**PROJECT REPORT**

**CHAT WITH BUDDY**

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD OF THE DEGREE OF

**BACHELOR OF TECHNOLOGY**

(Computer Science & Engineering)

**SUBMITTED BY:**

Aqir Ahmad Lone(7318107)

FEBUARY 2022



**Department of Computer Science & Engineering**

**Galaxy Global Educational Trust’s Group of Institutions**

**DECLARATION**

I hereby declare that the project Report entitled ("CHAT WITH BUDDY") is an authentic record of my own work as requirements of project during the period from October to January for the award of degree of B.Tech. (Computer Science & Engineering), GALAXY GLOBAL GROUP OF INSTITUTIONS, DINARPUR.

**(Aqir Ahmad Lone)**

**( 7318107)**

**Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**ACKNOWLEDGEMENT**

I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

I express my sincere thanks to **Dr. Saurabh Gupta**, Director, Galaxy Global Educational Trust’s Group of Institutions, Dinarpur , Ambala and other staff members for their support and encouragement.

I would like to express my sincere gratitude to my supervisor Er. Tarun Kumar, Head, Department of Computer Science & Engineering, Galaxy Global Educational Trust’s Group of Institutions, Dinarpur, Ambala for his constant support, inspiration and guidance in both academic and personal life. I am extremely grateful to him/her for being an excellent advisor and a wonderful teacher.

Last but not the least; I would like to express my gratitude towards my parents for their kind co-operation and encouragement which help me in completion of this project.

|  |  |  |
| --- | --- | --- |
| **S. NO.** | **TOPIC** | **PAGE NO.** |
| **1** | **Introduction to the project** | **6** |
| **2** | **Working of project** | **6-12** |
| **3** | **Technologies Used** | **13** |
| **4** | **Android Studio** | **14** |
| **5** | **Java** | **15-16** |
| **6** | **XML** | **17** |
| **7** | **Firebase** | **18-21** |
| **8** | **Project Structure** | **21 - 23** |
| **9** | **Server side** | **24-33** |
| **10** | **Client side** | **34 - 42** |
| **11** | **Conclusion** | **43** |
| **12** | **Future Vision** | **44** |

**List of figures**

|  |  |  |  |
| --- | --- | --- | --- |
| **S. NO.** | **FIGURE NO.** | **DESCRIPTION** | **PAGE NO.** |
| 1 | Fig 1 | Splash Screen | 6 |
| 2 | Fig 2 | Phone Number Input | 7 |
| 3 | Fig 3 | OTP Verification | 8 |
| 4 | Fig 4 | Profile Details | 9 |
| 5 | Fig 5 | Update Details | 10 |
| 6 | Fig 6 | Users | 11 |
| 7 | Fig 7 | Chat | 12 |
| 8 | Fig 8 | Project Files | 19 |

**Introduction to Project**

**Talk With Buddy** is a real time chat application can be used for chatting with friends and it will automatically delete the chatting once user refreshes his/her browser. A Application chat is a system that allows users to communicate in real-time using easily accessible Applicatin interfaces. It is a type of Internet online chat distinguished by its simplicity and accessibility to. This trait allows users instantaneous. Users will always get the latest version of a chat service

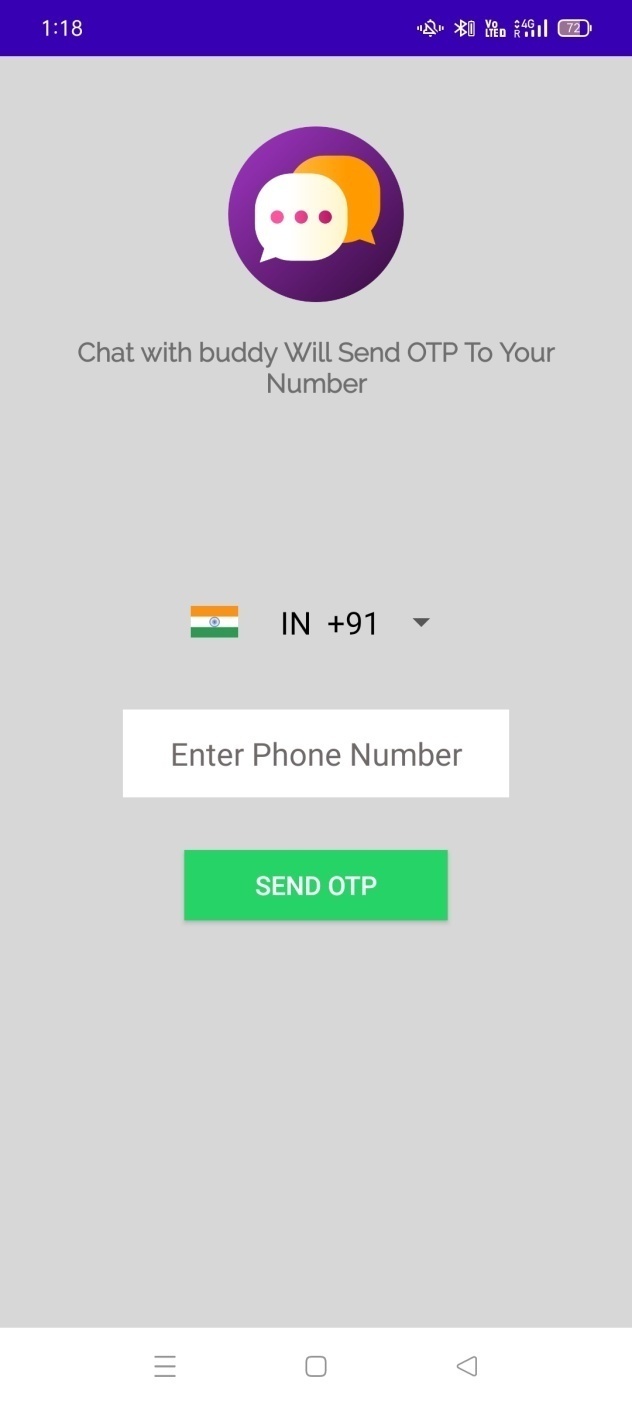
**The working of the project is explained below step by step.**

* **Step 1:** User Installs the app first and then spalash screen appears:

****

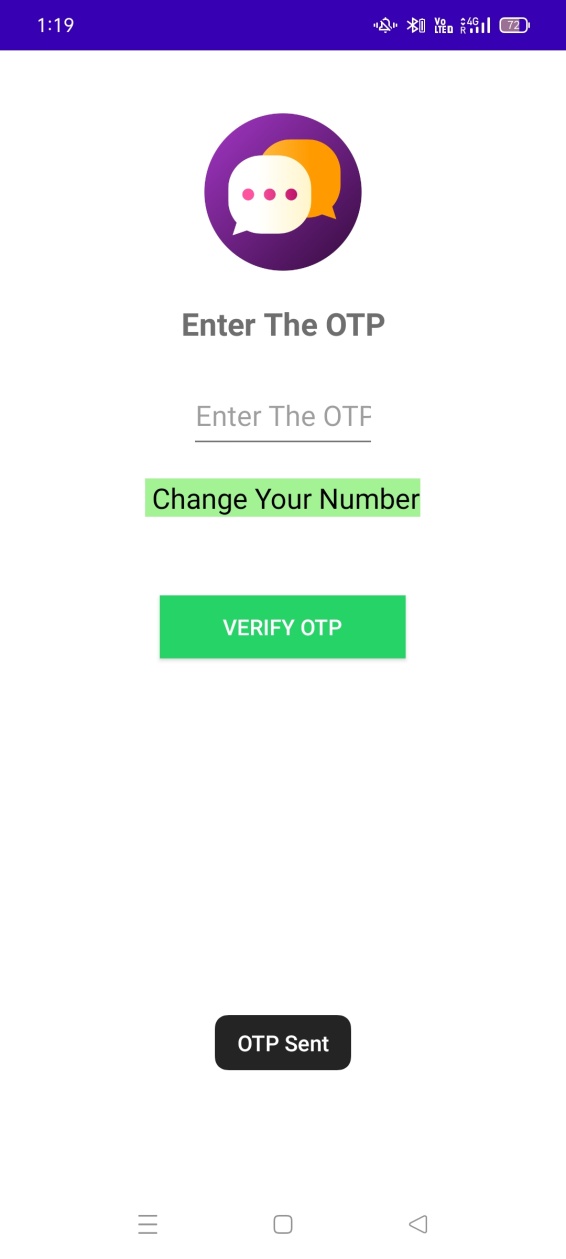
**Fig 1**

* **Step 2:** User Enters the mobile number for OTP verification:

