A SECURE CLOUD COMPUTING BASED FRAMEWORK FOR THE BLOOD BANK

Main Project Report

Submitted by

RANJANA K

Reg no:FIT20MCA-2089

Submitted in partial fulfillment of the requirements for the award of the degree of

Master of Computer Applications

Of

A P J Abdul Kalam Technological University

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)®

ANGAMALY-683577, ERNAKULAM(DIST)

JULY 2022

DECLARATION

I, hereby declare that the report of this project work, submitted to the Department of Computer Applications, Federal Institute of Science and Technology (**FISAT**), Angamaly in partial fulfillment of the award of the degree of Master of Computer Application is an authentic record of my original work.

The report has not been submitted for the award of any degree of this university or any other university.

Date :

Ranjana K

Place: Angamaly

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)®

(An ISO 9001:2015 Certified, NAAC ('A' Grade) Accredited Institution with NBA Accredited Programmes
(Approved by AICTE – Affiliated to APJ Abdul Kalam Technological University, Kerala)
Owned and Managed by Federal Bank Officers' Association Educational Society
HORMIS NAGAR, MOOKKANNOOR P.O., ANGAMALY - 683 577, ERNAKULAM DT., KERALA, S. INDIA.
Tel: (O) 0484-2725272 Fax: 0484 – 2725250 E-mail: mail@fisat.ac.in Website: www.fisat.ac.in

11TH July 2022

TO WHOMSOEVER IT MAY CONCERN

This is to certify that Mr./Ms. RANJANA K (Reg. No. FIT20MCA-2089) has successfully completed his/her Main Project with the title "A Secure Cloud Computing Based Framework for the Blood Bank.", in the Department of Computer Applications, FISAT, during the period from 30th March 2022 to 11th July 2022.

Dr DEEPA MARY MATHEWS

HEAD OF THE DEPARTMENT

FEDERAL INSTITUTE OF SCIENCE & TECHNOLOGY (FISAT) ANGAMALY
DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)®

ANGAMALY, ERNAKULAM-683577

Focus on Excellence

CERTIFICATE

This is to certify that the project report titled **"A SECURE CLOUD COMPUTING BASED FRAMEWORK FOR THE BLOOD BANK"** submitted by **RANJANA K** towards partial fulfillment of the requirements for the award of the degree of Master of Computer Applications is a record of bonafide work carried out by her during the year 2022.

12/7/22

Project Guide

Ms. Joice T

FEDERAL INSTITUTE OF SCIENCE & TECHNOLOGY (FISAT) ANGAMALY
DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS

Head of the Department

Dr. Deepa Mary Mathews

Submitted for the viva-voce held on ..... at .....

Examiner :

ACKNOWLEDGEMENT

Gratitude is a feeling which is more eloquent than words, more silent than silence. To complete this project work I needed the direction, assistance and co-operation of various individuals, which is received in abundance with the grace of God.

I hereby express my deep sense of gratitude to **Dr. Manoj George**, Principal of FISAT and **Dr. C Sheela**, Vice principal of FISAT, for allowing me to utilize all the facilities of the college.

My sincere thanks to **Dr.Deepa Mary Mathews**, Head of the department of Computer Applications FISAT ,who had been a source of inspiration.I express heartiest thanks to **Ms. Joice T** my internal guide for her encouragement and valuable suggestion.I express my heartiest gratitude to my scrum master **Ms. Senu Abi** and the faculty members in my Department for encouragement and constructive suggestions and comment during the project work.

Finally I wish to express my thanks to my parents, friends and well-wishers who extended their help in one way or other in preparation of my project. Besides all, I thank GOD for everything.

ABSTRACT

Cloud Computing is a modern technology and it gives access to the network upon request to required computing resources (network, servers, storage, applications and services). Cloud uses different models of service such as hybrid, community, private or public cloud model. Cloud creates a new understanding between companies, organisation and their information. This requires the existence of a third party to manage relationships. It is a named service provider in the cloud computing.

A blood Bank can be defined as a bank or storage place where blood is collected, preserved and used whenever needed or demanded. Everyone is aware that the traditional blood bank management system includes paperwork. Its way of working is not efficient enough at the time of emergency situations. The main aim of creating cloud-based blood bank system is to make the blood available on time to the people, even in emergency situations. With the help of this project, the user can be able to view information about every entity related to blood bank i.e. hospitals, donors, a location of another blood bank etc. The security factor is maintained properly. Whenever a new user accesses the system as a donor, he/she has to register himself/herself. This project consists of android application which can be used in smart phones; it will contain all the information of the donor and nearby hospitals. The application will also contain a GPS (Global Positioning System) system to track the location of the nearby blood banks or hospitals. The person did not need to go out far, for the search of the blood banks/hospitals, this application helps to save the time to a great extent. This also helps in correct and quick decision making.

Contents

Chapter 1

INTRODUCTION

Blood contribute to 7% of total body weight, so to maintain the specific amount of blood in the body is necessary for a human to survive. Studies show that for every moment, to save their life someone needs blood. Especially in the rural area the facility provided by the blood bank system is not appropriate due to lack of availability of information and amount of blood in one specific blood bank. The project is mainly used to improve the blood bank system working, management etc, with the help of cloud computing technology. The project provides a platform using which the information about the nearby blood bank are available for the requester requesting it. Location can be made visible to the user by the use of GPS technology.

This includes brief information about working with the blood bank management system, its services and various technologies like cloud computing, android application, web technology etc.

The system Consist of an android application that the user can access. The system gives unique identification to its every user. This unique identification can help the user in future correspondence. Administrator plays an important role in the system as he/she has right to accept the blood bank, also they can block and unblock blood banks if they are of no use. All this information will be collected in the central repository. The information in the android application that is the user will be stored using the cloud. The requester can get the information about blood blank as per his need which will help them in emergency situations.Cloud computing technology is used in this application because cloud comput-

ing is the latest and efficient way of server-based computing. Cloud provides good backup recovery, flexibility, and increased security.

Chapter 2

PROOF OF CONCEPT

2.1 Existing System

The information collected from the blood bank and hospitals describe the working of the blood bank system. Blood cannot be produced artificially in the laboratory. Thus to satisfy an increased need of blood, blood collection should be increased. Various promotional activities are conducted by the blood bank and hospitals to enhance the donor to donate blood, as the amount of blood in the bank depends on a number of donors. Information like phone number, email address etc is collected from the donor and stored after they complete their blood donation. The existing system is mainly dependent on paperwork. To donate and receive the blood from the bank, the donor and receiver has to fill the form consisting of the basic details.

There are some sources that provide an online platform for blood donation. These website provides various facilities like searching availability of blood, donor registration, and requesting blood. The website does not provide accurate location based search results and hence it will not be a reliable source in every scenario. There is no integration with blood banks. Also some website details include the availability of each blood group. But the information provided is not accurate.

