Messaging Chat Application

A PROJECT REPORT SUBMITTED

by

Aman Jain (2000290140016)

Rahul Kumar (2000290140094)

Shubhranshu Yadav (2000290140019)

Paras Sharma (2000290140083)

Submitted in partial fulfillment of the Requirements

for the Degree of

Master of Computer Application

Under the Supervision of

Mr. Ankit Verma

Assistant Professor

Submitted to

Faculty of MCA

DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad Uttar

Pradesh-201206

(14 JAN 2022)

STUDENT DECLARATION

We declare that minor project entitled "MESSAGING CHAT APPLICATION" is our own work conducted under the supervision of "Mr. Ankit Verma", Department of Master of Computer Application, KIET Group of Institutions, Ghaziabad.

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

Date:14-01-2022

AMAN JAIN(2000290140016)

RAHUL KUMAR (2000290140094)

SHUBHRANSHU YADAV(2000290140019)

PARAS SHARMA (2000290140083)

This is to certify that the above statement made by the candidate is correct to the best of our knowledge.

Signature of HOD

(Dr.) Ajay Shrivastava

HOD-MCA

Date: 14-01-2022

Signature of Internal Prof

Dr. Arun Tripathi

Associate Professor

ACKNOWLEDGEMENT

Any software project is not a work of an individual. It combines efforts, ideas, suggestions, reviews and hard work. We express our sincere gratitude to all those who initiated and helped us in the successful completion of our project. One of the most pleasing aspect in collecting the necessary information and compiling it is the opportunity to thanks those who have actively contributed to it.

We take this occasion to thank God, almighty for blessing us with his grace and taking our endeavor to a successful culmination. We extend our sincere and heartfelt thanks to our esteemed guide, **Mr. Ankit Verma**, for providing us with the right guidance and advice at the crucial junctures and for showing us the right way.

We are highly indebted to Prof **(Dr.) Ajay Shrivastava** (**HOD- MCA**) for his continuous effort in building a good infrastructure and develop a professional attitude within ourselves during the academic period of MCA.

We would like to thank the other faculty members also, at this occasion. Last but not the least, We would like to thank our friends and family for the support and encouragement they have given us during our work.

AMAN JAIN(2000290140016) MCA-III Semester

RAHUL KUMAR (2000290140094) MCA-III Semester

SHUBHRANSHU YADAV(2000290140019) MCA-III Semester

PARAS SHARMA (2000290140083) MCA-III Semester

CERTIFICATE

Certified that **AMAN JAIN(2000290140016), RAHUL KUMAR (2000290140094), SHUBHRANSHU YADAV(2000290140019)**, **PARAS SHARMA (2000290140083)** have carried out the project work having "**MESSAGING CHAT APPLICATION**" for Master of Computer Applications from Dr. A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Technical University, Lucknow under our supervision. The project report embodies original work, and studies are carried out by the students himself / herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Date: 14-01-2022

AMAN JAIN(2000290140016)

RAHUL KUMAR (2000290140094)

SHUBHRANSHU YADAV(2000290140019)

PARAS SHARMA (2000290140083)

This is to certify that the above statement made by the candidates is correct to the best of our knowledge.

Date: 14-01-2022

Mr. Ankit Verma

Assistant Professor
Department of Computer Applications
KIET Group of Institutions, Ghaziabad

Signature of Internal Examiner Signature of External Examiner

Dr. Ajay Shrivastava
Head, Department of Computer Applications KIET
Group of Institutions, Ghaziabad

ABSTRACT

Messaging apps now have more global users than traditional social networks—which means they will play an increasingly important role in the distribution of digital journalism in the future. Drawing upon our interviews and case studies, we identify a number of opportunities and challenges for organizations using—or hoping to use—messaging apps for news.

This project is a Android based chat application for the users. The objective of the project is to provide an easy way of communication between the users of the application.

Teleconferencing or chatting refers to any kind of communication that offers a real-time transmission of messages from sender to the receiver.

Chatting is a method of using technology to bring people and ideas together despite the geographical barriers. The technology to provide the chatting facility has been available for years, but the acceptance is quite recent.

Analysis of chatting provides an overview of the technologies used, available features, functions, system of the architecture of the application, the structure of database of an Instant Messaging application: IChat (IC). The objective of IC application is to facilitate text messaging, group chatting option, data transfer without size restriction which is commonly seen in most of the messaging applications.

Chatting Application is a social-networking tool that leverages on technology advancement thereby allowing its users communicate and share media. It offers a wonderful one stop shop experience for keeping in touch with people you know.

TABLE OF CONTENT

1.0) INTRODUCTION :

- There are numerous products available that allow for real time “chatting” over the Internet.

- The purpose of this project is to implement a Java based chat application that will allow users with an internet connection to engage in private and public conversations.

- The development of this project centered on the development of a message protocol that would allow the application to properly log in users, send messages, and perform system maintenance.

1.1) Problem Introduction:

- This project is to create a chat application with a server and clients to enable the clients to chat with many other clients in the same common chat group.

- This project is to simulate the multicast chatting. In the case of multicasting when a message is sent to a group of clients, then only a single message is sent to the router.

- The main purpose of this project is to provide multi chatting functionality through network.

- This project can play an important role in organizational field where employees can connect together through LAN.

1.2) Innovative Ideas of Project:

- **GUI**: Easy to use GUI (Graphical User Interface), hence any user with minimal knowledge of operating a system can use the software.

- **Platform independence**: The messenger operates on any system irrelevant of the underlying operating system.

- **Unlimited clients**: „n" number of users can be connected without any performance degradation of the server.

1.3) Project Objective:

- **Communication**: To develop an instant messaging solution to enable users to seamlessly communicate with each other

- **User friendliness**: The project should be very easy to use enabling even a novice person to use it.

1.4) Scope of the project:

- Broadcasting Chat Server Application is going to be a text communication software, it will be able to communicate between two computers using point to point communication.

- The limitation of Live Chat is it does not support audio conversations. To overcome this limitation we are concurrently working on developing better technologies.

- Companies would like to have a communication software wherein they can communicate instantly within their organization.

- The fact that the software uses an internal network setup within the organization makes it very secure from outside attacks.

1.5) Related Previous Work:

- When the existing system was studied, it was found having some problems, existing system was very time consuming and was not very efficient.

- The drawback of the existing system has resulted in to the development of new system, which is very user friendly and effective. Existing system was also very low in performance.

- There is no private chatting option had been available.

- Client can connect with the server with the only IP Address.

2.1) Project Perspective:

- The system to be developed here is an Chat facility. It is a centralized system. It is Client-Server system with centralized database server. All local clients are connected to the centralized server via LAN.

- There is a two way communication between different clients and server. This chat application can be used for group discussion. It allows users to find other logged in users.

- **No need of Internet connection**: Existing system requires Internet connection; whereas in the proposed system only Intranet connection i.e. only a LAN connection is required. This system is useful for those who can not afford to have an Internet connection. For example: schools, colleges, small companies, etc.

- **Conference possible on LAN**: Usually on LANs connections conferencing is not possible. The proposed system allows the LAN users to create and participate in conference. This makes communications possible among number of LAN users simultaneously.

2.1.1) Interface:

- This application interacts with the user through G.U.I. The interface is simple , easy to handle and self-explanatory.

- Once opened, user will easily come into the flow with the application and easily uses all interfaces properly.

- However the basic interface available in our application is Title panel , Content panel and Admin panel.

2.1.2) Hardware Interface:

Minimum requirements will be as follows:

- 128 MB RAM required.