****

**Fig 2**

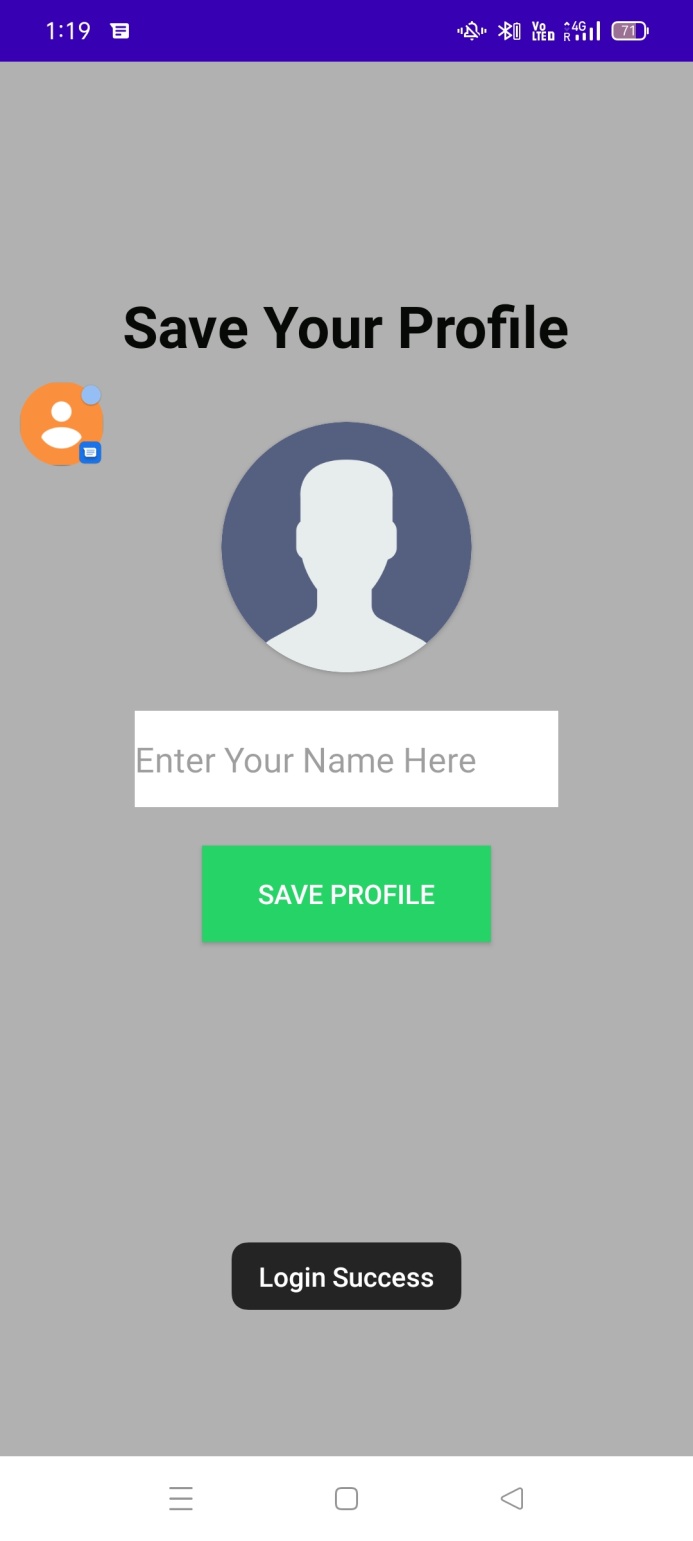
* **Step 3:** User Enters OTP for Verification.

****

**Fig 3**

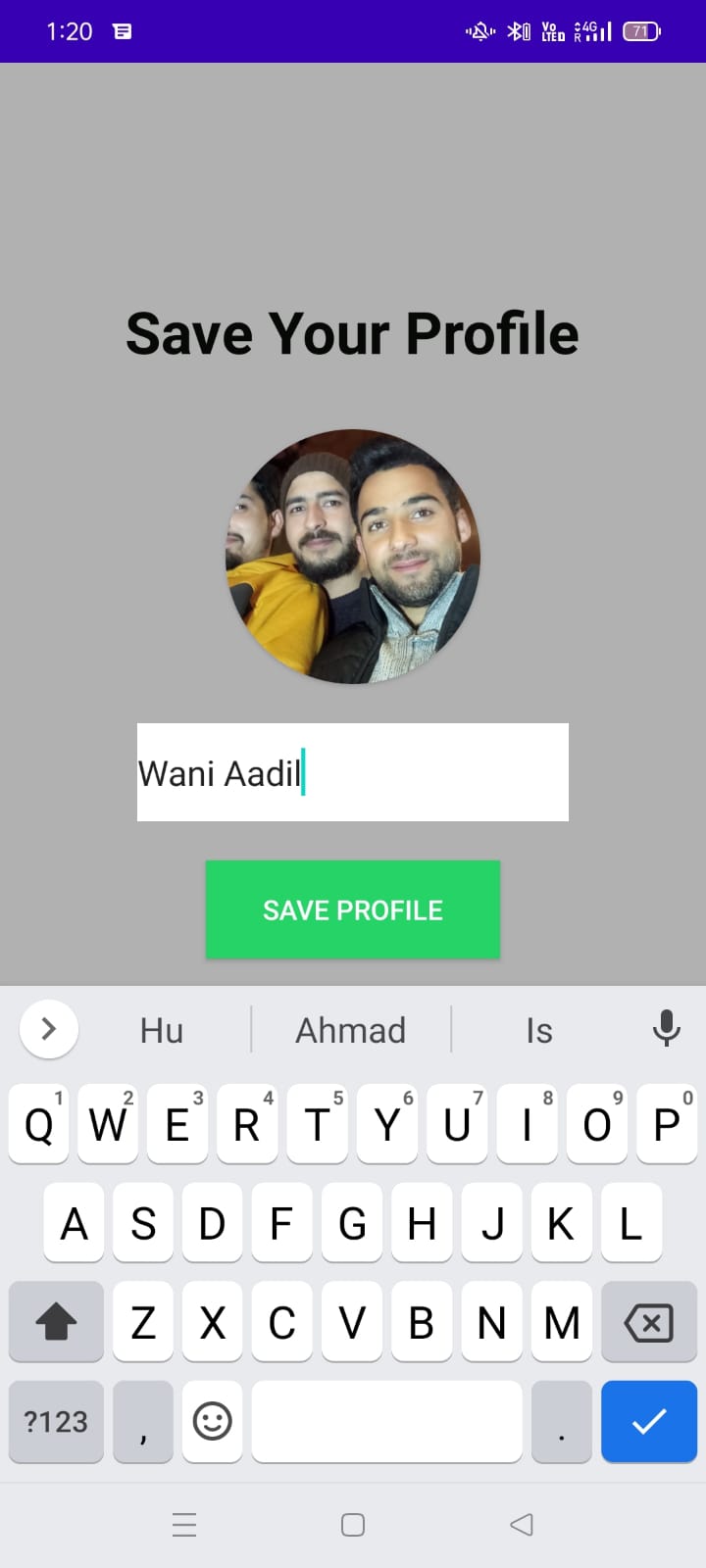
.

* **Step 4:** User updates profile.

