
# SOCIAL NETWORKING ANDROID APP

**A Report of Project 2**

***Submitted by***

**CHHAVI GAUR**

**17SCSE1044135**

**1713111001**

***in partial fulfilment for the award of the degree of***

**BACHELOR OF COMPUTER APLLICATION**

**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING**

**Under the Supervision of**

**MS. SAUMYA CHATURVEDI**

**ASSITANT PROFESSOR**


# SCHOOL OF COMPUTING AND SCIENCE AND ENGINEERING

## BONAFIDE CERTIFICATE

Certified that this project report **"INSTANT MESSAGING ANDROID APP USING FIREBASE"** is the bonafide work of **"CHHAVI GAUR (1713111001)"** who carried out the project work under my supervision.

**SIGNATURE OF HEAD**

Dr. MUNISH SHABARWAL

Professor & Dean,

**School of Computing Science & Engineering**

**SIGNATURE OF SUPERVISIOR**

MS. SAUMYA CHATURVEDI

ASSISTANT PROFESSOR,

**School of Computing Science & Engineering**

# ACKNOWLEDGEMENT

I take this occasion to thank God, almighty for blessing me with his grace and taking our endeavour to a successful Culmination. I extend my sincere and heartfelt thanks to my esteemed guide, Ms saumya Chaturvedi, for providing me with the right guidance and advice at the crucial junctures and for showing me the right way. I extend my sincere thanks to my respected Head of the division DR. Thirunavukkarasu k., for allowing me to use the facilities available. I would like to thank the other faculty members also at this occasion. Last but not the least, I would like to thank my friends and family for the support and encouragement they have given me during the course of this work. I also thanks to my senior Saurabh Sharma.

# ABSTRACT

Communication through internet is becoming vital these days. An online communication allows the users to communicate with other people in a fast and convenient way. Considering this, the online communication application must be able share the texts or images or any other files in a faster way with minimum delay or with no delay. Firebase is one of the platforms which provides a real-time database and cloud services which allows the developer to make these applications with ease. Instant messaging can be considered as platform to maintain communication. Android provides better platform to develop various applications for instant messaging compared to other platforms such as iOS. The main objective of this paper is to present a software application for the launching of a real time communication between operators/users. The system developed on android will enable the users to communicate with another users through text messages with the help of internet. The system requires both the device to be connected via internet. This application is based on Android with the backend provided by google Firebase.

Keywords: communication; firebase; android; Instant messaging; real-time databases; group messaging**.**

# TABLE OF CONTENTS


| CHAPTER NO. | TITLE | PAGE NO. |
|---|---|---|
| | **ABSTRACT** | **iv** |
| | **LIST OF FIGURES** | **vi** |
| **1.** | **INTRODUCTION** | **1** |
| **2.** | **SYSTEM ANALYSIS** | **3** |
| | 2.1 EXISTING SYSTEM | |
| | 2.2 LIMITATION | |
| | 2.3 PROPOSED SYSTEM | |
| | 2.4 FEASIBILITY STUDY | |
| | 2.5 ECONOMIC STUDY | |
| **3.** | **MODULES** | **5** |
| | 3.1 FUNCTIONAL REQUIREMENTS | |
| | 3.2 NON-FUNCTIONAL REQUIREMENTS | |
| | 3.3 IMPLEMENTATION REQUIREMENTS | |
| **4.** | **SYSTEM PROCESS** | **7** |
| **5.** | **SYSTEM DESIGN** | **8** |
| | 5.1 DATA FLOW DIAGRAM | |
| | 5.2 SEQUENCE DIAGRAM | |
| | 5.3 CLASS DIAGRAM | |
| | 5.4 ER DIAGRAM | |
| | 5.5 USE CASE DIAGRAM | |
| **6.** | **TESTING** | **16** |
| **7.** | **USER INTERFACES** | **17** |
| | 7.1 SCREENSHOTS | |
| | 7.2 CODING | |

| | | |
|---|---|---|
| **8.** | **RESULT** | **30** |
| **9.** | **CONCLUSION** | **31** |
| **10.** | **REFERENCES** | **32** |

# LIST OF FIGURES


| FIGURE NO. | FIGURE NAME |
|---|---|
| FIGURE 1 | DFD LEVEL-0 |
| FIGURE 2 | DFD LEVEL-1 |
| FIGURE 3 | SEQUENCE CHAT |
| FIGURE 4 | SEQUENCE LOGIN |
| FIGURE 5 | SEQUENCE RESET-PASSWORD |
| FIGURE 6 | CLASS DIAGRAM |
| FIGURE 7 | ENTITY – RELATIONSHIP DIAGRAM |
| FIGURE 8 | USE CASE DIAGRAM |
| FIGURE 9 | FIREBASE DATA CHATTING |
| FIGURE 10 | REAL & NORMAL DATABASE |
| FIGURE 11 | ANDRID APP ICON |
| FIGURE 12 | REGISTRATION PAGE |
| FIGURE 13 | LOGIN PAGE |
| FIGURE 14 | CHATTING PAGE |
| FIGURE 15 | GROUP CHATTING PAGE |
| FIGURE 16 | USER INFORMATION PAGE |
| FIGURE 17 | SOCIAL TENDING PAGE |
| FIGURE 18 | SOCIAL ISSUE REGISTRATION PAGE |
| FIGURE 19 | SOCIAL ISSUE POSTING PAGE |
| FIGURE 20 | SOCIAL ISSUE SHARING PAGE |

# CHAPTER I

# Introduction

Communication is a mean for people to exchange messages. It has started since the beginning of human creation. Distant communication began as early as 1800 century with the introduction of television, telegraph and then telephony. Interestingly enough, telephone communication stands out as the fastest growing technology, from fixed line to mobile wireless, from voice call to data transfer. The emergence of computer network and telecommunication technologies bears the same objective that is to allow people to communicate.

All this while, much efforts has been drawn towards consolidating the device into one and therefore indiscriminate the services. Chatting is a method of using technology to bring people and ideas together despite of the geographical barriers. The technology has been available for years but the acceptance it was quite recent. Our project is an example of a chat server. It is made up of applications the client application which runs on the users mobile and server application which runs on any pc on the network. To start chatting our client should get connected to server where they can do Group and private chatting.

The grounds based long range relational correspondence is made to offer understudies and staffs of a foundation immense proportion of information around one another. Likewise, when it is used as a sifting through device, it can give better ways to deal with understudy social affairs to react to people, share information, get some answers concerning grounds events and enact support and movement.

Firebase is a scalable, real-time support for web-based application. It allows developers to build rich, collaborative applications without the hassle of managing servers or writing server-side code. It is a mobile and web application platform with tools and infrastructure designed to help developers build high-quality apps. Firebase is made up of complementary features that developers can mix-and-match to fit their needs. Firebase got evolved from Envolve Co., a prior start-up founded by computer experts Templin and Lee in 2011 in San Francisco and Mountain View, California. Envolve provided developers an API that let them integrate online chat into their websites. But they founded that the service was being used to pass application data that wasn't chat messages. Developers were using Envolve to sync application data such as game state in realtime across their users.

Tamplin and Lee decided to separate the chat system and the real-time architecture that powered it, founding Firebase as a separate company in April 2012. In January 2014 Google acquired Firebase for $7 million. Firebase has support for the web, iOS, OS X, and Android clients. In addition, it has a Node.Js and a Java library designed for server-side use. The Firebase web client supports all mainstream browsers (IE 7+, Firefox 3+, Chrome, Safari, Opera, and major mobile web browsers), and it works on any network connection. Much of Firebase is built on top of Google Cloud. Firebase projects are also Google Cloud projects. You can use both Google Cloud features along with Firebase features in the same project. Google Cloud doesn’t really give tools to build your mobile app. You’ll just have to write all

the code that deals with getting data in and out of the project. With Firebase, you get the SDKs for mobile development, which makes building a mobile app a lot easier. Also, Google Cloud doesn't give you any realtime data operations like you get with Firebase Realtime Database.

In many cases, application logic is best controlled on the server in order to avoid tampering on the client side. Cloud Functions are fully insulated from the client so you can be sure they are private and secure and can't be reverse engineered.

**PROJECT OBJECTIVE AND SCOPE:**

- To have attractive and Secure Login page to access
- Make new user account in more user friendly and proper validation of details
- Search People easily on entire network
- Send Friend Request to other users to make friends
- Add friends to your friend box accept request
- Creating a public profile having social, professional and personal information
- Ease of editing of profile anytime
- Chat with Online friends
- Upload and Share Images on network
- Add, Search and shares videos of youtube
- Send messages to other friends
- Reply directly to incoming user messages
- Post Advertisement of products
- Administration page to keep eye on user operation
- Easily password recovery processing

# CHAPTER II
# SYSTEM ANALYSIS

## 2.1 EXISTING SYSTEM

In Existing System people doesn't have the option to name social control violators. Singular inhabitants can't take the measures to rate, control and screen the social obligation of a given individual. In an indirect manner the bad behavior is extending point of fact, everybody is dismissing the rules. No one is taking thought. The able individuals are dismissing an immediate aftereffect of pollution. The higher authorities not making any move towards indiscipline. No one is feeling commitment to fight. With this in our country the legality is going crazy.

## 2.2 LIMITATIONS OF EXISTING SYSTEM

At the present time if an individual need to give protesting, he needs to go to the concerned office and raise a complaint. In India various commonplace people haven't the foggiest about the standards and rules of Govt. From now on they are blackmailed by the savage world. As the bad behaviors are extended, it is getting difficult for a commonplace man to persevere. Here as the people don't have the foggiest thought regarding the real commitments of the authorities and they don't have the sources to show up at their voice to the higher pros, they are getting weak.