2.2 Proposed System

To get started with our proposed application user need to first download the application. Once the application is downloaded user will be provided with two options on screen. First is, log in and second is, sign in option. If the user is already registered, then he/she can go for the first option and login. If the user is using the application for the first time then he/she has to create an account by providing details like name, gender, contacts, post, and email id. After registration is done the user can access the application, provided the user has internet access. Once the user is signed in he/she will be provided with various options like:

- Nearby hospitals
- Request for blood
- View Request status
- View other requests
- Blood camp details
- Feedback
- Complaint and its Reply

Just by selecting any of the options mentioned above he /she will get the information accordingly like information about blood camp, the nearby hospital etc. All the detail of the blood donor, hospital is stored in database. Security care is taken. Data of each user is stored safely on cloud. By using this application the user will not have to search for the blood in case of emergency and can directly get the detail of nearby blood bank. The entities involved in the cloud-based blood bank management system are as follows:

- Requester: The person who needs the blood from blood bank because of accident, disease, surgery etc. The requester can login from the android application and can request for blood.

- Donor: The person who is healthy enough to donate the blood to the blood bank for saving a person's life is the donor. The person having appropriate body weight, hemoglobin and no acute or chronic disease can become the donor.The donor can login from the android application. Then they can view the other blood request that will be required for others. So they can accept that request and can donate the blood to that blood bank.

- Blood Bank: Blood bank can be simply defined as a section of the blood bank where the blood is stored and tested, to reduce the risk at the time of transfusion.The blood bank can login from the web application and will confirm whether the donor has donated the blood or not.

- Admin: Admin is the important part of the system.The Admin can accept/verify the blood bank, i.e. After the blood bank registration, the blood bank will be verified by admin and after that only the blood bank will be in use. The admin can block and unblock the blood bank according to their working. He is the person responsible for handling the efficient working of the system.In case any problem arises the admin must try to resolve it and make system work again.

The latest cloud computing technology will be used in this system. Databases will be created and managed using the SQL database.Normal database server would also provide facility to store large data but cloud provides additional features like flexibility, disaster recovery etc. GPS (Global Positioning System) will be used to get location of the nearby blood bank, which would make it easier for the seeker to find the blood when needed in emergency situations.

2.3 Objectives

The main objective of this cloud computing based web and android application is to help satisfy a blood request made from anywhere and anytime, by maintaining all information. This system provides transparency in this field, ie, makes the process of obtaining blood from a blood bank, corruption free and makes blood bank management effective. The

system intends to make the blood search process much more efficient and quick. Therefore, no permanent registration to the website is needed for the requester, they are only required to provide their basic details and contact information for verification. A single platform for maintaining all genuine blood related activities and information increases the trust of the public to get involved in these activities, and to participate in blood donations.

Chapter 3

IMPLEMENTATION

The project uses Pycharm to develop the web application for the blood bank and Flask server as framework to connect the code and the User Interface . It used Android Studio to develop the android application and it will be saved as an apk file for implementing it in the android phone. The back-end is python and server is python flask. The cloud used for storage is Amazon Web Services(AWS) S3

3.1 Technologies Used

3.1.1 PyCharm

PyCharm is a hybrid platform developed by JetBrains as an IDE for Python. It is commonly used for Python application development.An IDE consists of an editor and a compiler that we use to write and compile programs. It has a combination of features required for developing software.The presence of an IDE makes the development process and programming much easier. It interprets what we are typing and suggests the relevant keyword to insert. We can run PyCharm on Windows, Linux, or Mac OS. Additionally, it contains modules and packages that help programmers develop software using Python in less time and with minimal effort.Further, it can also be customized according to the requirements of developers.

3.1.2 Flask (Python Framework)

Flask is a web framework. This means flask provides you with tools, libraries and technologies that allow you to build a web application. This web application can be some web pages, a blog, a wiki or go as big as a web-based calendar application or a commercial website.

Flask is part of the categories of the micro-framework. Micro-framework are normally framework with little to no dependencies to external libraries. This has pros and cons. Pros would be that the framework is light, there are little dependency to update and watch for security bugs, cons is that some time you will have to do more work by yourself or increase yourself the list of dependencies by adding plugins.

3.1.3 SQL Server

SQL stands for Structured Query Language. SQL is used to communicate with a database. According to ANSI (American National Standards Institute), it is the standard language for relational database management systems. SQL statements are used to perform tasks

such as update data on a database, or retrieve data from a database.

Some common relational database management systems that use SQL are: Oracle, Sybase, Microsoft SQL Server, Microsoft Access, Ingres, etc.Although most database systems use SQL, most of them also have their own additional proprietary extensions that are usually only used on their system. However,the standard SQL commands such as ”Select”, ”Insert”, ”Update”, ”Delete”, ”Create”, and ”Drop” can be used to accomplish almost everything that one needs to do with a database.

3.1.4 Android Studio

Android Studio is the official integrated development environment (IDE) for Android application development. It is based on the IntelliJ IDEA, a Java integrated development environment for software, and incorporates its code editing and developer tools. To support application development within the Android operating system, Android Studio uses a Gradle-based build system, emulator, code templates, and Github integration. Every project in Android Studio has one or more modalities with source code and resource files. These modalities include Android app modules, Library modules, and Google App Engine modules.

Android Studio uses an Instant Push feature to push code and resource changes to a running application. A code editor assists the developer with writing code and offering code completion, refraction, and analysis. Applications built in Android Studio are then compiled into the APK format for submission to the Google Play Store.

3.1.5 Python

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages. Some of the key advantages of learning Python are:
Python is Interpreted : Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

Python is Interactive : You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
Python is Object-Oriented : Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
Python is a Beginner's Language : Python is a great language for the beginner level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

3.1.6 Java

Java is a language which is used at a major scale for the development of mobile applications. In our IDE i.e. Android Studio, Java is default language to code. JAVA is a programming language which is used in Android App Development. It is class based and object oriented programming whose syntax is influenced by C++. The primary goals of JAVA is to be simple, object-oriented, robust, secure and high level. JAVA application runs on JVM (JAVA Virtual Machine) but Android has it's own virtual machine called Dalvik Virtual Machine (DVM) optimized for mobile devices.

Java and XML are used in the development of a mobile application. It is used in Android development to write the back-end logic or business logic. It is a high-level language. XML is used to design the UI(User Interface). It is an object-oriented language but not pure object-oriented because of the presence of built-in data types in Java. This is called object oriented because it supports all of the features of object-oriented language.