****

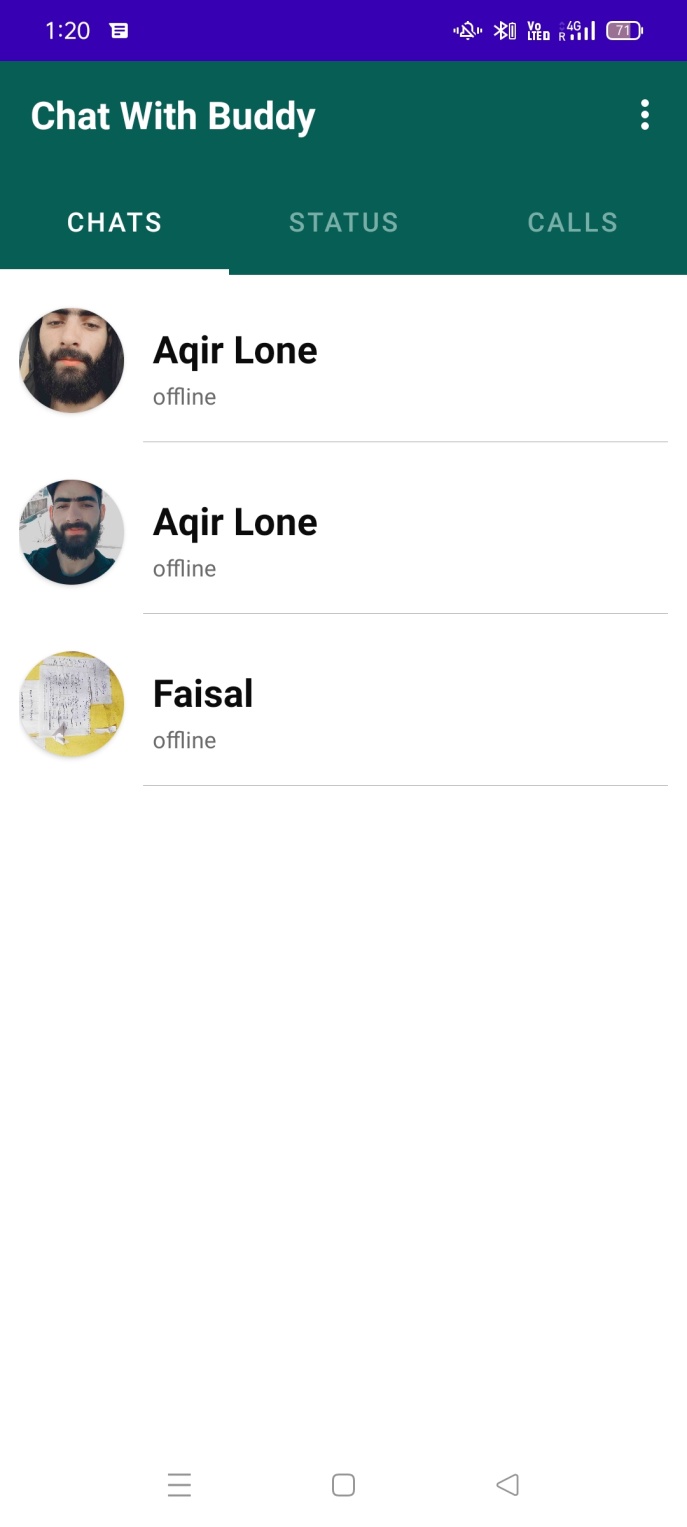
**Fig 4**

* **Step 6:** User fills the details and updates profile picture.



**Fig 5**

* **Step 6:** then the user can can easily select any other user for chatting **.**

****

**Fig 6**

**Step 7: Chat :**

****

**Fig 7**

**Tools and Technologies Used**

* **IDE:-** Android Studio.
* **Language :-** Java , XML.
* **Database:**-Firebase Database.

**Android Studio.**

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA. On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps, such as:

* A flexible Gradle-based build system
* A fast and feature-rich emulator
* A unified environment where you can develop for all Android devices
* Apply Changes to push code and resource changes to your running app without restarting your app
* Code templates and GitHub integration to help you build common app features and import sample code
* Extensive testing tools and frameworks
* Lint tools to catch performance, usability, version compatibility, and other problems
* C++ and NDK support
* Built-in support for [Google Cloud Platform](https://cloud.google.com/tools/android-studio/docs/), making it easy to integrate Google Cloud Messaging and App Engine

**Java**

JAVA was developed by James Gosling at Sun Microsystems Inc in the year 1991, later acquired by Oracle Corporation. It is a simple programming language. Java makes writing, compiling, and debugging programming easy. It helps to create reusable code and modular programs.

[Java](https://www.geeksforgeeks.org/java/) is a class-based, object-oriented programming language and is designed to have as few implementation dependencies as possible. A general-purpose programming language made for developers to write once run anywhere that is compiled Java code can run on all platforms that support Java. Java applications are compiled to byte code that can run on any Java Virtual Machine. The syntax of Java is similar to c/c++.

History

Java’s history is very interesting. It is a programming language created in 1991. James Gosling, Mike Sheridan, and Patrick Naughton, a team of Sun engineers known as the Green team initiated the Java language in 1991. Sun Microsystems released its first public implementation in 1996 as Java 1.0. It provides no-cost -run-times on popular platforms. Java1.0 compiler was re-written in Java by Arthur Van Hoff to strictly comply with its specifications. With the arrival of Java 2, new versions had multiple configurations built for different types of platforms.

In 1997, Sun Microsystems approached the ISO standards body and later formalized Java, but it soon withdrew from the process. At one time, Sun made most of its Java implementations available without charge, despite their proprietary software status. Sun generated revenue from Java through the selling of licenses for specialized products such as the Java Enterprise System.

On November 13, 2006, Sun released much of its Java virtual machine as free, open-source software. On May 8, 2007, Sun finished the process, making all of its JVM’s core code available under open-source distribution terms.

The principles for creating java were simple, robust, secured, high performance, portable, multi-threaded, interpreted, dynamic, etc. James Gosling in 1995 developed Java, who is known as the Father of Java. Currently, Java is used in mobile devices, internet programming, games, e-business, etc.

Java programming language is named JAVA. Why?

After the name OAK, the team decided to give a new name to it and the suggested words were Silk, Jolt, revolutionary, DNA, dynamic, etc. These all names were easy to spell and fun to say, but they all wanted the name to reflect the essence of technology. In accordance with James Gosling, Java the among the top names along with Silk, and since java was a unique name so most of them preferred it.

Java is the name of an island in Indonesia where the first coffee(named java coffee) was produced. And this name was chosen by James Gosling while having coffee near his office. Note that Java is just a name, not an acronym.

Java Terminology

Before learning Java, one must be familiar with these common terms of Java.

1.  Java Virtual Machine(JVM):  This is generally referred to as [JVM](https://www.geeksforgeeks.org/jvm-works-jvm-architecture/#:~:text=JVM(Java%20Virtual%20Machine)%20acts,(Write%20Once%20Run%20Anywhere).). There are three execution phases of a program. They are written, compile and run the program.

* Writing a program is done by a java programmer like you and me.
* The compilation is done by the JAVAC compiler which is a primary Java compiler included in the Java development kit (JDK). It takes Java program as input and generates bytecode as output.
* In the Running phase of a program, JVM executes the bytecode generated by the compiler.

Now, we understood that the function of Java Virtual Machine is to execute the bytecode produced by the compiler. Every Operating System has a different JVM but the output they produce after the execution of bytecode is the same across all the operating systems. This is why Java is known as a platform-independent language.

2. Bytecode in the Development process:  As discussed, the Javac compiler of JDK compiles the java source code into bytecode so that it can be executed by JVM. It is saved as .class file by the compiler. To view the bytecode, a disassembler like [javap](https://www.geeksforgeeks.org/javap-tool-in-java-with-examples/) can be used.

3. Java Development Kit(JDK): While we were using the term JDK, when we learn about bytecode and JVM . So, as the name suggests, it is a complete Java development kit that includes everything including compiler, Java Runtime Environment (JRE), java debuggers, java docs, etc. For the program to execute in java, we need to install JDK on our computer in order to create, compile and run the java program.

4. Java Runtime Environment (JRE): JDK includes JRE. JRE installation on our computers allows the java program to run, however, we cannot compile it. JRE includes a browser, JVM, applet supports, and plugins. For running the java program, a computer needs JRE.

5. Garbage Collector: In Java, programmers can’t delete the objects. To delete or recollect that memory JVM has a program called [Garbage Collector](https://www.geeksforgeeks.org/garbage-collection-java/). Garbage Collectors can recollect the of objects that are not referenced. So Java makes the life of a programmer easy by handling

**XML**

XML stands for Extensible Markup Language. It is a text-based markup language derived from Standard Generalized Markup Language (SGML).