-
- Processor with speed of 500mhz.
  Internet or LAN connection.
- KEYBOARD: 101 key Keyboard

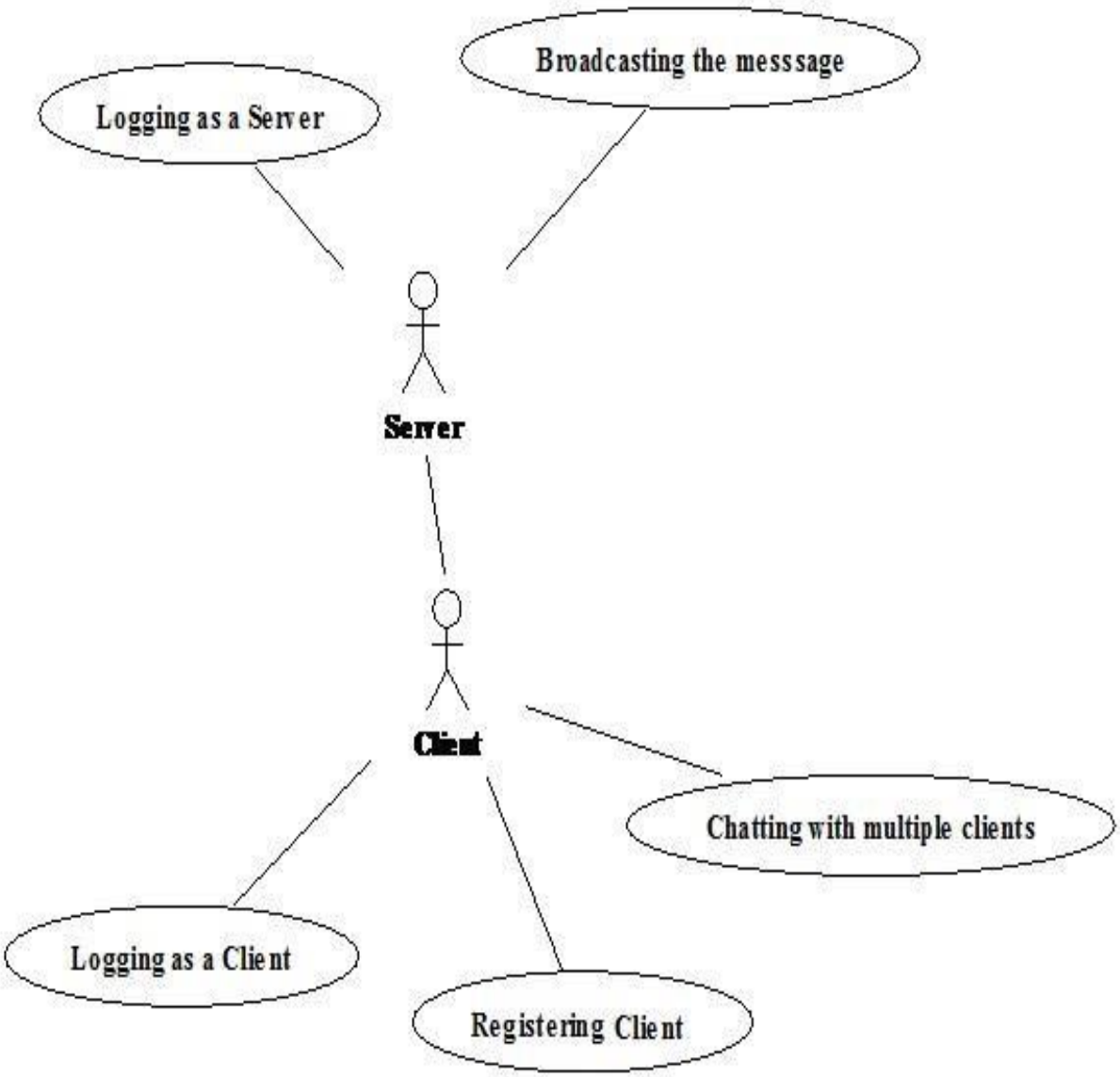
2.1.3 Software Interface:

- Notepad++ is a text editor and source code editor and provides an environment for developing HTML, jsp, JavaScript many other editing purposes.
- Coding done in java so required JDK 1.4 and above for run java programs.
- Operating system (such as window 7, 8, 10 and Linux etc).

2.1.5 Assumption and dependencies:

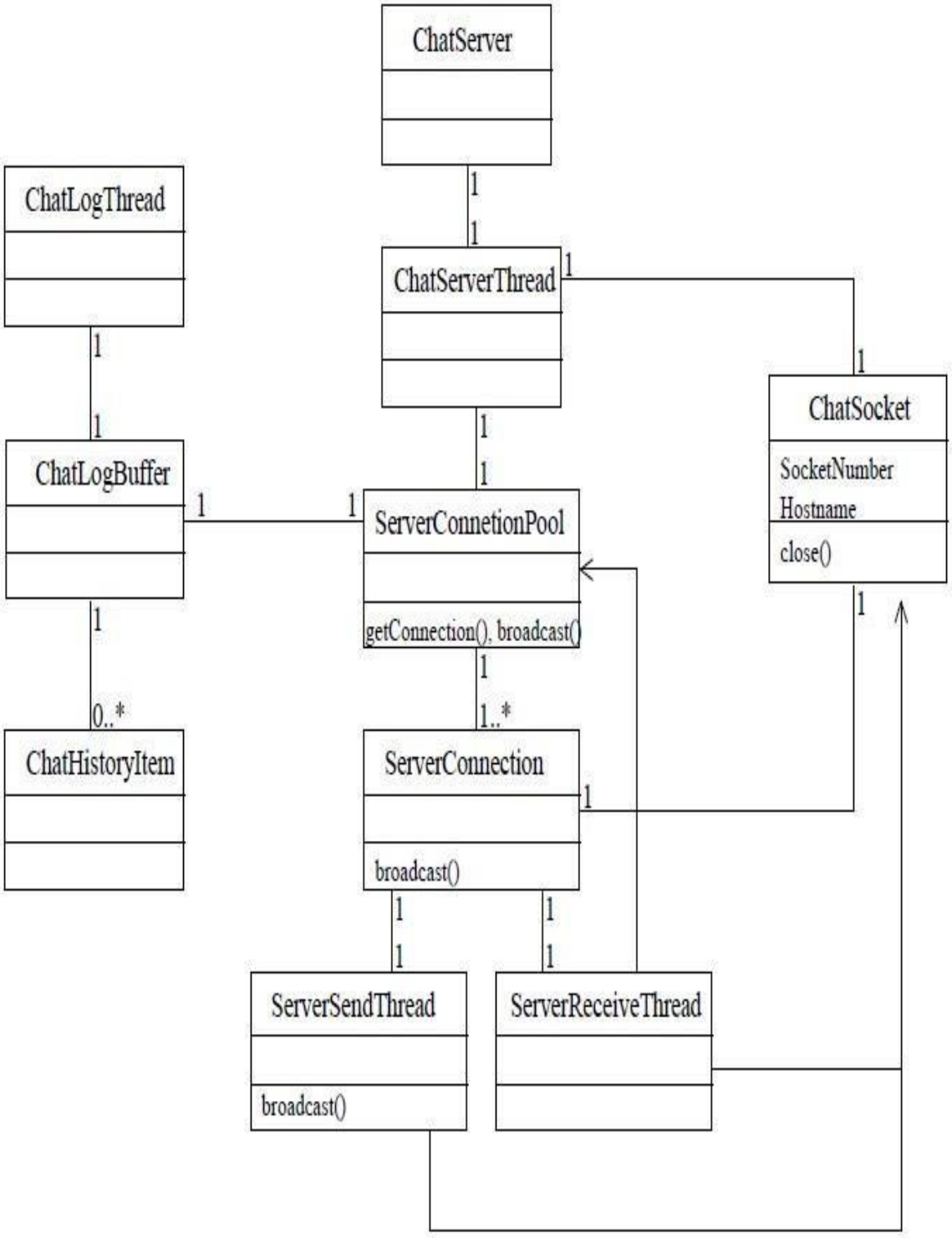
- There should be LAN or internet connection.
- Clients should know each other.
- There can be multiple client