3.1.7 Amazon Web Services(AWS) S3

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. Customers of all sizes and industries can use Amazon S3 to store and protect any amount of data for a range of use cases, such as data lakes, websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides management features so that you can optimize, organize, and configure access to your

data to meet your specific business, organizational, and compliance requirements.

Amazon S3 is an object storage service that stores data as objects within buckets. An object is a file and any metadata that describes the file. A bucket is a container for objects.To store your data in Amazon S3, you first create a bucket and specify a bucket name and AWS Region. Then, you upload your data to that bucket as objects in Amazon S3. Each object has a key (or key name), which is the unique identifier for the object within the bucket.

S3 provides features that you can configure to support your specific use case. For example, you can use S3 Versioning to keep multiple versions of an object in the same bucket, which allows you to restore objects that are accidentally deleted or overwritten.Buckets and the objects in them are private and can be accessed only if you explicitly grant access permissions. You can use bucket policies, AWS Identity and Access Management (IAM) policies, access control lists (ACLs), and S3 Access Points to manage access.

3.2 System Architecture

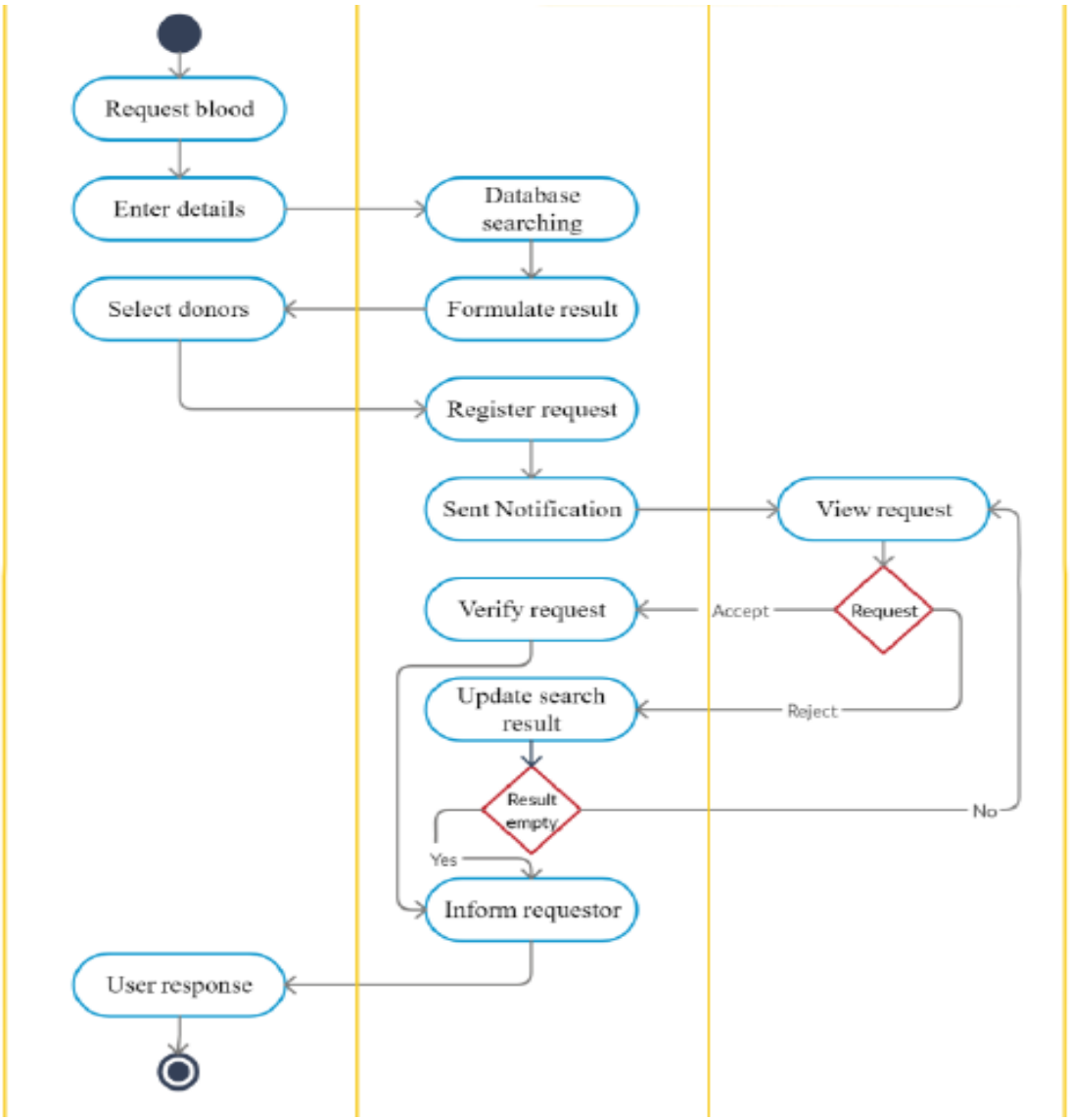


Figure 3.1: System architecture

3.3 DataBase

The data requirements is very high for the project.The database is created and maintained as it provides the mechanism of storage and retrieval of data.The database will contain the information regarding all the entities. This database server would also provide facility to store large data.

The project include a database . This system is supported by a MySQL database to store blood bank and user specific details. So our system maintains a database to store the details of blood banks, blood requests etc. and this database is updated consistently.

Our website makes use of this database to locate and find the nearest blood bank in case of emergency blood requests. So a database is created and it's name is blooddonation . There are eight tables in the database. They are:

• login

• bloodbank

• bloodrequest

• camp

• complaint

• feedback

• location

• userregistration

3.4 Modules

3.4.1 login and logout

The admin and blood bank can login and logout from the web-application. The user can login and logout from the android application.

3.4.2 Manage Blood Bank

The admin can to the blood bank. Only Admin can accept/verify the blood bank, i.e. After the blood bank registration, the blood bank will be verified by admin and after that only the blood bank will be in use. The admin can block and unblock the blood bank according to their working.

3.4.3 Request for blood

The user can request for blood from the android application from their user account. The user details will be taken from their account and they need to give the requesting blood group and their required unit.

3.4.4 Registration

The blood bank can register from the web-application. The user can register from the android application.

3.4.5 Contact Information

The contact information of the blood bank and the user were given

3.4.6 Searching Nearby hospitals

The nearby hospitals are searched by user from android application. The location is calculated using the latitude and longitude of the blood bank. In android it uses Global Positioning Service (GPS).

Chapter 4

RESULT ANALYSIS

The project was successfully finished on schedule. Witnessing someone desperately searching for a blood donor, when a particular blood group is unavailable in the blood bank and the donors in mind are out of reach, is a painful and helpless situation. Losing a life just because a donor was not available at the most needed moment is an unbearable experience. Our mission is, to fulfill every blood request in the country with a promising web portal and, try to motivate individuals who are willing to donate blood. So our idea of a cloud-based blood bank management system solves most of the key issues existing in this sector.More the people who participate and use the system, the more efficient the system becomes. The system creates a direct bridge between the donor, blood bank and the recipient. The health sector will be definitely benefited by the services provided by the system as the patient's safety and life is considered valuable.