XML tags identify the data and are used to store and organize the data, rather than specifying how to display it like HTML tags, which are used to display the data. XML is not going to replace HTML in the near future, but it introduces new possibilities by adopting many successful features of HTML.

There are three important characteristics of XML that make it useful in a variety of systems and solutions −

* XML is extensible − XML allows you to create your own self-descriptive tags, or language, that suits your application.
* XML carries the data, does not present it − XML allows you to store the data irrespective of how it will be presented.
* XML is a public standard − XML was developed by an organization called the World Wide Web Consortium (W3C) and is available as an open standard.

XML Usage

A short list of XML usage says it all −

* XML can work behind the scene to simplify the creation of HTML documents for large web sites.
* XML can be used to exchange the information between organizations and systems.
* XML can be used for offloading and reloading of databases.
* XML can be used to store and arrange the data, which can customize your data handling needs.
* XML can easily be merged with style sheets to create almost any desired output.
* Virtually, any type of data can be expressed as an XML document.

What is Markup?

XML is a markup language that defines set of rules for encoding documents in a format that is both human-readable and machine-readable. So *what exactly is a markup language?* Markup is information added to a document that enhances its meaning in certain ways, in that it identifies the parts and how they relate to each other. More specifically, a markup language is a set of symbols that can be placed in the text of a document to demarcate and label the parts of that document.

**Firebase**

Firebase is a Backend-as-a-Service (Baas). It provides developers with a variety of tools and services to help them develop quality apps, grow their user base, and earn profit. It is built on Google’s infrastructure.

Firebase is categorized as a [NoSQL](https://www.educative.io/edpresso/whats-the-difference-betweensql-and-nosql) database program, which stores data in JSON-like documents.



In Firebase, a document is a set of key-value pairs defined by a schema. A group of documents makes up a collection.

Key Features

1. Authentication

It supports authentication using passwords, phone numbers, Google, Facebook, Twitter, and more. The Firebase Authentication (SDK) can be used to manually integrate one or more sign-in methods into an app.

2. Realtime database

Data is synced across all clients in realtime and remains available even when an app goes offline.

3. Hosting

Firebase Hosting provides fast hosting for a web app; content is cached into content delivery networks worldwide.

4. Test lab

The application is tested on virtual and physical devices located in Google’s data centers.

5. Notifications

Notifications can be sent with firebase with no additional coding.

**Project structure**

****

**Fig 8**

Figure 1. The project files in Android view.

Each project in Android Studio contains one or more modules with source code files and resource files. Types of modules include:

* Android app modules
* Library modules
* Google App Engine modules

By default, Android Studio displays your project files in the Android project view, as shown in figure 1. This view is organized by modules to provide quick access to your project's key source files.

All the build files are visible at the top level under Gradle Scripts and each app module contains the following folders:

* manifests: Contains the AndroidManifest.xml file.
* java: Contains the Java source code files, including JUnit test code.
* res: Contains all non-code resources, such as XML layouts, UI strings, and bitmap images.

The Android project structure on disk differs from this flattened representation. To see the actual file structure of the project, select Project from the Project dropdown (in figure 1, it's showing as Android).

You can also customize the view of the project files to focus on specific aspects of your app development. For example, selecting the Problems view of your project displays links to the source files containing any recognized coding and syntax errors, such as a missing XML element closing tag in a layout file.



**Fig 8**

 The project files in Problems view, showing a layout file with a problem.

For more information, see [Projects overview](https://developer.android.com/studio/projects).

## The user interface

The Android Studio main window is made up of several logical areas identified in figure 3.



**Fig 9**

The Android Studio main window.

1. The **toolbar** lets you carry out a wide range of actions, including running your app and launching Android tools.
2. The **navigation bar** helps you navigate through your project and open files for editing. It provides a more compact view of the structure visible in the **Project** window.
3. The **editor window** is where you create and modify code. Depending on the current file type, the editor can change. For example, when viewing a layout file, the editor displays the Layout Editor.
4. The **tool window bar** runs around the outside of the IDE window and contains the buttons that allow you to expand or collapse individual tool windows.
5. The **tool windows** give you access to specific tasks like project management, search, version control, and more. You can expand them and collapse them.

**Client Side**

**ChatActivity.java**

*package* com.aqirlone.talkwithbuddies;  
  
*import* androidx.annotation.NonNull;  
*import* androidx.appcompat.app.AppCompatActivity;  
*import* androidx.core.content.ContextCompat;  
*import* androidx.viewpager.widget.ViewPager;  
  
*import* android.content.Intent;  
*import* android.graphics.drawable.Drawable;  
*import* android.os.Bundle;  
*import* android.view.*Menu*;  
*import* android.view.MenuInflater;  
*import* android.view.*MenuItem*;  
*import* android.widget.Toast;  
  
*import* com.google.android.gms.tasks.*OnSuccessListener*;  
*import* com.google.android.material.tabs.TabItem;  
*import* com.google.android.material.tabs.TabLayout;  
*import* com.google.firebase.auth.FirebaseAuth;  
*import* com.google.firebase.database.FirebaseDatabase;  
*import* com.google.firebase.firestore.DocumentReference;  
*import* com.google.firebase.firestore.FirebaseFirestore;  
*import* com.google.firebase.storage.FirebaseStorage;  
  
*public class* chatActivity *extends* AppCompatActivity {  
  
 TabLayout tabLayout;  
 TabItem mchat,mcall,mstatus;  
 ViewPager viewPager;  
 PagerAdapter pagerAdapter;  
 androidx.appcompat.widget.Toolbar mtoolbar;  
  
 FirebaseAuth firebaseAuth;  
 FirebaseFirestore firebaseFirestore;  
  
 @Override  
 *protected void* onCreate(Bundle savedInstanceState) {  
 *super*.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_chat*);  
  
 tabLayout=findViewById(R.id.*include*);  
 mchat=findViewById(R.id.*chats*);  
 mcall=findViewById(R.id.*Calls*);  
 mstatus=findViewById(R.id.*Status*);  
 mtoolbar=findViewById(R.id.*toolbar*);  
 viewPager=findViewById(R.id.*fragmentcontainer*);  
 setSupportActionBar(mtoolbar);  
 firebaseAuth=FirebaseAuth.*getInstance*();  
 firebaseFirestore=FirebaseFirestore.*getInstance*();  
  
 Drawable drawable= ContextCompat.*getDrawable*(getApplicationContext(),R.drawable.*ic\_baseline\_more\_vert\_24*);  
 mtoolbar.setOverflowIcon(drawable);  
 pagerAdapter=*new* PagerAdapter(getSupportFragmentManager(),tabLayout.getTabCount());  
 viewPager.setAdapter(pagerAdapter);  
  
 tabLayout.setOnTabSelectedListener(*new* TabLayout.OnTabSelectedListener() {  
 @Override  
 *public void* onTabSelected(TabLayout.Tab tab) {  
 viewPager.setCurrentItem(tab.getPosition());  
 *if* (tab.getPosition()==0||tab.getPosition()==1||tab.getPosition()==2){  
 pagerAdapter.notifyDataSetChanged();  
 }  
  
 }  
  
 @Override  
 *public void* onTabUnselected(TabLayout.Tab tab) {  
  
 }  
  
 @Override  
 *public void* onTabReselected(TabLayout.Tab tab) {  
  
 }  
 });  
 viewPager.addOnPageChangeListener(*new* TabLayout.TabLayoutOnPageChangeListener(tabLayout));  
  
 }  
  
 @Override  
 *public boolean* onOptionsItemSelected(@NonNull *MenuItem* item) {  
  
 *switch*(item.getItemId()) {  
 *case* R.id.*profile*:  
 Intent intent = *new* Intent(chatActivity.*this*, ProfileActivity.*class*);  
 startActivity(intent);  
 *break*;  
  
 *case* R.id.*Settings*:  
 Toast.*makeText*(getApplicationContext(), "Settings", Toast.*LENGTH\_SHORT*).show();  
 *break*;  
 }  
 *return true*;  
 }  
  
 @Override  
 *public boolean* onCreateOptionsMenu(*Menu* menu) {  
  
 MenuInflater menuInflater=getMenuInflater();  
 menuInflater.inflate(R.menu.*menu*,menu);  
 *return true*;  
 }  
 @Override  
 *protected void* onStop() {  
  