## 2.3 PROPOSED SYSTEM

By this no one will be cheated. Accordingly, all the encroachment of rules can be diminished taking everything into account.
In our current structure people drawing nearer, checking out the conversations and engaging against the bad behaviors in their area.
Everybody feels their obligation to restrain the degradation.

Here each and every individual has the situation to look at regarding any social issue and matter .
With these conversations various hooligans can come into the picture of veritable world.With these things the fair people can get careful. By then basing on these discussions the right decision can be taken by the concerned position.

## 2.4 FEASIBILITY STUDY

A feasibility study looks at the viability of an idea with an emphasis on identifying potential problems and attempts to answer one main question: Will the idea work and should you proceed with it?
Before you begin writing your business plan you need to identify how, where, and to whom you intend to sell a service or product. You also need to assess your competition and figure out how much money you need to start your business and keep it running until it ++is established.

Feasibility studies address things like where and how the business will operate. They provide in-depth details about the business to determine if and how it can succeed, and serve as a valuable tool for developing a winning business plan.

## 2.5 Economic feasibility

This is an analysis of the costs to be incurred in the system and the benefits derivable out of the system.
Time-based study: This is an analysis of the time required to achieve a return on investments. The future value of a project is also a factor.

JUSTIFICATION AND NEED FOR THE SYSTEM

The site is build so that users can interact with others.

1. To provide better service to their users.
2. The ti*9/8me of the user is being saved.
3. Allowing the user to contribute to the environment.

## 2.6 ADVANTAGES OF THE PROPOSED SYSTEM

•Efficient usage of resources.
•It provides security through verification process.
•Performance is high.
•Reduces the effort and time in gathering the information about the users.
•Provides a complete record of all the available vehicles for pooling.
•The constraints and checks lead to a valid database.

# CHAPTER III
# MODULES

## MODULES:

1. Administrator

2. Friends

3. Users

4. Blogs

5. Comments Rating

6. Registration

## USERS:

1. Administrator

2. Friends

3. Authenticate Users

4. Public User

## 3.1 FUNCTIONAL REQUIREMENTS

Should provide a common platform where people of India can

- Voice out violations, injustice, inhumanity, corruption happening in their vicinity

- Endorse someone else's concern and augment with more proofs, details etc.

- Call for an online debate or discussion on certain topics of broad applicability

Should be highly dynamic, with minimal static content as framework and maximum content created by site participants

Should have the ability to tag social discipline violators using their UID

Authenticity for adding users is utmost important for such a website. Definitely one should not be allowed to have more than one profile.

## 3.2 NON-FUNCTIONAL REQUIREMENT

Secure access of confidential data (user's details). SSL can be used. 24 X 7 availability Better component design to get better performance at peak time

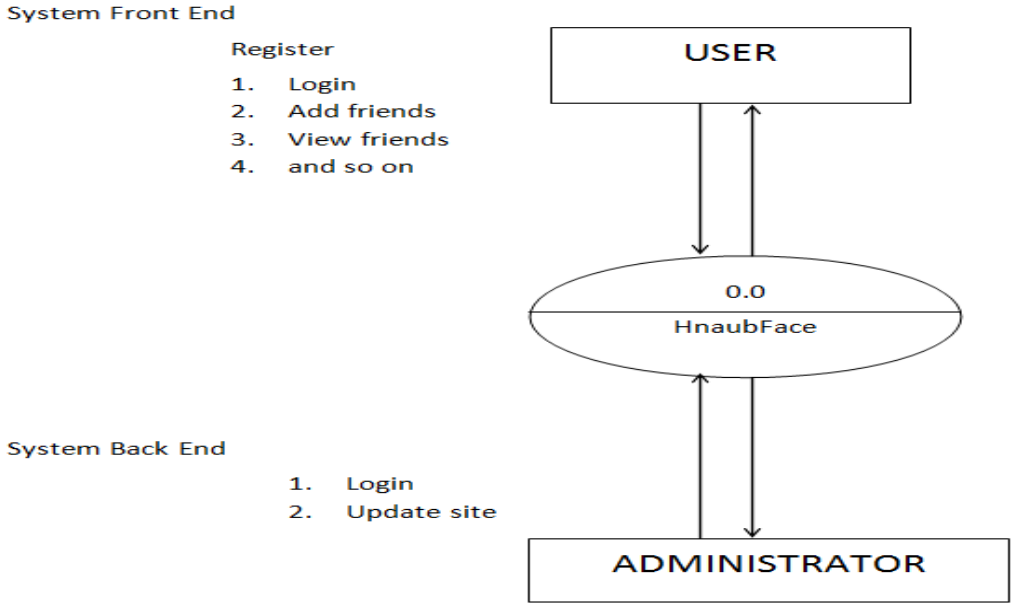
Flexible service based architecture will be highly desirable for future extension

## *RESEARCH MEATHODOLOGY*

Campus based social network was developed using android studio , java, firebase data .

In these application we add a new feature social tending option and verification of user through email address . in these one user can make one id only.

## 3.3 Implementation Front end & Back end



# CHAPTER IV

# SYSTEM PROCESS

## 4.1 SYSTEM PROCESS ILUSTRATION

The structure administrator can login to revive the system from the back end or to perceive what has being the latest substance on the grounds or to check who is doing what and bring to the warning of the association authority any sensible platform of security.

New customers of the grounds based long range relational correspondence approaches join page as understudy or staff of the foundation and as such there is opportunity to send and get message, see profile, search for partners or incorporate sidekicks, square or unblock mates, and besides to view or add photos to assortment, make gathering and soon.

## 4.2 SYSTEM ENGINEERING

The product advancement life cycle (SDLC) is the whole procedure of formal, intelligent advances taken to build up a product item. Inside the more extensive setting of Application Lifecycle Management (ALM), the SDLC is essentially the piece of procedure wherein coding/writing computer programs is applied to the issue being fathomed by the current or arranged application.

The phases of SDLC can vary somewhat but generally include the following:

- Conceptualization
- Requirements and cost/benefits analysis
- Detailed specification of the software requirements
- Software design
- Programming
- Testing
- User and technical training
- Maintenance

There are numerous systems or models that can be utilized to control the product improvement lifecycle either as a centre model to the SDLC or as a correlative strategy.

# CHAPTER V

# SYSTEM DESIGN

System design is a solution, how to approach to the creation of a new system. This important phase is composed of several steps. It provides the understanding and procedural details necessary for implementation of the system recommended in the feasibility study. Emphasis is on translating the performance requirements in to design specification. Design goes through logical and physical stages of development. Logical design reviews the present physical system. Prepares input and output specification, makes edit security and control specification details, and the implementation plan prepares a logical design walkthrough. The design phase is transition from user-oriented document to a document to the programmers or database personnel. It describes the input, compatible to database and procedures all in a format compatible to user requirements. The logical design also specifies input, output, files and screen layouts. Physical design produces the working system by defining the design specification that tells the programmers exactly what the system should do in turn.

## 5.1 DATAFLOW DIAGRAM

A data flow diagram (DFD) is a graphical representation of the " flow" of data through an information system.

