
16	anjali	1234	1244	A+	7674366746
17	2134	Gupta	2163	8746	8337
18	anjali	Gupta	1234	A+	8475676873
19	anjali	1234	Delhi	A+	1234
20	anjali	Gupta	Delhi	1234	1234
21	anjali	564	786	766	6556
22	anjali	Gupta	Delhi	A+	76377
23	1234	1234	Delhi	1439	73637
24	anjali	Gupta	Delhi	A+	7.6474E+11
25	anjali	Gupta	Delhi	A+	7.6484E+10
26	anjali	Gupta	Delhi	delhi	delhi
27	anjali	Gupta	jg	hjg	hffg
28	agjg	hftfh	jhg	764	8476
29	priya	saroha	shimla	Abcd	8934767488
30	anjali	saroha	shimla	A+	8374789738

For field ‘First name’, possible inputs = ‘all digits’+’all characters’+’all aphabets’+’mixed input’ = 5+5+5

For field ‘Last name’, possible inputs =‘all digits’+’all characters’+’all aphabets’+’mixed input’ = 5+5+5

For field ‘city’, possible inputs = ‘all digits’+’all characters’+’all aphabets’+’mixed input’ = 5+5+5

For field ‘the red bloodgroup’, possible inputs = ‘all digits’+’all characters’+’all aphabets’+’mixed input’ = 5+5+5

For field ‘Mobile number’, possible inputs = ‘all digits’+’all characters’+’all aphabets’+’mixed input’ = 5+5+5

Correct input= 5+5+5+5+5+5 = 30
Total input = 75
Error= (75-30)/75 =45/75 = 0.6

By putting constraints on the fields error was minimized to 0.0%.

5.4 REAL TIME STREAMING

The android application was tested on a batch of 30 people using android operated devices. With permissions to internet & GPS, the android application didn’t fail & performed 100% accurate real time streaming.

5.5 GROUP MESSAGING

The server was shared with 750 users at a time with login access to the android application & 100% performance was observed with a viable internet connection.

5.6 COMPARISON BETWEEN JSON AND XML

Similarities between JSON & XML

- Both are simple & open.
- Both supports unicode. So, internationalization supported by JSON & XML both.
- Both represents self describing data.
- Both are interoperable or language-independent.

JSON is Like XML Because

- Both JSON & XML are "self describing" (human readable)
- Both JSON & XML are hierarchical (values within values)
- Both JSON & XML can be parsed & used by lots of programming languages
- Both JSON & XML can be fetched with an XML Http Request

JSONUnlike XML Because

- JSON doesn't use end tag
- JSON is shorter
- JSON is quicker to read & write
- JSON can use arrays

The biggest difference is:XML has to be parsed with an XML parser. JSON can be parsed by standard JavaScript functions.

Why JSON is Better Than XML

- XML is much more difficult to parse than JSON.
- JSON is parsed nto a ready-to-use JavaScript object.
- For AJAX android applications, JSON is faster & easier than XML.

Using XML

- Fetch an XML document
- Use the XML DOM to loop through the document
- Extract values & store in variables

Using JSON

- Fetch a JSON string
- JSON.Parse the JSON string

No.	JSON	XML
1)	JSON stands for JavaScriptObjectNotation.	XML stands for eXtensible Markup Language.
2)	JSON is simpler to read & write.	XML is comparitively less simple than JSON.
3)	JSON is easier to learn.	XML is comparatively less easy than JSON.
4)	JSON is data-oriented.	XML is document-oriented.
5)	JSON doesn't provide display capabilities.	XML provides the capability to display data because it is a markup language.
6)	JSON supports array.	XML doesn't support array.
7)	JSON is less secured than XML.	XML is more secured.
8)	JSON files are morehuman readable than XML.	XML files are less human readable.
9)	JSON supports only text & number data type.	XML support many data types such as text, number, images, charts, graphs etc. Moreover, XML offers options for transferring the format or structure of the data with actual data.