 *super*.onStop();  
 DocumentReference documentReference=firebaseFirestore.collection("Users").document(firebaseAuth.getUid());  
 documentReference.update("status","offline").addOnSuccessListener(*new* OnSuccessListener<Void>() {  
 @Override  
 *public void* onSuccess(Void unused) {  
 *//user is now offline* }  
 });  
 }  
  
 @Override  
 *protected void* onStart() {  
 *super*.onStart();  
 DocumentReference documentReference=firebaseFirestore.collection("Users").document(firebaseAuth.getUid());  
 documentReference.update("status","Online").addOnSuccessListener(*new* OnSuccessListener<Void>() {  
 @Override  
 *public void* onSuccess(Void unused) {  
 *//user is now offline* }  
 });  
  
 }  
}

**SetProfile.java**

*package* com.aqirlone.talkwithbuddies;  
  
*import* androidx.annotation.NonNull;  
*import* androidx.annotation.Nullable;  
*import* androidx.appcompat.app.AppCompatActivity;  
  
*import* android.content.Intent;  
*import* android.graphics.Bitmap;  
  
*import* android.net.Uri;  
*import* android.os.Bundle;  
*import* android.provider.MediaStore;  
*import* android.util.Log;  
*import* android.view.View;  
*import* android.widget.Toast;  
  
*import* com.aqirlone.talkwithbuddies.databinding.ActivitySetProfileBinding;  
*import* com.google.android.gms.tasks.*OnFailureListener*;  
*import* com.google.android.gms.tasks.*OnSuccessListener*;  
*import* com.google.firebase.auth.FirebaseAuth;  
*import* com.google.firebase.database.DatabaseReference;  
*import* com.google.firebase.database.FirebaseDatabase;  
*import* com.google.firebase.firestore.DocumentReference;  
*import* com.google.firebase.firestore.FirebaseFirestore;  
*import* com.google.firebase.storage.FirebaseStorage;  
*import* com.google.firebase.storage.StorageReference;  
*import* com.google.firebase.storage.UploadTask;  
  
*import* java.io.ByteArrayOutputStream;  
  
*import* java.io.IOException;  
*import* java.util.HashMap;  
*import* java.util.*Map*;  
  
*public class* setProfile *extends* AppCompatActivity {  
  
 ActivitySetProfileBinding binding;  
 *private final static int PICK\_IMAGE*=123;  
 *private* Uri imagepath;  
 *private* FirebaseAuth firebaseAuth;  
 *private* String name;  
 *private* FirebaseStorage firebaseStorage;  
 *private* StorageReference storageReference;  
 *private* String ImageUriAccessToken;  
 *private* FirebaseFirestore firebaseFirestore;  
  
 @Override  
 *protected void* onCreate(Bundle savedInstanceState) {  
 *super*.onCreate(savedInstanceState);  
 binding=ActivitySetProfileBinding.*inflate*(getLayoutInflater());  
 setContentView(binding.getRoot());  
 firebaseAuth=FirebaseAuth.*getInstance*();  
 firebaseStorage=FirebaseStorage.*getInstance*();  
 storageReference=firebaseStorage.getReference();  
 firebaseFirestore=FirebaseFirestore.*getInstance*();  
  
  
 binding.getuserimage.setOnClickListener(*new* View.OnClickListener() {  
 @Override  
 *public void* onClick(View view) {  
 Intent intent=*new* Intent(Intent.*ACTION\_PICK*, MediaStore.Images.Media.*INTERNAL\_CONTENT\_URI*);  
 startActivityForResult(intent,*PICK\_IMAGE*);  
 }  
 });  
 binding.saveprofilebtn.setOnClickListener(*new* View.OnClickListener() {  
 @Override  
 *public void* onClick(View view) {  
 name=binding.getusername.getText().toString();  
 *if*(name.isEmpty()){  
 Toast.*makeText*(getApplicationContext(), "Name Is Empty", Toast.*LENGTH\_SHORT*).show();  
 }*else if*(imagepath==*null*){  
 Toast.*makeText*(getApplicationContext(), "Image is empty", Toast.*LENGTH\_SHORT*).show();  
 }*else*{  
  
 binding.progressbarofsetprofile.setVisibility(View.*VISIBLE*);  
 sendDataForNewUser();  
 binding.progressbarofsetprofile.setVisibility(View.*INVISIBLE*);  
 Intent intent=*new* Intent(setProfile.*this*,chatActivity.*class*);  
 startActivity(intent);  
 finish();  
  
 }  
 }  
 });  
  
  
 }  
  
  
 @Override  
 *protected void* onActivityResult(*int* requestCode, *int* resultCode, @Nullable Intent data) {  
  
 *if*(requestCode==*PICK\_IMAGE* && resultCode==*RESULT\_OK*){  
  
 imagepath=data.getData(); *////looooooooooooooooooooooooook heeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeerrreeeeeeeeee* binding.getuserimageinimageview.setImageURI(imagepath);  
 }  
 *super*.onActivityResult(requestCode, resultCode, data);  
 }  
  
 *private void* sendDataForNewUser(){  
 sendDataToRealTimeDatabase();  
 }  
  
  
  
 *private void* sendDataToRealTimeDatabase(){  
 name=binding.getusername.getText().toString();  
 FirebaseDatabase firebaseDatabase=FirebaseDatabase.*getInstance*();  
 DatabaseReference databaseReference=  
 firebaseDatabase.getReference(firebaseAuth.getUid());  
 UserProfileModel userProfileModel=*new* UserProfileModel(name,firebaseAuth.getUid());  
 databaseReference.setValue(userProfileModel);  
 *// Toast.makeText(getApplicationContext(), "User Profile Added Successfully", Toast.LENGTH\_SHORT).show();* sendImageToStorage();  
  
 }  
 *private void* sendImageToStorage(){  
  
 StorageReference imagrref=storageReference.child("Images").child(firebaseAuth.getUid()).child("profile Pic");  
 *//Image Compression Code* Bitmap bitmap=*null*;  
 *try* {  
 bitmap = MediaStore.Images.Media.*getBitmap*(getContentResolver(), imagepath);  
 }*catch* (IOException e){  
  
 e.printStackTrace();  
 }  
 ByteArrayOutputStream byteArrayOutputStream=*new* ByteArrayOutputStream();  
 bitmap.compress(Bitmap.CompressFormat.*JPEG*,25,byteArrayOutputStream);  
 *byte*[] data=byteArrayOutputStream.toByteArray();  
  
 *// putting image to storage* UploadTask uploadTask=imagrref.putBytes(data);  
  
uploadTask.addOnSuccessListener(*new* OnSuccessListener<UploadTask.TaskSnapshot>() {  
  
 @Override  
 *public void* onSuccess(UploadTask.TaskSnapshot taskSnapshot) {  
 imagrref.getDownloadUrl().addOnSuccessListener(*new* OnSuccessListener<Uri>() {  
 @Override  
 *public void* onSuccess(Uri uri) {  
 ImageUriAccessToken=uri.toString();  
 *// Toast.makeText(getApplicationContext(), "uri get successful", Toast.LENGTH\_SHORT).show();* sendDataToCloudFireStore();  
  
 }  
 }).addOnFailureListener(*new* OnFailureListener() {  
 @Override  
 *public void* onFailure(@NonNull Exception e) {  
 Toast.*makeText*(getApplicationContext(), "uri get Failed", Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
 Toast.*makeText*(getApplicationContext(), "Image Uploaded Successfully", Toast.*LENGTH\_SHORT*).show();  
 }  
}).addOnFailureListener(*new* OnFailureListener() {  
 @Override  
 *public void* onFailure(@NonNull Exception e)  
 {  
 Toast.*makeText*(getApplicationContext(), "Image not uploaded", Toast.*LENGTH\_SHORT*).show();  
 }  
});  
  
  
  