Dfd 0-level

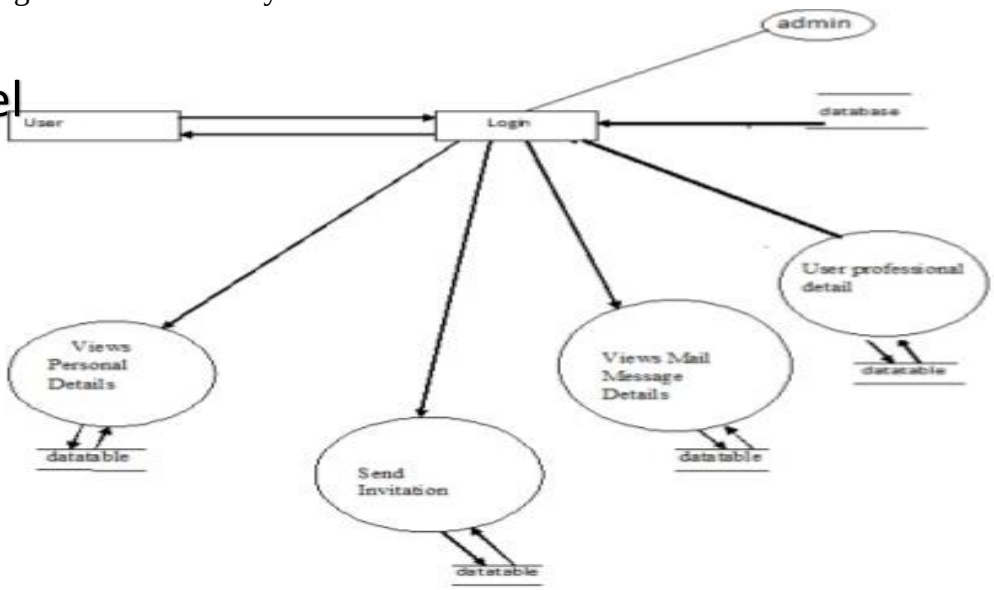



***Fig 1: DFD 0-LEVEL***

Dfd level-1

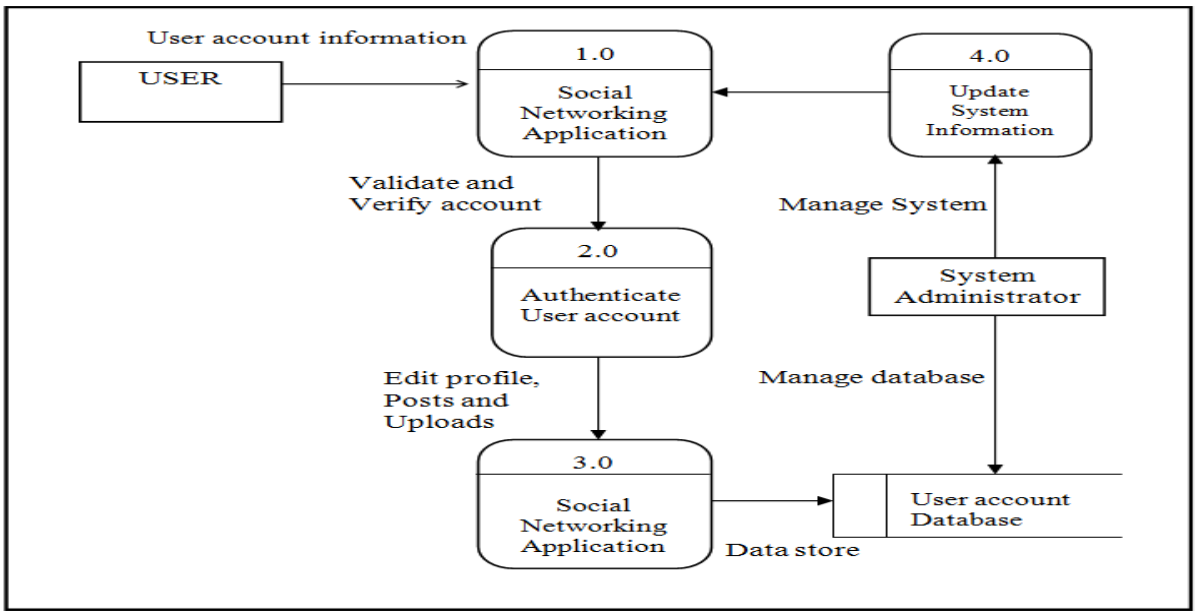



**Fig 2: DFD LEVEL-1**

## 5.2 Sequence Diagram

A sequence diagram, in the context of UML, represents object collaboration and is used to define event sequences between objects for a certain outcome. A sequence diagram is an essential component used in processes related to analysis.

