Android Software Development for

ITC Applications

Project Leader

Dr Zheng Xie

Report by

Bipin Paneru

Contents

[Executive Summary 1](#_Toc80992142)

[Introduction 1](#_Toc80992143)

[Methodology 1](#_Toc80992144)

[Home Task 1 1](#_Toc80992145)

[Home Task 2 2](#_Toc80992146)

[Demonstration 4](#_Toc80992147)

[Home Task 1 Application 4](#_Toc80992148)

[Home Task 2 Application 6](#_Toc80992149)

[Conclusion 9](#_Toc80992150)

[Appendix A 10](#_Toc80992151)

[Home Task 1 Java Code – Talking to itself (MainActivity.java) 10](#_Toc80992152)

[Home Task 1 XML Script – Talking to itself (MainActivity.xml) 11](#_Toc80992153)

[Home Task 1 Manifest Script – Talking to itself (AndroidManifest.xml) 12](#_Toc80992154)

[Home Task 1 Xml Script – Talking to itself (Strings.xml) 12](#_Toc80992155)

[Home Task 1 Java Code – Advanced Talking to itself (LoginScreen.java) 13](#_Toc80992156)

[Home Task 1 Java Code – Advanced Talking to itself (ChoiceScreen.java) 13](#_Toc80992157)

[Home Task 1 Java Code – Advanced Talking to itself (MainActivity.java) 14](#_Toc80992158)

[Home Task 1 Java Code – Advanced Talking to itself (DiceRoller.java) 15](#_Toc80992159)

[Home Task 1 XML Script – Advanced Talking to itself (activity\_login\_screen.xml) 16](#_Toc80992160)

[Home Task 1 XML Script – Advanced Talking to itself (activity\_choice\_screen.xml) 17](#_Toc80992161)

[Home Task 1 XML Script – Advanced Talking to itself (activity\_main.xml) 17](#_Toc80992162)

[Home Task 1 XML Script – Advanced Talking to itself (activity\_dice\_roller.xml) 19](#_Toc80992163)

[Home Task 1 – Advanced Talking to itself Manifest Script 20](#_Toc80992164)

[Home Task 1 – Advanced Talking to itself (Strings.xml) 20](#_Toc80992165)

[Home Task 2 Java Code (WAVConverter.java) 22](#_Toc80992166)

[Home Task 2 Java Code (WAVAudioRecorder.java) 23](#_Toc80992167)

[Home Task 2 Java Code (AudioFileManager.java) 25](#_Toc80992168)

[Home Task 2 Java Code (ErrorManager.java) 26](#_Toc80992169)

[Home Task 2 Java Code (LoadScreen.java) 26](#_Toc80992170)

[Home Task 2 Java Code (RecordScreen.java) 27](#_Toc80992171)

[Home Task 2 Java Code (ListScreen.java) 30](#_Toc80992172)

[Home Task 2 XML Scripts (activity\_load\_screen.xml) 32](#_Toc80992173)

[Home Task 2 XML Scripts(activity\_record\_screen.xml) 33](#_Toc80992174)

[Home Task 2 XML Scripts(activity\_list\_screen.xml) 34](#_Toc80992175)

[Home Task 2 Manifest Script 35](#_Toc80992176)

[Home Task 2 String.xml 36](#_Toc80992177)

[Appendix B 36](#_Toc80992178)

[Python Server Code 36](#_Toc80992179)

[References 38](#_Toc80992180)

# Executive Summary

The undergraduate research internship program titled “Android software development for ICT applications” comprised of tasks involving android studio and jupyter notebook covering Java, Xml and Python programming / scripting languages. For 9 weeks various tasks were given focusing on application development on android studio and simple server script on jupyter notebook, assuming that at the end of the internship a working application will be produced capable of exchanging strings of text, recorded text messages, images, MP3 files, and real time speech between application and server. During internship period the python server was used to communicate while the final product will be communicating to firebase server to store and retrieve the audio.

# Introduction

Although software development for conventional platforms is a mainstream activity in undergraduate teaching and postgraduate research, the development of application software for mobile phones is still not a widely taught skill. This is surprising in view of the amount of software development that is now required for the mobile market. Many research opportunities exist for graduates with skills in mobile software. Many of these opportunities involve the use of Internet Communications Technology (ICT) along with basic computing activities and graphical user interfaces (GUI). This internship being a part of bigger project, aims to gain some experience with the development of distributed ICT applications for mobile platforms whose result will be used in order to produce an instruction manual for introducing undergraduates and new graduate students to this type of software development environment.

# Methodology

For this internship, the goal was to develop a messaging system application capable of exchanging strings of text, recorded text messages, images, MP3 files, and real time speech. In order to achieve these 2 home tasks were given with multiple revisions. The codes for each home task can be found in Appendix A.

## Home Task 1

This task started with instructions to install and set up the android studio along with instructions to setup Android virtual device (AVD). As Android studio makes use of java language Java Development Kit (JDK) was required to be installed beforehand. After installing the android studio successfully by following the instructions it is required to create a new project in order to setup the AVD. For simplicity all the options were left as it was and empty project was created. Android studio involves a lot of downloading and a lot of ways to modify the working environment as per the preference of developer. In this case the downloaded AVD was for android 11. This internship also required to do testing on physical device. The physical device had android 10.1 OS. One of the helpful settings for new learners would be not having to worry about the classes and interfaces that needed to be imported on every activity, as java is an object-oriented language and specifically in terms of android studio it contains a vast number of libraries for ease of application development. For this automatic import was turned on so that android studio would handle importing.

Android studio basically has separate parts for front-end or designing and for back-end or developing. For front-end XML is used and for back-end java is used. IT also has a separate manifest file for setting up the whole project. There are separate resource files which can be used by both front-end and back-end. There are many other files like Gradle file can be used to import any third-party package to the project but that is out of scope for this internship.

After getting familiar with the android studio and understanding how android studio organizes the files in various directories then there were series of task to be performed. Starting with simple hello world program and the program to talk with itself the final task was to make an application with dice rolling function. The two parts of dice rolling function was one was totally random number generation between 1 to 6 while other was to create the cheating system where certain number was to have higher probability of getting rolled. Home task 1 ended after completion of above-mentioned application development. The application is demonstrated in detail below with attached screenshots.

## Home Task 2

Home task two started with introduction to python programming language. Python was used to develop a server for android application to communicate with. For such purpose Jupyter notebook was required to be installed which allowed hosting of a server in localhost (127.0.0.1). In order to install jupyter for this internship anaconda was installed which came with Jupyter notebook and other useful tools. In addition, pyaudio library was downloaded using conda as we were dealing with audio. Starting with basic python codes in python and getting familiar with Jupyter Notebook the home task two had demo code for python server. The python client developed in Jupyter notebook was used as a reference to build the android client as well as to verify if everything was working as intended.

The communication between server and client was established via TCP communication set up with Socket. The localhost was used as ip and the server accepted communication through port number 1234. First the test was conducted between a simple server that was able to accept a client required resetting every time the connection was closed. Then there was a second server that maintained the queue of requests from clients using multithreading and accepted multiple communication in a sequence. There were some minor errors while implementing the codes but were fixed after debugging.

After achieving successful communication between client and server the next step was to be able to record the audio in android client and play it back. First it was necessary to verify that the audio was being recorded and saved in such a manner that it can be played back as the audio was recorded in raw format. Recording audio in raw format gives us flexibility as such it can be converted into any final common format. Raw audio also contains many meta data which can be processed before sending to server or at server. But most importantly raw audio is uncompressed unlike common formats such as mp3.

The first attempt was to develop a naughty version of android which did not use multithreading and had an issue with the UI as it kept freezing. Once the raw audio was recorded and successfully stored, we had to add a header file in order to make it playable. The method shown in home task two was resulting in error, so a new method was implemented after some online research. This new method was easy to understand and was clear what was being added in header and it produced the playable audio without errors. The naughty version was not usable because of its issues with unresponsive UI hence with additional research a threaded version was implemented.

While the threaded version was able to record and play the audio flawlessly there were still some problems for establishing the communication between android client and the server. Either the communication was not being established or communication was not being closed. This was later mitigated after revised home task two. The first test was conducted by sending just a text message from client to server. After successful attempt the audio file was recorded and sent to the server. For some reason the audio was being received by server but was not able to play it back. The problem was the recorded audio was stored in byte array in android client, but the server was expecting byte string. After final revision of the home task two the problem was identified and was fixed for the final version of the application.

For home task 2 most of the task were referenced to the documents provided by tutor whereas certain part of the code was looked up on internet for updated version or in order to tackle the issues faced during application development phase. The referenced topics are: WAVE file header data (seanxiaoxiao, 2012), Creating custom app icon (developers, 2021) and multithreading in android (developers, android developers, 2021). The detailed reference can be found at the end of the document.

# Demonstration

## Home Task 1 Application

Graphical user interface

Description automatically generatedGraphical user interface, application

Description automatically generatedThe first application of home task 1 was simple application which takes input from user in EditText and displays the same in the TextView.

Graphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, application, Teams

Description automatically generatedGraphical user interface

Description automatically generatedGraphical user interface, application, Teams

Description automatically generatedThe second application in home task 1 loads with a login screen. The login screen accepts only valid username and password i.e. Username: tom Password: meow. After logging in the option screen has two buttons. The sum finder will launch the interface to find sum of the numbers entered in the textbox.