Table no. 5.6.1. JSON vs XML

CHAPTER 6
CODING

6.1 CODING

<u>HOME ACTIVITY</u>

```
package com.example.panacea.Activities;

import android.os.Bundle;
import android.view.View;
import android.view.Menu;

import com.example.panacea.R;
import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.android.material.snackbar.Snackbar;
import com.google.android.material.navigation.NavigationView;

import androidx.navigation.NavController;
import androidx.navigation.Navigation;
import androidx.navigation.ui.AppBarConfiguration;
import androidx.navigation.ui.NavigationUI;
import androidx.drawerlayout.widget.DrawerLayout;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;

public class HomeActivity extends AppCompatActivity {

    private AppBarConfiguration mAppBarConfiguration;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_home);
    Toolbar toolbar = findViewById(R.id.toolbar);
    setSupportActionBar(toolbar);
    FloatingActionButton fab = findViewById(R.id.fab);
    fab.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Snackbar.make(view, "Replace with your own action", Snackbar.LENGTH_LONG)
                    .setAction("Action", null).show();
        }
    });
    DrawerLayout drawer = findViewById(R.id.drawer_layout);
    NavigationView navigationView = findViewById(R.id.nav_view);
    // Passing each menu ID as a set of Ids because each
    // menu should be considered as top level destinations.
    mAppBarConfiguration = new AppBarConfiguration.Builder(
            R.id.nav_home, R.id.nav_myprofile,R.id.nav_bloodBank,R.id.nav_searchDonor,R.id.nav_logout)
            .setDrawerLayout(drawer)
            .build();
    NavController navController = Navigation.findNavController(this, R.id.nav_host_fragment);
    NavigationUI.setupActionBarWithNavController(this, navController, mAppBarConfiguration);
    NavigationUI.setupWithNavController(navigationView, navController);
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present.
    getMenuInflater().inflate(R.menu.home, menu);
    return true;
}

@Override
public boolean onSupportNavigateUp() {
        NavController navController = Navigation.findNavController(this, R.id.nav_host_fragment);
        return NavigationUI.navigateUp(navController, mAppBarConfiguration)
                || super.onSupportNavigateUp();
    }
}
```

LOGIN ACTIVITY

```
package com.example.panacea.Activities;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

import com.example.panacea.R;

public class LoginActivity extends AppCompatActivity {
Button OpenActivity;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        OpenActivity=findViewById(R.id.button2);
        OpenActivity.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent i=new Intent(getApplicationContext(),HomeActivity.class);
                startActivity(i);
            }
        });
    }
}
```

MAIN ACTIVITY

```
package com.example.panacea.Activities;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

import com.example.panacea.R;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

REGISTER ACTIVITY

```
package com.example.panacea.Activities;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.graphics.Canvas;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.Spinner;
import android.widget.Toast;

import com.example.panacea.R;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

public class RegisterActivity extends AppCompatActivity {
    EditText Name,Email,Phone,Password,Password2,Address2;
    RadioButton radioButton,radioButton2;
    Spinner spinner;
    String BloodGroup[]={"Blood Group","A+","A-","B+","B-","O+","O-","AB+","AB-"};
    ArrayAdapter<String> adapter;
    Button OpenActivity;

    FirebaseDatabase firebaseDatabase;
    DatabaseReference databaseReference;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);
        OpenActivity=findViewById(R.id.button);
        Name=findViewById(R.id.editTextTextPersonName);
        Email=findViewById(R.id.editTextTextEmailAddress);
        Phone=findViewById(R.id.editTextPhone);
        Password=findViewById(R.id.editTextTextPassword);
        Password2=findViewById(R.id.editTextTextPassword2);
        Address2=findViewById(R.id.editTextTextPostalAddress2);
        radioButton=findViewById(R.id.radioButton);
        radioButton2=findViewById(R.id.radioButton2);

        firebaseDatabase=firebaseDatabase.getInstance();
        databaseReference=firebaseDatabase.getReference().child("Data");



        spinner=findViewById(R.id.spinner);
        adapter=new                                         ArrayAdapter<>(getApplicationContext(),
android.R.layout.simple_spinner_item,BloodGroup);
        spinner.setAdapter(adapter);
        spinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
            @Override
            public void onItemSelected(AdapterView<?> parent, View view, int i, long id) {
                Toast.makeText(getApplicationContext(),adapter.getItem(i), Toast.LENGTH_SHORT).show();
            }

            @Override
            public void onNothingSelected(AdapterView<?> parent) {

            }
        });



        OpenActivity.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                Intent i=new Intent(getApplicationContext(),LoginActivity.class);
                startActivity(i);