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

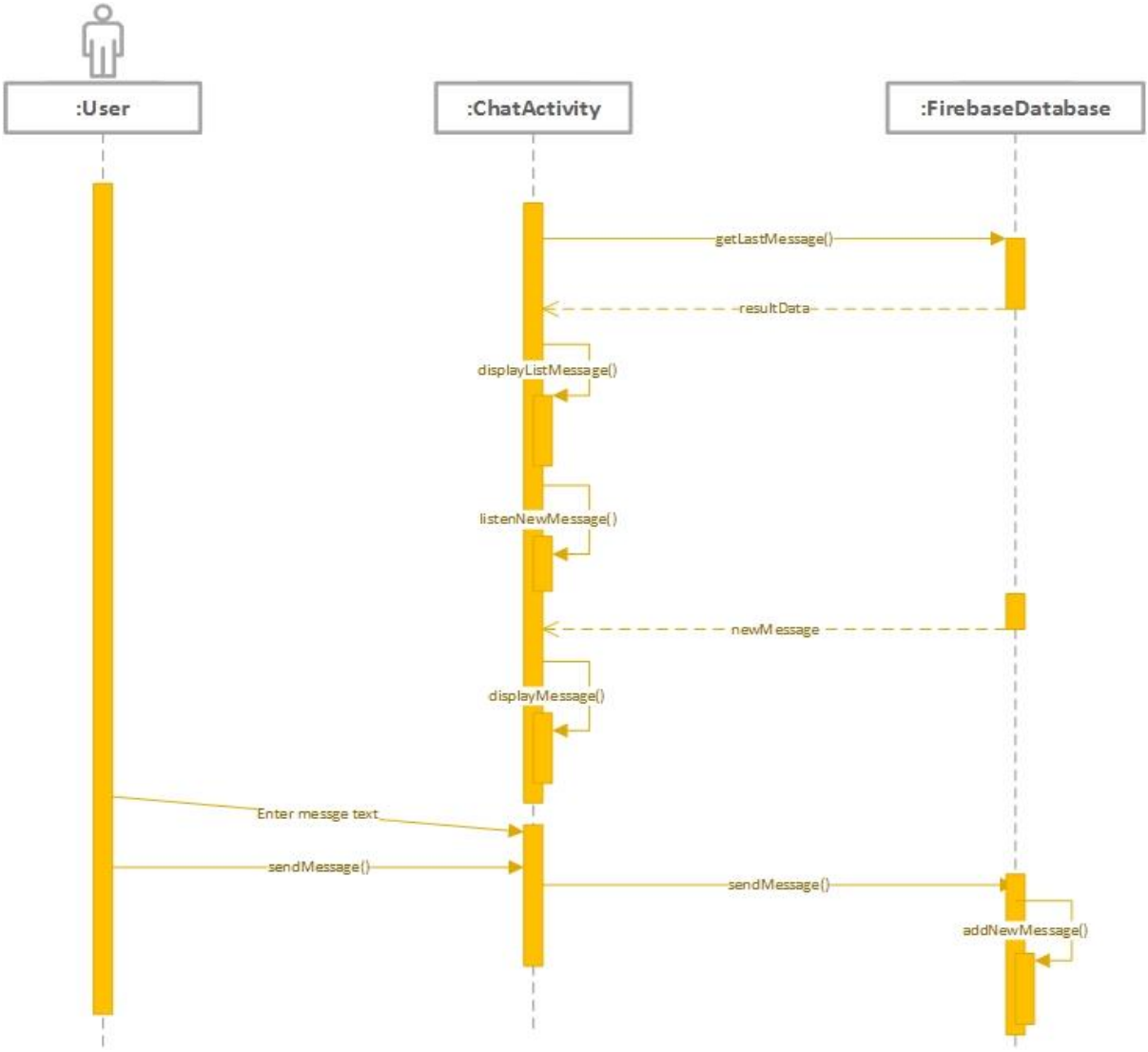



**Fig 3: sequence chat**

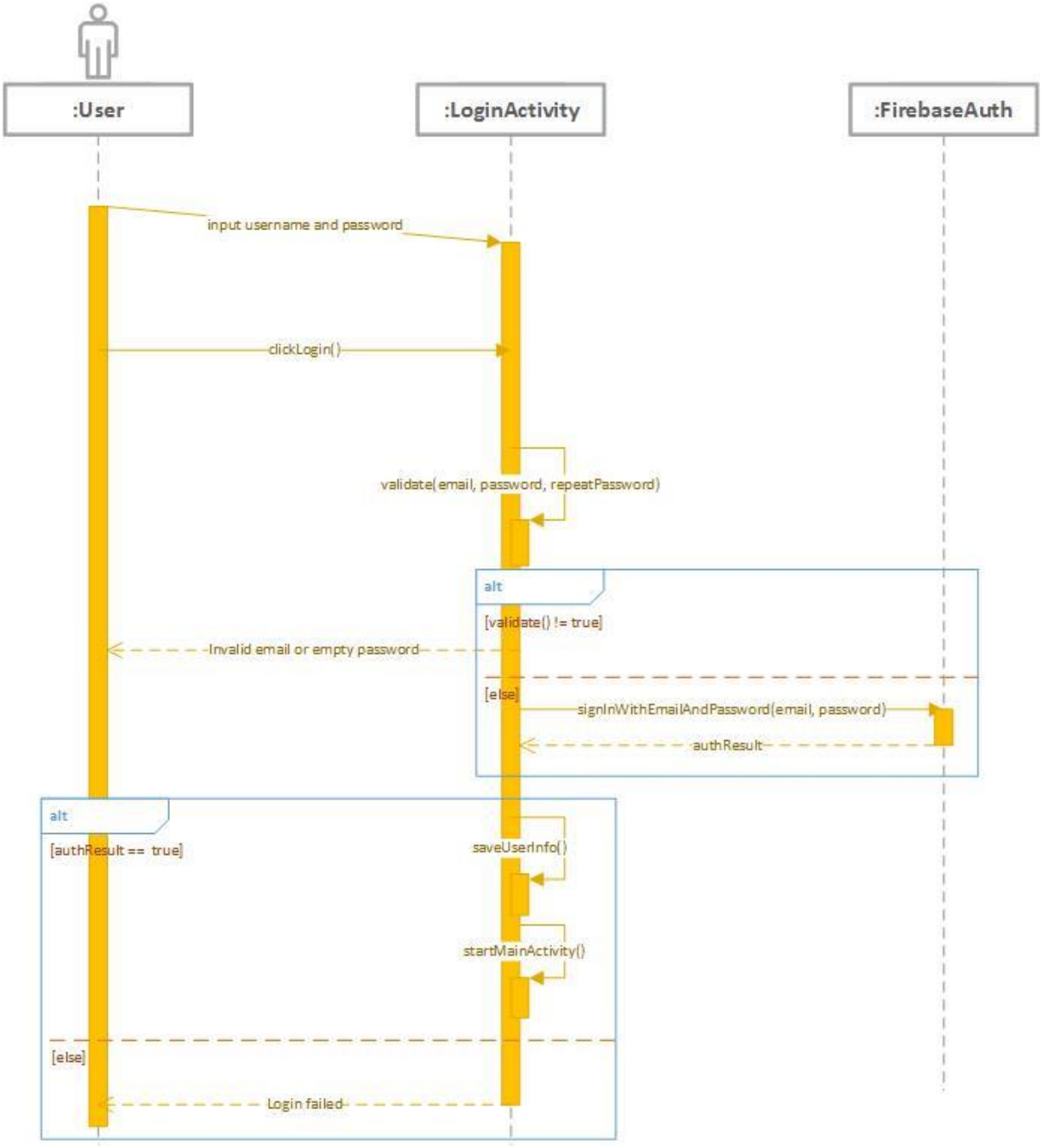



**Fig 4: sequence login**

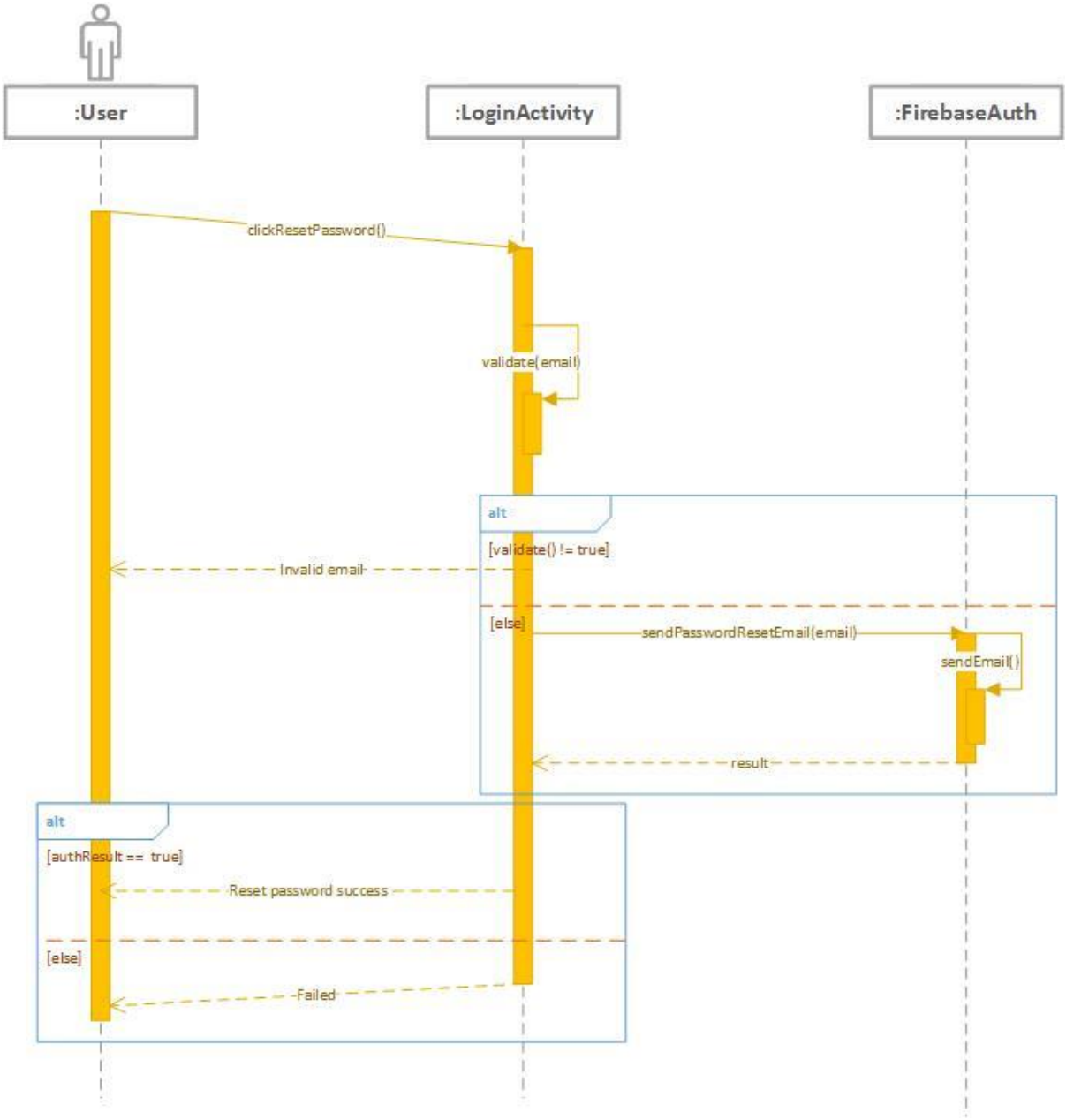



**Fig 5: sequence reset password**

## 5.3 Class diagram

A class diagram is an illustration of the relationships and source code dependencies among classes in the Unified Modelling Language (UML). In this context, a class defines the methods and variables in an object, which is a specific entity in a program or the unit of code representing that entity.

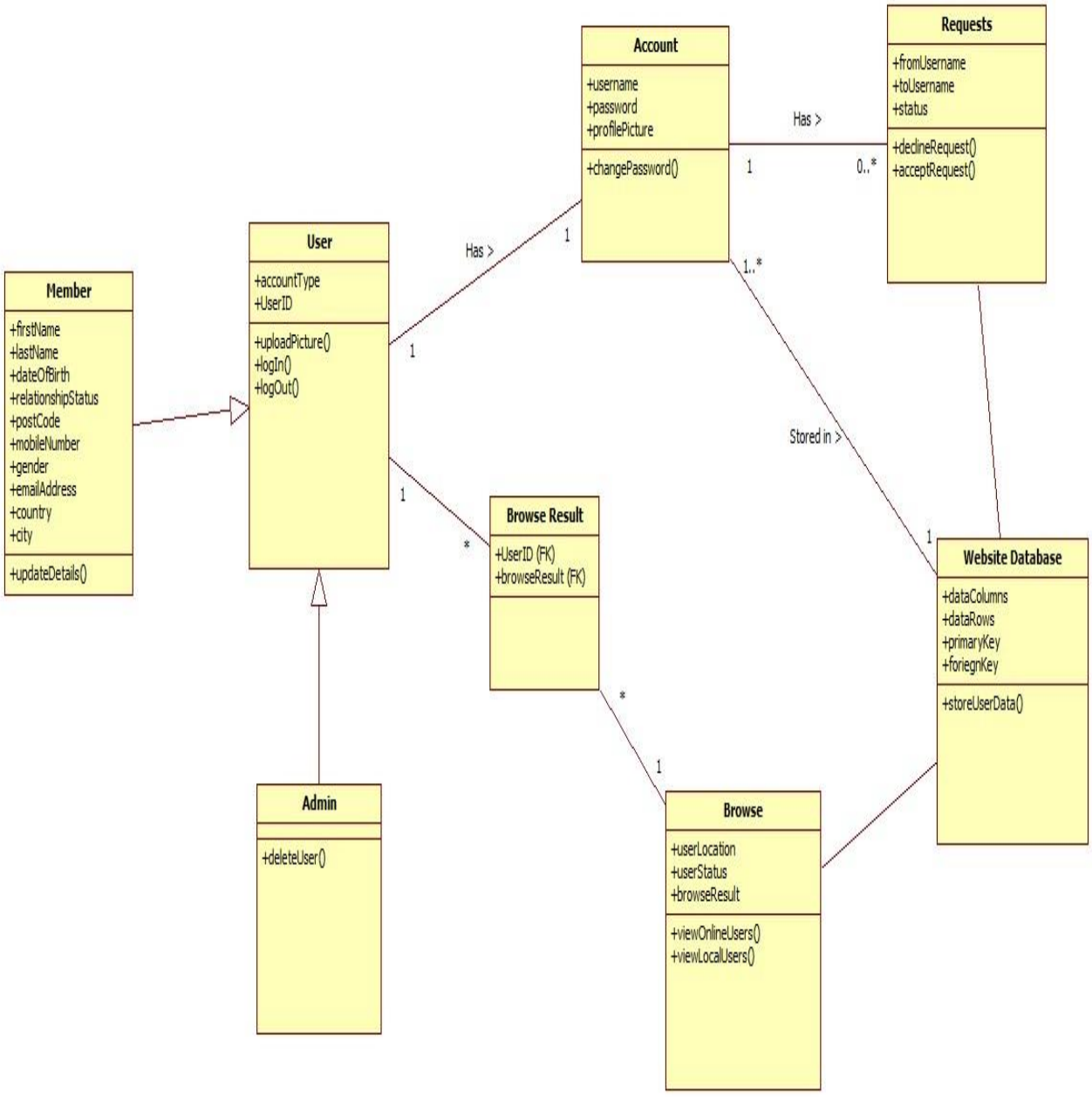



**Fig 6: class diagram**

## 5.4 ER diagram

An entity-relationship (ER) diagram is a specialized graphic that illustrates the interrelationships between entities in a database. A type of diagram used in data modeling for relational data bases.