Graphical user interface, application, Teams

Description automatically generated

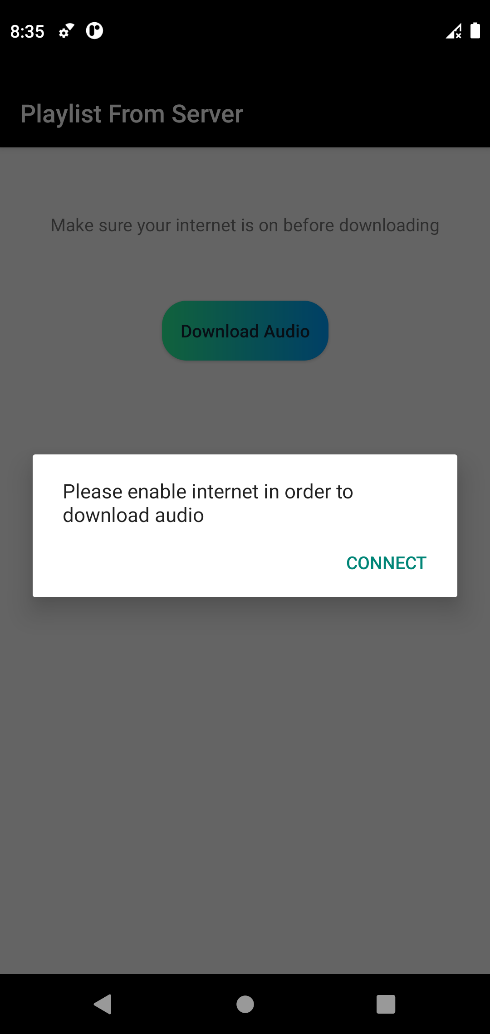
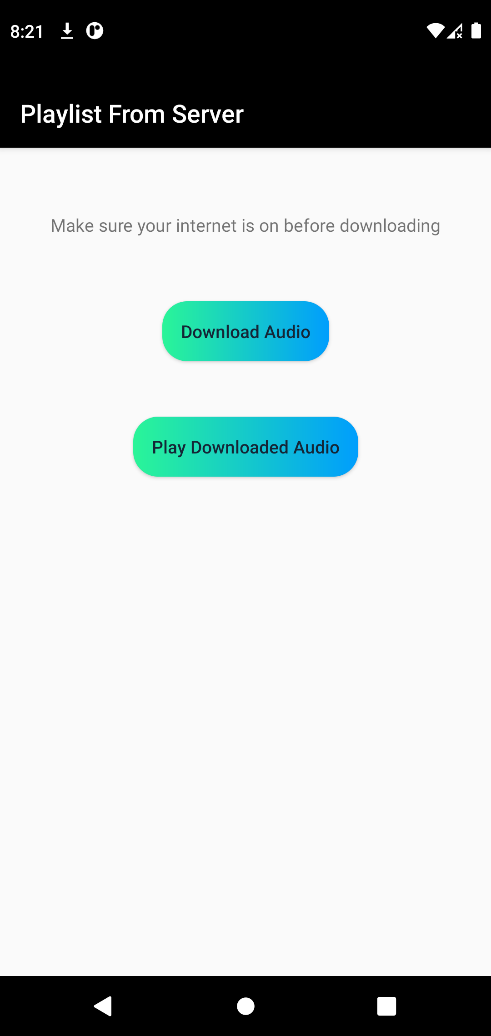
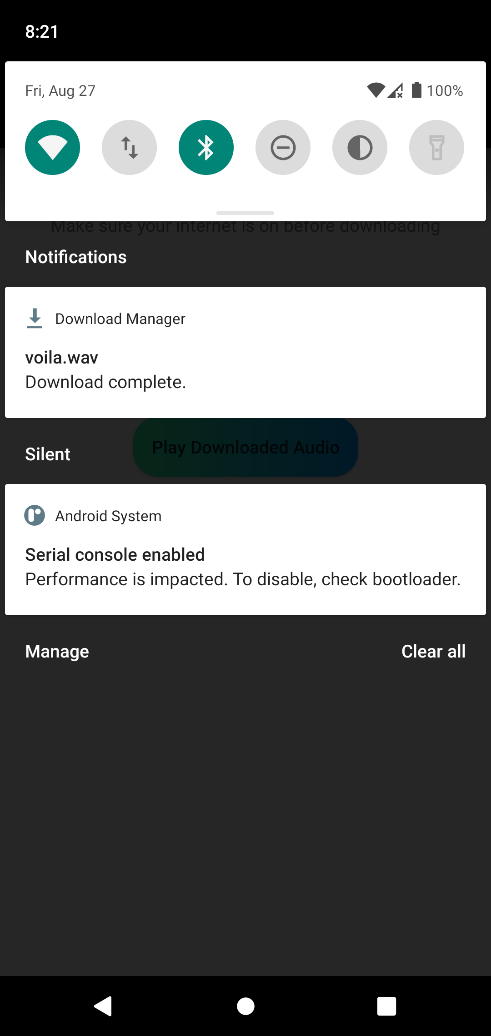
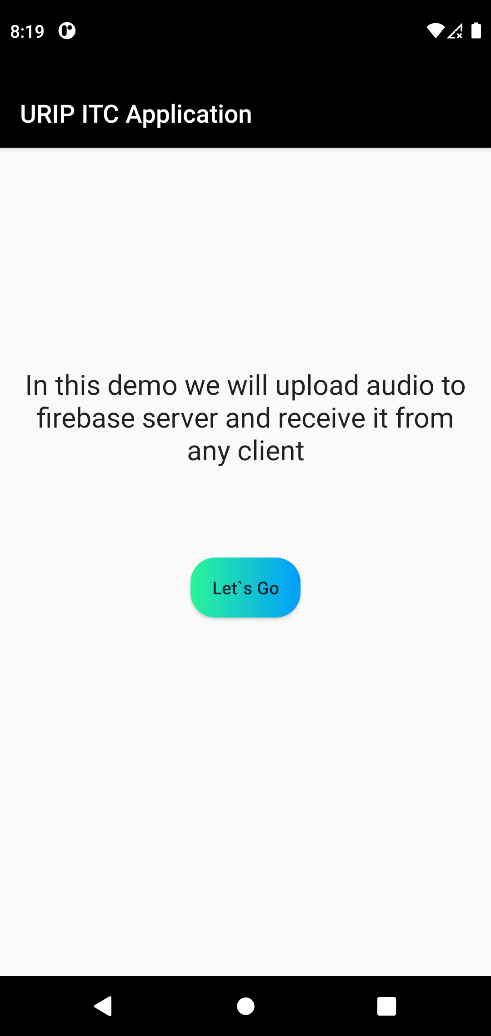
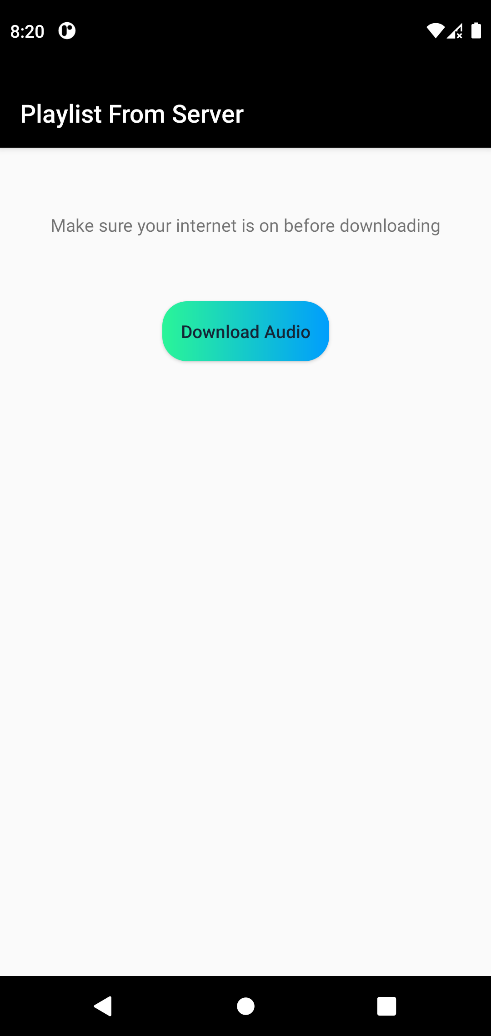
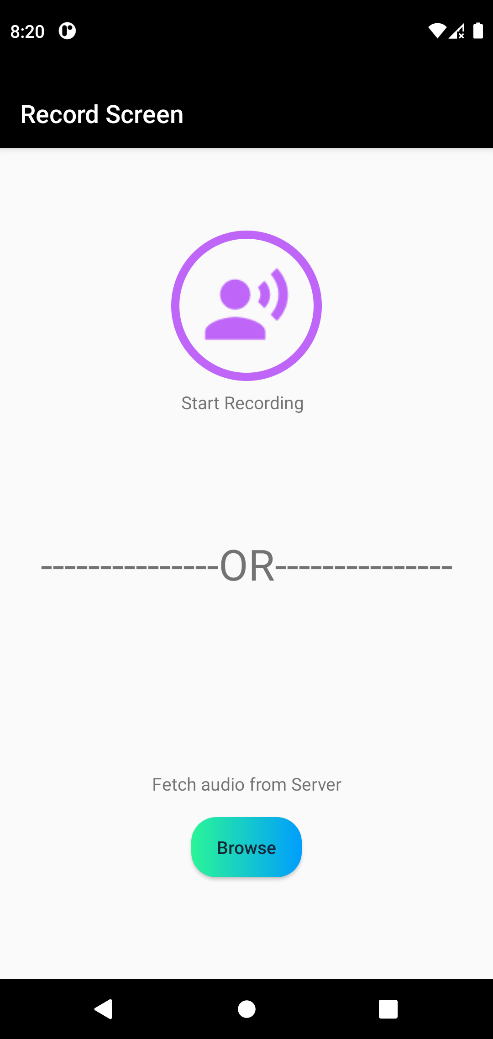
The second button launches the new screen that prompts user to input the bet and the predected number. When the roll button is pressed the a random number is generated between 1 to 6 and displayed. When the cheat option is enabled the generated number between 1 to 6 will be generated according to the probability of the respective number. In our case the number with higher probability is 3.