Chapter 5

CONCLUSION AND FUTURE SCOPE

5.1 Conclusion

This paper proposed the reliable online cloud-based blood bank system. Latest technology and information system plays a vital role in blood bank system and its services, as its quality improves. The system is beneficial for both requester and donor too. Due to this System, the bridge between donor and the requester and blood bank is reduced and their communication improves. Thus, providing the requested blood on time to the requester, when needed. The health sector will be definitely benefited by the services provided by the system as patients safety and life is considered valuable.The purpose of the project is, sometimes patients life is at risk if the appropriate amount of blood is not made available to him whenever needed. Even if blood units are present in the blood bank and the requester is not aware of it, then it is of no use. This system prevents such situations, as every requester will be able to know about the blood bank and blood unit nearby. The GPS Technology will be used to make the nearby blood bank location visible to the requester. The database containing all information about the blood bank's location, diffrent information etc will be maintained and updated.

5.2 Future Scope

In future, the service provided by the system is needed to be carried on with the SMS services. In the area where still people are not connected to the internet, this SMS service will be useful for them. The donor will receive an SMS from the seeker. The contact detail of the seeker will be encoded in some other form. The main purpose is to provide this blood bank facility without internet access.

.

Chapter 6

APPENDIX

6.1 Source code

webcode.py

```
from flask import*
app=Flask(__name__)
app.secret_key="aaa"

from  src.dbconnection import *

import functools
def login_required(func):
    @functools.wraps(func)
    def secure_function():
        if "lid" not in session:
            return redirect ("/")
        return func()
    return secure_function
@app.route('/logout')
```

```
def logout():
    session.clear()
    return render_template("login.html")

@app.route('/')
def main():
    return render_template("login.html")

@app.route('/login',methods=['post'])
def login():
    uname=request.form['uname']
    pswd=request.form['password']
    qry="select*from login where username=%s and
                                password=%s"
    val=(uname,pswd)
    res=selectone(qry,val)
    print(res)
    if res is None:
        return'''<script>alert("invalid");window.
        location='/'</script>'''
    elif res[3]=='admin':
        session['lid']=res[0]
        return'''<script>alert("welcome admin");
        window.location='/admin_home'</script>'''
    elif res[3]=='blood_bank':
        session['lid']=res[0]

  return'''<script>alert("welcome Blood bank");
  window.location='/h_home'</script>'''
```

```
    else:
    return '''<script>alert("invalid");
    window.location='/'</script>'''


@app.route('/admin_home')
@login_required
def admin_home():
    return render_template("admin/admin home.html")


@app.route('/h_home')
@login_required
def h_home():
    return render_template
    ("blood bank/blood bank home.html")
@app.route('/reg1',methods=['post'])
def reg1():
    try:
        name=request.form['textfield']
        place=request.form['textfield0']
        email=request.form['textfield3']
        phone=request.form['textfield2']
        uname=request.form['textfield6']
        passd=request.form['textfield8']
        longitude=request.form['textfield4']
        latitude=request.form['textfield5']

        q="insert into login values
            (null,%s,%s,'pending')"
        v=uname,passd
```

```
            id=iud(q,v)
            q2="insert into location values
                    (null,%s,%s,%s)"
            vv=id,latitude,longitude
            iud(q2,vv)
            q1="insert into blood_bank values
             (null,%s,%s,%s,%s,%s)"
            v1=str(id),name,place,email,phone
            iud(q1,v1)
            return '''<script>alert("success...");
            window.
        location='/'</script>'''
        except Exception as e:
            print(e)
            q="delete from login where id=%s"
            iud(q,id)
            q1="delete from location where h_lid=%s"
            iud(q1,id)
            return '''<script>alert("duplicate entry...");
            window.location='/'</script>'''


@app.route('/reg')
def reg():
    return render_template("registration.html")


@app.route('/acceptbank')
@login_required
def acceptbank():
    q="SELECT `blood_bank`.* FROM `blood_bank` JOIN
```

```
    `login` ON `login`.`id`=`blood_bank`.`login_id`
    WHERE `login`.`type`='pending'"
    res=select(q)
    return render_template("admin/blood bank
    accept or reject.html",val=res)

@app.route('/blocktbank')
@login_required
def blocktbank():
    q="SELECT `blood_bank`.*,login.type
    FROM `blood_bank`
    JOIN `login` ON `login`.`id`=`blood_bank`
    .`login_id` WHERE `login`.`type`='blood_bank' or
    `login`.`type`='block'"
    res=select(q)
    return render_template
    ("admin/block or unblock.html",val=res)

@app.route('/block')
@login_required

def block():
    id=request.args.get('id')
    q="update login set type='block'
    where id=%s" iud(q,id)
    return '''<script>alert("Accepted");
    window.location='/blocktbank#ntro'</script>'''
@app.route('/unblock')
@login_required
```

```
def unblock():
    id=request.args.get('id')
    q="update login set type='blood_bank'
    where id=%s"
    iud(q,id)
    return '''<script>alert("Accepted");window.
    location='/blocktbank#ntro'</script>'''



@app.route('/accept')
@login_required

def accept():
    id=request.args.get('id')
    q="update login set type='blood_bank'
    where id=%s"
    iud(q,id)
    return '''<script>alert("Accepted");window.
    location='/acceptbank'</script>'''

@app.route('/reject')
@login_required
def reject():
    id=request.args.get('id')
    q="update login set type='reject'
    where id=%s"
    iud(q,id)
```

```
    return '''<script>alert("Accepted");window.
    location='/acceptbank'</script>'''



@app.route('/viewuser')
@login_required
def viewuser():
    q="select * from user_registration"
    res=select(q)
    return render_template("admin/admin view user.
    html",val=res)



@app.route('/reply')
@login_required
def reply():
    id = request.args.get('id')
    session['cmpid']=id
    return render_template("admin/reply.html")



@app.route('/replys',methods=['post'])
@login_required
def replys():
    reply=request.form['text']
    q="update complaint set reply=%s where cid=%s"
    v=reply,session['cmpid']
    iud(q,v)
```

```
    return '''<script>alert("Replied");window.
    location='/viewcomplaint#intro'</script>'''



@app.route('/reply1')
@login_required
def reply1():
    id = request.args.get('id')
    session['cmpid']=id
    return render_template("bood bank/reply.html")


@app.route('/replys1',methods=['post'])
@login_required
def replys1():
    reply=request.form['text']
    q="update table set complaint=%s where cid=%s"
    v=reply,session['cmpid']
    iud(q,v)

    return '''<script>alert("Replied");window.
    location='/viewcomplaint1'</script>'''


@app.route('/updatecamp',methods=['post'])
@login_required
def updatecamp():
```

```
    name=request.form['textfield']
    place=request.form['textfield2']
    date=request.form['textfield3']
    time=request.form['textfield4']
    q="update camp set camp=%s,place=%s,
     date=%s,time=%s
    where campid=%s"
    v=name,place,date,time,session['campid']
    iud(q,v)
    return '''<script>alert("updated");window.
    location='/viewcamp#intro'</script>'''


@app.route('/addcamp1',methods=['post'])
@login_required
def addcamp1():
    name=request.form['textfield']
    place=request.form['textfield2']
    date=request.form['textfield3']
    time=request.form['textfield4']
    q="insert into camp values(null,%s,%s,%s,%s,%s)"
    v=session['lid'],name,place,date,time
    iud(q,v)
    return '''<script>alert("Added");window.
    location='/viewcamp#intro'</script>'''

app.run(debug=True)
```

