COLLEGE OF ENGINEERING & TECHNOLOGY
SRM INSTITUTE OF SCIENCE & TECHNOLOGY
S.R.M. NAGAR, KATTANKULATHUR – 603 203

BONAFIDE
CERTIFICATE

Certified that this project report **"Blood bank and Donor Management System"** is the bonafide work of **" Pranav Tatavarthy (RA2011031010011) Prithvi Singh Kirar (RA2011031010023) and Arghya Pahar (RA2011031010029)"**of III Year/VI Sem B.tech(CSE) who carried out the mini project work under my supervision for the course 18CSC303J- Database Management systems in SRM Institute of Science and Technology during the academic year 2022-2023(Even sem).

SIGNATURE
Dr.S.Thenmalar
Associate Professor
Networking and Communications

ABSTRACT

Blood transfusion safety is a relevant and significant public health issue in the India. Since most blood banks are still in a paper-based system, various disadvantages are experienced by various stakeholders, which endanger the lives of patients and deter the healthcare system. As such, the researchers aimed to design, develop, and implement a blood bank management system (BBMS). This web-based application allows hospitals in India to make inventories of their blood bags online, subsequently, allowing each hospital to check the availability of blood bags anytime. The researchers designed and administered a questionnaire that assessed the perceptions of various stakeholders in both manual-based and BBMS. Based on the findings and results, it was found out that these stakeholders perceived the blood bank management system is much better than the manual system. Therefore, with the use of a blood bank management system, the blood transfusion process is safe and secured. Threats on improper blood donor documentation, or misplaced records will be totally eradicated. Also, processes involving recording about blood donors, blood bag collection, storage, and inventory will be systematized and organized, hence, improving the healthcare management for blood banks.

TABLE OF CONTENTS

LIST OF FIGURES

INTRODUCTION

System Development:

The process of building systems has always been complex with system becoming larger, the costs and complexities get multiplied. So the need for better methods for developing systems is widely recognized to be effective and the applied model should meet a few basic requirements.

- The model should be structured and cover the entire system development process from feasibility study to programming, testing and implementation.
- The model should utilize established methods and techniques like database designs, normalizations and structured programming techniques.
- The model should consist of building blocks, which define tasks, results and interfaces.
- The model should separate the logical system from the physical system.
- Documentation should be a direct result of the development work and should be concise, precise and as non-redundant as possible.

Based on the above requirements of the system model, system study has been made. Various methodologies have been applied for system study.

Project:

The persons who like to donate blood registers in my site. The persons in need of blood searches for the persons having the same blood group and within the city. If he found a donor in his city then he gets the total details of the donor, if he doesn't find any donor then he is given the contact numbers and addresses of the Life Saving Contact Persons for major cities. In this projects we use PHP and Mysql and it contains two modules i.e Admin and Donor.

Admin Module

Dashboard: In this section, admin can view all the details in brief like total blood group listed, registered donor list and total enquiries received.

Blood Group: In this section, admin can manage blood group(Add/ Delete).

Donor List: In this section, admin can view list of donor and have right to delete and hide the detail of donor.

Manage Contact us Query: In this section, admin can manage query which is received by users.

Manage Pages: In this section, admin can website pages.

Update Contact info: In this section, admin can update the contact details of website.

Request Received by Donor: In this section, admin can view the request of blood which is received by donor.

Admin can also update his profile, change the password and recover the password.

Donor Module

Home: Its is welcome page for users and donor. If any users want to donate the blood they must register with us.

About Us: Users can view the about us page.

Contact Us: Users can contact with admin the through contact us page.

Donor List: Users can view and contact with donor.

Search Donor: Users can search the donor according to city and blood group.

Registered Users(Donor)

Home: Its is welcome page for users and donor. If any users want to donate the blood they must register with us.

About Us: Users can view the about us page.

Contact Us: Users can contact with admin the through contact us page.

Donor List: Users can view and contact with donor.

Search Donor: Users can search the donor according to city and blood group.

My Account:

- Profile
- Change Password
- Request Received
- Logout

OBJECTIVES

1. The people who like to donate blood register on the site.
2. The person in need of blood searches for the people having the same blood group and within the city.
3. If he finds a donor in his city then he gets the total details of the donor, if he doesn’t find any donor then he is given the contact numbers and addresses of the Life Saving Contact Persons for major cities.
4. In this project we will use PHP and MySQL and it contains two modules i.e Admin and Donor.

SCOPE OF PROJECT

1. More enhanced search algorithms can be implemented
2. The Encryption standards can also be used to make the transactions more secure.
3. More modules can be introduced in the backend to have better relationships between tables
4. News panels can be added – (regarding Blood donation camps , Blood testing centres , etc.)

SYSTEM REQUIREMENTS

Software Environment:

Software Environment is a technical specification of requirement of software product. This specifies the environment for development, operation and maintenance of the product

Technology used:

- Android Studio
- Kotlin
- Firebase
- XML
- Javascript
- Jetpack Compose
- MongoDB

Literature Survey

. Blood bank and donor management systems are critical components of the healthcare system, providing blood products to patients who need them. Effective management of blood donations and the blood supply chain is essential for ensuring that patients receive the right blood products at the right time. To achieve this, blood banks and donor management systems must overcome a range of challenges, including managing the donor population, ensuring the safety and quality of blood products, and improving the efficiency of the supply chain.

One of the key challenges facing blood banks is managing the donor population. Donor recruitment and retention are critical components of this process, as blood banks rely on a steady supply of donors to meet patient needs. Donor recruitment involves identifying potential donors, educating them about the importance of blood donation, and encouraging them to donate. Effective donor retention strategies are also necessary to maintain a stable donor population, as donors may not donate regularly or may stop donating altogether. To achieve these goals, blood banks often use a range of outreach strategies, such as advertising campaigns, community events, and social media, to raise awareness about the importance of blood donation and encourage donors to participate.

Another important aspect of blood bank and donor management is donor screening. Donors must be screened for a range of infectious diseases, such as HIV, hepatitis B and C, and syphilis, to ensure the safety and quality of the blood products. Donor screening involves a series of tests and

questionnaires to identify potential risks and ensure that donors are healthy and safe to donate. Blood banks also provide post-donation care to donors, including monitoring their health and providing support and advice if any issues arise.

In addition to managing the donor population, blood banks must also ensure the safety and quality of blood products. This involves a range of processes, including blood collection, testing, processing, and storage. Blood collection involves taking a sample of blood from a donor, which is then tested for infectious diseases and processed into various blood products, such as red blood cells, plasma, and platelets. These products must be stored in appropriate conditions to ensure their safety and quality. Blood banks must also manage the supply chain, ensuring that blood products are transported and distributed to hospitals and other healthcare facilities in a timely and efficient manner.