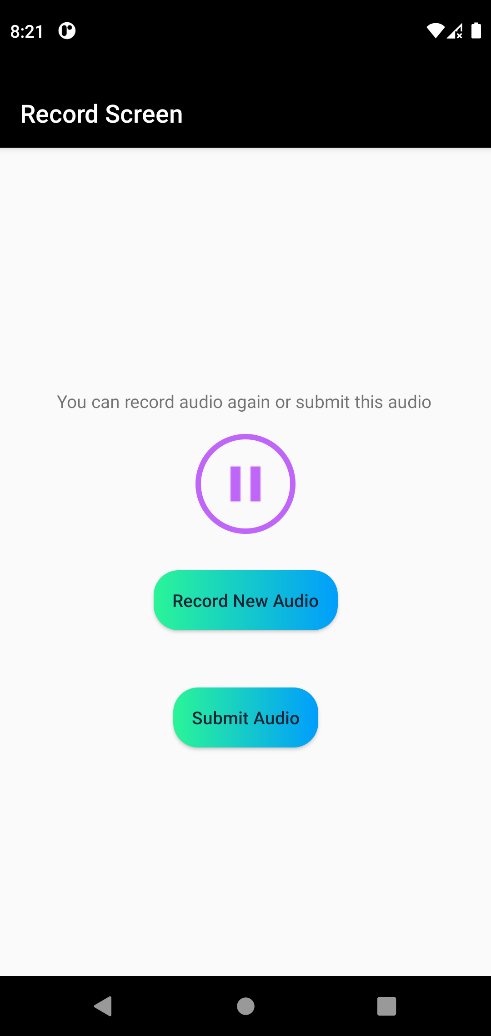
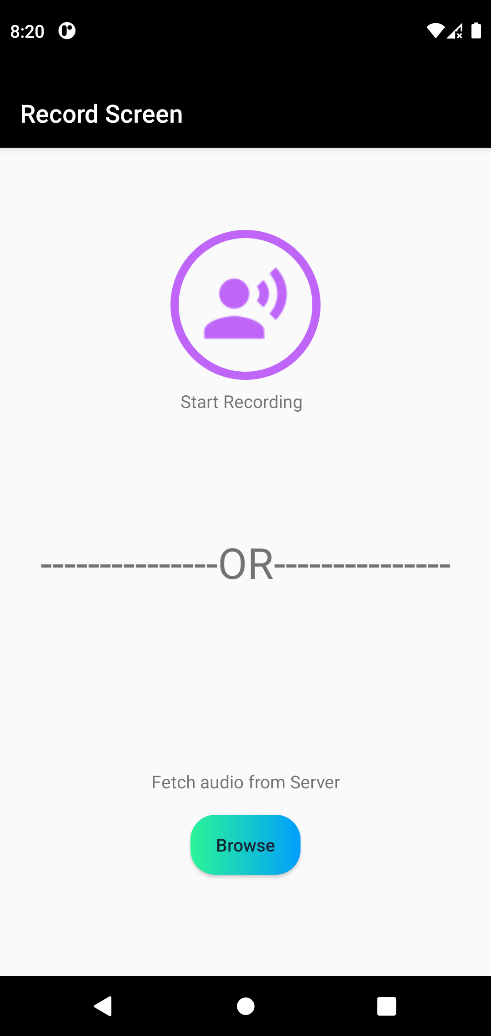
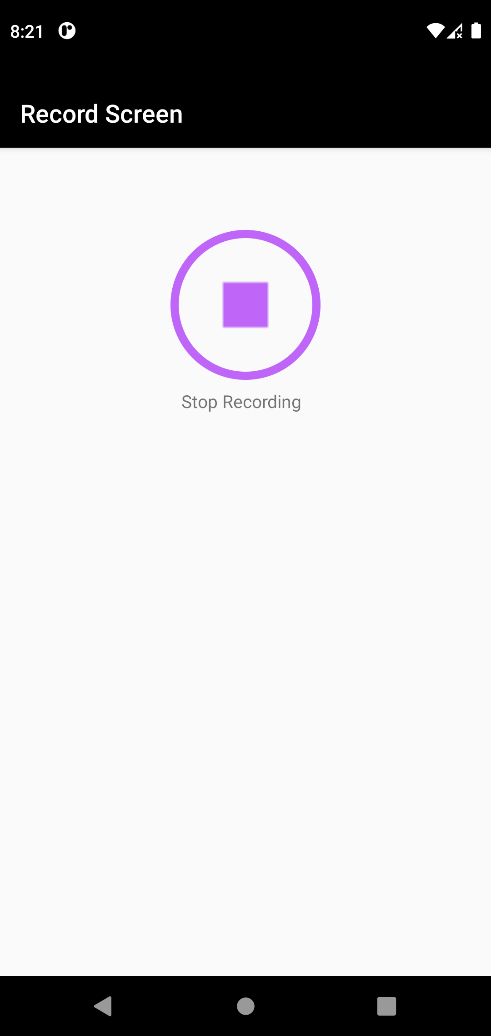
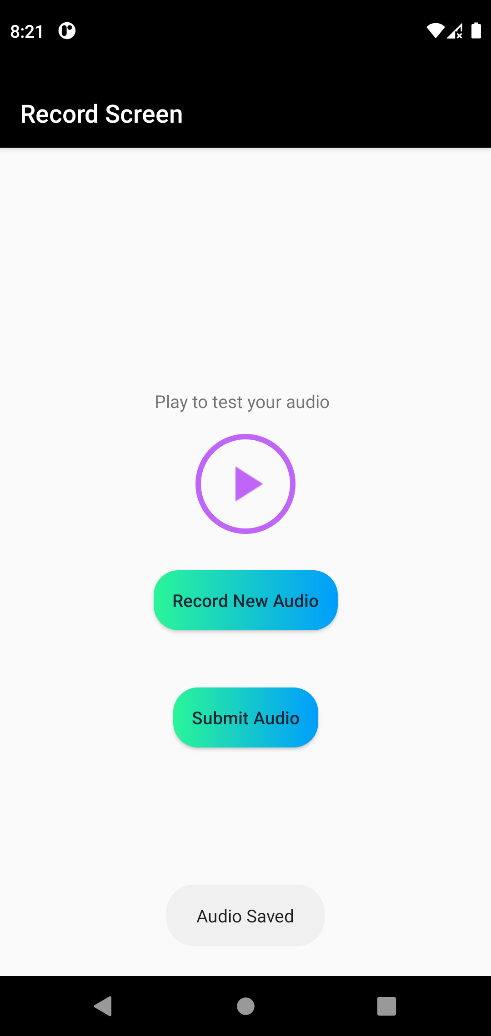
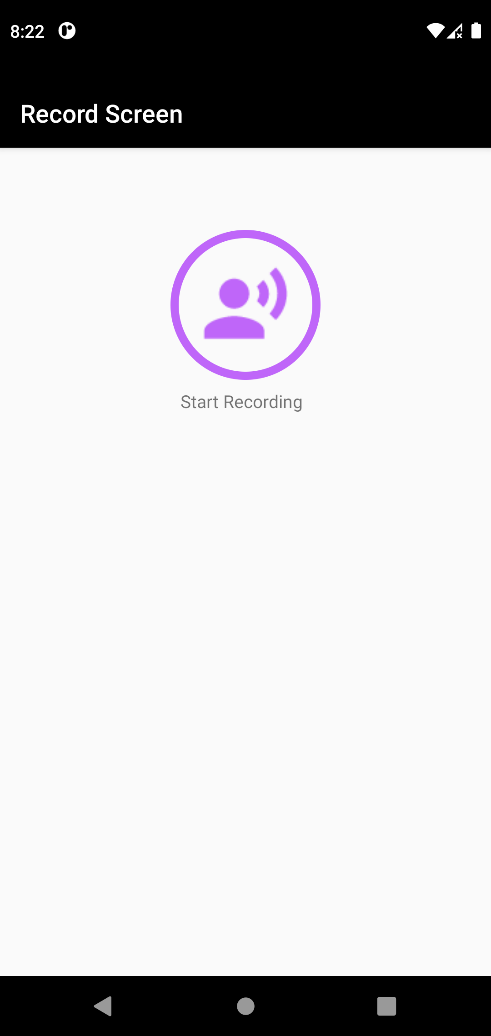
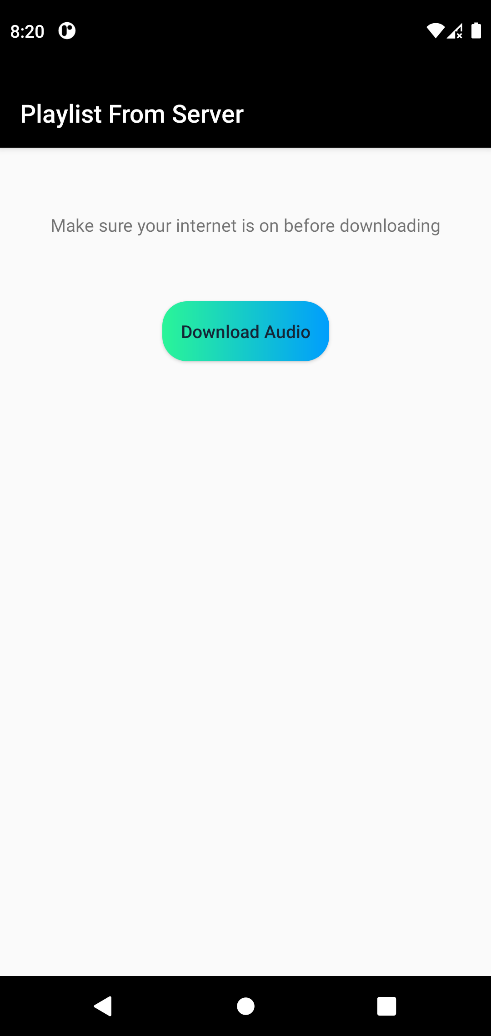
## Graphical user interface, application, Teams Description automatically generatedGraphical user interface, application, Teams Description automatically generatedHome Task 2 Application

Although there were multiple versions of the application developed for home task 2 only the final working version has been included in this report file.