            }
        });
    }
/*
    private void getValue() {
        databaseReference.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot snapshot) {
                spinner.clearFocus();
                for(DataSnapshot dataSnapshot: snapshot.getChildren()){
                     //getvalue
                    String svalue=dataSnapshot.child("Value").getValue(String.class);
                }

            }
            @Override
            public void onCancelled(@NonNull DatabaseError error) {

            }
        })
    }
*/
}
```

SPLASH SCREEN

```
package com.example.panacea.Activities;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.os.Handler;

import com.example.panacea.R;

public class SplashScreen extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_splash_screen);
        Handler handler=new Handler();
        handler.postDelayed(new Runnable() {
            @Override
            public void run() {
                startActivity(new Intent(SplashScreen.this,RegisterActivity.class));
                finish();
            }
        },2500);
    }
}
```

MYPROFILE FRAGMENT

```
package com.example.panacea.Activities.ui.myprofile;

import androidx.lifecycle.ViewModelProvider;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;
import androidx.lifecycle.Observer;

import com.example.panacea.Activities.ui.searchdonor.SearchDonorViewModel;
import com.example.panacea.R;

public class MyProfileFragment extends Fragment {

    private MyProfileViewModel MyProfileViewModel;

    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle savedInstanceState) {
        MyProfileViewModel =
                new ViewModelProvider(this).get(MyProfileViewModel.class);
        View root = inflater.inflate(R.layout.fragment_my_profile, container, false);
        final TextView textView = root.findViewById(R.id.text_myprofile);
        MyProfileViewModel.getText().observe(getViewLifecycleOwner(), new Observer<String>() {
            @Override
            public void onChanged(@Nullable String s) {
                textView.setText(s);
            }
        });
        return root;
    }
}
```

LOGOUT FRAGMENT

```
package com.example.panacea.Activities.ui.logout;

import androidx.lifecycle.ViewModelProvider;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;
import androidx.lifecycle.Observer;

import com.example.panacea.Activities.ui.searchdonor.SearchDonorViewModel;
import com.example.panacea.R;

public class LogoutFragment extends Fragment {
    private LogoutViewModel LogoutViewModel;

    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle savedInstanceState) {
        LogoutViewModel =
                new ViewModelProvider(this).get(LogoutViewModel.class);
        View root = inflater.inflate(R.layout.fragment_logout, container, false);
        final TextView textView = root.findViewById(R.id.text_logout);
        LogoutViewModel.getText().observe(getViewLifecycleOwner(), new Observer<String>() {
            @Override
            public void onChanged(@Nullable String s) {
                textView.setText(s);
            }
        });
        return root;
    }
}
```

SEARCH DONOR FRAGMENT

```
package com.example.panacea.Activities.ui.searchdonor;

import androidx.lifecycle.ViewModelProvider;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;
import androidx.lifecycle.Observer;
import com.example.panacea.Activities.ui.slideshow.SlideshowViewModel;
import com.example.panacea.R;

public class SearchDonorFragment extends Fragment {

    private SearchDonorViewModel SearchDonorViewModel;

    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle savedInstanceState) {
        SearchDonorViewModel =
                new ViewModelProvider(this).get(SearchDonorViewModel.class);
        View root = inflater.inflate(R.layout.fragment_search_donor, container, false);
        final TextView textView = root.findViewById(R.id.text_searchdonor);
        SearchDonorViewModel.getText().observe(getViewLifecycleOwner(), new Observer<String>() {
            @Override
            public void onChanged(@Nullable String s) {
                textView.setText(s);
            }
        });
        return root;
    }
}
```

HOME FRAGMENT

```
package com.example.panacea.Activities.ui.home;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;
import androidx.lifecycle.Observer;
import androidx.lifecycle.ViewModelProvider;

import com.example.panacea.R;

public class HomeFragment extends Fragment {

    private HomeViewModel homeViewModel;

    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle savedInstanceState) {
        homeViewModel =
                new ViewModelProvider(this).get(HomeViewModel.class);
        View root = inflater.inflate(R.layout.fragment_home, container, false);
        final TextView textView = root.findViewById(R.id.text_home);
        homeViewModel.getText().observe(getViewLifecycleOwner(), new Observer<String>() {
            @Override
            public void onChanged(@Nullable String s) {
                textView.setText(s);
            }
        });
        return root;
    }
}
```

BLOODBANK FRAGMENT