To address these challenges, blood banks and donor management systems are increasingly turning to technology. Automated systems, such as electronic health records, donor management software, and supply chain management software, can help to improve the efficiency and effectiveness of blood bank and donor management systems. These systems can streamline the donation process, reduce errors and delays, and provide real-time data on the blood supply chain.

In conclusion, blood bank and donor management systems play a critical role in the healthcare system, providing blood products to patients who need them. Effective management of blood donations and the blood supply chain is essential for ensuring that patients receive the right blood products at the right time. To achieve this, blood banks and donor management

systems must overcome a range of challenges, including managing the donor population, ensuring the safety and quality of blood products, and improving the efficiency of the supply chain. Automated systems are increasingly being used to address these challenges, providing a range of benefits to blood banks and donor management systems.

2.1 Existing Systems:

There are several existing systems that are used for blood bank and donor management. These systems are designed to streamline the donation process, manage the donor database, and monitor the blood supply. In this section, we will discuss some of the existing systems used for blood bank and donor management.

1. Hematos I

Hematos I is a web-based blood bank and donor management system that is widely used in many countries. It provides comprehensive features for managing donor records, blood inventory, and blood processing. Hematos I also provides a user-friendly interface for managing blood bank operations and generates reports to provide insights into the blood bank's performance. The system can be accessed from any web-enabled device, and it is highly scalable, making it suitable for both small and large blood banks.

2. HemaGo

HemaGo is a mobile application designed to help donors manage their donation history, track their blood type, and receive alerts about upcoming blood drives. The app also provides users with information about their eligibility to donate blood and helps them find nearby blood donation centers. HemaGo can be downloaded on both iOS and Android devices, and it is free to use.

3. Blood Bank Management System (BBMS)

BBMS is a comprehensive blood bank management system designed to manage all aspects of blood bank operations, including blood donation, blood processing, blood testing, and blood distribution. BBMS provides a user-friendly interface for managing blood bank operations and generates reports to provide insights into the blood bank's performance. The system also integrates with blood testing laboratories, allowing for real-time monitoring of the blood supply.

4. LifeServe Blood Center

LifeServe Blood Center is a nonprofit organization that operates blood donation centers across the United States. The organization uses a cloud-based donor management system that helps manage donor records, track donation history, and send alerts to donors about upcoming blood drives. The system also provides real-time data about the blood supply, allowing LifeServe to monitor the inventory and ensure that an adequate supply of blood is available.

5. DonorPerfect

DonorPerfect is a donor management system designed for nonprofits, including blood banks. The system provides a comprehensive set of features for managing donor records, tracking donations, and generating reports. DonorPerfect is highly customizable, allowing blood banks to tailor the system to their specific needs. The system also provides integration with various marketing and communication tools, allowing blood banks to reach out to donors and promote blood donation.

Blood bank and donor management systems are essential tools for ensuring the availability of safe blood for patients in need. These systems are designed to streamline the donation process, manage the donor database, and monitor the blood supply. There are several existing systems used for blood bank and donor management, including Hematos I, HemaGo, BBMS, LifeServe Blood Center, and DonorPerfect. These systems provide comprehensive features for managing blood bank operations and generating reports to provide insights into the blood bank's performance. Blood banks must choose the system that best suits their needs and ensure that it is implemented correctly to ensure the efficient functioning of the blood bank.

2.2 Comparison of Existing vs Proposed system

Enhanced Donor Experience:

The proposed solution can enhance the donor experience by providing real-time information about blood drives and donation centers in their area. This feature can help donors make informed decisions about where and when to donate blood, resulting in increased donations and improved blood supply. In contrast, existing systems may not provide such detailed information, leading to donors being unaware of blood drives and donation centers in their area.

Improved Inventory Management:

The proposed solution can improve inventory management by providing real-time data on blood supply and demand in each location. This feature can help blood banks manage their inventory more efficiently, ensuring that an adequate supply of blood is available at all times. In contrast, existing systems may not provide such granular data, leading to difficulties in managing inventory and resulting in shortages or wastage of blood.

Faster Response Times:

The proposed solution can provide faster response times to emergencies by quickly identifying the nearest blood donation centers and available blood supplies. This feature can help save lives in critical situations where time is of the essence. In contrast, existing systems may take longer to identify the nearest blood donation centers and available blood supplies, resulting in delays that can have serious consequences.

Customizable Alerts:

The proposed solution can provide customizable alerts to donors, notifying them about blood drives and donation centers in their area. This feature can help blood banks reach out to potential donors more effectively, resulting in increased donations and improved blood supply. In contrast, existing systems may not provide such customizable alerts, leading to donors being unaware of blood drives and donation centers in their area.

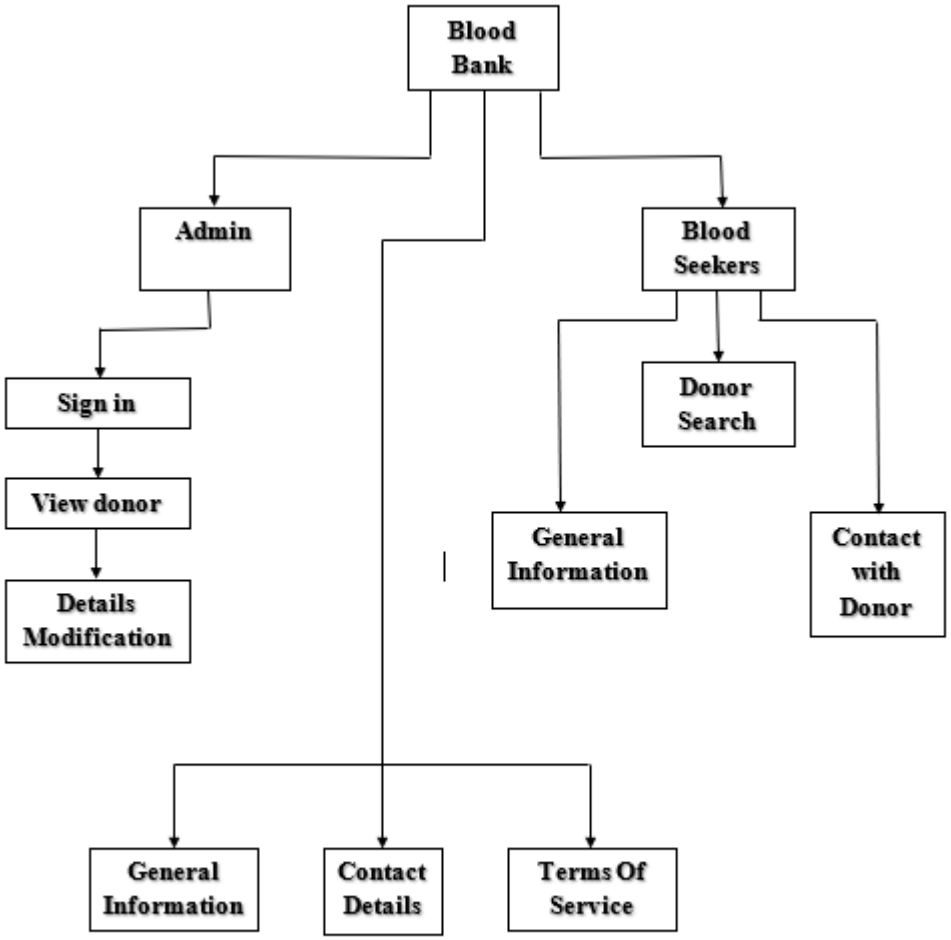
User-friendly Interface:

The proposed solution can provide a user-friendly interface for managing blood bank operations, making it easier for blood bank staff to manage donor records, blood inventory, and blood processing. This feature can improve the efficiency of blood bank operations, resulting in improved blood supply and better patient outcomes. In contrast, existing systems may not provide such a user-friendly interface, leading to difficulties in managing blood bank operations and resulting in inefficiencies.