2.1.6) Use Case Diagram:

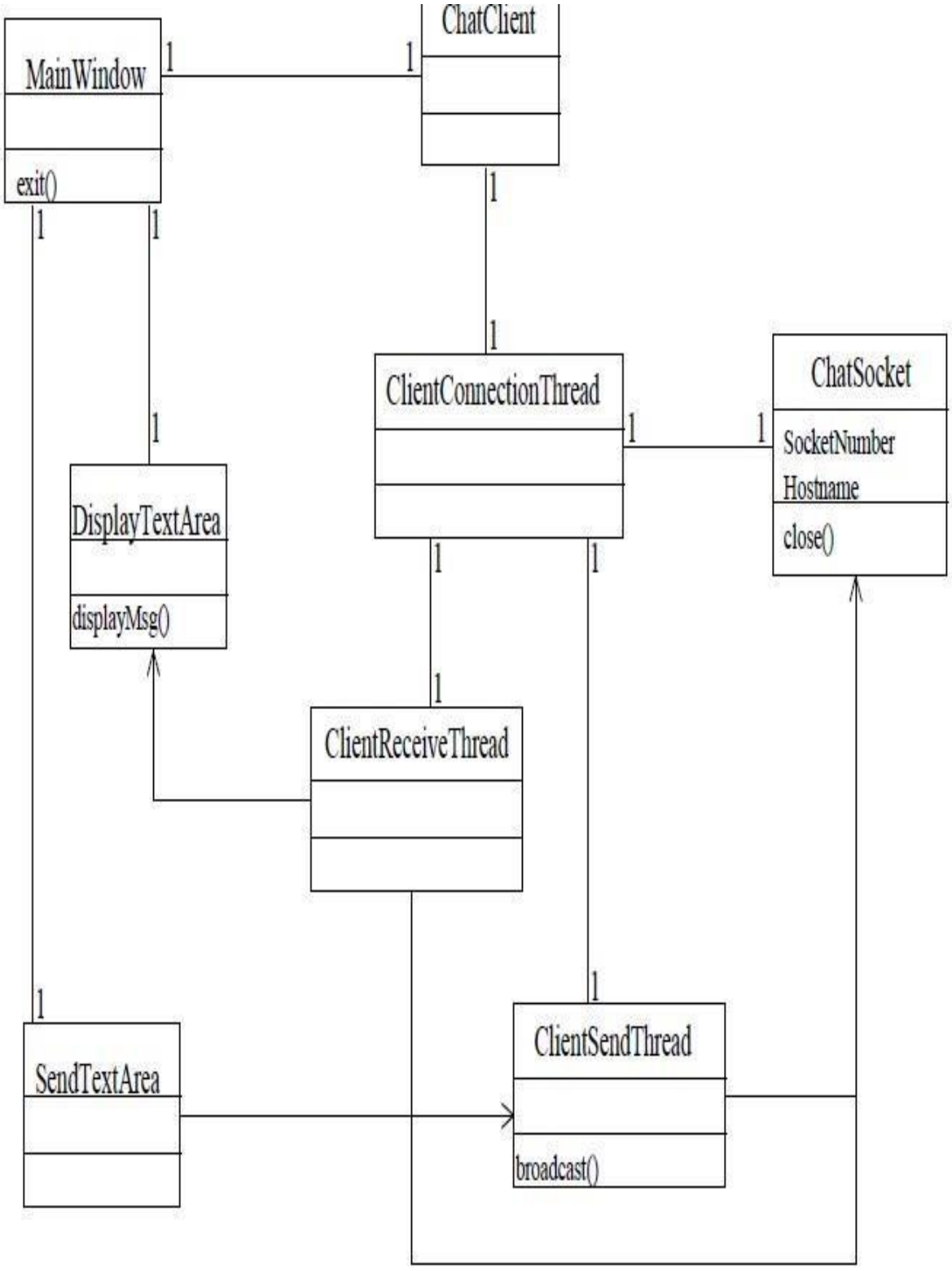


This use case model how client interacts with the server and start getting communicating with other clients.