android.py

```
from src.sample import cloud_upload
from flask import *
from src.dbconnection import *

app=Flask(__name__)
@app.route("/logincode",methods=['post'])
def logincode():
    uname=request.form['username']
    password=request.form['password']
    q="select * from login where username=%s
    and password=%s "
    val=uname,password
    res=selectone(q,val)
    if res is None:
        return jsonify({'task': 'invalid'})
    else:
        return jsonify({'task': 'success',
        'lid':res[0],'type':res[3]})

@app.route("/register",methods=['post'])
def register():
    try:
        print(request.form)
        print(request.files)
        fname=request.form['fname']
        lname=request.form['lname']
        Gender=request.form['Gender']
```

```
            Place=request.form['Place']
            Post=request.form['Post']
            Pin=request.form['Pin']
            Phone=request.form['Phone']
            Email=request.form['Email']
            username=request.form['username']
            password=request.form['password']
            lid=0
            qry = "insert into login values
               (null,%s,%s,'user')"
            value = (username, password)
            lid = iud(qry, value)
            qry1 = "insert into user_registrationvalues
              (null,%s,%s,%s,%s,%s,%s,%s,%s,%s)"
            val = (str(lid),fname, lname,Gender,Place,
            Post,Pin,Phone,Email)
            iud(qry1, val)
            return jsonify({'task': 'success'})
        except Exception as e:
            print(e)
            q="delete from login where id=%s"
            iud(q,lid)
            return jsonify({'task': 'error'+e})

@app.route('/nearestbank',methods=['post'])
def nearestbank():
    print(request.form)
    lati=request.form['lati']
    longi=request.form['longi']
```

```
    v=lati ,longi , lati
    q="SELECT `blood_bank`.*,(3959 * ACOS
    ( COS ( RADIANS(%s) ) * COS( RADIANS
    ( `location`.`lattitude`) ) * COS
    ( RADIANS(`location`.`longitude` ) -
    RADIANS(%s) ) + SIN ( RADIANS(%s) )
    * SIN( RADIANS( `location`.`lattitude` ) )))
    AS user_distance FROM `blood_bank` JOIN `location`
    ON `location`.`h_lid`=`blood_bank`.`login_id`
    HAVING user_distance  < 31.068"
    res=androidselectall(q,v)
    print(res)
    return jsonify(res)

@app.route('/bloodrequest',methods=['post'])
def bloodrequest():
    uid=request.form['uid']
    bankid=request.form['bnkid']
    blood=request.form['blood']
    unit=request.form['unit']
    txt=str(uid)+" "+str(bankid)+" "+str(blood)+" "
    +unit from datetime import datetime
    fn="static/file/"+datetime.now().strftime
    ("%Y%m%d%H%M%S")+".txt"
    with open(fn, 'w') as f:
        f.write(txt)
    cloud_upload(fn)
    q="insert into blood_request values
    (null,%s,%s,%s,%s,curdate(),'pending','0')"
```

```
    v=uid,bankid,blood,unit
    iud(q,v)
    return jsonify({'task': 'success'})

@app.route('/requeststatus',methods=['post'])
def requeststatus():
    uid=request.form['lid']
    q="SELECT `blood_bank`.*,`blood_request`.*
    FROM `blood_request` JOIN `blood_bank`
    ON `blood_bank`.`login_id`=`blood_request`.
    `bloodbank_id` WHERE `blood_request`.`uid`=%s "
    res=androidselectall(q,uid)
    return jsonify(res)

@app.route('/user',methods=['post'])
def user():
    rid=request.form['rid']
    print(rid)
    q="SELECT `firstname`,`lastname`,`place`,`phone`
    FROM `user_registration` JOIN `blood_request`
    ON `blood_request`.`accepted_id`=`user_registration`.
    `lid` WHERE `blood_request`.`rid`=%s"
    res=androidselectall(q,rid)
    print(res)
    return jsonify(res)

@app.route('/viewrequest',methods=['post'])
def viewrequest():
    print(request.form)
```

```
    lid=request.form['lid']
    lati=request.form['lati']
    longi=request.form['longi']
    v=lati,longi,lati,lid
    q="SELECT `blood_bank`.*,`user_registration`.
    `firstname`,`user_registration`.`lastname`,
    `user_registration`.`phone`,blood_request.*,
    (3959 * ACOS ( COS ( RADIANS(%s) ) *
    COS( RADIANS( `location`.`lattitude`) ) *
    COS( RADIANS(`location`.`longitude` ) -
    RADIANS(%s) ) + SIN ( RADIANS(%s) ) *
    SIN( RADIANS( `location`.`lattitude` ) )))
    AS user_distance FROM `blood_bank` JOIN `location`
    ON `location`.`h_lid`=`blood_bank`.`login_id`
    JOIN `blood_request` ON `blood_request`.
    `bloodbank_id`=`blood_bank`.`login_id`
    JOIN `user_registration` ON
    `user_registration`.`lid`=`blood_request`.`uid`
    where blood_request.status='pending' and
    blood_request.uid!=%s HAVING user_distance < 2031"
    res=androidselectall(q,v)
    print(res)
    return jsonify(res)

@app.route('/acceptrequest',methods=['post'])
def acceptrequest():
    acceptid=request.form['lid']
    rid=request.form['rid']
    v=acceptid,rid
```

```
    q="update blood_request set accepted_id=%s,
    status='accept' where rid=%s"
    iud(q,v)
    return jsonify({'task': 'success'})

@app.route('/feedback',methods=['post'])
def feedback():
    id=request.form['lid']
    feedback=request.form['feedback']
    q="insert into feedback values(null,%s,%s,curdate())"
    v=id,feedback
    iud(q,v)
    return jsonify({'task': 'success'})

@app.route('/complaint',methods=['post'])
def complaint():
    id=request.form['lid']
    complaint=request.form['complaint']
    q="insert into complaint values(null,%s,curdate(),
        %s,'pending')"
    v=id,complaint
    iud(q,v)
    return jsonify({'task': 'success'})

@app.route('/reply',methods=['post'])
def reply():
    lid=request.form['lid']
    q="select * from complaint where lid=%s "
    res=androidselectone(q,lid)
```

```
    return jsonify(res)

@app.route('/camp',methods=['post'])
def camp():
    q="select * from camp  "
    res=androidselectallnew(q)
    return jsonify(res)

@app.route('/search',methods=['post'])
def search():
    print(request.form)
    name=request.form['name']
    print("SELECT * FROM `blood_bank`  WHERE
          blood_bank_name like '"+name+"%'")
    q="SELECT * FROM `blood_bank`  WHERE
       blood_bank_name like '"+name+"%'"
    res=androidselectallnew(q)
    print(res)
    return jsonify(res)

app.run(port=5000,host="0.0.0.0")
```