Conclusion:

In conclusion, the proposed solution for blood bank and donor management that is location-based can offer several advantages over existing systems. The proposed solution can enhance the donor experience, improve inventory management, provide faster response times to emergencies, provide customizable alerts, and offer a user-friendly interface for managing blood bank operations. These advantages can result in increased donations, improved blood supply, better patient outcomes, and overall efficiency in blood bank operations. Blood banks must consider adopting the proposed solution to ensure that they can provide safe and timely blood to patients in need.

3. System Architecture and Design



3.1.1 Front End Design

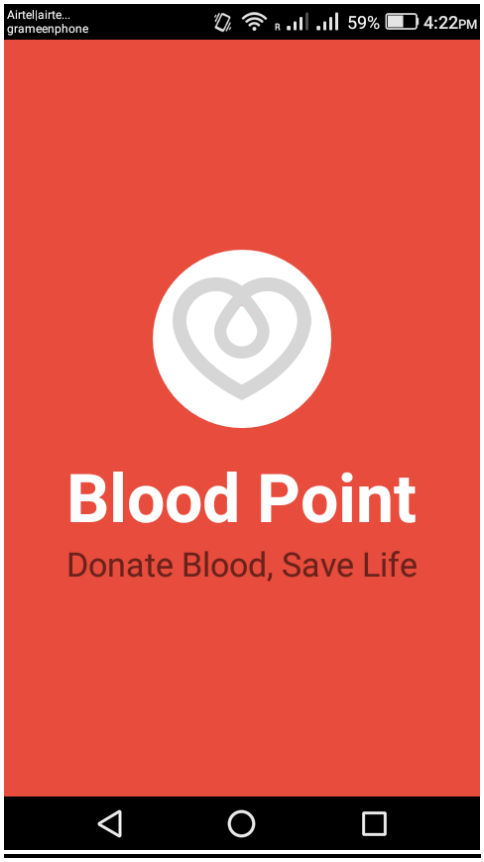


Fig 1

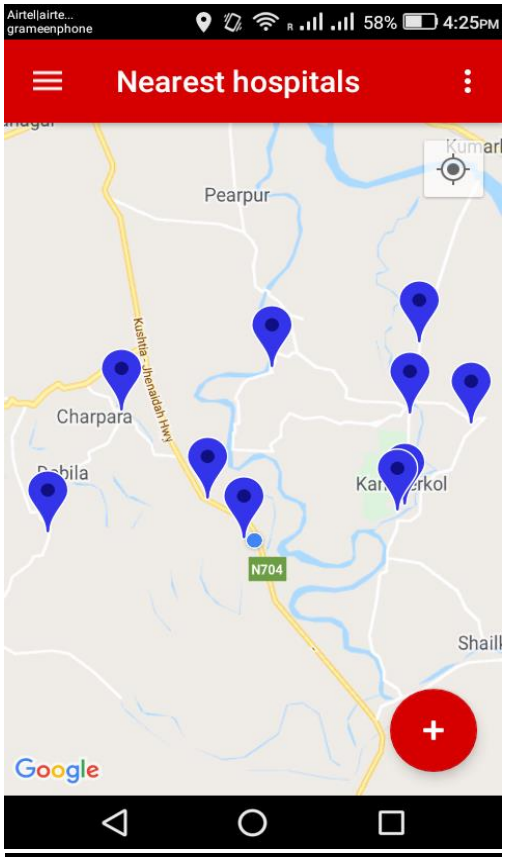

Airtel|airte...
grameenphone
58%
4:25PM
Nearest hospitals
Pearpur
Charpara
Kushtia - Jhenaidah Hwy
N704
Google

Fig 2

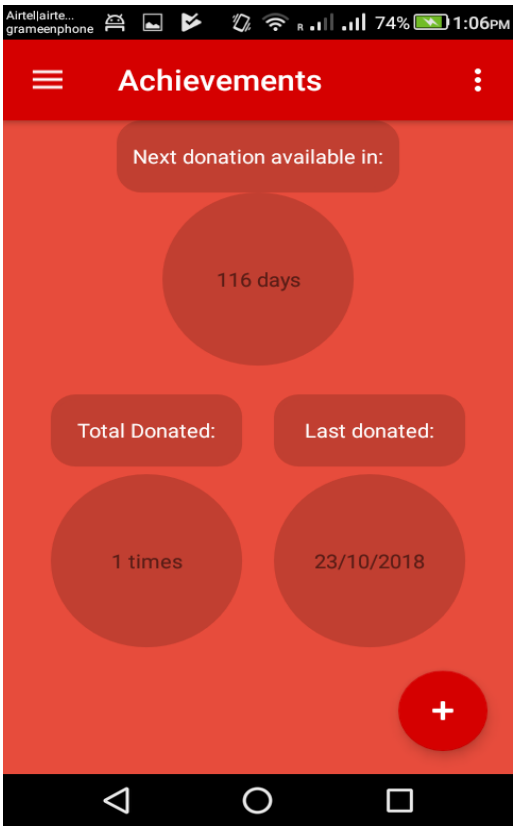

Airtel|airte...
grameenphone
74%
1:06PM
Achievements
Next donation available in:
116 days
Total Donated:
Last donated:
1 times
23/10/2018

Fig 3

3.1.2 Back end (Database) design

The backend design of a blood bank management system is a critical component in ensuring the efficient and effective management of blood inventory and donor information. It involves the design and implementation of a database schema, data storage and retrieval mechanisms, as well as data processing and analysis functions.

Database schema design:
The first step in designing the backend of a blood bank management system is to design a database schema that will efficiently store all relevant information about donors, blood units, and inventory levels. The schema should be designed to optimize data retrieval and processing, and ensure data integrity and security. It should also be scalable to accommodate the growth of the blood bank over time.

Data storage and retrieval:
Once the database schema is designed, the next step is to implement the data storage and retrieval mechanisms. This involves setting up a database server and configuring it to store and retrieve data according to the schema. It also involves setting up backup and recovery mechanisms to ensure that data is not lost in the event of a hardware or software failure.