In Home task 2 we simulate the audio data transferred between android client and web server with the use of internet service. The application used for the demonstration is simple and minimalistic in design. The first page of the application gives a brief description of what is going to happen in the application. The next screen will allow users to either record the audio and upload it to the server or download the audio and play. Once the audio has been recorded and uploaded to the server it can be downloaded and played from any device that has the application installed.

The screens presented to user who will simulate the receiving end of the application will see following screens:

The internet must be enabled in order for download audio button to work otherwise it will redirect to data connection menu where users will be promped to wither connect to a wifi or mobile data. Once the internet is accissible the audio can be downloaded from server and played.

If the user wishes to simulate the sending end they can simply record the audio and submit it. Once the audio is recorded they can test it and if they wish to they can rerecord the audio before submitting. After submittig the audio they will be directed to download screen as well.

# Conclusion

The final application from Home Task 2 is fully functioning working prototype which can be demonstrated with internet connection. This final application can be used as foundation or motivation for the application that will be developed in the main project. The design aspect of the applicaation have been kept minimal while focusing on the actual working of the application i.e. The audio is being recorded in raw format and stores as a uncompressed wave format which is then sent to the firebase server and can be retrieved back without any loss in original audio data being recorded.

The link for github repo is: <https://github.com/VoidofLimbo/SummerInternship2021>

# Appendix A

## Home Task 1 Java Code – Talking to itself (MainActivity.java)

import android.os.Bundle;  
import android.text.method.ScrollingMovementMethod;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.TextView;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
public class MainActivity extends AppCompatActivity {  
  
 TextView outputReceiver;  
 EditText inputProvider;  
 Button buttonSend;  
 Button buttonKill;  
 String receivedText, newText, initialText, newline = "\n";  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 outputReceiver = (TextView) findViewById(R.id.*myTextViewBox*);  
 inputProvider = (EditText) findViewById(R.id.*myEditTextBox*);  
 buttonSend = (Button) findViewById(R.id.*mySendBtn*);  
 buttonKill = (Button) findViewById(R.id.*myKillBtn*);  
  
 outputReceiver.setMovementMethod(new ScrollingMovementMethod());  
 initialText = newText = outputReceiver.getText().toString() + newline;  
 }  
  
// // New text is added at the end old item stays at top  
// public void sendMethod(View view) {  
// receivedText = inputProvider.getText().toString(); //myText  
// outputReceiver.setText(">" + receivedText + newline + newText);  
// inputProvider.setText(null);  
// }  
  
// // New text on top old item moves down display text also moves down  
// public void sendMethod(View view) {  
// receivedText = inputProvider.getText().toString();  
// if(!receivedText.isEmpty()){  
// newText = receivedText + newline + newText;  
// }  
// outputReceiver.setText(newText);  
// inputProvider.setText(null);  
// }  
  
 // New text is added at the end old item stays at top display text does not move  
 public void sendMethod(View view) {  
 receivedText = inputProvider.getText().toString();  
 if(!receivedText.isEmpty()){  
 newText += receivedText + newline;  
 }  
 outputReceiver.setText(newText);  
 inputProvider.setText(null);  
 }  
  
// // New text on top old item moves down display text stays on top  
// public void sendMethod(View view) {  
// receivedText = inputProvider.getText().toString();  
// if(!receivedText.isEmpty()){  
// if(!newText.equals(initialText)){  
// newText = receivedText + newline + newText;  
// } else {  
// newText = receivedText + newline;  
// }  
// }  
// outputReceiver.setText(initialText + newText);  
// inputProvider.setText(null);  
// }  
  
// public void killMethod(View view) {  
// finish(); // not requiring to press button twice  
// onDestroy();  
// }  
  
 public void killMethod(View view) {  
 finish(); // closes any thread or process running before exiting the app.  
 System.*exit*(0);  
 }  
  
  
//// code will work even without the following override  
// @Override  
// protected void onDestroy() {  
// super.onDestroy();  
// }  
}

## Home Task 1 XML Script – Talking to itself (MainActivity.xml)

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout  
 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"  
 tools:context=".MainActivity">  
  
 <TextView  
 android:id="@+id/myTextViewBox"  
 android:layout\_width="300dp"  
 android:layout\_height="150dp"  
 android:background="#00BCD4"  
 android:padding="5dp"  
 android:scrollbars="vertical"  
 android:text="@string/textView"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.10" />  
  
 <EditText  
 android:id="@+id/myEditTextBox"  
 android:layout\_width="300dp"  
 android:layout\_height="50dp"  
 android:background="#8BC34A"  
 android:hint="@string/editText"  
 android:padding="5dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/myTextViewBox"  
 app:layout\_constraintVertical\_bias="0.05"  
 />  
  
 <Button  
 android:id="@+id/myKillBtn"  
 android:layout\_width="100dp"  
 android:layout\_height="50dp"  
 android:onClick="killMethod"  
 android:text="@string/kill"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.75"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/myEditTextBox"  
 app:layout\_constraintVertical\_bias="0.05"  
 />  
  
 <Button  
 android:id="@+id/mySendBtn"  
 android:layout\_width="100dp"  
 android:layout\_height="50dp"  
 android:onClick="sendMethod"  
 android:text="@string/send"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.25"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/myEditTextBox"  
 app:layout\_constraintVertical\_bias="0.05"  
 />  
  
</androidx.constraintlayout.widget.ConstraintLayout>

## Home Task 1 Manifest Script – Talking to itself (AndroidManifest.xml)

<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="{your package name}">  
  
 <application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:roundIcon="@mipmap/ic\_launcher\_round"  
 android:supportsRtl="true"  
 android:theme="@style/Theme.MumboJumbo">  
 <activity android:name=".MainActivity">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
 </application>  
  
</manifest>

## Home Task 1 Xml Script – Talking to itself (Strings.xml)

<resources>  
 <string name="app\_name">MumboJumbo</string>  
 <string name="textView">Data:</string>  
 <string name="editText">Enter your text here</string>  
 <string name="kill">Kill</string>  
 <string name="send">Send</string>  
</resources>

## Home Task 1 Java Code – Advanced Talking to itself (LoginScreen.java)

import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.EditText;  
import android.widget.Toast;  
  
public class LoginScreen extends AppCompatActivity {  
  
 EditText username,password;  
 Toast message;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_login\_screen*);  
  
 username = (EditText) findViewById(R.id.*UserName*);  
 password = (EditText) findViewById(R.id.*Password*);  
 }  
  
 public void authorise(View view) {  
 String appPassword = "meow", appUsername = "tom";  
  
 if(username.getText().toString().equals(appUsername)){  
 if(password.getText().toString().equals(appPassword)){  
 message = Toast.*makeText*(getApplicationContext(),"Logged In successfully",Toast.*LENGTH\_SHORT*);  
 Intent intent = new Intent(this, ChoiceScreen.class);  
 startActivity(intent);  
 }else{  
 message = Toast.*makeText*(getApplicationContext(),"Password incorrect",Toast.*LENGTH\_SHORT*);  
 }  
 } else {  
 message = Toast.*makeText*(getApplicationContext(),"Username does not exist",Toast.*LENGTH\_SHORT*);  
 }  
 message.show();  
 }  
}

## Home Task 1 Java Code – Advanced Talking to itself (ChoiceScreen.java)

import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
  
public class ChoiceScreen extends AppCompatActivity {  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_choice\_screen*);  
 }  
  
 public void openSum(View view) {  
 Intent intent = new Intent(this, MainActivity.class);  
 startActivity(intent);  
 }  
  
 public void openDice(View view) {  
 Intent intent = new Intent(this, DiceRoller.class);  
 startActivity(intent);  
 }  
}

## Home Task 1 Java Code – Advanced Talking to itself (MainActivity.java)

import androidx.appcompat.app.AppCompatActivity;  
  
import android.os.Bundle;  
import android.text.method.ScrollingMovementMethod;  
import android.util.Log;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.TextView;  
  
public class MainActivity extends AppCompatActivity {  
  
 TextView outputReceiver;  
 EditText inputFetcher;  
 Button buttonSend;  
 Button buttonKill;  
 String receivedText;  
 String[] filteredInput;  
 Integer total = 0;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 outputReceiver = (TextView) findViewById(R.id.*myTextViewBox*);  
 inputFetcher = (EditText) findViewById(R.id.*myEditTextBox*);  
 buttonSend = (Button) findViewById(R.id.*mySendBtn*);  
 buttonKill = (Button) findViewById(R.id.*myKillBtn*);  
  
 outputReceiver.setMovementMethod(new ScrollingMovementMethod());  
 }  
  
 public void sendMethod(View view) {  
 receivedText = inputFetcher.getText().toString();  
 filteredInput = receivedText.split("[ ]{1,}");  
 for (String number:filteredInput  
 ) {  
 total += Integer.*parseInt*(number);  
 }  
 outputReceiver.setText("Sum = " + total.toString());  
 total = 0;  
 }  
  
 public void killMethod(View view) {  
 finish();  
 onDestroy();  
 }  
}

## Home Task 1 Java Code – Advanced Talking to itself (DiceRoller.java)

import androidx.appcompat.app.AppCompatActivity;  
  
import android.annotation.SuppressLint;  
import android.os.Bundle;  
import android.util.Log;  
import android.view.View;  
import android.widget.CompoundButton;  
import android.widget.EditText;  
import android.widget.Switch;  
import android.widget.TextView;  
  
import java.util.ArrayList;  
import java.util.Random;  
  
public class DiceRoller extends AppCompatActivity {  
  
 TextView rolledValue, betResult;  
 EditText playerBet, playerGuess;  
 @SuppressLint("UseSwitchCompatOrMaterialCode") // support in lower version of android  
 Switch cheatGame;  
  
 boolean givePriority = true;  
 Integer[] priorityArray = {1,1,95,1,1,1}; // total = 100 to make it easier (out of 100%)  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_dice\_roller*);  
  
 rolledValue = (TextView) findViewById(R.id.*displayRolledValue*);  
 betResult = (TextView) findViewById(R.id.*displayResult*);  
 cheatGame = (Switch) findViewById(R.id.*cheater*);  
 playerBet = (EditText) findViewById(R.id.*userBet*);  
 playerGuess = (EditText) findViewById(R.id.*userGuess*);  
  
 cheatGame.setChecked(givePriority);  
  