  
 }  
  
 *private void* sendDataToCloudFireStore() {  
  
 DocumentReference documentReference=firebaseFirestore.collection("Users").document(firebaseAuth.getUid());  
 *Map*<String ,Object> userData=*new* HashMap<>();  
 userData.put("name",name);  
 userData.put("image",ImageUriAccessToken);  
 userData.put("uid",firebaseAuth.getUid());  
 userData.put("status","Online");  
  
 documentReference.set(userData).addOnSuccessListener(*new* OnSuccessListener<Void>() {  
 @Override  
 *public void* onSuccess(Void unused) {  
 *// Toast.makeText(getApplicationContext(), "Data On Cloud FireStorage Send Successfully ", Toast.LENGTH\_SHORT).show();* }  
 });  
  
  
 }  
}

**SpecficChat.java**

*package* com.aqirlone.talkwithbuddies;  
  
*import* androidx.appcompat.app.AppCompatActivity;  
  
*import* android.os.Bundle;  
*import* androidx.annotation.NonNull;  
*import* androidx.appcompat.app.AppCompatActivity;  
*import* androidx.cardview.widget.CardView;  
*import* androidx.recyclerview.widget.LinearLayoutManager;  
*import* androidx.recyclerview.widget.RecyclerView;  
  
*import* android.content.Intent;  
*import* android.content.pm.PackageInfo;  
*import* android.os.Bundle;  
*import* android.view.View;  
*import* android.widget.EditText;  
*import* android.widget.ImageButton;  
*import* android.widget.ImageView;  
*import* android.widget.TextView;  
*import* android.widget.Toast;  
  
*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.DataSnapshot;  
*import* com.google.firebase.database.DatabaseError;  
*import* com.google.firebase.database.DatabaseReference;  
*import* com.google.firebase.database.FirebaseDatabase;  
*import* com.google.firebase.database.*ValueEventListener*;  
*import* com.google.firebase.firestore.DocumentReference;  
*import* com.google.firebase.firestore.FirebaseFirestore;  
*import* com.google.firebase.storage.FirebaseStorage;  
*import* com.squareup.picasso.Picasso;  
  
*import* java.text.SimpleDateFormat;  
*import* java.util.ArrayList;  
*import* java.util.Calendar;  
*import* java.util.Date;  
  
  
*public class* specificchat *extends* AppCompatActivity {  
  
 EditText mgetmessage;  
 ImageButton msendmessagebutton;  
 FirebaseFirestore firebaseFirestore;  
  
 CardView msendmessagecardview;  
 androidx.appcompat.widget.Toolbar mtoolbarofspecificchat;  
 ImageView mimageviewofspecificuser;  
 TextView mnameofspecificuser;  
  
 *private* String enteredmessage;  
 Intent intent;  
 String mrecievername,sendername,mrecieveruid,msenderuid;  
 *private* FirebaseAuth firebaseAuth;  
 FirebaseDatabase firebaseDatabase;  
 String senderroom,recieverroom;  
  
 ImageButton mbackbuttonofspecificchat;  
  
 RecyclerView mmessagerecyclerview;  
  
 String currenttime;  
 Calendar calendar;  
 SimpleDateFormat simpleDateFormat;  
  
 MessagesAdapter messagesAdapter;  
 ArrayList<Messages> messagesArrayList;  
  
  
  
 @Override  
 *protected void* onCreate(Bundle savedInstanceState) {  
 *super*.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_specificchat*);  
  
 mgetmessage=findViewById(R.id.*getmessage*);  
 msendmessagecardview=findViewById(R.id.*carviewofsendmessage*);  
 msendmessagebutton=findViewById(R.id.*imageviewsendmessage*);  
 mtoolbarofspecificchat=findViewById(R.id.*toolbarofspecificchat*);  
 mnameofspecificuser=findViewById(R.id.*Nameofspecificuser*);  
 mimageviewofspecificuser=findViewById(R.id.*specificuserimageinimageview*);  
 mbackbuttonofspecificchat=findViewById(R.id.*backbuttonofspecificchat*);  
  
 messagesArrayList=*new* ArrayList<>();  
 mmessagerecyclerview=findViewById(R.id.*recyclerviewofspecific*);  
  
 LinearLayoutManager linearLayoutManager=*new* LinearLayoutManager(*this*);  
 linearLayoutManager.setStackFromEnd(*true*);  
 mmessagerecyclerview.setLayoutManager(linearLayoutManager);  
 messagesAdapter=*new* MessagesAdapter(specificchat.*this*,messagesArrayList);  
 mmessagerecyclerview.setAdapter(messagesAdapter);  
 intent=getIntent();  
  
 setSupportActionBar(mtoolbarofspecificchat);  
 mtoolbarofspecificchat.setOnClickListener(*new* View.OnClickListener() {  
 @Override  
 *public void* onClick(View view) {  
 Toast.*makeText*(getApplicationContext(),"Toolbar is Clicked",Toast.*LENGTH\_SHORT*).show();  
  
  
 }  
 });  
  
 firebaseAuth=FirebaseAuth.*getInstance*();  
 firebaseDatabase=FirebaseDatabase.*getInstance*();  
 calendar=Calendar.*getInstance*();  
 simpleDateFormat=*new* SimpleDateFormat("hh:mm a");  
  
  
 msenderuid=firebaseAuth.getUid();  
 mrecieveruid=getIntent().getStringExtra("receiveruid");  
 mrecievername=getIntent().getStringExtra("name");  
  
  
  
 senderroom=msenderuid+mrecieveruid;  
 recieverroom=mrecieveruid+msenderuid;  
  
  
  
 DatabaseReference databaseReference=firebaseDatabase.getReference().child("chats").child(senderroom).child("messages");  
 messagesAdapter=*new* MessagesAdapter(specificchat.*this*,messagesArrayList);  
 databaseReference.addValueEventListener(*new* ValueEventListener() {  
 @Override  
 *public void* onDataChange(@NonNull DataSnapshot snapshot) {  
 messagesArrayList.clear();  
 *for*(DataSnapshot snapshot1:snapshot.getChildren())  
 {  
 Messages messages=snapshot1.getValue(Messages.*class*);  
 messagesArrayList.add(messages);  
 }  
 messagesAdapter.notifyDataSetChanged();  
 }  
  
 @Override  
 *public void* onCancelled(@NonNull DatabaseError error) {  
  
 }  
 });  
  
  
  
  
 mbackbuttonofspecificchat.setOnClickListener(*new* View.OnClickListener() {  
 @Override  
 *public void* onClick(View view) {  
  
 mgetmessage.setText(*null*);  
 *//finish();* }  
 });  
  
  
 mnameofspecificuser.setText(mrecievername);  
 String uri=intent.getStringExtra("imageuri");  
 *if*(uri.isEmpty())  
 {  
 Toast.*makeText*(getApplicationContext(),"null is recieved",Toast.*LENGTH\_SHORT*).show();  
 }  
 *else* {  
 Picasso.*get*().load(uri).into(mimageviewofspecificuser);  
 }  
  
  
 msendmessagebutton.setOnClickListener(*new* View.OnClickListener() {  
 @Override  
 *public void* onClick(View view) {  
  
 enteredmessage=mgetmessage.getText().toString();  
 *if*(enteredmessage.isEmpty())  
 {  
 Toast.*makeText*(getApplicationContext(),"Enter message first",Toast.*LENGTH\_SHORT*).show();  
 }  
  
 *else* {  
 Date date=*new* Date();  
 currenttime=simpleDateFormat.format(calendar.getTime());  
 Messages messages=*new* Messages(enteredmessage,firebaseAuth.getUid(),date.getTime(),currenttime);  
 mgetmessage.setText("");  
 firebaseDatabase=FirebaseDatabase.*getInstance*();  
 firebaseDatabase.getReference().child("chats")  
 .child(senderroom)  
 .child("messages")  
 .push().setValue(messages).addOnCompleteListener(*new* OnCompleteListener<Void>() {  
 @Override  
 *public void* onComplete(@NonNull Task<Void> task) {  
 firebaseDatabase.getReference()  
 .child("chats")  
 .child(recieverroom)  
 .child("messages")  
 .push()  
 .setValue(messages).addOnCompleteListener(*new* OnCompleteListener<Void>() {  
 @Override  
 *public void* onComplete(@NonNull Task<Void> task) {  
  
 }  
 });  
 }  
 });  
  
 mgetmessage.setText("");  
*// messagesAdapter.notifyDataSetChanged();* }  
  
  
  