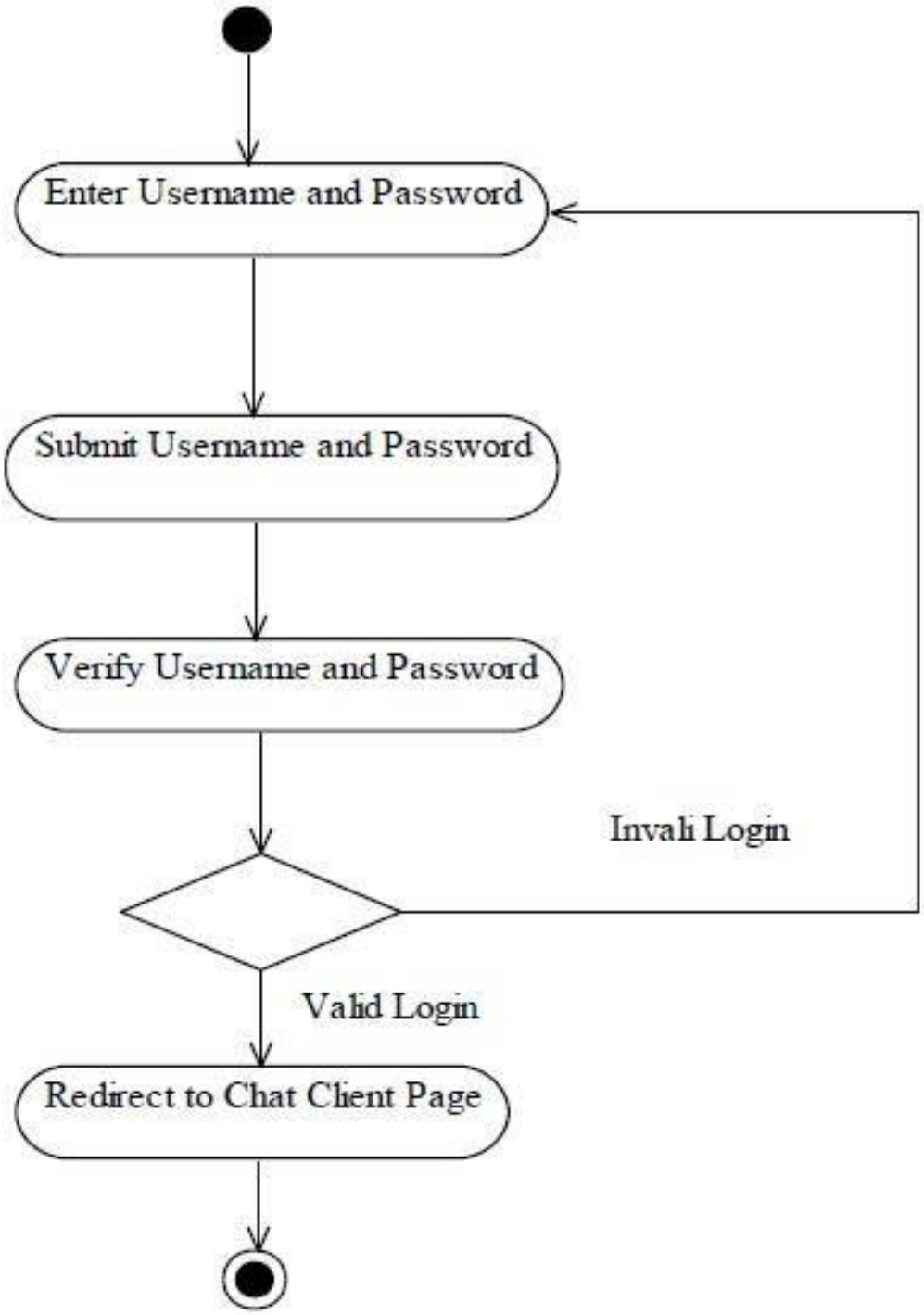
2.1.8) **Class Diagram Of Server:**



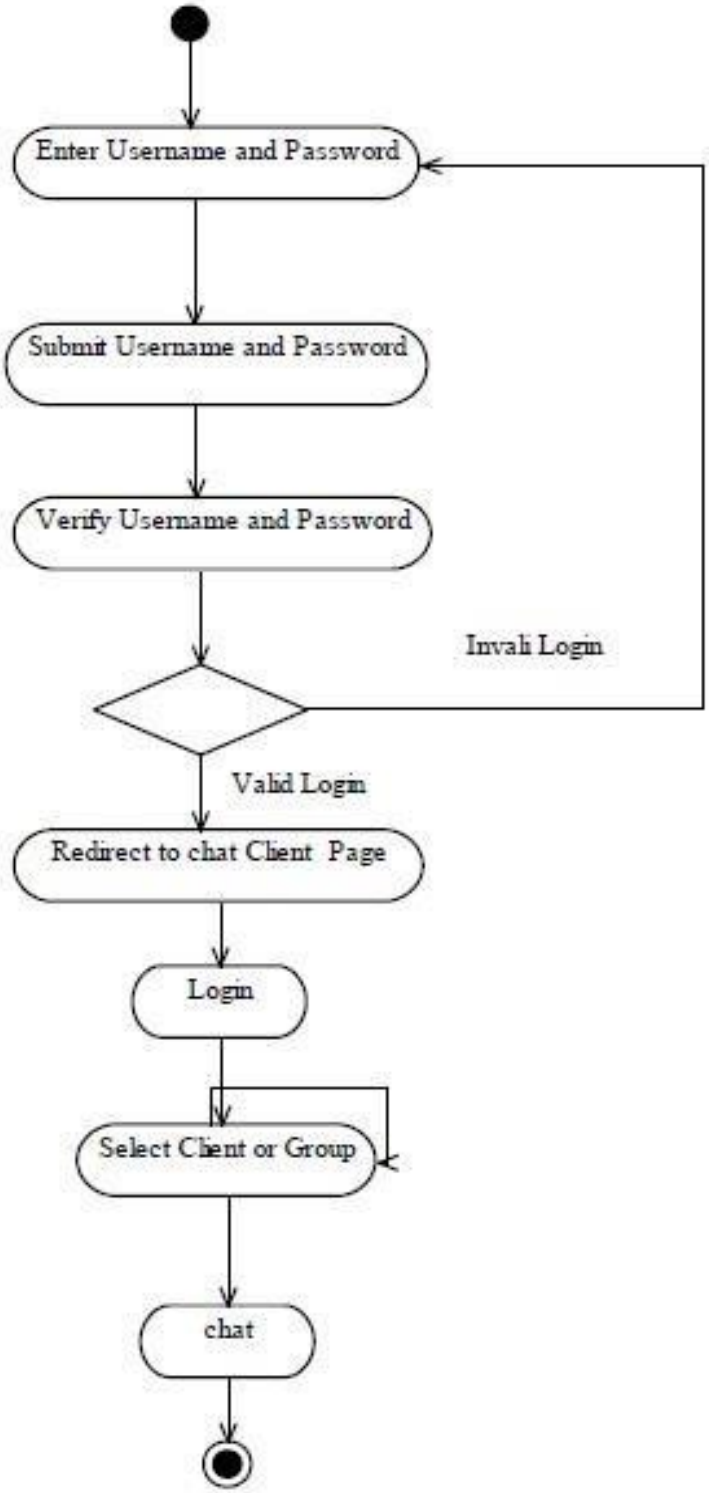
Class Diagram for Client:



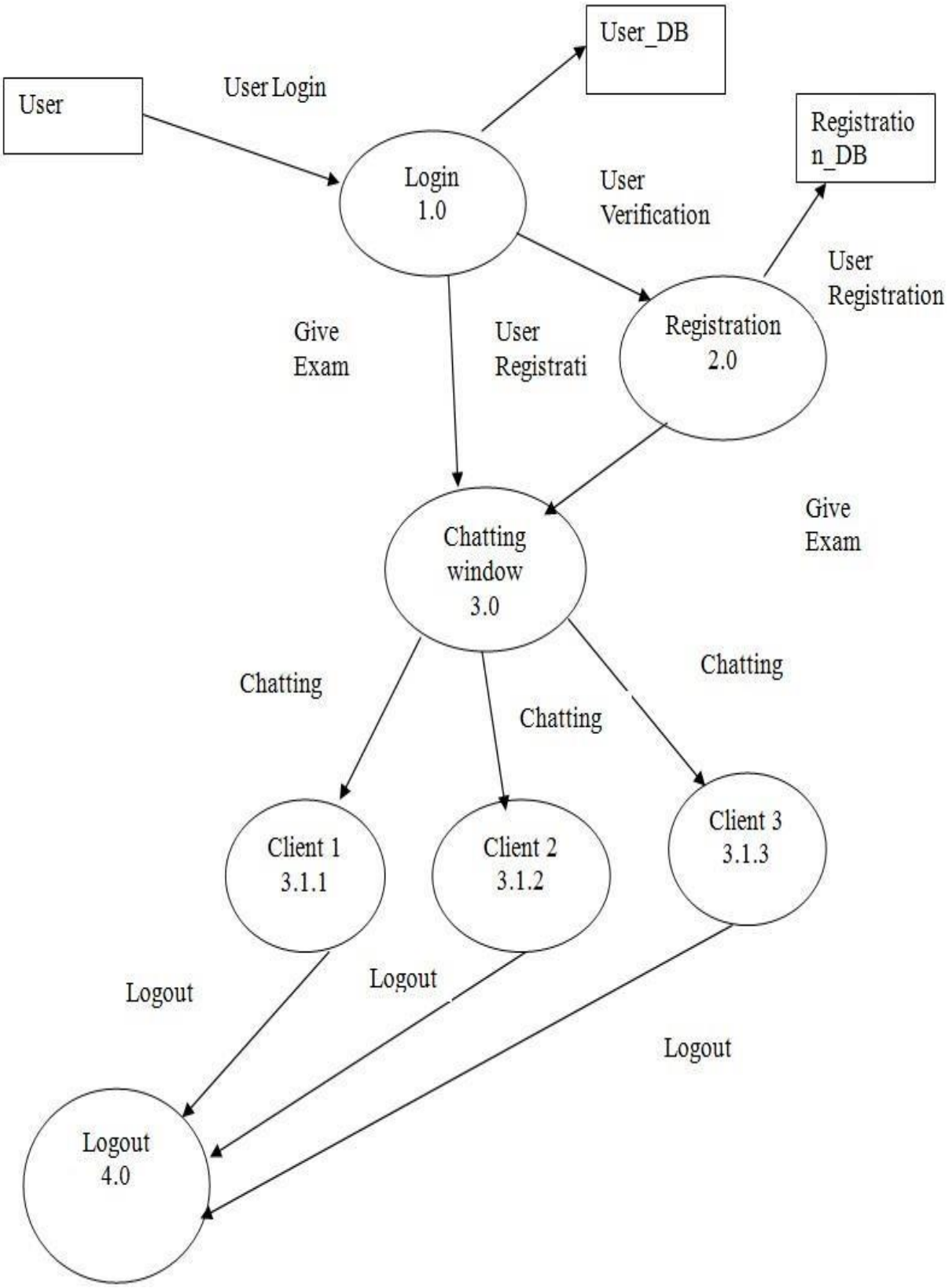
2.1.9) **Activity Diagram For Login:**



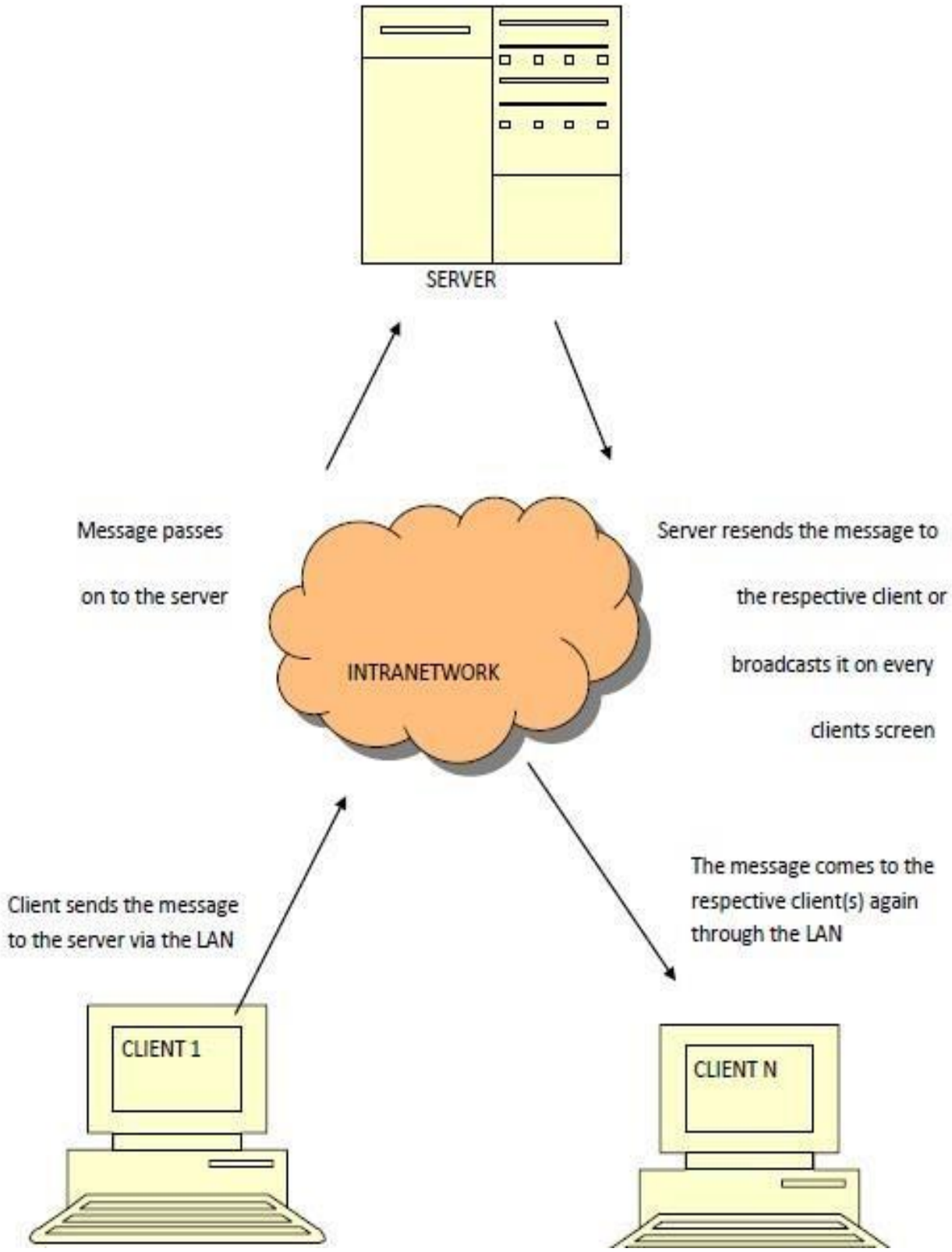
• Activity Diagram For Chatting:



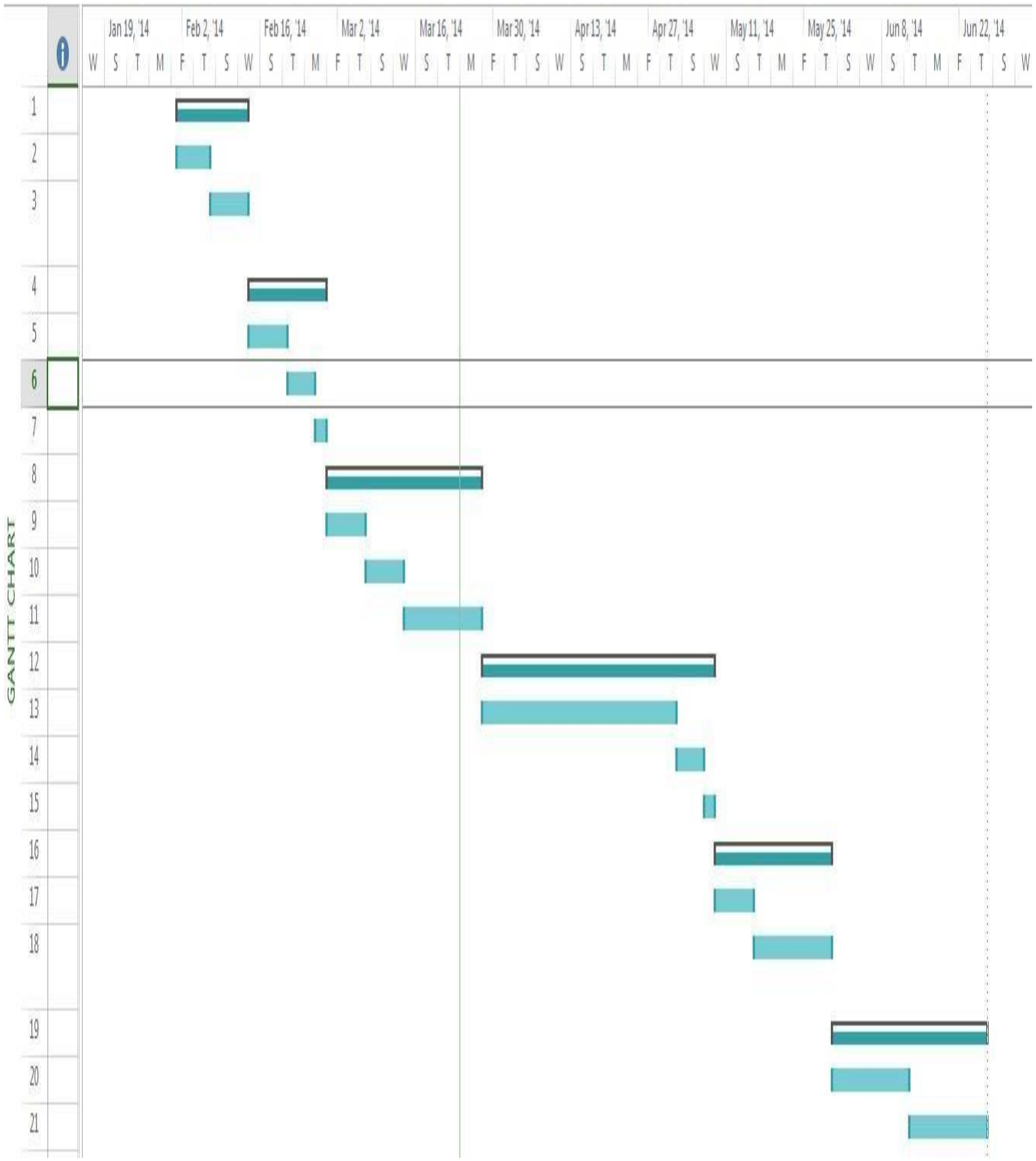
2.9) Data Flow Diagram:



2.10) E-R Diagram:



3.1) Gantt Chart:



3.2) Work Breakdown Structure:

GANTT CHART

	Task Mode	Task Name	Duration	Start	Finish	Predecess
1		**0. Initiation**	**10 days**	**Sat 2/1/14**	**Thu 2/13/14**	
2		0.1 Feasibility Study	5 days	Sat 2/1/14	Thu 2/6/14	
3		0.2 Appoint the Project Team	5 days	Fri 2/7/14	Thu 2/13/14	
4		**1. Planning**	**10 days**	**Fri 2/14/14**	**Thu 2/27/14**	
5		1.1 Requirement	5 days	Fri 2/14/14	Thu 2/20/14	
6		1.2 Resource Plan	3 days	Fri 2/21/14	Tue 2/25/14	
7		1.3 Risk Plan	2 days	Wed 2/26/14	Thu 2/27/14	
8		**2. Designing**	**20 days**	**Fri 2/28/14**	**Thu 3/27/14**	
9		2.1 System Design	5 days	Fri 2/28/14	Thu 3/6/14	
10		2.2 Database Design	5 days	Fri 3/7/14	Thu 3/13/14	
11		2.3 Program Design	10 days	Fri 3/14/14	Thu 3/27/14	
12		**3. Implementing**	**30 days**	**Fri 3/28/14**	**Thu 5/8/14**	
13		3.1 Perform Codding	25 days	Fri 3/28/14	Thu 5/1/14	
14		3.2 Build Deliverable	3 days	Fri 5/2/14	Tue 5/6/14	
15		3.3 Time Management	2 days	Wed 5/7/14	Thu 5/8/14	
16		**4. Testing**	**15 days**	**Fri 5/9/14**	**Thu 5/29/14**	
17		4.1 Develop Test Cases	5 days	Fri 5/9/14	Thu 5/15/14	
18		4.2 Implement Test Cases	10 days	Fri 5/16/14	Thu 5/29/14	
19		**5. Maintainance**	**20 days**	**Fri 5/30/14**	**Thu 6/26/14**	
20		5.1 Project Closure	10 days	Fri 5/30/14	Thu 6/12/14	
21		5.2 Review	10 days	Fri 6/13/14	Thu 6/26/14	

Code

PhoneNumberActivity.java

```
package com.example.one2manychatapp.Activities;



import androidx.appcompat.app.AppCompatActivity;



import android.content.Intent;

import android.os.Bundle;

import android.view.View;



import com.example.one2manychatapp.databinding.ActivityPhoneNumberBinding;

import com.google.firebase.auth.FirebaseAuth;
```

```
public class PhoneNumberActivity extends AppCompatActivity {



    ActivityPhoneNumberBinding binding;

    FirebaseAuth auth;



    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        binding = ActivityPhoneNumberBinding.inflate(getLayoutInflater());

        setContentView(binding.getRoot());

/*

       auth = FirebaseAuth.getInstance();



        if(auth.getCurrentUser() != null)
```

```
    {
        Intent intent = new Intent(PhoneNumberActivity.this, MainActivity.class);
        startActivity(intent);
        finish();
    }
*/
    getSupportActionBar().hide();


    binding.nameBox.requestFocus();


    binding.continueBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Intent intent = new Intent(PhoneNumberActivity.this, OTPActivity.class);
```

```
            intent.putExtra("phoneNumber", binding.nameBox.getText().toString());

            startActivity(intent);

        }

    });

  }

}
```

OTPactivity.java

```
package com.example.one2manychatapp.Activities;



import androidx.annotation.NonNull;

import androidx.appcompat.app.AppCompatActivity;
```

```
import android.app.ProgressDialog;

import android.content.Intent;

import android.os.Bundle;

import android.widget.Toast;



import com.example.one2manychatapp.databinding.ActivityOtpactivityBinding;

import com.google.android.gms.tasks.OnCompleteListener;

import com.google.android.gms.tasks.Task;

import com.google.firebase.FirebaseException;

import com.google.firebase.auth.AuthResult;

import com.google.firebase.auth.FirebaseAuth;

import com.google.firebase.auth.PhoneAuthCredential;

import com.google.firebase.auth.PhoneAuthOptions;

import com.google.firebase.auth.PhoneAuthProvider;
```

```
import com.mukesh.OnOtpCompletionListener;



import java.util.concurrent.TimeUnit;



public class OTPActivity extends AppCompatActivity {



    ActivityOtpactivityBinding binding;

    FirebaseAuth auth;



    String verificationId;



    ProgressDialog dialog;



    @Override
```

```
protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    binding = ActivityOtpactivityBinding.inflate(getLayoutInflater());

    setContentView(binding.getRoot());



    dialog = new ProgressDialog(this);

    dialog.setMessage("Sending OTP.....");

    dialog.setCancelable(false);

    dialog.show();



    auth = FirebaseAuth.getInstance();



    getSupportActionBar().hide();
```

```
        String phoneNumber = getIntent().getStringExtra("phoneNumber");



        binding.phoneLbl.setText("Verify " + phoneNumber);



        PhoneAuthOptions options = PhoneAuthOptions.newBuilder(auth)

                .setPhoneNumber(phoneNumber)

                .setTimeout(60L, TimeUnit.SECONDS)

                .setActivity(OTPActivity.this)

                .setCallbacks(new PhoneAuthProvider.OnVerificationStateChangedCallbacks() {

                    @Override

                    public void onVerificationCompleted(@NonNull PhoneAuthCredential
phoneAuthCredential) {



                    }
```

```
            @Override
            public void onVerificationFailed(@NonNull FirebaseException e) {


            }


            @Override
            public void onCodeSent(@NonNull String verifyId, @NonNull
PhoneAuthProvider.ForceResendingToken forceResendingToken) {
                super.onCodeSent(verifyId, forceResendingToken);
                dialog.dismiss();


                verificationId = verifyId;
            }
```

```
            }).build();



        PhoneAuthProvider.verifyPhoneNumber(options);



        binding.otpView.setOtpCompletionListener(new OnOtpCompletionListener() {

            @Override

            public void onOtpCompleted(String otp) {

                PhoneAuthCredential credential = PhoneAuthProvider.getCredential(verificationId, otp);



                auth.signInWithCredential(credential).addOnCompleteListener(new
OnCompleteListener<AuthResult>()

                {

                    @Override

                    public void onComplete(@NonNull Task<AuthResult> task)
```

```
            {

                if(task.isSuccessful()){

                    Intent intent = new Intent(OTPActivity.this, SetupProfileActivity.class);

                    startActivity(intent);

                    finishAffinity();

                }

                else{

                    Toast.makeText(OTPActivity.this, "Failed", Toast.LENGTH_SHORT).show();

                }

            }

        });

    }

});
```

```
    }

}
```