Data processing and analysis:
In addition to data storage and retrieval, the backend of a blood bank management system should also include data processing and analysis functions. These functions can include inventory management, donor screening, blood typing and cross-matching, and reporting and analytics. Inventory management functions can include tracking inventory levels,

expiration dates, and shipment and delivery of blood units. Donor screening functions can include tracking donor eligibility and screening results, as well as ensuring compliance with regulatory requirements. Blood typing and cross-matching functions can include automated testing and analysis to ensure that the correct blood type is matched with the correct recipient. Reporting and analytics functions can include generating reports on inventory levels, donor demographics, and blood usage trends, as well as providing data visualizations to help identify patterns and trends.

Security and access control:
Finally, the backend of a blood bank management system should include robust security and access control mechanisms to ensure the confidentiality, integrity, and availability of sensitive donor and inventory information. This can include user authentication and authorization, data encryption, and audit trails to track access and usage of the system.

In conclusion, the backend design of a blood bank management system is critical in ensuring the efficient and effective management of blood inventory and donor information. A well-designed database schema, data storage and retrieval mechanisms, data processing and analysis functions, and security and access control mechanisms are essential components of a robust blood bank management system.

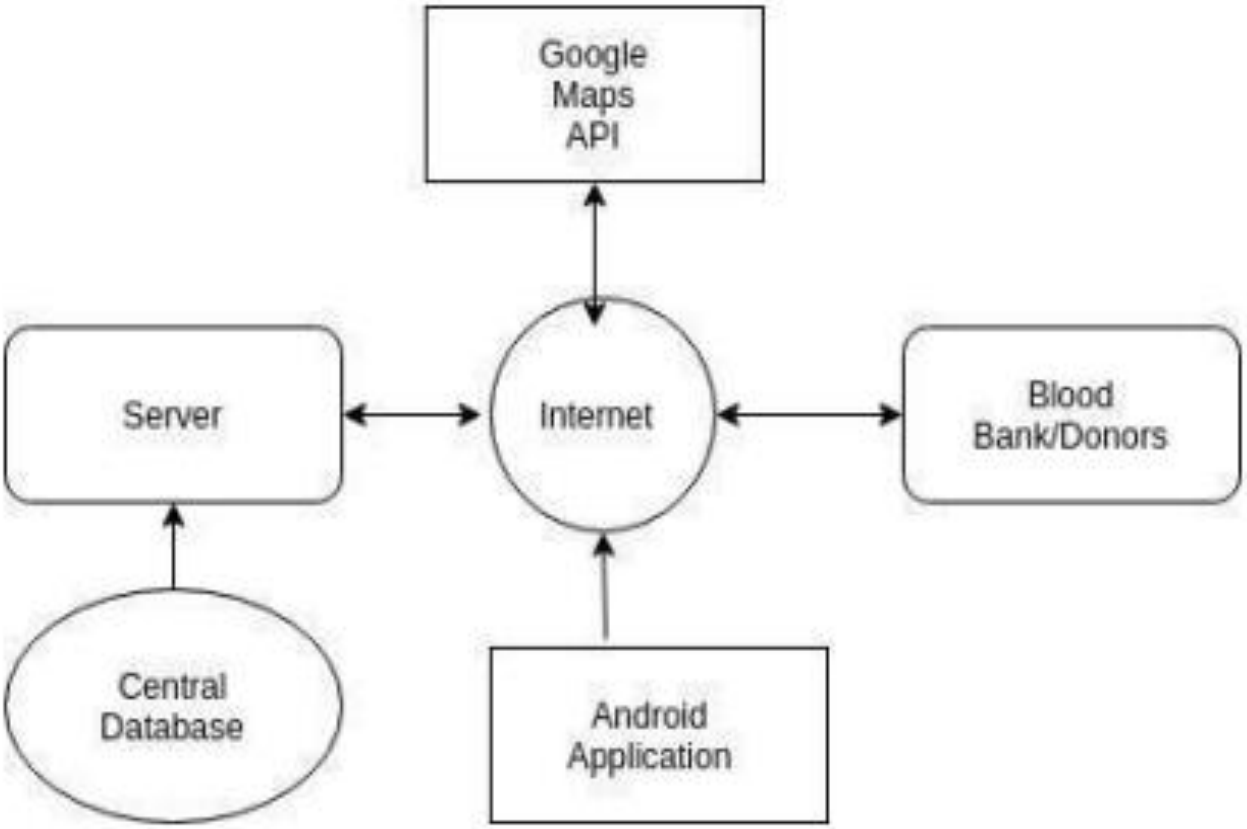

Google
Maps
API
Server
Internet
Blood
Bank/Donors
Central
Database
Android
Application