  
 }  
 });  
  
  
  
  
 }  
  
  
 @Override  
 *public void* onStart() {  
 *super*.onStart();  
 firebaseFirestore = FirebaseFirestore.*getInstance*();  
 messagesAdapter.notifyDataSetChanged();  
 DocumentReference documentReference = firebaseFirestore.collection("Users").document(firebaseAuth.getUid());  
 documentReference.update("status", "Online").addOnSuccessListener(*new* OnSuccessListener<Void>() {  
 @Override  
 *public void* onSuccess(Void unused) {  
 *//user is now offline* }  
 });  
 }  
  
 @Override  
 *public void* onStop() {  
 *super*.onStop();  
 *if*(messagesAdapter!=*null*)  
 {  
 messagesAdapter.notifyDataSetChanged();  
 DocumentReference documentReference=firebaseFirestore.collection("Users").document(firebaseAuth.getUid());  
 documentReference.update("status","offline").addOnSuccessListener(*new* OnSuccessListener<Void>() {  
 @Override  
 *public void* onSuccess(Void unused) {  
 *//user is now offline* }  
 });  
  
 }  
 }  
  
  
}

**Server Side**

**activity\_specfic\_chat.xml**

<?*xml version*="1.0" *encoding*="utf-8"?>  
<RelativeLayout *xmlns:android*="http://schemas.android.com/apk/res/android"  
 *xmlns:app*="http://schemas.android.com/apk/res-auto"  
 *xmlns:tools*="http://schemas.android.com/tools"  
 *android:layout\_width*="match\_parent"  
 *android:layout\_height*="match\_parent"  
 *android:background*="#ECE5DD"  
 *tools:context*=".specificchat">  
  
  
  
 <androidx.appcompat.widget.Toolbar  
 *android:layout\_width*="match\_parent"  
 *android:layout\_height*="?attr/actionBarSize"  
 *android:layout\_marginTop*="0dp"  
 *android:background*="#075e54"  
 *android:id*="@+id/toolbarofspecificchat">  
  
 <RelativeLayout  
 *android:layout\_width*="match\_parent"  
 *android:layout\_height*="match\_parent"  
 *android:gravity*="center\_vertical">  
  
 <ImageButton  
 *android:layout\_width*="40dp"  
 *android:layout\_height*="40dp"  
 *android:id*="@+id/backbuttonofspecificchat"  
 *app:tint*="@color/white"  
 *android:background*="@android:color/transparent"  
 *android:src*="@drawable/ic\_baseline\_arrow\_back\_24"  
 *android:layout\_centerVertical*="true">  
  
 </ImageButton>  
  
  
 <androidx.cardview.widget.CardView  
 *android:layout\_width*="35dp"  
 *android:layout\_height*="35dp"  
 *android:id*="@+id/cardviewofspeficuser"  
 *android:layout\_marginLeft*="5dp"  
 *android:layout\_toRightOf*="@id/backbuttonofspecificchat"  
 *android:layout\_centerVertical*="true"  
 *app:cardCornerRadius*="35dp">  
  
  
 <ImageView  
 *android:layout\_width*="match\_parent"  
 *android:layout\_height*="match\_parent"  
 *android:src*="@drawable/user"  
 *android:id*="@+id/specificuserimageinimageview"  
 *android:scaleType*="centerCrop">  
  
 </ImageView>  
  
  
  
 </androidx.cardview.widget.CardView>  
  
  
  
  
 <TextView  
 *android:layout\_width*="wrap\_content"  
 *android:layout\_height*="wrap\_content"  
 *android:text*="Name of User"  
 *android:textSize*="20sp"  
 *android:layout\_toRightOf*="@id/cardviewofspeficuser"  
 *android:id*="@+id/Nameofspecificuser"  
 *android:layout\_marginLeft*="10dp"  
 *android:layout\_centerVertical*="true"  
 *android:textStyle*="bold"  
 *android:textColor*="@color/white">  
  
 </TextView>  
  
  
 </RelativeLayout>  
  
  
  
  
 </androidx.appcompat.widget.Toolbar>  
  
 <androidx.recyclerview.widget.RecyclerView  
 *android:layout\_width*="match\_parent"  
 *android:layout\_height*="match\_parent"  
 *android:layout\_above*="@id/getmessage"  
 *android:id*="@+id/recyclerviewofspecific"  
 *android:layout\_below*="@id/toolbarofspecificchat"  
 *android:padding*="5dp">  
  
 </androidx.recyclerview.widget.RecyclerView>  
  
 <EditText  
 *android:layout\_width*="match\_parent"  
 *android:layout\_height*="50dp"  
 *android:hint*="Type a message"  
 *android:layout\_marginLeft*="5dp"  
 *android:layout\_marginBottom*="5dp"  
 *android:paddingLeft*="20dp"  
 *android:paddingEnd*="10dp"  
 *android:textSize*="18sp"  
 *android:background*="@drawable/messagebackground"  
 *android:textAlignment*="textStart"  
 *android:layout\_alignParentBottom*="true"  
 *android:layout\_marginRight*="55dp"  
 *android:textColor*="@color/black"  
 *android:textColorHint*="#A8A7A7"  
 *android:id*="@+id/getmessage" />  
  
  
 <androidx.cardview.widget.CardView  
 *android:layout\_width*="45dp"  
 *android:layout\_height*="45dp"  
 *android:id*="@+id/carviewofsendmessage"  
 *android:layout\_toRightOf*="@id/getmessage"  
 *android:layout\_marginLeft*="-50dp"  
 *android:layout\_marginBottom*="5dp"  
 *android:backgroundTint*="#0D8F80"  
 *android:layout\_alignParentBottom*="true"  
 *app:cardCornerRadius*="45dp">  
  
  
 <ImageButton  
 *android:layout\_width*="match\_parent"  
 *android:layout\_height*="match\_parent"  
 *android:src*="@drawable/ic\_baseline\_arrow\_forward\_ios\_24"  
 *android:padding*="5dp"  
 *android:backgroundTint*="@android:color/transparent"  
 *android:background*="@android:color/transparent"  
 *android:id*="@+id/imageviewsendmessage"  
 *android:layout\_gravity*="center"  
 *android:scaleType*="centerCrop"  
 *app:tint*="@color/white">  
  
 </ImageButton>  
  
  
  
 </androidx.cardview.widget.CardView>

**Updateprofile.java**

*package* com.aqirlone.talkwithbuddies;  
  
*import* androidx.annotation.NonNull;  
*import* androidx.annotation.Nullable;  
*import* androidx.appcompat.app.AppCompatActivity;  
  
*import* android.content.Intent;  
*import* android.graphics.Bitmap;  
  
*import* android.net.Uri;  
*import* android.os.Bundle;  
*import* android.provider.MediaStore;  
*import* android.util.Log;  
*import* android.view.View;  
*import* android.widget.Toast;  
  
*import* com.aqirlone.talkwithbuddies.databinding.ActivitySetProfileBinding;  
*import* com.aqirlone.talkwithbuddies.databinding.ActivityUpdateProfileBinding;  
*import* com.google.android.gms.tasks.*OnFailureListener*;  
*import* com.google.android.gms.tasks.*OnSuccessListener*;  
*import* com.google.firebase.auth.FirebaseAuth;  
*import* com.google.firebase.database.DatabaseReference;  
*import* com.google.firebase.database.FirebaseDatabase;  
*import* com.google.firebase.firestore.DocumentReference;  
*import* com.google.firebase.firestore.FirebaseFirestore;  
*import* com.google.firebase.storage.FirebaseStorage;  
*import* com.google.firebase.storage.StorageReference;  
*import* com.google.firebase.storage.UploadTask;  
  
*import* java.io.ByteArrayOutputStream;  
  
*import* java.io.IOException;  
*import* java.util.HashMap;  
*import* java.util.*Map*;  
  
*public class* UpdateProfile *extends* AppCompatActivity {  
  
 ActivityUpdateProfileBinding binding;  
 *private final static int PICK\_IMAGE*=123;  
 *private* Uri imagepath;  
 *private* FirebaseAuth firebaseAuth;  
 *private* String name;  
 *private* FirebaseStorage firebaseStorage;  
 *private* StorageReference storageReference;  
 *private* String ImageUriAccessToken;  
 *private* FirebaseFirestore firebaseFirestore;  
  