```
package com.example.panacea.Activities.ui.bloodbank;

import androidx.lifecycle.ViewModelProvider;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;
import androidx.lifecycle.Observer;

import com.example.panacea.Activities.ui.searchdonor.SearchDonorViewModel;
import com.example.panacea.R;

public class BloodBankFragment extends Fragment {

    private BloodBankViewModel BloodBankViewModel;

    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle savedInstanceState) {
        BloodBankViewModel =
                new ViewModelProvider(this).get(BloodBankViewModel.class);
        View root = inflater.inflate(R.layout.fragment_blood_bank, container, false);
        final TextView textView = root.findViewById(R.id.text_bloodbank);
        BloodBankViewModel.getText().observe(getViewLifecycleOwner(), new Observer<String>() {
            @Override
            public void onChanged(@Nullable String s) {
                textView.setText(s);
            }
        });
        return root;
    }

}
```

CHAPTER 7

CONCLUSIONS

The following chapter settles this report. The summary of the overall research has been presented, & findings of the overall study have been discussed as well as interpreted. The implication of the examination in the immediate framework of Blood Bank-android application and in the field android development is examined. Commendations for further examination culminate the chapter.

The opportunity of the following inferences is limited to the characteristics and the contextof the application. Hence, applied to some other situation, these conclusions can yield improper conventions. Still, the conclusions are applicable to the processes of dwelling the evolution in other progressive development projects.

6.1 CONCLUSION

One may have proposed this dependable and effective android the blood donation centre android application. The administrative give away by this proposed framework requires being significant to one's wellbeing with the part where the nature of the blood's considered for the security purposes of the patients. The contributor can then get himself/herself enrolled into this enhanced framework where there should be some occurrence of some crisis pre-requisite that the bloodgive away/donor can always put some demand. The remote strategy empowers stream of info that'll work a lot more quickly and advantageously. The forthcoming work of the framework's to make this android application into IOS stage. Acknowledgment is that it gives away us the extra-ordinary delight within this exhibiting undertaking report titled "Blood Bank"& one wishes to offer our huge thanks to the population which gave usimportant learning & provision in culmination of this particular venture. The direction & inspiration provided helped us in making the task into an incredible achievement. One offers our thanks to our task control Prof Pradeep Kumar Singh, who gave us the direction consolation all throughout the undertaking advancement.

6.2 FUTURE ENHANCEMENT

In future, the above proposed idea may be used on a   huge scale to provide the blood bank, the board framework. Further, this android application can be tinted with free manual area, following of contributors & recipients. Checking the authenticity of the benefactor must be possible with some more confirmed process. The distribution of the message or telecom of the promotion process might be in future presented for all the prominent android based life rather than just on social media like Facebook, Whatsapp, IG, TV, Radio etc. The database executive for this android application depends on the Firebase Parse server which can be taken into consideration by some best database servers. The data-base can be relocated to the cloud servers which can really expand proficiency of this undertaking.

REFERENCES

[1] P Priya, V Saranya,Shabana, Kavitha Subramani, "The Optimization of The blood Donor nformation&Management System by Technopedia" Department of Computer Science&Engineering, Panimalar Engineering College, Chennai, ndia, Volume 3, Special ssue 1, February 2014

[2] Tushar Pandit, Satish Niloor & A. S Shinde, "A Survey Paper on E-The blood Bank & idea to use on Smartphone" Dept of .T Sinhgad Academy of Engineering, Pune, India Year 2015

[3] Narendra Gupta, Ramakant Gawande & Nikhil Thengadi, "MBB: A L Check life Saving android app" Final Year, CSE Dept., JDIET, Yavatmal, India.VOLUME-2, SPECIAL SSUE-1, MARCH-2015

[4] Vikas Kulshreshtha, Dr Sharad Maheshwari, "The blood donation centre Management Information System in India" international Journal of Engineering Research&Android applications (IJERA) SSN: 2248-9622 Vol 1, ssue 2, pp.260-263

[5] Sultan Turhan, "AN ANDROID ANDROID APPLICATION FOR VOLUNTEER THE BLOODPEOPLE CANING TO DONATE THE BLOOD"

[6] T. Hilda Jenipha, R. Backiyalakshmi, "Android The blood Donor L Check life Saving Android application Cloud Computing" Department of Computer Science & Engineering, PRIST University, Puducherry, India