 // switching whether to cheat or not during runtime  
 cheatGame.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener() {  
 @Override  
 public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {  
 givePriority = isChecked;  
 }  
 });  
 }  
  
 public void rollDice(View view) {  
 // calculate and get the rolled number  
 String result = calculateRoll(givePriority, priorityArray), finalMessage;  
  
 // messages to show  
 String message = "You have rolled: " + result,  
 winMessage = "Congratulation you have won: $",  
 lossMessage = "Oh no! you lost $";  
  
 // check if player guessed correct and set message accordingly  
 if(playerGuess.getText().toString().equals(result)){  
 finalMessage = winMessage + String.*valueOf*(Integer.*parseInt*(playerBet.getText().toString()) \* 2);  
 }else {  
 finalMessage = lossMessage + playerBet.getText().toString();  
 }  
  
 // display messages  
 rolledValue.setText(message);  
 betResult.setText(finalMessage);  
 }  
  
 private String calculateRoll(boolean priority, Integer[] priorityArray) {  
 int rolledNumber = 0, counter = 0, currentNumber = 1;  
 ArrayList<Integer> randomPool = new ArrayList<>();  
  
 // generate and store a number pool from where we will select a random number  
 for (int x:priorityArray  
 ) {  
 counter = x;  
 while(counter != 0){  
 randomPool.add(currentNumber);  
 counter--;  
 Log.*d*("Created random pool",String.*valueOf*(currentNumber));  
 }  
 currentNumber++;  
 }  
  
 // deciding whether to use priority or not to generate new number  
 if (!priority) {  
 rolledNumber = new Random().nextInt(6) + 1;  
 } else {  
 int index = new Random().nextInt(100);  
 rolledNumber = randomPool.get(index);  
 }  
  
 // returning the generated numebr  
 return String.*valueOf*(rolledNumber);  
 }  
}

## Home Task 1 XML Script – Advanced Talking to itself (activity\_login\_screen.xml)

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout 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"  
 tools:context=".LoginScreen">  
  
 <EditText  
 android:id="@+id/UserName"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:hint="@string/username"  
 android:inputType="textPersonName"  
 app:layout\_constraintBottom\_toTopOf="@+id/Password"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.7" />  
  
 <EditText  
 android:id="@+id/Password"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:hint="@string/passwordHint"  
 android:inputType="textPassword"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.3" />  
  
 <Button  
 android:id="@+id/button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/log\_in"  
 android:onClick="authorise"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/UserName"  
 app:layout\_constraintVertical\_bias="0.3" />  
  
  
</androidx.constraintlayout.widget.ConstraintLayout>

## Home Task 1 XML Script – Advanced Talking to itself (activity\_choice\_screen.xml)

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout 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"  
 tools:context=".ChoiceScreen">  
  
 <Button  
 android:id="@+id/button2"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/sum\_finder"  
 android:onClick="openSum"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.33" />  
  
 <Button  
 android:id="@+id/button3"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/dice\_roller"  
 android:onClick="openDice"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/button2"  
 app:layout\_constraintVertical\_bias="0.33"/>  
</androidx.constraintlayout.widget.ConstraintLayout>

## Home Task 1 XML Script – Advanced Talking to itself (activity\_main.xml)

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout  
 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"  
 tools:context=".MainActivity">  
  
 <TextView  
 android:id="@+id/myTextViewBox"  
 android:layout\_width="300dp"  
 android:layout\_height="150dp"  
 android:background="#00BCD4"  
 android:padding="5dp"  
 android:scrollbars="vertical"  
 android:text="@string/textView"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.10" />  
  
 <EditText  
 android:id="@+id/myEditTextBox"  
 android:layout\_width="300dp"  
 android:layout\_height="50dp"  
 android:background="#8BC34A"  
 android:hint="@string/editText"  
 android:inputType="textPersonName"  
 android:padding="5dp"  
 android:singleLine="true"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/myTextViewBox"  
 app:layout\_constraintVertical\_bias="0.05" />  
  
 <Button  
 android:id="@+id/myKillBtn"  
 android:layout\_width="100dp"  
 android:layout\_height="50dp"  
 android:onClick="killMethod"  
 android:text="@string/kill"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.75"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/myEditTextBox"  
 app:layout\_constraintVertical\_bias="0.05"  
 />  
  
 <Button  
 android:id="@+id/mySendBtn"  
 android:layout\_width="100dp"  
 android:layout\_height="50dp"  
 android:onClick="sendMethod"  
 android:text="@string/send"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.25"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/myEditTextBox"  
 app:layout\_constraintVertical\_bias="0.05"  
 />  
  
</androidx.constraintlayout.widget.ConstraintLayout>

## Home Task 1 XML Script – Advanced Talking to itself (activity\_dice\_roller.xml)

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout 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"  
 tools:context=".DiceRoller">  
  
 <TextView  
 android:id="@+id/displayRolledValue"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/dice\_roll"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.120000005" />  
  
 <Button  
 android:id="@+id/rollTheDice"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:onClick="rollDice"  
 android:text="@string/roll"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/displayRolledValue"  
 app:layout\_constraintVertical\_bias="0.6" />  
  
 <Switch  
 android:id="@+id/cheater"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/cheat"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/rollTheDice"  
 app:layout\_constraintVertical\_bias="0.26"  
 tools:ignore="UseSwitchCompatOrMaterialXml" />  
  
 <EditText  
 android:id="@+id/userBet"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:hint="@string/bet"  
 android:inputType="number"  
 app:layout\_constraintBottom\_toTopOf="@+id/rollTheDice"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/displayRolledValue"  
 app:layout\_constraintVertical\_bias="0.19999999" />  
  
 <TextView  
 android:id="@+id/displayResult"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/status"  
 app:layout\_constraintBottom\_toTopOf="@+id/rollTheDice"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/userGuess"  
 app:layout\_constraintVertical\_bias="0.48000002" />  
  
 <EditText  
 android:id="@+id/userGuess"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:hint="@string/guess"  
 android:inputType="number"  
 app:layout\_constraintBottom\_toTopOf="@+id/rollTheDice"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/userBet"  
 app:layout\_constraintVertical\_bias="0.26" />  
  
  
</androidx.constraintlayout.widget.ConstraintLayout>

## Home Task 1 – Advanced Talking to itself Manifest Script

<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.voidoflimbo.talkingtoyourself">  
  
 <application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:roundIcon="@mipmap/ic\_launcher\_round"  
 android:supportsRtl="true"  
 android:theme="@style/Theme.TalkingToYourself">  
 <activity android:name=".ChoiceScreen" />  
 <activity  
 android:name=".DiceRoller"  
 android:parentActivityName=".ChoiceScreen" />  
 <activity  
 android:name=".MainActivity"  
 android:parentActivityName=".ChoiceScreen" />  
 <activity  
 android:name=".LoginScreen"  
 android:noHistory="true">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
 </application>  
  
</manifest>

## Home Task 1 – Advanced Talking to itself (Strings.xml)

<resources>  
 <string name="app\_name">Talking To Yourself</string>  
 <string name="textView">[Use space to indicate separate numbers]</string>  
 <string name="editText">Enter your numbers here</string>  
 <string name="kill">Kill</string>  
 <string name="send">Send</string>  
 <string name="username">Username</string>  
 <string name="passwordHint">\*\*\*\*\*\*\*</string>  
 <string name="log\_in">LOG IN</string>  
 <string name="sum\_finder">Sum Finder</string>  
 <string name="dice\_roller">Dice Roller</string>  
 <string name="dice\_roll">Press Roll to roll the dice</string>  
 <string name="roll">Roll</string>  
 <string name="cheat">Cheat</string>  
 <string name="bet">Place your bet</string>  
 <string name="status">status</string>  
 <string name="guess">Place your guess here</string>  
</resources>

## Home Task 2 Java Code (WAVConverter.java)

import java.io.FileInputStream;  
import java.io.FileOutputStream;  
import java.io.IOException;  
  
public class WAVConverter {  
  
 // will store the data that will be used to make the raw data into wav format  
 private byte[] header, rawData;  
  
 //Sampling frequency  
 //44100 is the current standard, but some devices still support 22050，16000，11025  
 public final static int *AUDIO\_SAMPLE\_RATE* = 8000; // 44.1KHz,Commonly used frequency  
  
 // will return the byte array of header  
 public byte[] getHeader(){ return this.header;}  
  
 public byte[] getRawData(){ return this.rawData; }  
  
 WAVConverter(){}  
  
 // Here to get a playable audio file  
 public void convertPCMToWaveFile(String inFilename, String outFilename, int bufferSizeInBytes) {  
 FileInputStream in = null;  
 FileOutputStream out = null;  
 long totalAudioLen = 0, totalDataLen = 0;  
 long longSampleRate = *AUDIO\_SAMPLE\_RATE*;  
 int channels = 2, bitsPerSample = 16; // 1 = mono || 2 = stereo  
 long byteRate = (bitsPerSample \* longSampleRate \* channels) / 8;  
 rawData = new byte[bufferSizeInBytes];  
  
 // read from raw data and write it to wav file  
 try {  
 in = new FileInputStream(inFilename);  
 out = new FileOutputStream(outFilename);  
 totalAudioLen = in.getChannel().size();  
 totalDataLen = totalAudioLen + 44;  
  
 // add header to the wav file before audio data  
 WriteWaveFileHeader(out, totalAudioLen, totalDataLen,  
 longSampleRate, channels, byteRate);  
  