Figure 4

3.2 E-R AND USE CASE DIAGRAM

3.2.1 USE CASE DIAGRAM

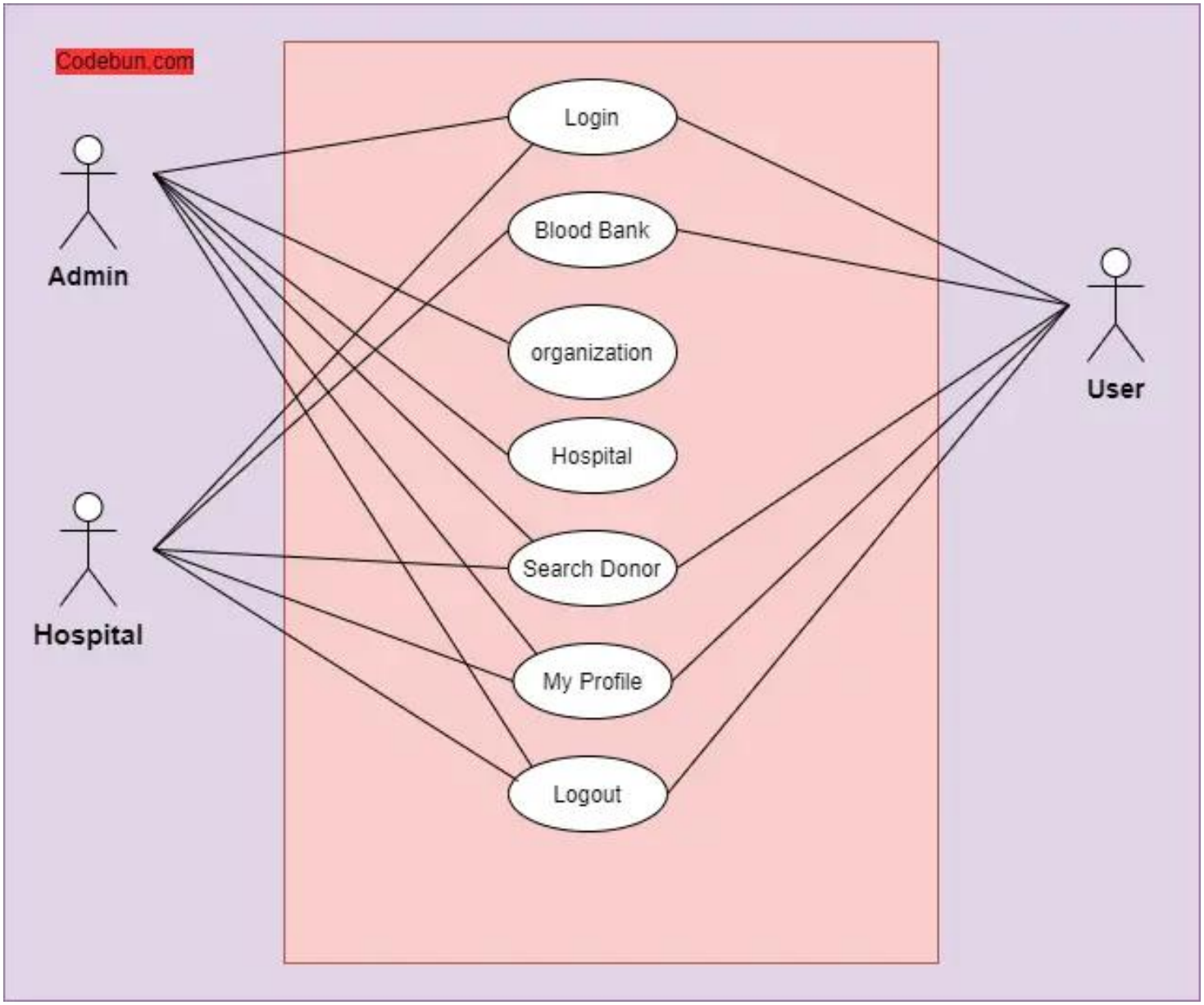


Figure 5

3.2.2 E-R DIAGRAM

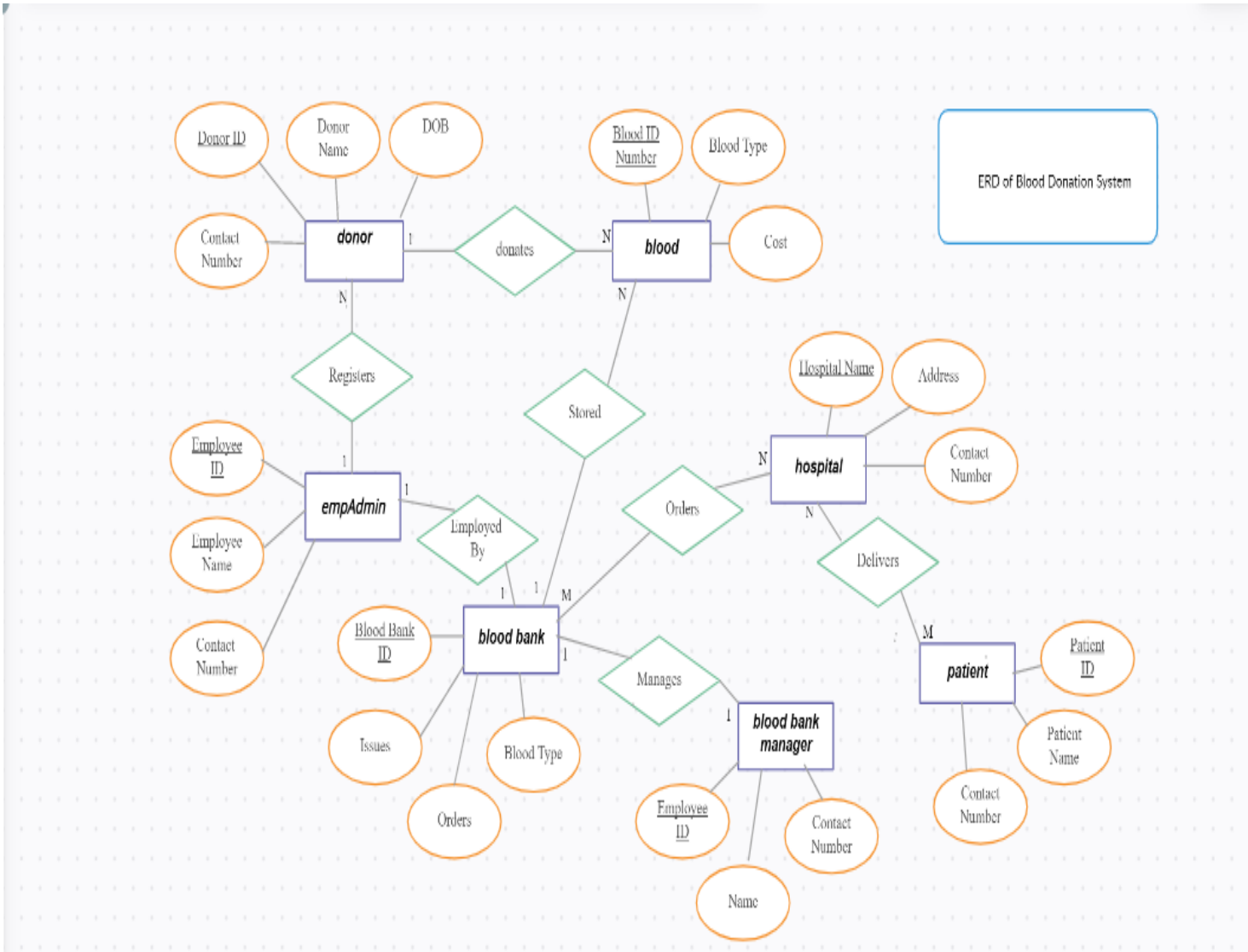


Figure 6

4.MODULES AND FUNCTIONALITY

In the proposed blood bank and donor management system, user login/signup functionality is an essential feature. This feature enables users to create an account and log in to the application to access their profiles, donation history, and other related features. Below is an explanation of the user login/signup functionality in the proposed application.

User Signup:

The user signup process is a simple process that requires the user to provide basic information such as name, email address, and password. The user can also provide additional information such as their blood group, contact number, and address. Once the user has entered their information, the system verifies their email address and password, and the user account is created.

User Login:

The user login process allows users to access their profiles, donation history, and other related features. The user enters their registered email address and password, and the system verifies their credentials. Once the system verifies the user's credentials, the user is redirected to their dashboard, where they can view their donation history, edit their profile information, and view upcoming blood drives in their area.

Forgot Password:

The application also provides a 'Forgot Password' feature to help users recover their account in case they forget their password. The user enters their registered email address, and the system sends a password reset link to their email address. The user can then use the link to reset their password and regain access to their account.

Profile Management:

Once the user has logged in, they can manage their profile information by editing their personal details, contact information, and blood group. This feature enables users to update their profile information in real-time, ensuring that their information is accurate and up to date.