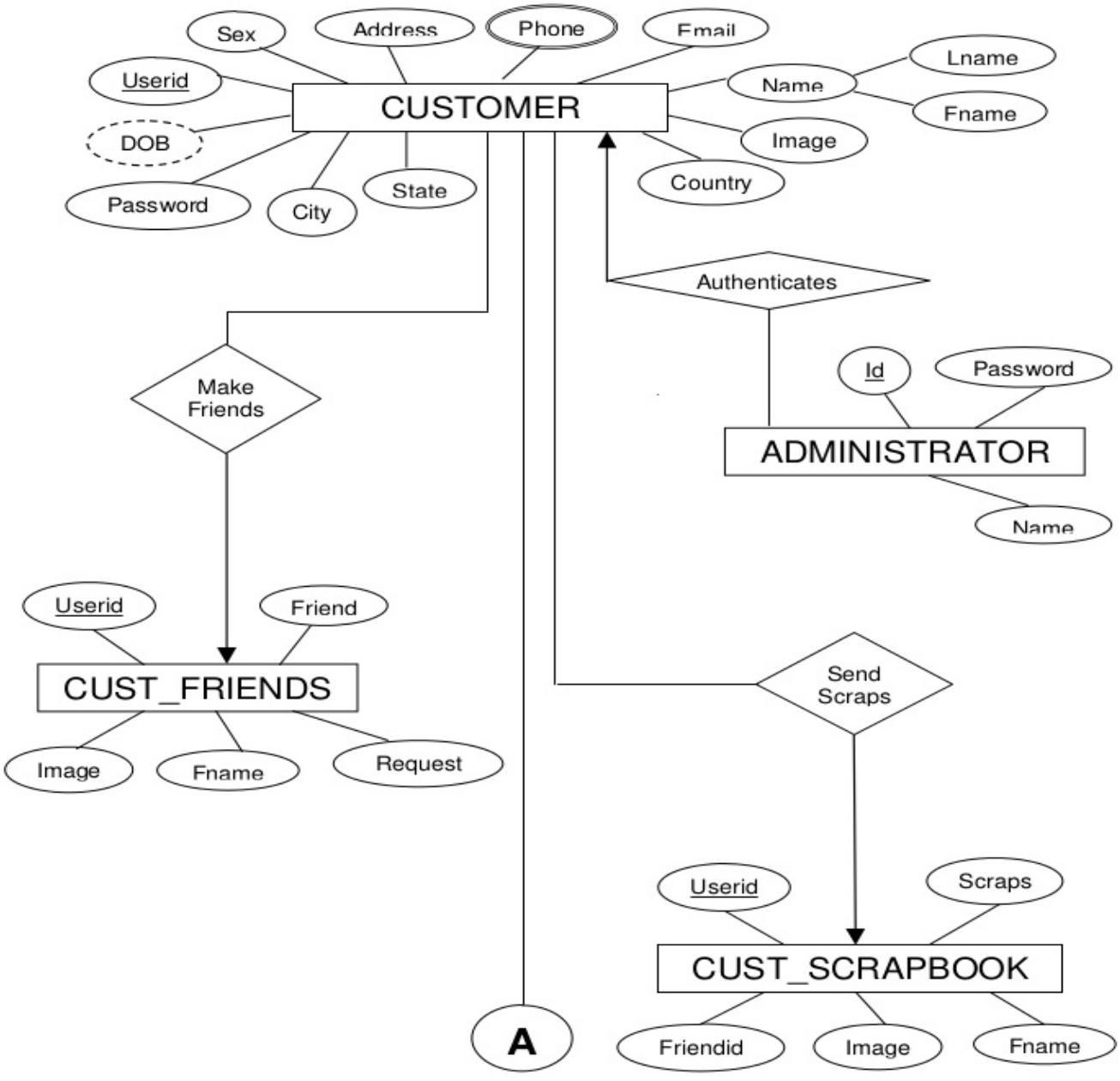



**Fig 7: ENTITY-RELATIONSHIP DIAGRAM**

## 5.5 Use Case diagram

A use case diagram is a graphic depiction of the interactions among the elements of a system. A use case is a methodology used in system analysis to identify, clarify, and organize system requirements.

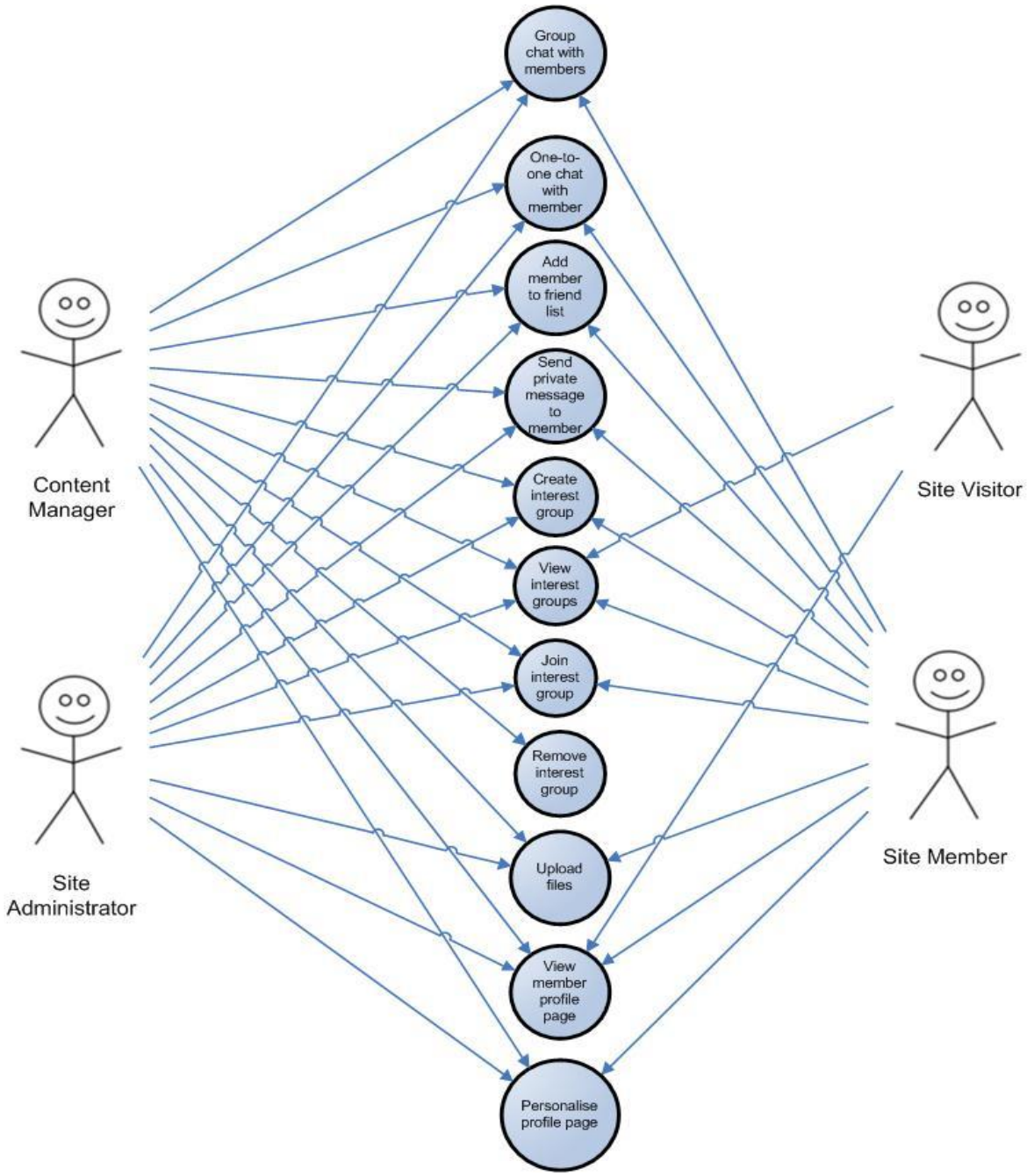



**Fig 8: use case diagram**

# CHAPTER VI

# TESTING

## TEST CASES:

In any endeavor, testing is always important before the actual implementation. SDLC is no different, in fact testing in SDLC is so important these days that developers have to work in testing longer that most stages in SDLC.

Types of Testing:

**Testing by developers** – Developers will have to test the software themselves extensively. Even though they are also the one who have developed the software, they still have to run it extensively. These are the usual steps developers follow in testing their software:

**1. Checking of Syntax** – The software is checked as it is. Developers just run software and look for possible errors. A very basic checking to know if there are major functions and errors that will occur.

**2. Module Testing** – After the walk through, developers will again check the software in terms of individual modules. Each module will be tested extensively for possible errors.

**3. Integration Testing** – Once the module has been tested, it is time to test them as being integrated to other modules.

**4. System Testing** – After the integration, the whole program will again be tested. Case studies will again be applied with integrated software.

**Testing by Users -** Once the developers have finished their own testing with the help of other professional developers, the software is ready for release to the public or to the intended users.

The stages usually followed are:

Alpha Testing – Developers usually choose the users who will try out the program. For the general public, it is often tested on known users or even technology bloggers who can honestly give their opinion of the software. It is also the time where developers try to create scenarios for their software such as:

Recovery – Developers will try recovering the software or the program in case it crashes.