 // read data from one file and write it on top of another file.  
 while (in.read(rawData) != -1) {  
 out.write(rawData);  
 }  
 in.close();  
 out.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 */\*\*  
 \* Here is a header information. Insert this information to get a file that can be played.  
 \* The reason for creating 44 byte header was referenced from  
 \* Reference https://stackoverflow.com/questions/13039846/what-do-the-bytes-in-a-wav-file-represent  
 \* http://www.topherlee.com/software/pcm-tut-wavformat.html  
 \*/* private void WriteWaveFileHeader(FileOutputStream out, long totalAudioLen, long totalDataLen, long longSampleRate, int channels, long byteRate) throws IOException {  
 header = new byte[44];  
 // RIFF / WAVE header  
 header[0] = 'R'; header[1] = 'I'; header[2] = 'F'; header[3] = 'F';  
 header[4] = (byte) (totalDataLen & 0xff);  
 header[5] = (byte) ((totalDataLen >> 8) & 0xff);  
 header[6] = (byte) ((totalDataLen >> 16) & 0xff);  
 header[7] = (byte) ((totalDataLen >> 24) & 0xff);  
 header[8] = 'W'; header[9] = 'A'; header[10] = 'V'; header[11] = 'E';  
 header[12] = 'f'; header[13] = 'm'; header[14] = 't'; header[15] = ' '; // 'fmt ' chunk  
 header[16] = 16; // 4 bytes: size of 'fmt ' chunk  
 header[17] = 0; header[18] = 0; header[19] = 0;  
 header[20] = 1; // format = 1  
 header[21] = 0;  
 header[22] = (byte) channels;  
 header[23] = 0;  
 header[24] = (byte) (longSampleRate & 0xff);  
 header[25] = (byte) ((longSampleRate >> 8) & 0xff);  
 header[26] = (byte) ((longSampleRate >> 16) & 0xff);  
 header[27] = (byte) ((longSampleRate >> 24) & 0xff);  
 header[28] = (byte) (byteRate & 0xff);  
 header[29] = (byte) ((byteRate >> 8) & 0xff);  
 header[30] = (byte) ((byteRate >> 16) & 0xff);  
 header[31] = (byte) ((byteRate >> 24) & 0xff);  
 header[32] = (byte) (2 \* 16 / 8); // block align  
 header[33] = 0;  
 header[34] = 16; // bits per sample  
 header[35] = 0;  
 header[36] = 'd'; header[37] = 'a'; header[38] = 't'; header[39] = 'a';  
 header[40] = (byte) (totalAudioLen & 0xff);  
 header[41] = (byte) ((totalAudioLen >> 8) & 0xff);  
 header[42] = (byte) ((totalAudioLen >> 16) & 0xff);  
 header[43] = (byte) ((totalAudioLen >> 24) & 0xff);  
 out.write(header, 0, 44);  
 out.flush();  
 }  
}

## Home Task 2 Java Code (WAVAudioRecorder.java)

import android.content.Context;  
import android.media.AudioFormat;  
import android.media.AudioRecord;  
import android.media.MediaRecorder;  
  
import java.io.File;  
import java.io.FileOutputStream;  
import java.io.IOException;  
  
public class WAVAudioRecorder {  
  
 // Buffer size in bytes  
 private int bufferSizeInBytes = 0;  
  
 private byte[] audioData;  
  
 private boolean isRecord = false;  
 private AudioRecord recorder;  
  
 WAVAudioRecorder(){}  
  
 WAVConverter converter = new WAVConverter();  
  
 // Raw audio data file from microphone & Playable audio file  
 private String RawFileAddress = "", WaveFileAddress = "";  
  
 public String getWAVFileAddress() { return this.WaveFileAddress; }  
  
 public void startRecording(Context context) {  
 if (AudioFileManager.*isSdcardExit*()) {  
 if (!isRecord) {  
 if (recorder == null) {  
 installRecorder(context);  
 }  
  
 recorder.startRecording();  
 // Let the recording status be true  
 isRecord = true;  
 //Start the audio file writing thread  
 new Thread(new AudioRecordThread()).start();  
  
 }  
 }  
 }  
  
 public void stopRecording() {  
 if (recorder != null) {  
 System.*out*.println("stopped Recording");  
 isRecord = false;//Stop file writing  
 recorder.stop();  
 recorder.release();//Release resources  
 recorder = null;  
 } else {  
 System.*out*.println("nothing is being recorded");  
 }  
 }  
  
 private void installRecorder(Context context) {  
 // create both raw and WAV files  
 prepareFiles(context);  
  
 // Get the size of the buffer in bytes  
 bufferSizeInBytes = AudioRecord.*getMinBufferSize*(  
 WAVConverter.*AUDIO\_SAMPLE\_RATE*,  
 AudioFormat.*CHANNEL\_IN\_STEREO*,  
 AudioFormat.*ENCODING\_PCM\_16BIT* );  
  
 System.*out*.println("buffer size: " + bufferSizeInBytes);  
  
 // Create AudioRecord object  
 recorder = new AudioRecord(  
 MediaRecorder.AudioSource.*MIC*,  
 WAVConverter.*AUDIO\_SAMPLE\_RATE*,  
 AudioFormat.*CHANNEL\_IN\_STEREO*,  
 AudioFormat.*ENCODING\_PCM\_16BIT*,  
 bufferSizeInBytes );  
 }  
  
 private void prepareFiles(Context context) {  
 RawFileAddress = AudioFileManager.*getRawFilePath*(context);  
 WaveFileAddress = AudioFileManager.*getWavFilePath*(context);  
 }  
  
 private class AudioRecordThread implements Runnable{  
 @Override  
 public void run() {  
 writeDataTOFile(); // Write raw data to the file  
 converter.convertPCMToWaveFile(RawFileAddress, WaveFileAddress,bufferSizeInBytes); // Add header files to raw data  
 }  
 }  
  
 */\*\*  
 \* The data is written into the file here, but it cannot be played, because the audio obtained by AudioRecord is the original raw audio.  
 \*/* private void writeDataTOFile() {  
 // new A byte array is used to store some byte data, the size is the buffer size  
 audioData = new byte[bufferSizeInBytes];  
 FileOutputStream fos = null;  
 int readSize;  
 try {  
 File file = new File(RawFileAddress);  
 if (file.exists()) { file.delete(); }  
 fos = new FileOutputStream(file);// Create a file with accessible bytes  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 while (isRecord) {  
 readSize = recorder.read(audioData, 0, bufferSizeInBytes);  
 if (AudioRecord.*ERROR\_INVALID\_OPERATION* != readSize && fos!=null) {  
 try {  
 fos.write(audioData);  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 try {  
 if(fos != null)  
 fos.close();// Close write stream  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
}

## Home Task 2 Java Code (AudioFileManager.java)

import android.content.Context;  
import android.content.ContextWrapper;  
import android.os.Environment;  
  
import java.io.File;  
  
public class AudioFileManager {  
  
 //Recording output file  
 private final static String *AUDIO\_RAW\_FILENAME* = "test.raw";  
 private final static String *AUDIO\_WAV\_FILENAME* = "finalAudio.wav";  
 */\*\*  
 \* Determine whether there is an external storage device sdcard  
 \* returns true | false  
 \*/* public static boolean isSdcardExit(){  
 return (Environment.*getExternalStorageState*().equals(android.os.Environment.*MEDIA\_MOUNTED*) || !Environment.*isExternalStorageRemovable*());  
 }  
  
 */\*\*  
 \* Get the original audio stream file path of the microphone input  
 \*/* public static String getRawFilePath(Context context){  
 ContextWrapper contextWrapper = new ContextWrapper(context);  
 File musicDirectory = contextWrapper.getExternalFilesDir(Environment.*DIRECTORY\_MUSIC*);  
 File file = new File(musicDirectory, *AUDIO\_RAW\_FILENAME*);  
 return file.getPath();  
 }  
  
 */\*\*  
 \* Get the encoded WAV format audio file path  
 \*/* public static String getWavFilePath(Context context){  
 ContextWrapper contextWrapper = new ContextWrapper(context);  
 File musicDirectory = contextWrapper.getExternalFilesDir(Environment.*DIRECTORY\_MUSIC*);  
 File file = new File(musicDirectory, *AUDIO\_WAV\_FILENAME*);  
 return file.getPath();  
 }  
}

## Home Task 2 Java Code (ErrorManager.java)

import android.content.Context;  
import android.content.res.Resources.NotFoundException;  
  
  
public class ErrorManager {  
 public final static int *SUCCESS* = 1000;  
 public final static int *E\_NOSDCARD* = 1001;  
 public final static int *E\_STATE\_RECODING* = 1002;  
 public final static int *E\_UNKOWN* = 1003;  
  
  
 public static String getErrorInfo(Context vContext, int vType) throws NotFoundException  
 {  
 switch(vType)  
 {  
 case *SUCCESS*:  
 return "success";  
 case *E\_NOSDCARD*:  
 return vContext.getResources().getString(R.string.*error\_no\_sdcard*);  
 case *E\_STATE\_RECODING*:  
 return vContext.getResources().getString(R.string.*error\_state\_record*);  
 case *E\_UNKOWN*:  
 default:  
 return vContext.getResources().getString(R.string.*error\_unknown*);  
  
 }  
 }  
}

## Home Task 2 Java Code (LoadScreen.java)

import androidx.annotation.NonNull;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.core.app.ActivityCompat;  
  
import android.Manifest;  
import android.content.Context;  
import android.content.Intent;  
import android.content.pm.PackageManager;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.Toast;  
  
public class LoadScreen extends AppCompatActivity {  
  
 private static final int *PERMISSION\_CODE* = 100;  
 Button TryApp;  
 private String[] AppPermissions;  
   
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_load\_screen*);  
  