cloud.py

```
import sys
from boto.s3.key import Key
import boto
import boto.s3
import  ast
import os
from flask.globals import request
from werkzeug.utils import secure_filename
from flask import Flask,jsonify

AWS_ACCESS_KEY_ID = 'AKIAVTA624LZNEGZPSAQ'
AWS_SECRET_ACCESS_KEY =
        'nRF9lOBROOhBQmAmkFc+xq9e6oR1BPiBeXgKteRV'

def cloud_down( fname,sname):
    bucket_name = 'samplebucket1riit'
    conn = boto.connect_s3(AWS_ACCESS_KEY_ID,
            AWS_SECRET_ACCESS_KEY)
    bucket = conn.create_bucket
    (bucket_name,location=boto.s3.
       connection.Location.DEFAULT)
    key = bucket.get_key(fname)
    key.get_contents_to_filename(sname)
    print("okkkkkkkkk")

def cloud_upload(fname):
    bucket_name = 'samplebucket1riit'
    conn = boto.connect_s3(AWS_ACCESS_KEY_ID,
```

```
                                AWS_SECRET_ACCESS_KEY)
    bucket = conn.create_bucket
    (bucket_name, location=boto.s3.connection.
                                    Location.DEFAULT)
    testfile =fname
    tes = testfile.split("/")
    leng = len(tes)
    namef = tes[leng - 1]
    print(namef,"========")

    def percent_cb(complete, total):
        sys.stdout.write('.')
        sys.stdout.flush()
    k = Key(bucket)
    k.key = namef
    k.set_contents_from_filename
    (testfile ,cb=percent_cb, num_cb=10)
    print("okkkkk++++++")
# path=r"C:\Users\Fayas\PycharmProjects\dedupserver
                                        \src\RSA.py"
# # path="C:\Users\Fayas\PycharmProjects\dedupserver
                                      \src\static\RSA.p"
# cloud_upload(path)
#
# cloud_down("RSA.py","RSA11.py")
```