Donation History:

The application also maintains a record of the user's donation history, enabling them to track their previous donations, including the date, location, and type of blood donated. This feature can help users stay informed about their donation history and encourage them to continue donating blood regularly.

Conclusion:

In conclusion, the proposed blood bank and donor management system provide users with a simple and intuitive login/signup process that enables

them to access their profiles, donation history, and other related features. The application's user-friendly interface makes it easy for users to manage their profile information and track their donation history. With these features, users can stay informed about their donation history, making them more likely to continue donating blood and ensuring a steady supply of blood for those in need.

5. Coding and Testing

5.1 Dashboard

```
package
com.android.iunoob.bloodbank.activities;

import android.app.ProgressDialog;
import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.NonNull;
import
android.support.design.widget.FloatingActionBu
tton;
import android.support.v4.app.Fragment;
import android.util.Log;
import android.view.View;
import
android.support.design.widget.NavigationView;
import android.support.v4.view.GravityCompat;
import
android.support.v4.widget.DrawerLayout;
```

```
import
android.support.v7.app.ActionBarDrawerToggle
;
import
android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.TextView;

import com.android.iunoob.bloodbank.R;
import
com.android.iunoob.bloodbank.fragments.About
Us;
import
com.android.iunoob.bloodbank.fragments.Achie
vmentsView;
import
com.android.iunoob.bloodbank.fragments.Blood
Info;
import
com.android.iunoob.bloodbank.fragments.Home
View;
import
com.android.iunoob.bloodbank.fragments.NearB
yHospitalActivity;
import
com.android.iunoob.bloodbank.fragments.Searc
hDonorFragment;
```

```
import
com.android.iunoob.bloodbank.viewmodels.Use
rData;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import
com.google.firebase.database.DataSnapshot;
import
com.google.firebase.database.DatabaseError;
import
com.google.firebase.database.DatabaseReferenc
e;
import
com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.Query;
import
com.google.firebase.database.ValueEventListen
er;

import static
com.android.iunoob.bloodbank.R.id.home;

public class Dashboard extends
AppCompatActivity
        implements
NavigationView.OnNavigationItemSelectedList
ener {

    private FirebaseAuth mAuth;
```

```
    private TextView getUserName;
    private TextView getUserEmail;
    private FirebaseDatabase user_db;
    private FirebaseUser cur_user;
    private DatabaseReference userdb_ref;

    private ProgressDialog pd;

    @Override
    protected void onCreate(Bundle
savedInstanceState) {
        super.onCreate(savedInstanceState);

setContentView(R.layout.activity_dashboard);

        pd = new ProgressDialog(this);
        pd.setMessage("Loading...");
        pd.setCancelable(true);
        pd.setCanceledOnTouchOutside(false);

        mAuth = FirebaseAuth.getInstance();
        user_db = FirebaseDatabase.getInstance();
        cur_user = mAuth.getCurrentUser();
        userdb_ref =
user_db.getReference("users");

        getUserEmail =
findViewById(R.id.UserEmailView);
```

```
        getUserName =
findViewById(R.id.UserNameView);
        Toolbar toolbar = (Toolbar)
findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        FloatingActionButton fab =
(FloatingActionButton) findViewById(R.id.fab);
        fab.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new
Intent(Dashboard.this, PostActivity.class));
            }
        });

        DrawerLayout drawer = (DrawerLayout)
findViewById(R.id.drawer_layout);
        ActionBarDrawerToggle toggle = new
ActionBarDrawerToggle(
                this, drawer, toolbar,
R.string.navigation_drawer_open,
R.string.navigation_drawer_close);
        drawer.addDrawerListener(toggle);
        toggle.syncState();
```

```
        NavigationView navigationView =
(NavigationView)
findViewById(R.id.nav_view);

navigationView.setNavigationItemSelectedListe
ner(this);
        View header =
navigationView.getHeaderView(0);

        getUserEmail = (TextView)
header.findViewById(R.id.UserEmailView);
        getUserName = (TextView)
header.findViewById(R.id.UserNameView);

        Query singleuser =
userdb_ref.child(cur_user.getUid());
        pd.show();

singleuser.addListenerForSingleValueEvent(ne
w ValueEventListener() {

            @Override
            public void onDataChange(@NonNull
DataSnapshot dataSnapshot) {
                //pd.show();
                String name =
dataSnapshot.getValue(UserData.class).getNam
e();
```

```
                getUserName.setText(name);

getUserEmail.setText(cur_user.getEmail());

                pd.dismiss();
            }

            @Override
            public void onCancelled(@NonNull
DatabaseError databaseError) {
                Log.d("User",
databaseError.getMessage());

            }
        });


        if(savedInstanceState == null)
        {

getSupportFragmentManager().beginTransaction
().replace(R.id.fragmentcontainer, new
HomeView()).commit();

navigationView.getMenu().getItem(0).setCheck
ed(true);

        }
```

```
    }


    @Override
    public void onBackPressed() {
        DrawerLayout drawer = (DrawerLayout)
findViewById(R.id.drawer_layout);
        if
(drawer.isDrawerOpen(GravityCompat.START)
) {

drawer.closeDrawer(GravityCompat.START);
        } else {
            super.onBackPressed();
        }
    }

    @Override
    public boolean onCreateOptionsMenu(Menu
menu) {
        // Inflate the menu; this adds items to the
action bar if it is present.

getMenuInflater().inflate(R.menu.dashboard,
menu);
        return true;
    }

    @Override
```

```
    public boolean
onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The
action bar will
        // automatically handle clicks on the
Home/Up button, so long
        // as you specify a parent activity in
AndroidManifest.xml.
        int id = item.getItemId();

        if (id == R.id.donateinfo) {

getSupportFragmentManager().beginTransaction
().replace(R.id.fragmentcontainer, new
BloodInfo()).commit();
        }
        if (id == R.id.devinfo) {

getSupportFragmentManager().beginTransaction
().replace(R.id.fragmentcontainer, new
AboutUs()).commit();
        }

        return super.onOptionsItemSelected(item);
    }


@SuppressWarnings("StatementWithEmptyBod
y")
```

```
    @Override
    public boolean
onNavigationItemSelected(MenuItem item) {
        // Handle navigation view item clicks here.
        int id = item.getItemId();

        if (id == home) {

getSupportFragmentManager().beginTransaction
().replace(R.id.fragmentcontainer, new
HomeView()).commit();

        } else if (id == R.id.userprofile) {
            startActivity(new
Intent(getApplicationContext(),
ProfileActivity.class));

        }
        else if (id == R.id.user_achiev) {

getSupportFragmentManager().beginTransaction
().replace(R.id.fragmentcontainer, new
AchievmentsView()).commit();

        }
        else if (id == R.id.logout) {
            mAuth.signOut();
```

```
            Intent intent = new
Intent(getApplicationContext(),
LoginActivity.class);
            startActivity(intent);
        }
        else if (id == R.id.blood_storage){


getSupportFragmentManager().beginTransaction
().replace(R.id.fragmentcontainer, new
SearchDonorFragment()).commit();

        } else if (id == R.id.nearby_hospital) {

getSupportFragmentManager().beginTransaction
().replace(R.id.fragmentcontainer, new
NearByHospitalActivity()).commit();

        }

        DrawerLayout drawer = (DrawerLayout)
findViewById(R.id.drawer_layout);

drawer.closeDrawer(GravityCompat.START);
        return true;
    }

    @Override
    protected void onStart() {
```

```
        super.onStart();
        FirebaseUser currentUser =
mAuth.getCurrentUser();
        if(currentUser == null)
        {
            Intent intent = new
Intent(getApplicationContext(),
LoginActivity.class);
            startActivity(intent);
            finish();
        }
    }

    @Override
    protected void onResume() {
        super.onResume();
        FirebaseUser currentUser =
mAuth.getCurrentUser();
        if(currentUser == null)
        {
            Intent intent = new
Intent(getApplicationContext(),
LoginActivity.class);
            startActivity(intent);
            finish();
        }
    }

}
```