# CHAPTER VII

# USER INTERFACES

## 7.1 IMPLEMENTATION SCREENSHOT

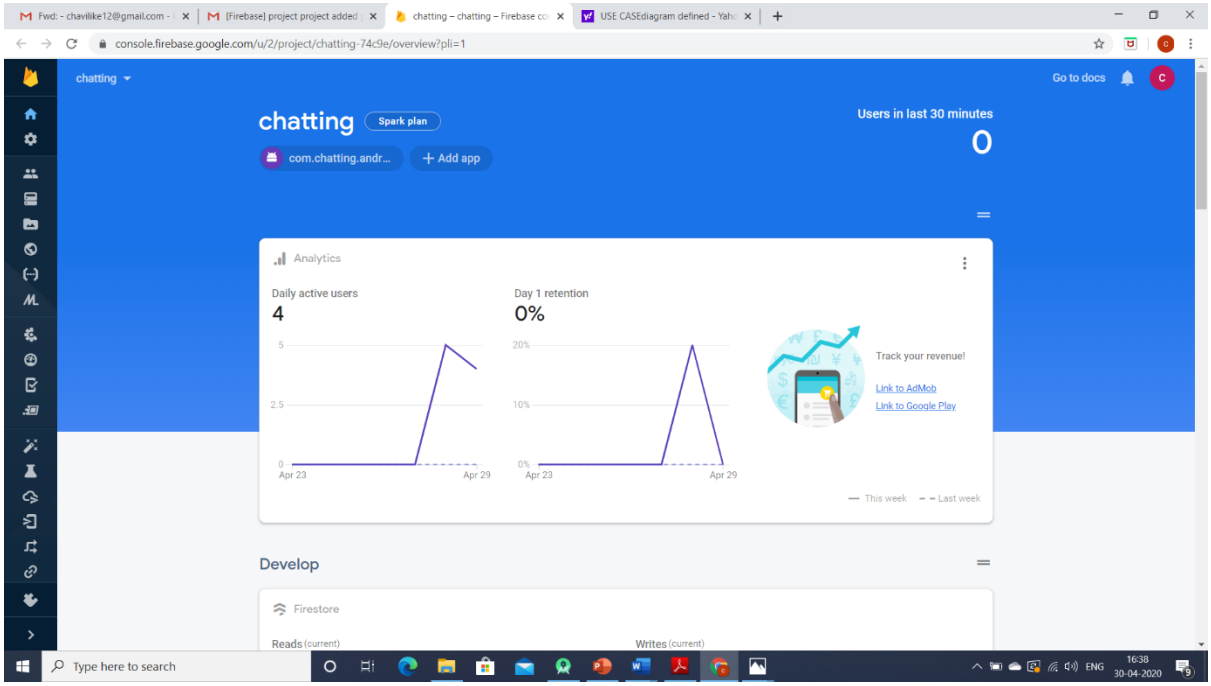



**Fig 9: firebase data of chatting**

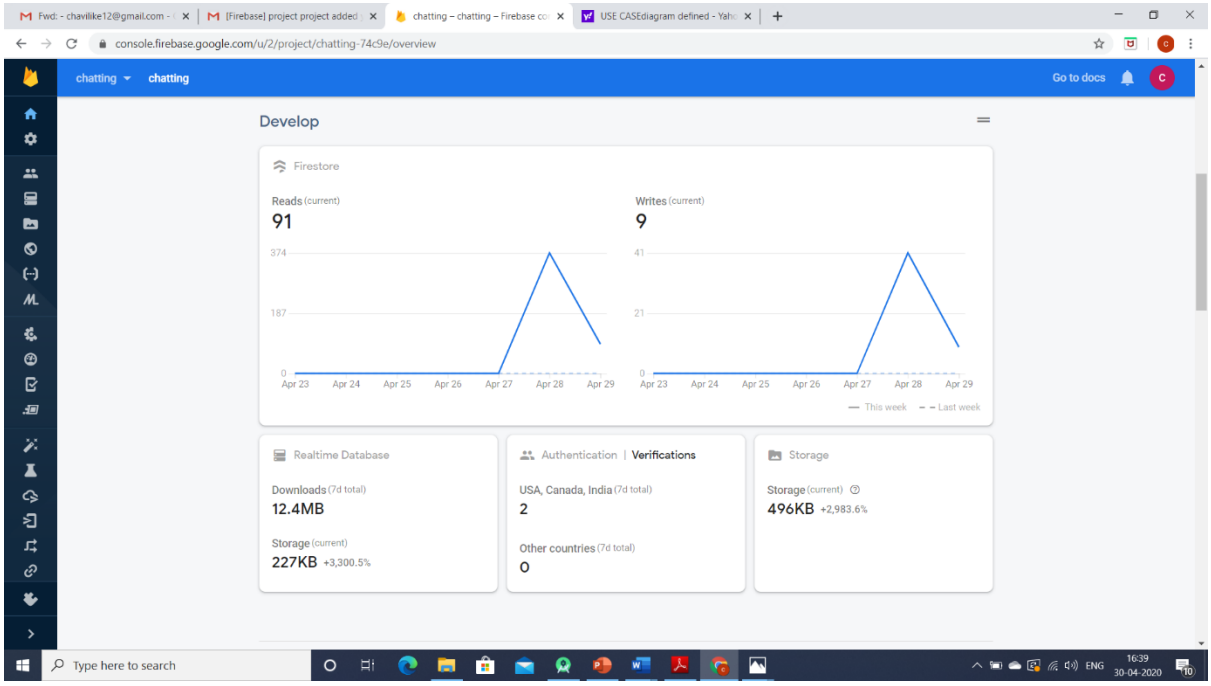



**Fig 10: Real time database & normal database**


**Fig 11: ANDROID APP ICON**

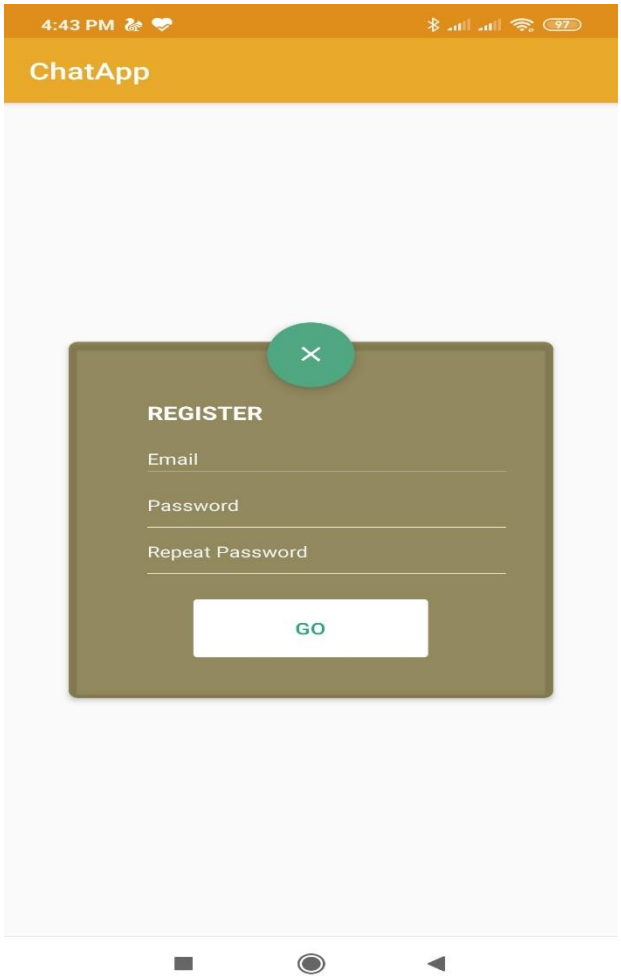



**Fig 12: REGISTERATION PAGE**

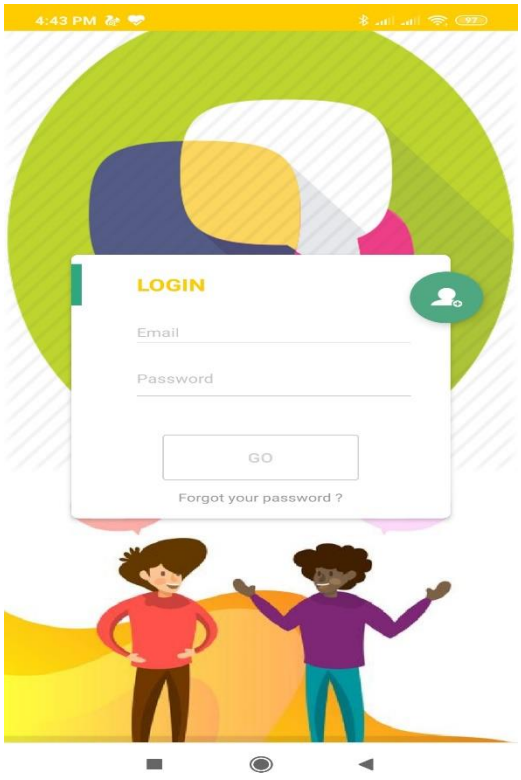



**Fig 13:LOGIN PAGE**


**Fig 14: CHATTING PAGE**

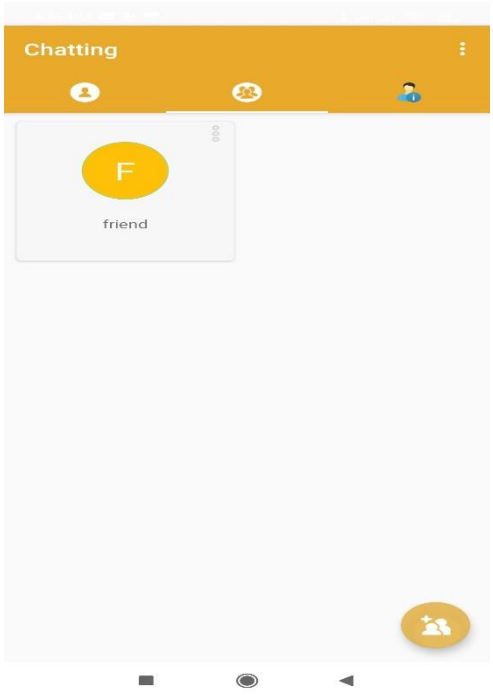



**Fig 15: GROUP CHATTING PAGE**

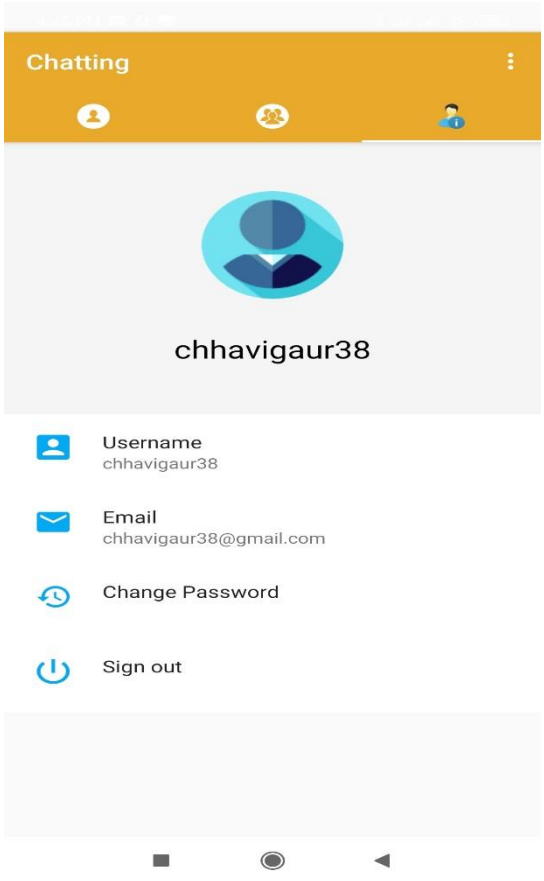



**Fig 16:USER INFORMATION PAGE**

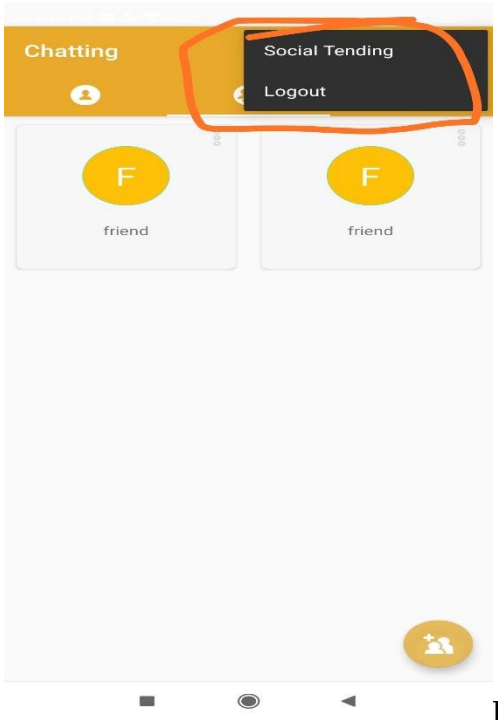



**FIG 17: SOCIAL TENDING**

***(THESE IS USED FOR CURRENT SOCIAL ISSUES).***

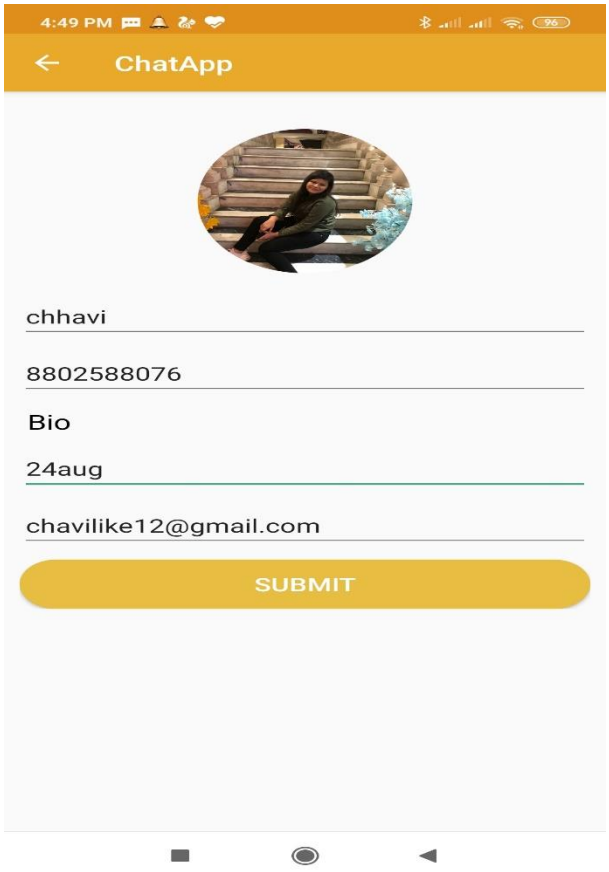



**FIG 18: SOCIAL ISSUE REGISTRATION PAGE**


**FIG 19: SOCIAL ISSUE POSTING PAGE**

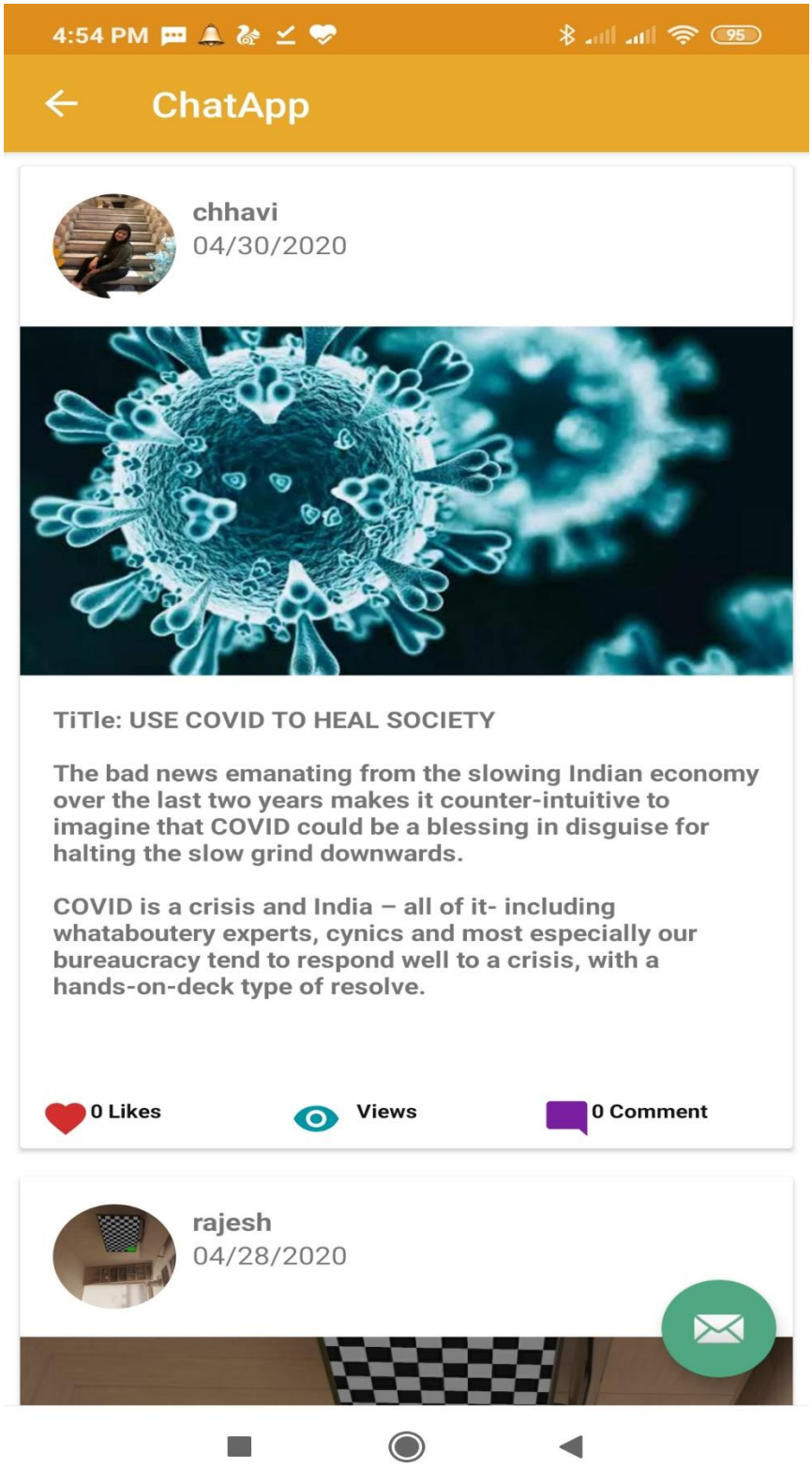



**FIG 20: SOCIAL ISSUE SHARING PAGE**

## 7.2 CODING

```
Package com.chatting.androidfirebasechat;

import android.content.Intent;
import android.os.Bundle;
import androidx.annotation.NonNull;

import com.chatting.androidfirebasechat.MySocial.HomeFragment;
import com.chatting.androidfirebasechat.MySocial.PostActivity;
import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.android.material.tabs.TabLayout;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentManager;
import androidx.fragment.app.FragmentPagerAdapter;
import androidx.viewpager.widget.ViewPager;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Toast;

import com.chatting.androidfirebasechat.data.FriendDB;
import com.chatting.androidfirebasechat.data.GroupDB;
import com.chatting.androidfirebasechat.data.StaticConfig;
import com.chatting.androidfirebasechat.service.ServiceUtils;
import com.chatting.androidfirebasechat.ui.FriendsFragment;
import com.chatting.androidfirebasechat.ui.GroupFragment;
import com.chatting.androidfirebasechat.ui.LoginActivity;
import com.chatting.androidfirebasechat.ui.UserProfileFragment;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;

import java.util.ArrayList;
import java.util.List;

public class MainActivity extends AppCompatActivity {
    private static String TAG = "MainActivity";
    private ViewPager viewPager;
    private TabLayout tabLayout = null;
    public static String STR_FRIEND_FRAGMENT = "FRIEND";
    public static String STR_GROUP_FRAGMENT = "GROUP";
    public static String STR_INFO_FRAGMENT = "INFO";
   // public static String STR_SOCIAL_FRAGMENT = "SOCIAL";

    private FloatingActionButton floatButton;
    private ViewPagerAdapter adapter;
    private FirebaseAuth mAuth;
```

```
private FirebaseAuth.AuthStateListener mAuthListener;
private FirebaseUser user;

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

    Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
    if (toolbar != null) {
        setSupportActionBar(toolbar);
        getSupportActionBar().setTitle("Chatting ");
    }

    viewPager = (ViewPager) findViewById(R.id.viewpager);
    floatButton = (FloatingActionButton) findViewById(R.id.fab);
    initTab();
    initFirebase();
}

private void initFirebase() {
    // Start the program to start, close
    mAuth = FirebaseAuth.getInstance();
    mAuthListener = new FirebaseAuth.AuthStateListener() {
        @Override
        public void onAuthStateChanged(@NonNull FirebaseAuth firebaseAuth) {
            user = firebaseAuth.getCurrentUser();
            if (user != null) {
                StaticConfig.UID = user.getUid();
            } else {
                MainActivity.this.finish();
                // User is signed in
                startActivity(new Intent(MainActivity.this, LoginActivity.class));
                Log.d(TAG, "onAuthStateChanged:signed_out");
            }
            // ...
        }
    };
}


@Override
protected void onStart() {
    super.onStart();
    mAuth.addAuthStateListener(mAuthListener);
    ServiceUtils.stopServiceFriendChat(getApplicationContext(), false);
}

@Override
protected void onStop() {
```

```