6.2 Screenshots

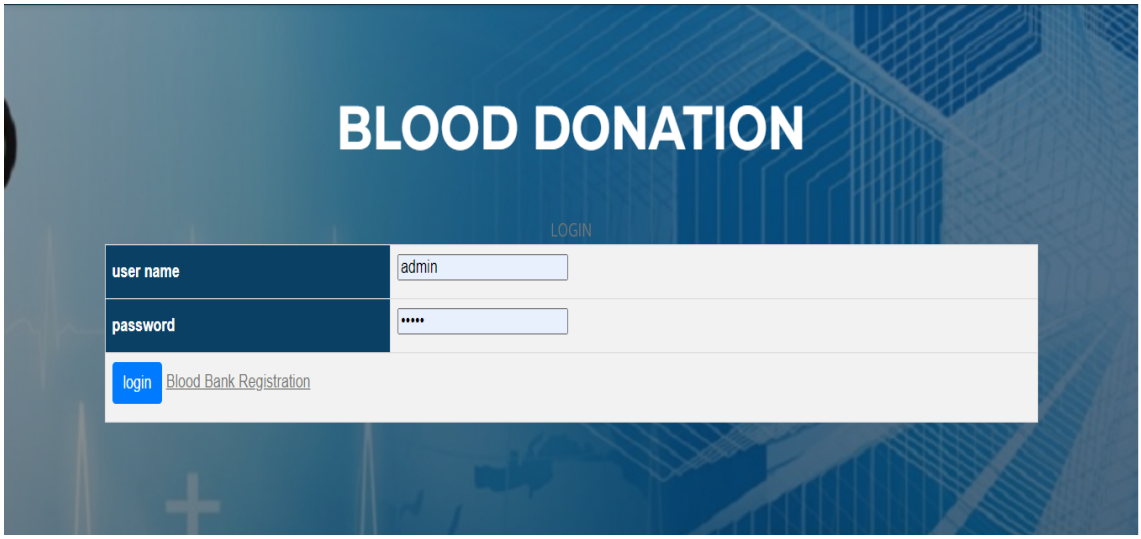


Figure 6.1: Login Page

Figure 6.2: Admin Page

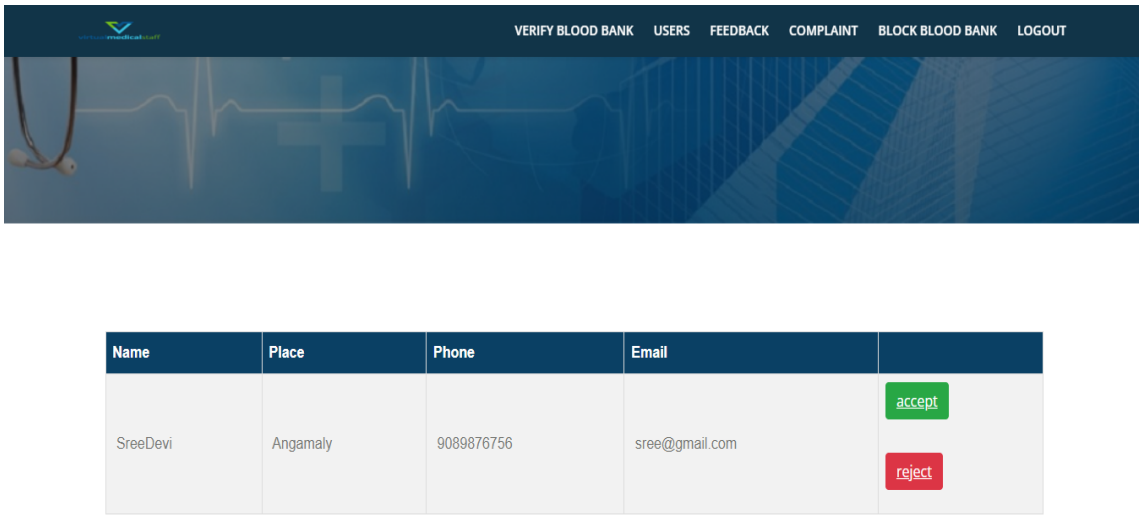


Figure 6.3: Blood Bank Verification

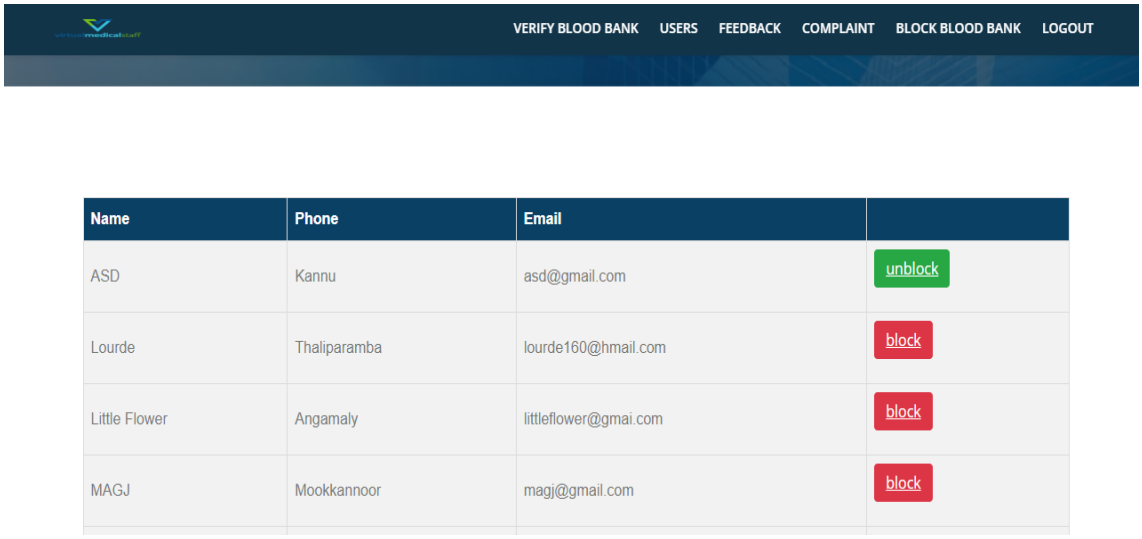


Figure 6.4: Block/Unblock Blood Bank

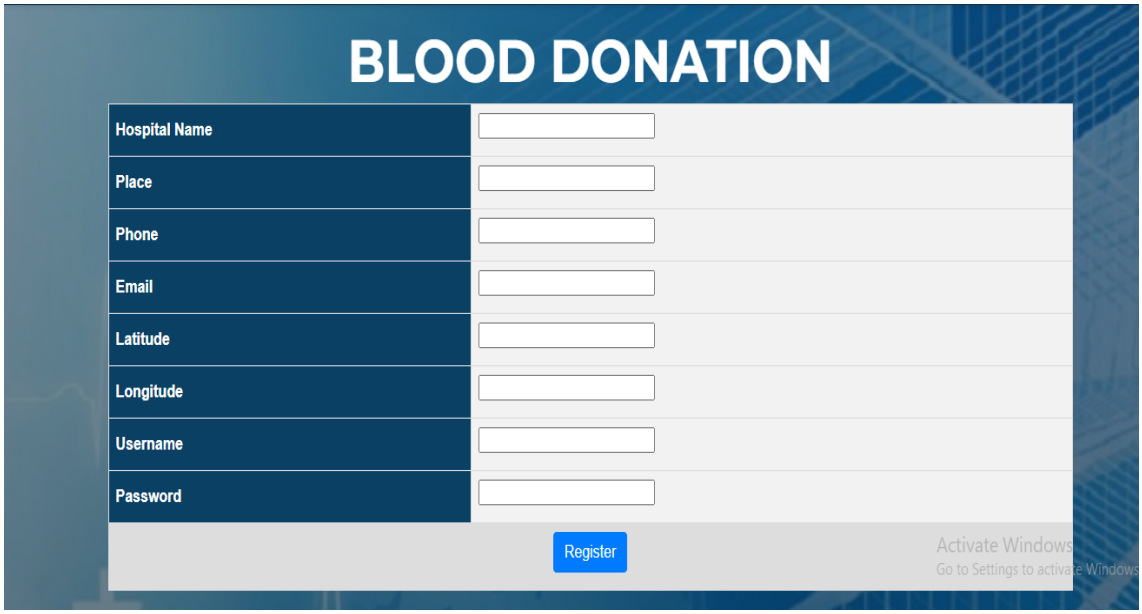


Figure 6.5: Blood Bank registration

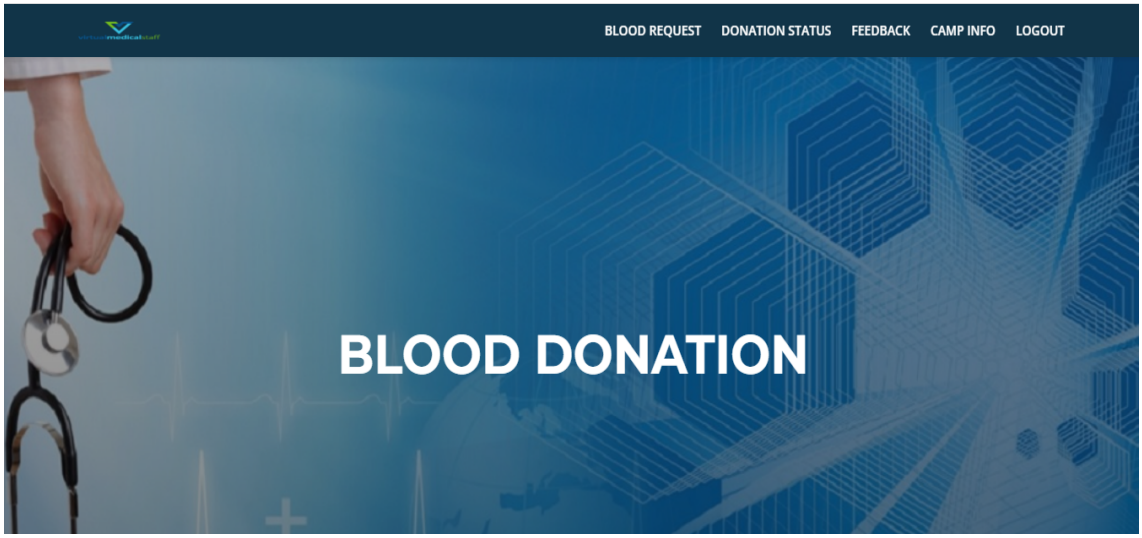


Figure 6.6: Blood Bank Page

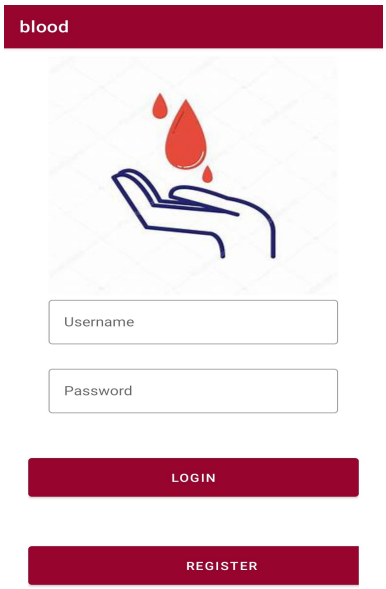


Figure 6.7: User Login

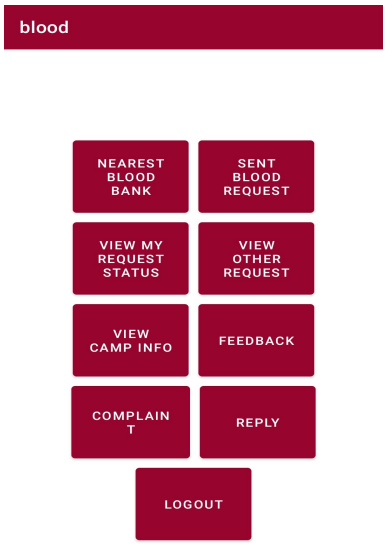
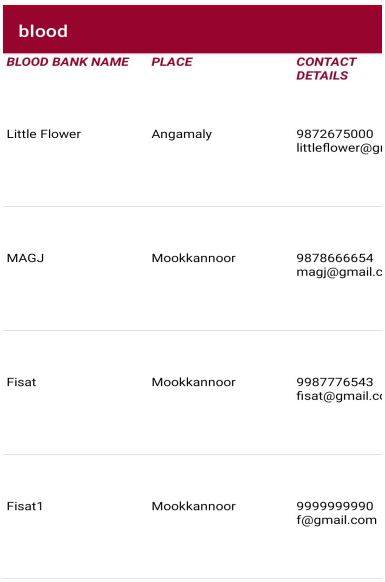


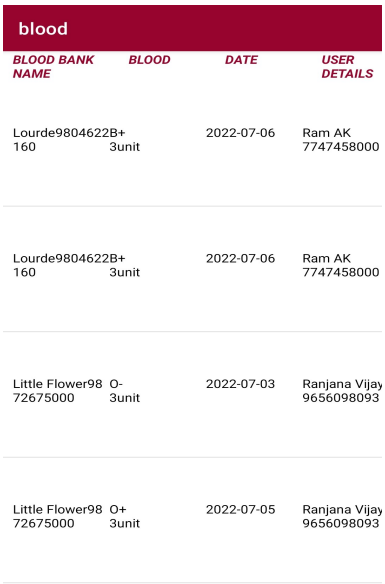
Figure 6.8: User Page



blood

BLOOD BANK NAME	PLACE	CONTACT DETAILS
Little Flower	Angamaly	9872675000 littleflower@gı
MAGJ	Mookkannoor	9878666654 magj@gmail.c
Fisat	Mookkannoor	9987776543 fisat@gmail.cc
Fisat1	Mookkannoor	9999999990 f@gmail.com

Figure 6.9: Nearby Blood bank



blood

BLOOD BANK NAME	BLOOD	DATE	USER DETAILS
Lourde9804622160	B+ 3unit	2022-07-06	Ram AK 7747458000
Lourde9804622160	B+ 3unit	2022-07-06	Ram AK 7747458000
Little Flower9872675000	O- 3unit	2022-07-03	Ranjana Vijay 9656098093
Little Flower9872675000	O+ 3unit	2022-07-05	Ranjana Vijay 9656098093

Figure 6.10: Accept Blood Requests

Figure 6.11: Camp Information

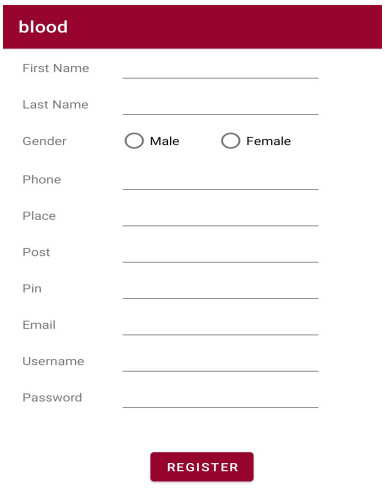


Figure 6.12: User Registration

Chapter 7

REFERENCES