 TryApp = findViewById(R.id.*button\_try\_it*);  
 AppPermissions = new String[]{  
 Manifest.permission.*INTERNET*,  
 Manifest.permission.*ACCESS\_NETWORK\_STATE*,  
 Manifest.permission.*READ\_EXTERNAL\_STORAGE*,  
 Manifest.permission.*WRITE\_EXTERNAL\_STORAGE*,  
 Manifest.permission.*RECORD\_AUDIO* };  
 }  
  
 private boolean checkPermission(Context context, String... appPermissions){  
 for(String singlePermission:appPermissions){  
 if(ActivityCompat.*checkSelfPermission*(context,singlePermission) != PackageManager.*PERMISSION\_GRANTED*){  
 System.*out*.println("permission not granted");  
 return false;  
 }  
 }  
 System.*out*.println(" all permissions granted");  
 return true;  
 }  
  
 @Override  
 public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {  
 super.onRequestPermissionsResult(requestCode, permissions, grantResults);  
  
 int index = 0;  
 if(requestCode == *PERMISSION\_CODE*){  
 for(int grantResult : grantResults){  
 String permissionName = permissions[index];  
 permissionName = permissionName.substring(permissionName.lastIndexOf(".")+1);  
 if(grantResult == PackageManager.*PERMISSION\_GRANTED*){  
 Toast.*makeText*(this, permissionName + " permission has been Granted" , Toast.*LENGTH\_SHORT*).show();  
 } else{  
 Toast.*makeText*(this, permissionName + " permission has been Denied" , Toast.*LENGTH\_SHORT*).show();  
 }  
 index++;  
 }  
 }  
 }  
  
 public void openApp(View view) {  
 if(!checkPermission(this, AppPermissions)){  
 ActivityCompat.*requestPermissions*(this,AppPermissions,*PERMISSION\_CODE*);  
 } else {  
 Intent recordScreenActivity = new Intent(this, RecordScreen.class);  
 startActivity(recordScreenActivity);  
 }  
 }  
}

## Home Task 2 Java Code (RecordScreen.java)

import androidx.appcompat.app.AppCompatActivity;  
  
import android.app.ProgressDialog;  
import android.content.Intent;  
import android.media.AudioAttributes;  
import android.media.MediaPlayer;  
import android.net.Uri;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.ImageButton;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import com.google.firebase.storage.FirebaseStorage;  
import com.google.firebase.storage.StorageReference;  
  
import java.io.File;  
import java.io.IOException;  
import java.util.Calendar;  
  
public class RecordScreen extends AppCompatActivity {  
  
 private boolean audioRecorded = false;  
 private StorageReference storageReference;  
 private ProgressDialog progressDialog;  
  
 ImageButton recordButton, playButton;  
 Button recordNew, sendAudio, openDownloader;  
 TextView recordingStatus, gpSimulationStatus, orSeparator;  
 String finalFilename;  
  
 MediaPlayer mediaPlayer;  
  
 WAVAudioRecorder recorder;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_record\_screen*);  
  
 recordButton = findViewById(R.id.*buttonRecordAudio*);  
 playButton = findViewById(R.id.*buttonPlayAudio*);  
 recordNew = findViewById(R.id.*buttonRecordNew*);  
 sendAudio = findViewById(R.id.*buttonSendAudio*);  
 openDownloader = findViewById(R.id.*buttonOpenDownloader*);  
 recordingStatus = findViewById(R.id.*recordingStatus*);  
 gpSimulationStatus = findViewById(R.id.*gpSimulationLabel*);  
 orSeparator = findViewById(R.id.*orSeparator*);  
  
 recorder = new WAVAudioRecorder();  
 storageReference = FirebaseStorage.*getInstance*().getReference();  
 progressDialog = new ProgressDialog(this);  
  
 playButton.setImageResource(R.drawable.*ic\_baseline\_play\_arrow\_24*);  
 recordButton.setImageResource(R.drawable.*ic\_baseline\_record\_voice\_over\_24*);  
  
 playButton.setVisibility(View.*INVISIBLE*);  
 sendAudio.setVisibility(View.*INVISIBLE*);  
 recordNew.setVisibility(View.*INVISIBLE*);  
  
 }  
  
 public void micPressed(View view) {  
 if(!audioRecorded){  
 // record audio  
 recordingStatus.setText(R.string.*recording\_started*);  
 recorder.startRecording(getApplicationContext());  
 audioRecorded = true;  
 openDownloader.setVisibility(View.*INVISIBLE*);  
 gpSimulationStatus.setVisibility(View.*INVISIBLE*);  
 recordButton.setImageResource(R.drawable.*ic\_baseline\_stop\_24*);  
 } else {  
 // stop recording audio  
 recordingStatus.setText(R.string.*recording\_stopped*);  
 playButton.setVisibility(View.*VISIBLE*);  
 recordButton.setVisibility(View.*INVISIBLE*);  
 recorder.stopRecording();  
 prepareAudio();  
 sendAudio.setVisibility(View.*VISIBLE*);  
 recordNew.setVisibility(View.*VISIBLE*);  
 }  
 orSeparator.setVisibility(View.*INVISIBLE*);  
 System.*out*.println("audioRecorded: " + audioRecorded);  
 }  
  
 public void playPressed(View view) {  
 recordingStatus.setText(R.string.*testingAudio*);  
 if(mediaPlayer.isPlaying()){  
 mediaPlayer.pause();  
 playButton.setImageResource(R.drawable.*ic\_baseline\_play\_arrow\_24*);  
 System.*out*.println("Audio Paused");  
 }else{  
 playButton.setImageResource(R.drawable.*ic\_baseline\_pause\_24*);  
 System.*out*.println("Audio Playing");  
 mediaPlayer.start();  
 }  
 mediaPlayer.setOnCompletionListener(mp -> playButton.setImageResource(R.drawable.*ic\_baseline\_play\_arrow\_24*));  
 }  
  
 public void recordAgain(View view) {  
 playButton.setVisibility(View.*INVISIBLE*);  
 recordButton.setVisibility(View.*VISIBLE*);  
 audioRecorded = false;  
 recordingStatus.setText(R.string.*recording\_notification*);  
 sendAudio.setVisibility(View.*INVISIBLE*);  
 recordNew.setVisibility(View.*INVISIBLE*);  
 recordButton.setImageResource(R.drawable.*ic\_baseline\_record\_voice\_over\_24*);  
  
 }  
  
 private void prepareAudio(){  
 mediaPlayer = new MediaPlayer();  
 mediaPlayer.setAudioAttributes(  
 new AudioAttributes  
 .Builder()  
 .setContentType(AudioAttributes.*CONTENT\_TYPE\_MUSIC*)  
 .build());  
 try {  
 mediaPlayer.setDataSource(recorder.getWAVFileAddress());  
 mediaPlayer.prepareAsync();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 mediaPlayer.setOnPreparedListener(mp -> {  
 Toast.*makeText*(getApplicationContext(),"Audio Saved",Toast.*LENGTH\_SHORT*).show();  
 recordingStatus.setText(R.string.*recording\_stopped*);  
 });  
 }  
  
 public void uploadAudioToFirebase(View view) {  
  
 progressDialog.setMessage("Uploading Audio");  
  
 progressDialog.show();  
  
 String filename = "User\_Audio\_";  
 filename += Calendar.*getInstance*().get(Calendar.*YEAR*) + "\_" +  
 Calendar.*getInstance*().get(Calendar.*MONTH*) + "\_" +  
 Calendar.*getInstance*().get(Calendar.*DATE*) + "\_" +  
 Calendar.*getInstance*().get(Calendar.*HOUR*) + "\_" +  
 Calendar.*getInstance*().get(Calendar.*MINUTE*) + "\_" +  
 Calendar.*getInstance*().get(Calendar.*SECOND*) + ".wav";  
 System.*out*.println(filename);  
  
 finalFilename = "voila.wav";  
// String finalFilename = filename; // it will create a separate files in server.  
  
 StorageReference filepath = storageReference.child("Audio").child(finalFilename);  
 Uri uri = Uri.*fromFile*(new File(recorder.getWAVFileAddress()));  
  
  
 filepath.putFile(uri).addOnSuccessListener(taskSnapshot -> {  
 progressDialog.dismiss();  
 Intent listScreenActivity = new Intent(getApplicationContext(), ListScreen.class);  
 startActivity(listScreenActivity);  
 });  
 }  
  
 public void openListScreen(View view) {  
 orSeparator.setVisibility(View.*INVISIBLE*);  
 Intent listScreenActivity = new Intent(getApplicationContext(), ListScreen.class);  
 startActivity(listScreenActivity);  
 }  
}

## Home Task 2 Java Code (ListScreen.java)

import android.app.AlertDialog;  
import android.app.DownloadManager;  
import android.content.Context;  
import android.content.ContextWrapper;  
import android.content.Intent;  
import android.media.MediaPlayer;  
import android.net.ConnectivityManager;  
import android.net.NetworkInfo;  
import android.os.Bundle;  
import android.os.Environment;  
import android.provider.Settings;  
import android.view.View;  
import android.widget.Button;  
import android.widget.Toast;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import com.google.firebase.storage.FirebaseStorage;  
import com.google.firebase.storage.StorageReference;  
  
import java.io.File;  
import java.io.IOException;  
  
public class ListScreen extends AppCompatActivity {  
  
 StorageReference storageReference, downloadFileName;  
 String filename,filepath;  
  