        super.onStop();
        if (mAuthListener != null) {
            mAuth.removeAuthStateListener(mAuthListener);
        }
    }

    @Override
    protected void onDestroy() {
        ServiceUtils.startServiceFriendChat(getApplicationContext());
        super.onDestroy();
    }

    /**
     * Start me 3 tab
     */
    private void initTab() {
        tabLayout = (TabLayout) findViewById(R.id.tabs);

tabLayout.setSelectedTabIndicatorColor(getResources().getColor(R.color.colorIndivateTab
));
        setupViewPager(viewPager);
        tabLayout.setupWithViewPager(viewPager);
        setupTabIcons();
    }


    private void setupTabIcons() {
        int[] tabIcons = {
                R.drawable.ic_tab_person,
                R.drawable.ic_tab_group,
                R.drawable.ic_tab_infor

        };

        tabLayout.getTabAt(0).setIcon(tabIcons[0]);
        tabLayout.getTabAt(1).setIcon(tabIcons[1]);
        tabLayout.getTabAt(2).setIcon(tabIcons[2]);
       // tabLayout.getTabAt(3).setIcon(tabIcons[3]);
    }

    private void setupViewPager(ViewPager viewPager) {
        adapter = new ViewPagerAdapter(getSupportFragmentManager());
        adapter.addFrag(new FriendsFragment(), STR_FRIEND_FRAGMENT);
        adapter.addFrag(new GroupFragment(), STR_GROUP_FRAGMENT);
        adapter.addFrag(new UserProfileFragment(), STR_INFO_FRAGMENT);
       // adapter.addFrag(new HomeFragment(), STR_SOCIAL_FRAGMENT);
        floatButton.setOnClickListener(((FriendsFragment)
adapter.getItem(0)).onClickFloatButton.getInstance(this));
        viewPager.setAdapter(adapter);
        viewPager.setOffscreenPageLimit(3);
```

```
        viewPager.addOnPageChangeListener(new ViewPager.OnPageChangeListener() {
            @Override
            public void onPageScrolled(int position, float positionOffset, int
positionOffsetPixels) {

            }

            @Override
            public void onPageSelected(int position) {
                ServiceUtils.stopServiceFriendChat(MainActivity.this.getApplicationContext(),
false);
                if (adapter.getItem(position) instanceof FriendsFragment) {
                    floatButton.setVisibility(View.VISIBLE);
                    floatButton.setOnClickListener(((FriendsFragment)
adapter.getItem(position)).onClickFloatButton.getInstance(MainActivity.this));
                    floatButton.setImageResource(R.drawable.plus);
                } else if (adapter.getItem(position) instanceof GroupFragment) {
                    floatButton.setVisibility(View.VISIBLE);
                    floatButton.setOnClickListener(((GroupFragment)
adapter.getItem(position)).onClickFloatButton.getInstance(MainActivity.this));
                    floatButton.setImageResource(R.drawable.ic_float_add_group);
                }
//                else if (adapter.getItem(position) instanceof HomeFragment) {
//                    floatButton.setVisibility(View.VISIBLE);
//                    //floatButton.setOnClickListener(((HomeFragment)
adapter.getItem(position)).onClickFloatButton.getInstance(MainActivity.this));
//                    floatButton.setImageResource(R.drawable.ic_float_add_group);
//                }
                else {
                    floatButton.setVisibility(View.GONE);
                }
            }

            @Override
            public void onPageScrollStateChanged(int state) {

            }
        });
    }

    //    @Override
//    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
//        super.onActivityResult(requestCode, resultCode, data);
//        if (requestCode == REQUEST_CODE_LOGIN && resultCode == RESULT_OK) {
//            if
(data.getStringExtra(STR_EXTRA_ACTION).equals(LoginActivity.STR_EXTRA_ACTION_L
OGIN)) {
//                authUtils.signIn(data.getStringExtra(STR_EXTRA_USERNAME),
data.getStringExtra(STR_EXTRA_PASSWORD));
//            } else if
```

```
(data.getStringExtra(STR_EXTRA_ACTION).equals(RegisterActivity.STR_EXTRA_ACTION
_REGISTER)) {
//              authUtils.createUser(data.getStringExtra(STR_EXTRA_USERNAME),