[1] T.Hilda Jenipha and R.Backiyalakshmi, ”Android Blood Donor Life Saving Application in Cloud Computing,” American Journal of Engineering Research (AJER) 2014.

[2] The Optimization of Blood Donor Information and Management System by Technopedia P. Priya1, V. Saranya2, S. Shabana3, Kavitha Subramani4 Department of Computer Science and Engineering

[3] Shreyas Anil Chaudhari, Shrutika Subhash Walekar, Khushboo Ashok Ruparel,Vrushali Milind Pandagale (2018) A Secure Cloud Computing Based Framework for the Blood bank.,International Conference on Smart City and Emerging Technology.

[4] Sulaiman, S., Abdul Hamid, A. A. K., Najihah Yusri, N. A. (2015). Development of a Blood Bank Management System. Procedia - Social and Behavioral Sciences, 195, 2008-2013.

[5] Sagar Shrinivas, Vasaikar Vijay and Suresh Yennam, ”Online Blood Bank Using Cloud Computing,” International Journal of Advanced Research, Ideas and Innovation In Technology,(volume 3, Issue 1)

[6] Almetwally M. Mostafa, Ahmed E. Youssef, ".A Framework for a Smart Social Blood Donation System based on Mobile Cloud Computing,"

[7] Deepak Pandey, Achal Umare and Dr.R.S.Mangrulkar, "Requirement Based Blood Storage and Distribution System," International Journal of Research In Science Engineering Volume: 3 Issue: 2 March-April 2017

[8] Marinescu, D. C. (2018). Chapter 7 - Cloud Applications. In D. C. Marinescu (Ed.), Cloud Computing (Second Edition) (pp. 237-279): Morgan Kaufmann.