ChatActivity.java

```
package com.example.one2manychatapp.Activities;



import androidx.annotation.NonNull;

import androidx.appcompat.app.AppCompatActivity;

import androidx.recyclerview.widget.LinearLayoutManager;



import android.os.Bundle;

import android.view.View;
```

```
import com.example.one2manychatapp.Adapters.MessagesAdapter;

import com.example.one2manychatapp.Models.Message;

import com.example.one2manychatapp.databinding.ActivityChatBinding;

import com.google.android.gms.tasks.OnSuccessListener;

import com.google.firebase.auth.FirebaseAuth;

import com.google.firebase.database.DataSnapshot;

import com.google.firebase.database.DatabaseError;

import com.google.firebase.database.FirebaseDatabase;

import com.google.firebase.database.ValueEventListener;



import java.util.ArrayList;

import java.util.Date;



public class ChatActivity extends AppCompatActivity {
```

```
ActivityChatBinding binding;

MessagesAdapter adapter;

ArrayList<Message> messages;



String senderRoom, receiverRoom;



FirebaseDatabase database;



@Override

protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    binding = ActivityChatBinding.inflate(getLayoutInflater());

    setContentView(binding.getRoot());
```

```
messages = new ArrayList<>();



String name = getIntent().getStringExtra("name");

String receiverUid = getIntent().getStringExtra("uid");

String senderUid = FirebaseAuth.getInstance().getUid();



senderRoom = senderUid + receiverUid;

receiverRoom = receiverUid + senderUid;



adapter = new MessagesAdapter(this, messages, senderRoom, receiverRoom);

binding.recyclerView.setLayoutManager(new LinearLayoutManager(this));

binding.recyclerView.setAdapter(adapter);
```

```
database = FirebaseDatabase.getInstance();



database.getReference().child("chats")

        .child(senderRoom)

        .child("messages")

        .addValueEventListener(new ValueEventListener() {

            @Override

            public void onDataChange(@NonNull DataSnapshot snapshot) {

                messages.clear();

                for(DataSnapshot snapshot1 : snapshot.getChildren())

                {

                    Message message = snapshot1.getValue(Message.class);

                    message.setMessageId(snapshot1.getKey());

                    messages.add(message);
```

```
            }



            adapter.notifyDataSetChanged();

        }



        @Override

        public void onCancelled(@NonNull DatabaseError error) {



        }

    });



binding.sendBtn.setOnClickListener(new View.OnClickListener()

{

    @Override
```

```
public void onClick(View v)

{

    String messageTxt = binding.messageBox.getText().toString();



    Date date = new Date();

    Message message = new Message(messageTxt, senderUid, date.getTime());

    binding.messageBox.setText("");



    String randomKey = database.getReference().push().getKey();



    database.getReference().child("chats")

            .child(senderRoom)

            .child("messages")

            .child(randomKey)
```

```
        .setValue(message).addOnSuccessListener(new OnSuccessListener<Void>() {
    @Override
    public void onSuccess(Void aVoid) {
        database.getReference().child("chats")
                .child(receiverRoom)
                .child("messages")
                .child(randomKey)
                .setValue(message).addOnSuccessListener(new OnSuccessListener<Void>() {
            @Override
            public void onSuccess(Void aVoid) {


            }
        });
    }
```

```
            });

        }

    });


    getSupportActionBar().setTitle(name);


    getSupportActionBar().setDisplayHomeAsUpEnabled(true);

}


@Override

public boolean onSupportNavigateUp() {

    finish();

    return super.onSupportNavigateUp();
```

```
    }

}
```

MainActivity.java

```
package com.example.one2manychatapp.Activities;



import androidx.annotation.NonNull;

import androidx.appcompat.app.AppCompatActivity;



import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.widget.Toast;
```

```
import com.example.one2manychatapp.R;

import com.example.one2manychatapp.Models.User;

import com.example.one2manychatapp.Adapters.UsersAdapter;

import com.example.one2manychatapp.databinding.ActivityMainBinding;

import com.google.firebase.database.DataSnapshot;

import com.google.firebase.database.DatabaseError;

import com.google.firebase.database.FirebaseDatabase;

import com.google.firebase.database.ValueEventListener;



import java.util.ArrayList;



public class MainActivity extends AppCompatActivity {
```

```
ActivityMainBinding binding;

FirebaseDatabase database;

ArrayList<User> users;

UsersAdapter usersAdapter;



@Override

protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    binding = ActivityMainBinding.inflate(getLayoutInflater());

    setContentView(binding.getRoot());



    database = FirebaseDatabase.getInstance();

    users = new ArrayList<>();
```

```
    usersAdapter = new UsersAdapter(this, users);
//    binding.recyclerView.setLayoutManager(new LinearLayoutManager(this));
    binding.recyclerView.setAdapter(usersAdapter);


    database.getReference().child("users").addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot snapshot) {
            users.clear();
            for(DataSnapshot snapshot1 : snapshot.getChildren())
            {
                User user = snapshot1.getValue(User.class);
                users.add(user);
            }
            usersAdapter.notifyDataSetChanged();
```

```
        }


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


        }
    });
}


@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    switch (item.getItemId())
    {
        case R.id.search:
```

```
            Toast.makeText(this,"Search Clicked.", Toast.LENGTH_SHORT).show();

            break;

        case R.id.settings:

            Toast.makeText(this, "Settings Clicked.", Toast.LENGTH_SHORT).show();

            break;

        case R.id.invite:

            Toast.makeText(this,"Invite Clicked.", Toast.LENGTH_SHORT).show();

            break;

        case R.id.groups:

            Toast.makeText(this,"Groups Clicked.", Toast.LENGTH_SHORT).show();

            break;

    }

    return super.onOptionsItemSelected(item);

}
```

```
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.topmenu, menu);
        return super.onCreateOptionsMenu(menu);
    }
}
```