5.2 Login Activity

```
package
com.android.iunoob.bloodbank.activities;

import android.app.ProgressDialog;
import android.content.Intent;
import android.support.annotation.NonNull;
import
android.support.v7.app.AppCompatActivity
;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.android.iunoob.bloodbank.R;
import
com.google.android.gms.tasks.OnComplete
Listener;
import com.google.android.gms.tasks.Task;
import
com.google.firebase.auth.AuthResult;
import
com.google.firebase.auth.FirebaseAuth;
```

```
public class LoginActivity extends
AppCompatActivity {

    private Button signin, signup, resetpass;
    private EditText inputemail,
inputpassword;
    private FirebaseAuth mAuth;
    private ProgressDialog pd;

    @Override
    protected void onCreate(Bundle
savedInstanceState) {
        super.onCreate(savedInstanceState);

setContentView(R.layout.activity_login);

        pd = new ProgressDialog(this);
        pd.setMessage("Loading...");
        pd.setCancelable(true);
        pd.setCanceledOnTouchOutside(false);

        mAuth = FirebaseAuth.getInstance();

        if(mAuth.getCurrentUser() != null)
        {
            Intent intent = new
Intent(getApplicationContext(),
Dashboard.class);
            startActivity(intent);
```

```
            finish();
        }


        inputemail =
findViewById(R.id.input_username);
        inputpassword =
findViewById(R.id.input_password);

        signin =
findViewById(R.id.button_login);
        signup =
findViewById(R.id.button_register);
        resetpass =
findViewById(R.id.button_forgot_password
);

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

                final String email =
inputemail.getText().toString()+"";
                final String password =
inputpassword.getText().toString()+"";

                try {
```

```
                if(password.length()>0 &&
email.length()>0) {
                    pd.show();

mAuth.signInWithEmailAndPassword(emai
l, password)

.addOnCompleteListener(LoginActivity.this
, new OnCompleteListener<AuthResult>() {
                            @Override
                            public void
onComplete(@NonNull Task<AuthResult>
task) {
                                if
(!task.isSuccessful()) {

Toast.makeText(getApplicationContext(),

"Authentication Failed",

Toast.LENGTH_LONG).show();
                                    Log.v("error",
task.getException().getMessage());
                                } else {
                                    Intent intent =
new Intent(getApplicationContext(),
Dashboard.class);

startActivity(intent);
```

```
                                finish();
                            }
                            pd.dismiss();
                        }
                    });
                }
                else
                {

Toast.makeText(getApplicationContext(),
"Please fill all the field.",
Toast.LENGTH_LONG).show();
                }

            } catch (Exception e)
            {
                e.printStackTrace();
            }
        }
    });

    signup.setOnClickListener(new
View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent intent = new
Intent(getApplicationContext(),
ProfileActivity.class);
            startActivity(intent);
```

```
            }
        });

        resetpass.setOnClickListener(new
    View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new
    Intent(getApplicationContext(),
    RestorePassword.class);
                startActivity(intent);
            }
        });


    }

}
```