data.getStringExtra(STR_EXTRA_PASSWORD));
//          }else
if(data.getStringExtra(STR_EXTRA_ACTION).equals(LoginActivity.STR_EXTRA_ACTION_
RESET)){
//              authUtils.resetPassword(data.getStringExtra(STR_EXTRA_USERNAME));
//          }
//      } else if (resultCode == RESULT_CANCELED) {
//          this.finish();
//      }
//  }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.menu_main, 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();

        //noinspection SimplifiableIfStatement
        if (id == R.id.Social_Tending) {
            Intent intent= new Intent(MainActivity.this, PostActivity.class);
            intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK |
Intent.FLAG_ACTIVITY_CLEAR_TASK);
            startActivity(intent);

ServiceUtils.stopServiceFriendChat(getApplicationContext().getApplicationContext(), true);
            finish();
            return true;
        } else if (id == R.id.logout) {

            FirebaseAuth.getInstance().signOut();
            FriendDB.getInstance(getApplicationContext()).dropDB();
            GroupDB.getInstance(getApplicationContext()).dropDB();

ServiceUtils.stopServiceFriendChat(getApplicationContext().getApplicationContext(), true);
            finish();

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

```
    class ViewPagerAdapter extends FragmentPagerAdapter {
        private final List<Fragment> mFragmentList = new ArrayList<>();
        private final List<String> mFragmentTitleList = new ArrayList<>();

        public ViewPagerAdapter(FragmentManager manager) {
            super(manager);
        }

        @Override
        public Fragment getItem(int position) {
            return mFragmentList. Get(position);
        }

        @Override
        public int getCount() {
            return mFragmentList.size();
        }

        public void addFrag(Fragment fragment, String title) {
            mFragmentList.add(fragment);
            mFragmentTitleList.add(title);
        }

        @Override
        public CharSequence getPageTitle(int position) {

            // return null to display only the icon
            return null;
        }
    }
}
```

# CHAPTER VIII

## RESULT

The final system will result as a real time communication application which provides the users to communicate to each other with an ease. The application will have a login page through which the user can register and login themselves. Home page of the application contains the previous messages if any. The user can be
able to search for the other user. User can send and receive text messages. The user can create chat rooms and can search for the content or information. With these chat rooms users can exchange views and information about various topics. The identity of the user can also be made hidden in these public chat rooms.

# CHAPTER IX

## Conclusion

There is always a room for improvements in any apps. Right now we are just dealing with text communication. There are several android apps which serve similar purpose as this project, but these apps were rather difficult to use and provide confusing interfaces. A positive first impression is essential in human relationship as well as in human computer interaction. This project hopes to develop a chat service Android app with high quality user interface.

In future we may be extended to include features such as:

1. File Transfer
2. Voice Message
3. Video Message
4. Audio Call
5. Video Call
6. Group Call

# CHAPTER X

## References


1. http://www.fit.vutbr.cz/study/courses/ITS/public/ieee829.html

2. https://techwhirl.com/user-guide-template/

3.https://www.onetonline.org/search/t2/examples/43231507?s=management%20software.

4. http://www.projectinsight.net/project-management-basics/project-management-schedule.

5. https://techwhirl.com/business-requirements-document-brd-template/

6. https://web.cs.dal.ca/hawkey/3130/srstemplate−ieee.dochttps : //bia.ca/risk−management − the − what − why − and − how/.

7. WWW.GOOGLE.COM