 @Override  
 *protected void* onCreate(Bundle savedInstanceState) {  
 *super*.onCreate(savedInstanceState);  
 binding=ActivityUpdateProfileBinding.*inflate*(getLayoutInflater());  
 setContentView(binding.getRoot());  
 firebaseAuth=FirebaseAuth.*getInstance*();  
 firebaseStorage=FirebaseStorage.*getInstance*();  
 storageReference=firebaseStorage.getReference();  
 firebaseFirestore=FirebaseFirestore.*getInstance*();  
  
  
 binding.getuserimage.setOnClickListener(*new* View.OnClickListener() {  
 @Override  
 *public void* onClick(View view) {  
 Intent intent=*new* Intent(Intent.*ACTION\_PICK*, MediaStore.Images.Media.*INTERNAL\_CONTENT\_URI*);  
 startActivityForResult(intent,*PICK\_IMAGE*);  
 }  
 });  
 binding.saveprofilebtn.setOnClickListener(*new* View.OnClickListener() {  
 @Override  
 *public void* onClick(View view) {  
 name=binding.getusername.getText().toString();  
 *if*(name.isEmpty()){  
 Toast.*makeText*(getApplicationContext(), "Name Is Empty", Toast.*LENGTH\_SHORT*).show();  
 }*else if*(imagepath==*null*){  
 Toast.*makeText*(getApplicationContext(), "Image is empty", Toast.*LENGTH\_SHORT*).show();  
 }*else*{  
  
 binding.progressbarofsetprofile.setVisibility(View.*VISIBLE*);  
 sendDataForNewUser();  
 binding.progressbarofsetprofile.setVisibility(View.*INVISIBLE*);  
 Intent intent=*new* Intent(UpdateProfile.*this*,ProfileActivity.*class*);  
 startActivity(intent);  
 finish();  
  
 }  
 }  
 });  
  
  
 }  
  
  
 @Override  
 *protected void* onActivityResult(*int* requestCode, *int* resultCode, @Nullable Intent data) {  
  
 *if*(requestCode==*PICK\_IMAGE* && resultCode==*RESULT\_OK*){  
  
 imagepath=data.getData(); *////looooooooooooooooooooooooook heeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeerrreeeeeeeeee* binding.getuserimageinimageview.setImageURI(imagepath);  
 }  
 *super*.onActivityResult(requestCode, resultCode, data);  
 }  
  
 *private void* sendDataForNewUser(){  
 sendDataToRealTimeDatabase();  
 }  
  
  
  
 *private void* sendDataToRealTimeDatabase(){  
 name=binding.getusername.getText().toString();  
 FirebaseDatabase firebaseDatabase=FirebaseDatabase.*getInstance*();  
 DatabaseReference databaseReference=  
 firebaseDatabase.getReference(firebaseAuth.getUid());  
 UserProfileModel userProfileModel=*new* UserProfileModel(name,firebaseAuth.getUid());  
 databaseReference.setValue(userProfileModel);  
 *// Toast.makeText(getApplicationContext(), "User Profile Added Successfully", Toast.LENGTH\_SHORT).show();* sendImageToStorage();  
  
 }  
 *private void* sendImageToStorage(){  
  
 StorageReference imagrref=storageReference.child("Images").child(firebaseAuth.getUid()).child("profile Pic");  
 *//Image Compression Code* Bitmap bitmap=*null*;  
 *try* {  
 bitmap = MediaStore.Images.Media.*getBitmap*(getContentResolver(), imagepath);  
 }*catch* (IOException e){  
 Log.*w*("TAG", "........................:failure.............................................");  
 e.printStackTrace();  
 }  
 ByteArrayOutputStream byteArrayOutputStream=*new* ByteArrayOutputStream();  
 bitmap.compress(Bitmap.CompressFormat.*JPEG*,25,byteArrayOutputStream);  
 *byte*[] data=byteArrayOutputStream.toByteArray();  
  
 *// putting image to storage* UploadTask uploadTask=imagrref.putBytes(data);  
  
 uploadTask.addOnSuccessListener(*new* OnSuccessListener<UploadTask.TaskSnapshot>() {  
  
 @Override  
 *public void* onSuccess(UploadTask.TaskSnapshot taskSnapshot) {  
 imagrref.getDownloadUrl().addOnSuccessListener(*new* OnSuccessListener<Uri>() {  
 @Override  
 *public void* onSuccess(Uri uri) {  
 ImageUriAccessToken=uri.toString();  
 *// Toast.makeText(getApplicationContext(), "uri get successful", Toast.LENGTH\_SHORT).show();* sendDataToCloudFireStore();  
  
 }  
 }).addOnFailureListener(*new* OnFailureListener() {  
 @Override  
 *public void* onFailure(@NonNull Exception e) {  
 Toast.*makeText*(getApplicationContext(), "uri get Failed", Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
 Toast.*makeText*(getApplicationContext(), "Image Uploaded Successfully", Toast.*LENGTH\_SHORT*).show();  
 }  
 }).addOnFailureListener(*new* OnFailureListener() {  
 @Override  
 *public void* onFailure(@NonNull Exception e)  
 {  
 Toast.*makeText*(getApplicationContext(), "Image not uploaded", Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
  
  
  
  
 }  
  
 *private void* sendDataToCloudFireStore() {  
  
 DocumentReference documentReference=firebaseFirestore.collection("Users").document(firebaseAuth.getUid());  
 *Map*<String ,Object> userData=*new* HashMap<>();  
 userData.put("name",name);  
 userData.put("image",ImageUriAccessToken);  
 userData.put("uid",firebaseAuth.getUid());  
 userData.put("status","Online");  
  
 documentReference.set(userData).addOnSuccessListener(*new* OnSuccessListener<Void>() {  
 @Override  
 *public void* onSuccess(Void unused) {  
 *// Toast.makeText(getApplicationContext(), "Data On Cloud FireStorage Send Successfully ", Toast.LENGTH\_SHORT).show();* }  
 });  
  
  
 }  
 @Override  
 *protected void* onStop() {  
  
 *super*.onStop();  
 DocumentReference documentReference=firebaseFirestore.collection("Users").document(firebaseAuth.getUid());  
 documentReference.update("status","offline").addOnSuccessListener(*new* OnSuccessListener<Void>() {  
 @Override  
 *public void* onSuccess(Void unused) {  
 *//user is now offline* }  
 });  
 }  
  
 @Override  
 *protected void* onStart() {  
 *super*.onStart();  
 DocumentReference documentReference=firebaseFirestore.collection("Users").document(firebaseAuth.getUid());  
 documentReference.update("status","Online").addOnSuccessListener(*new* OnSuccessListener<Void>() {  
 @Override  
 *public void* onSuccess(Void unused) {  
 *//user is now offline* }  
 });  
  
 }  
}

**UserProfilemodel,java**

*package* com.aqirlone.talkwithbuddies;  
  
*public class* UserProfileModel {  
  
 *public* String userName,userUID;  
  
 *public* UserProfileModel(String userName, String userUID) {  
 *this*.userName = userName;  
 *this*.userUID = userUID;  
 }  
  
 *public* UserProfileModel() {  
 }  
  
 *public* String getUserName() {  
 *return* userName;  
 }  
  
 *public void* setUserName(String userName) {  
 *this*.userName = userName;  
 }  
  
 *public* String getUserUID() {  
 *return* userUID;  
 }  
  
 *public void* setUserUID(String userUID) {  
 *this*.userUID = userUID;  
 }  
}

**CONCLUSION**

The main objective of the project is to develop a Secure Chat Application. I had taken a wide range of literature review in order to achieve all the tasks, where I came to know about some of the products that are existing in the market. I made a detailed research in that path to cover the loop holes that existing systems are facing and to eradicate them in our application. In the process of research I came to know about the latest technologies and different algorithms.

**FUTURE VISION**

With the knowledge I have gained by developing this application, I am confident that in the future I can make the application more effectively by adding this services.

* Extending this application by providing Authorisation service.
* Creating Database and maintaining users.
* Increasing the effectiveness of the application by providing Voice Chat.
* Extending it to Web Support.
* Allowing users to Create different rooms and join different chats.