5.3 Post Activity

```
package
com.android.iunoob.bloodbank.activities;

import android.app.ProgressDialog;
import android.content.Intent;
import android.support.annotation.NonNull;
```

```
import
android.support.v7.app.AppCompatActivity
;
import android.os.Bundle;
import android.util.Log;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.Toast;

import com.android.iunoob.bloodbank.R;
import
com.android.iunoob.bloodbank.viewmodels.
UserData;
import
com.google.firebase.auth.FirebaseAuth;
import
com.google.firebase.auth.FirebaseUser;
import
com.google.firebase.database.DataSnapshot;
import
com.google.firebase.database.DatabaseError
;
import
com.google.firebase.database.DatabaseRefe
rence;
```

```
import
com.google.firebase.database.FirebaseDatab
ase;
import com.google.firebase.database.Query;
import
com.google.firebase.database.ValueEventLi
stener;

import java.util.Calendar;

public class PostActivity extends
AppCompatActivity {

    ProgressDialog pd;

    EditText text1, text2;
    Spinner spinner1, spinner2;
    Button btnpost;

    FirebaseDatabase fdb;
    DatabaseReference db_ref;
    FirebaseAuth mAuth;

    Calendar cal;
    String uid;
    String Time, Date;

    @Override
```

```
    protected void onCreate(Bundle
savedInstanceState) {
        super.onCreate(savedInstanceState);

setContentView(R.layout.activity_post);

        pd = new ProgressDialog(this);
        pd.setMessage("Loading...");
        pd.setCancelable(true);
        pd.setCanceledOnTouchOutside(false);

        getSupportActionBar().setTitle("Post
Blood Request");

getSupportActionBar().setDisplayHomeAs
UpEnabled(true);

        text1 = findViewById(R.id.getMobile);
        text2 =
findViewById(R.id.getLocation);

        spinner1 =
findViewById(R.id.SpinnerBlood);
        spinner2 =
findViewById(R.id.SpinnerDivision);

        btnpost = findViewById(R.id.postbtn);

        cal = Calendar.getInstance();
```

```
        int day =
cal.get(Calendar.DAY_OF_MONTH);
        int month =
cal.get(Calendar.MONTH);
        int year = cal.get(Calendar.YEAR);
        int hour = cal.get(Calendar.HOUR);
        int min = cal.get(Calendar.MINUTE);
        month+=1;
        Time = "";
        Date = "";
        String ampm="AM";

        if(cal.get(Calendar.AM_PM) ==1)
        {
           ampm = "PM";
        }

        if(hour<10)
        {
           Time += "0";
        }
        Time += hour;
        Time +=":";

        if(min<10) {
           Time += "0";
        }
```

```
        Time +=min;
        Time +=(" "+ampm);

        Date = day+"/"+month+"/"+year;

        FirebaseUser cur_user =
mAuth.getInstance().getCurrentUser();

        if(cur_user == null)
        {
            startActivity(new
Intent(PostActivity.this,
LoginActivity.class));
        } else {
            uid = cur_user.getUid();
        }

        mAuth = FirebaseAuth.getInstance();
        fdb = FirebaseDatabase.getInstance();
        db_ref = fdb.getReference("posts");

        try {
            btnpost.setOnClickListener(new
View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    pd.show();
                    final Query findname =
fdb.getReference("users").child(uid);
```

```
                if(text1.getText().length() == 0)
                {

Toast.makeText(getApplicationContext(),
"Enter your contact number!",

Toast.LENGTH_LONG).show();
                }
                else if(text2.getText().length()
== 0)
                {

Toast.makeText(getApplicationContext(),
"Enter your location!",

Toast.LENGTH_LONG).show();
                }
                else {

findname.addListenerForSingleValueEvent(
new ValueEventListener() {
                        @Override
                        public void
onDataChange(@NonNull DataSnapshot
dataSnapshot) {

                            if
(dataSnapshot.exists()) {
```

```
db_ref.child(uid).child("Name").setValue(d
ataSnapshot.getValue(UserData.class).getN
ame());

db_ref.child(uid).child("Contact").setValue(
text1.getText().toString());

db_ref.child(uid).child("Address").setValue(
text2.getText().toString());

db_ref.child(uid).child("Division").setValue
(spinner2.getSelectedItem().toString());

db_ref.child(uid).child("BloodGroup").setV
alue(spinner1.getSelectedItem().toString());

db_ref.child(uid).child("Time").setValue(Ti
me);

db_ref.child(uid).child("Date").setValue(Da
te);

Toast.makeText(PostActivity.this, "Your
post has been created successfully",

Toast.LENGTH_LONG).show();
                        startActivity(new
Intent(PostActivity.this, Dashboard.class));
```

```
                        } else {

Toast.makeText(getApplicationContext(),
"Database error occured.",

Toast.LENGTH_LONG).show();
                        }


                    }

                    @Override
                    public void
onCancelled(@NonNull DatabaseError
databaseError) {
                        Log.d("User",
databaseError.getMessage());

                    }
                });
            }
        }
    });
}
catch (Exception e)
{
    e.printStackTrace();
}
pd.dismiss();
```

```
    }

    @Override
    public boolean
onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case android.R.id.home:
                onBackPressed();
                return true;
        }

        return
super.onOptionsItemSelected(item);
    }
}
```

6. Result and Discussion

The proposed blood bank and donor management system with a location-based approach offers several advantages over existing systems. The system's key features, including enhanced donor experience, improved inventory management, faster response times, customizable alerts, and user-friendly interface, have the potential to improve blood bank operations significantly. In this section, we will discuss the results of implementing this proposed idea and its impact on blood banks and donors.

Improved Blood Supply:

One of the most significant advantages of the proposed system is its potential to improve the blood supply. By providing real-time information about blood drives and donation centers in a donor's area, the system can increase the number of blood donations. The system's customizable alerts can notify potential donors about blood drives and donation centers, increasing awareness and participation in blood donation campaigns. With an increased number of donors, blood banks can maintain an adequate supply of blood and prevent shortages, ensuring that patients in need can access safe and timely blood.

Efficient Inventory Management:

The proposed system's location-based approach can also help blood banks manage their inventory more efficiently. The system can provide real-time data on blood supply and demand in each location, allowing blood banks to allocate their resources effectively. With this feature, blood banks can maintain optimal inventory levels, reducing the likelihood of shortages or

wastage of blood products. The system can also track blood donations, processing, and distribution, allowing blood banks to identify bottlenecks and inefficiencies in their operations and take corrective measures.

Enhanced Donor Experience:

The proposed system's location-based approach can enhance the donor experience significantly. The system can provide donors with real-time information about blood drives and donation centers in their area, making it easier for them to donate blood. The system's customizable alerts can also notify donors about upcoming blood drives and donation centers, making it more convenient for them to donate blood. With an enhanced donor experience, blood banks can attract more donors, resulting in increased blood supply and improved patient outcomes.

Faster Response Times:

The proposed system's location-based approach can also help blood banks respond to emergencies faster. The system can quickly identify the nearest blood donation centers and available blood supplies, allowing blood banks to provide timely blood transfusions to patients in critical situations. With faster response times, blood banks can save lives and improve patient outcomes.

User-friendly Interface:

The proposed system's user-friendly interface can improve blood bank operations significantly. The system can provide blood bank staff with an intuitive interface for managing donor records, blood inventory, and blood

processing. With this feature, blood bank staff can manage their operations more efficiently, reducing the likelihood of errors and inefficiencies. The system can also generate reports on blood supply and demand, donor participation, and inventory levels, allowing blood banks to make data-driven decisions.

Limitations:

Despite its many advantages, the proposed system has some limitations. One of the limitations is that it relies on donor participation, which can be unpredictable. Blood banks may face challenges in attracting donors to participate in blood donation campaigns, leading to fluctuations in blood supply. The system's location-based approach may also be limited in rural areas where blood donation centers and blood drives are scarce.

7. Conclusion and Future Scope

7.1 Conclusion

In conclusion, the proposed blood bank and donor management system with a location-based approach has significant potential for future development and improvement. By leveraging emerging technologies such as artificial intelligence, blockchain, and mobile applications, the system can further improve blood bank operations, increase blood supply, and improve patient outcomes. The future scope of the proposed system is promising, and continued research and development can bring about further advancements in blood bank operations and blood transfusion.

7.2 Future Scope

The proposed blood bank and donor management system with a location-based approach has significant potential for future development and improvement. Here are some potential future scope of the proposed system:

Integration with other Healthcare Systems:

The proposed system can be integrated with other healthcare systems, such as electronic health records and hospital information systems. Integration can improve communication and collaboration between healthcare providers, resulting in faster and more efficient blood transfusions for patients.

Artificial Intelligence and Machine Learning:

The proposed system can leverage artificial intelligence and machine learning technologies to improve blood bank operations. For example, the system can use machine learning algorithms to predict blood demand based on historical data and current trends. This feature can help blood banks maintain optimal inventory levels and reduce wastage.

Blockchain Technology:

The proposed system can also leverage blockchain technology to improve blood bank operations. Blockchain technology can provide a secure and transparent way to store and share blood donation data, reducing the likelihood of errors and fraud. Blockchain can also enable the tracking.

8.References

- **https://developer.mozilla.org/en-US/docs/Web**
- **https://firebase.google.com/docs/**
- **https://www.mongodb.com/docs/**
- **https://developer.android.com/docs/**
- **https://kotlinlang.org/docs/home.html**