SetupProfileActivity.java

```
package com.example.one2manychatapp.Activities;


import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
```

```
import androidx.appcompat.app.AppCompatActivity;



import android.app.ProgressDialog;

import android.content.Intent;

import android.net.Uri;

import android.os.Bundle;

import android.view.View;



import com.example.one2manychatapp.Models.User;

import com.example.one2manychatapp.databinding.ActivitySetupProfileBinding;

import com.google.android.gms.tasks.OnCompleteListener;

import com.google.android.gms.tasks.OnSuccessListener;

import com.google.android.gms.tasks.Task;

import com.google.firebase.auth.FirebaseAuth;
```

```
import com.google.firebase.database.FirebaseDatabase;

import com.google.firebase.storage.FirebaseStorage;

import com.google.firebase.storage.StorageReference;

import com.google.firebase.storage.UploadTask;



public class SetupProfileActivity extends AppCompatActivity {



    ActivitySetupProfileBinding binding;

    FirebaseAuth auth;

    FirebaseDatabase database;

    FirebaseStorage storage;

    Uri selectedImage;

    ProgressDialog dialog;
```

```java
@Override

protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    binding = ActivitySetupProfileBinding.inflate(getLayoutInflater());

    setContentView(binding.getRoot());



    dialog = new ProgressDialog(this);

    dialog.setMessage("Updating profile.....");

    dialog.setCancelable(false);




    database = FirebaseDatabase.getInstance();

    storage = FirebaseStorage.getInstance();

    auth = FirebaseAuth.getInstance();
```

```
getSupportActionBar().hide();



binding.imageView.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Intent intent = new Intent();
        intent.setAction(Intent.ACTION_GET_CONTENT);
        intent.setType("image/*");
        startActivityForResult(intent, 45);
    }
});



binding.continueBtn.setOnClickListener(new View.OnClickListener() {
```

```
    @Override
    public void onClick(View view) {
        String name = binding.nameBox.getText().toString();


        if(name.isEmpty())
        {
            binding.nameBox.setError("Please enter a name");
            return;
        }


        dialog.show();
        if(selectedImage != null)
        {
            StorageReference reference =
```

```
storage.getReference().child("Profiles").child(auth.getUid());

                reference.putFile(selectedImage).addOnCompleteListener(new

OnCompleteListener<UploadTask.TaskSnapshot>() {

                    @Override

                    public void onComplete(@NonNull Task<UploadTask.TaskSnapshot> task) {

                        if(task.isSuccessful()){

                            reference.getDownloadUrl().addOnSuccessListener(new

OnSuccessListener<Uri>() {

                                @Override

                                public void onSuccess(Uri uri) {

                                    String imageUrl = uri.toString();



                                    String uid = auth.getUid();

                                    String phone = auth.getCurrentUser().getPhoneNumber();
```

```
                        String name = binding.nameBox.getText().toString();



                        User user = new User(uid, name, phone, imageUrl);

                        database.getReference()

                                .child("users")

                                .child(uid)

                                .setValue(user)

                                .addOnSuccessListener(new OnSuccessListener<Void>() {

                                    @Override

                                    public void onSuccess(Void unused) {

                                        dialog.dismiss();

                                        Intent intent = new Intent(SetupProfileActivity.this,
MainActivity.class);

                                        startActivity(intent);
```

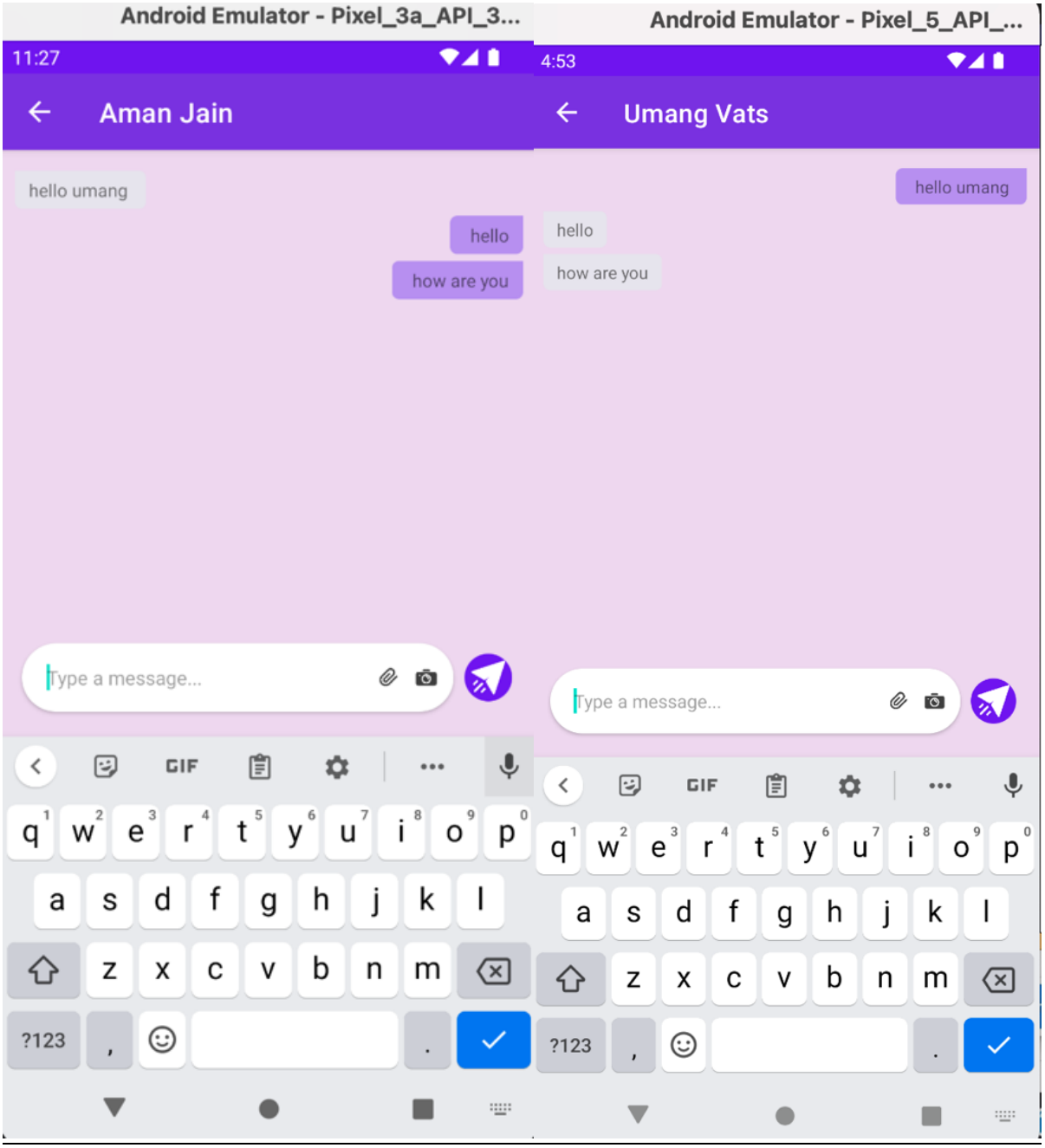
```
                                finish();

                            }

                        });

                    }

                });

            }

        }

    });

}

else

{



    String uid = auth.getUid();

    String phone = auth.getCurrentUser().getPhoneNumber();
```

```
User user = new User(uid, name, phone, "No Imagge");

database.getReference()

        .child("users")

        .child(uid)

        .setValue(user)

        .addOnSuccessListener(new OnSuccessListener<Void>() {

            @Override

            public void onSuccess(Void unused) {

                dialog.dismiss();

                Intent intent = new Intent(SetupProfileActivity.this, MainActivity.class);

                startActivity(intent);

                finish();

            }
```

```
                    });

            }

        }

    });

}



@Override

protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {

    super.onActivityResult(requestCode, resultCode, data);



    if(data != null)

    {

        if(data.getData() != null)

        {
```

```
            binding.imageView.setImageURI(data.getData());


            selectedImage = data.getData();


        }

    }

  }

}
```

Project Screenshots


Android Emulator - Pixel_3a_API_3...
11:27
Aman Jain
hello umang
hello
how are you
Type a message...
Android Emulator - Pixel_5_API_...
4:53
Umang Vats
hello umang
hello
how are you
Type a message...

Conclusion

We Developed application in Java by using Android Studio.

This software is portable, efficient, and easily maintainable for large number of users. Our developed Android-based Messaging Chat Application is unique in its features and more importantly easily customizable. The java packages provides a powerful and flexible set of classes for implementing applications.

Typically, programs running on client machines make requests to programs on a server Machine. These involve networking services provided by the transport layer. The most widely used transport protocols on the Internet are TCP (Transmission control Protocol) and UDP (User Datagram Protocol).

TCP is a connection-oriented protocol providing a reliable flow of data between two computers. On the other hand, UDP is a simpler messagebased connectionless protocol which sends packets of data known as datagrams from one computer to another with no guarantees of arrival.

Bibliography Internet reference

- https://www.geeksforgeeks.org/android-tutorial/
- https://www.youtube.com/
- https://www.javatpoint.co m/android-tutorial
- https://stackoverflow.com/