 Button playAudio,downloadAudio;  
 MediaPlayer mediaPlayer;  
 NetworkInfo activeNetworkInfo;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_list\_screen*);  
  
 downloadAudio = findViewById(R.id.*button\_download\_audio*);  
 playAudio = findViewById(R.id.*button\_play\_downloaded\_audio*);  
 playAudio.setVisibility(View.*INVISIBLE*);  
  
 storageReference = FirebaseStorage.*getInstance*().getReference("Audio");  
 storageReference.listAll()  
 .addOnSuccessListener(listResult -> {  
 // we have only one audio for all the time so only one for now  
 for (StorageReference item : listResult.getItems()) {  
 filename = item.getName();  
 }  
 });  
 }  
   
 public void downloadFromFirebase(View view) {  
 if (!isNetworkAvailable()) {  
 AlertDialog.Builder customAlert = new AlertDialog.Builder( this);  
 customAlert.setMessage("Please enable internet in order to download audio")  
 .setCancelable(true)  
 .setPositiveButton("Connect", (dialog, which) -> startActivity(new Intent(Settings.*ACTION\_WIFI\_SETTINGS*)))  
 .show();  
 System.*out*.println("i got called");  
 } else {  
 if(activeNetworkInfo != null && activeNetworkInfo.isConnected()){  
 downloadFileName = storageReference.child(filename);  
  
 System.*out*.println(filename);  
 System.*out*.println(downloadFileName);  
  
 removePreviousFile();  
  
 downloadFileName.getDownloadUrl().addOnSuccessListener(uri -> {  
 System.*out*.println(uri);  
 DownloadManager downloadManager = (DownloadManager) getBaseContext().getSystemService(Context.*DOWNLOAD\_SERVICE*);  
 DownloadManager.Request request = new DownloadManager.Request(uri);  
  
 request.setNotificationVisibility(DownloadManager.Request.*VISIBILITY\_VISIBLE\_NOTIFY\_COMPLETED*);  
 request.setDestinationInExternalFilesDir(getApplicationContext(), Environment.*DIRECTORY\_MUSIC*, filename);  
  
 downloadManager.enqueue(request);  
 }).addOnCompleteListener(task -> playAudio.setVisibility(View.*VISIBLE*));  
 } else {  
 Toast.*makeText*(this, "Internet not available please try again",Toast.*LENGTH\_SHORT*).show();  
 }  
 }  
 }  
  
 private void removePreviousFile() {  
 ContextWrapper contextWrapper = new ContextWrapper(this);  
 File musicDirectory = contextWrapper.getExternalFilesDir(Environment.*DIRECTORY\_MUSIC*);  
 File file = new File(musicDirectory, filename);  
 filepath = file.getPath();  
 if(file.exists()) file.delete();  
 }  
  
 public void playAudio(View view) {  
 mediaPlayer = new MediaPlayer();  
 try {  
 mediaPlayer.setDataSource(filepath);  
 mediaPlayer.prepare();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 mediaPlayer.start();  
 }  
  
 private boolean isNetworkAvailable() {  
 ConnectivityManager connectivityManager = (ConnectivityManager) getSystemService(Context.*CONNECTIVITY\_SERVICE*);  
 activeNetworkInfo = connectivityManager.getActiveNetworkInfo();  
  
 return activeNetworkInfo != null && activeNetworkInfo.isConnected();  
 }  
}

## Home Task 2 XML Scripts (activity\_load\_screen.xml)

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout 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"  
 tools:context=".LoadScreen">  
  
 <TextView  
 android:id="@+id/notice"  
 android:layout\_width="0sp"  
 app:layout\_constraintWidth\_percent="0.9"  
 android:layout\_height="wrap\_content"  
 android:text="@string/notice"  
 android:textAlignment="center"  
 android:textAppearance="@style/TextAppearance.AppCompat.Large"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.3" />  
  
 <Button  
 android:id="@+id/button\_try\_it"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/try\_it"  
 android:onClick="openApp"  
 android:textColor="@color/button\_text\_background"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/notice"  
 app:layout\_constraintVertical\_bias="0.2"  
 android:background="@drawable/app\_default\_buttons"  
 android:textAllCaps="false"/>  
  
</androidx.constraintlayout.widget.ConstraintLayout>

## Home Task 2 XML Scripts(activity\_record\_screen.xml)

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout 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"  
 tools:context=".RecordScreen">  
  
 <ImageButton  
 android:id="@+id/buttonRecordAudio"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="@drawable/round\_image\_button"  
 android:onClick="micPressed"  
 android:scaleType="center"  
 android:scaleX="3"  
 android:scaleY="3"  
 android:src="@drawable/ic\_baseline\_record\_voice\_over\_24"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.17" />  
  
 <TextView  
 android:id="@+id/recordingStatus"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/recording\_notification"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.49"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.3" />  
  
 <ImageButton  
 android:id="@+id/buttonPlayAudio"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:onClick="playPressed"  
 android:scaleType="center"  
 android:scaleX="2"  
 android:scaleY="2"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 android:background="@drawable/round\_image\_button"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.4"  
 app:srcCompat="@drawable/ic\_baseline\_play\_arrow\_24" />  
  
 <Button  
 android:id="@+id/buttonRecordNew"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:onClick="recordAgain"  
 android:textColor="@color/button\_text\_background"  
 android:text="@string/record\_new\_audio"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.55"  
 android:background="@drawable/app\_default\_buttons"  
 android:textAllCaps="false"/>  
  
 <Button  
 android:id="@+id/buttonSendAudio"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:onClick="uploadAudioToFirebase"  
 android:text="@string/submit\_audio"  
 android:textColor="@color/button\_text\_background"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/buttonRecordNew"  
 app:layout\_constraintVertical\_bias="0.2"  
 android:background="@drawable/app\_default\_buttons"  
 android:textAllCaps="false"/>  
  
 <Button  
 android:id="@+id/buttonOpenDownloader"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/open\_downloader"  
 android:onClick="openListScreen"  
 android:textColor="@color/button\_text\_background"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/buttonSendAudio"  
 app:layout\_constraintVertical\_bias="0.4"  
 android:background="@drawable/app\_default\_buttons"  
 android:textAllCaps="false"/>  
  
 <TextView  
 android:id="@+id/gpSimulationLabel"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:gravity="center\_horizontal"  
 android:text="@string/receive\_audio\_textview"  
 app:layout\_constraintBottom\_toTopOf="@+id/buttonOpenDownloader"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/buttonSendAudio"  
 app:layout\_constraintVertical\_bias="0.5" />  
  
 <TextView  
 android:id="@+id/orSeparator"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/or\_separator"  
 android:textSize="34sp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
</androidx.constraintlayout.widget.ConstraintLayout>

## Home Task 2 XML Scripts(activity\_list\_screen.xml)

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout 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"  
 tools:context=".ListScreen">  
  
 <Button  
 android:id="@+id/button\_download\_audio"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:onClick="downloadFromFirebase"  
 android:text="@string/download\_audio"  
 android:textColor="@color/button\_text\_background"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.2"  
 android:background="@drawable/app\_default\_buttons"  
 android:textAllCaps="false"/>  
  
 <Button  
 android:id="@+id/button\_play\_downloaded\_audio"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/play\_downloaded\_audio"  
 android:textColor="@color/button\_text\_background"  
 android:onClick="playAudio"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/button\_download\_audio"  
 app:layout\_constraintVertical\_bias="0.1"  
 android:background="@drawable/app\_default\_buttons"  
 android:textAllCaps="false"/>  
  
 <TextView  
 android:id="@+id/textView2"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/before\_downloading\_instruction"  
 app:layout\_constraintBottom\_toTopOf="@+id/button\_download\_audio"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.5" />  
  
</androidx.constraintlayout.widget.ConstraintLayout>

## Home Task 2 Manifest Script

<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.voidoflimbo.chatapplication">  
  
 <uses-permission android:name="android.permission.INTERNET" />  
 <uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />  
 <uses-permission android:name="android.permission.RECORD\_AUDIO" />  
 <uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE"/>  
 <uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE"/>  
  
 <application  
 android:allowBackup="true"  
 android:icon="@mipmap/appicon"  
 android:label="@string/app\_name"  
 android:roundIcon="@mipmap/appicon\_round"  
 android:supportsRtl="true"  
 android:theme="@style/Theme.ChatApplication">  
 <activity  
 android:name=".ListScreen"  
 android:label="@string/playlist\_title" />  
 <activity  
 android:name=".RecordScreen"  
 android:label="@string/record\_screen\_title"  
 android:noHistory="true"/>  
 <activity  
 android:name=".LoadScreen"  
 android:label="@string/load\_screen"  
 android:noHistory="true">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
 </application>  
  
</manifest>

## Home Task 2 String.xml

<resources>  
 <string name="app\_name">ChatApplication</string>  
 <string name="error\_no\_sdcard">No SD card, unable to store recording data</string>  
 <string name="error\_state\_record">Recording, please stop recording first</string>  
 <string name="error\_unknown">Unknown Error</string>  
 <string name="notice">In this demo we will upload audio to firebase server and receive it from any client</string>  
 <string name="try\_it">Let`s Go</string>  
 <string name="send\_audio">Send Audio</string>  
 <string name="playlist">Playlist</string>  
 <string name="playlist\_title">Playlist From Server</string>  
 <string name="record\_screen\_title">Record Screen</string>  
 <string name="recording\_notification">Start Recording</string>  
 <string name="record\_new\_audio">Record New Audio</string>  
 <string name="submit\_audio">Submit Audio</string>  
 <string name="recording\_stopped">Play to test your audio</string>  
 <string name="recording\_started">Stop Recording</string>  
 <string name="testingAudio">You can record audio again or submit this audio</string>download  
 <string name="download\_audio">Download Audio</string>  
 <string name="open\_downloader">Browse</string>  
 <string name="play\_downloaded\_audio">Play Downloaded Audio</string>  
 <string name="before\_downloading\_instruction">Make sure your internet is on before downloading</string>  
 <string name="receive\_audio\_textview">Fetch audio from Server</string>  
 <string name="load\_screen">URIP ITC Application</string>  
 <string name="or\_separator">---------------OR---------------</string>  
</resources>

# Appendix B

## Python Server Code

The python script used to test the communication between android client and server is:

# FirstServer (Python3) 2 july 2021 with 2 versions of receive\_from\_client function.

import socket

import time

def alt\_receive\_from\_client():

MTU =4096 ; cmsg = b"" ;

while True:

cmsg += Clientsocket.recv(MTU) # receive up to MTU bytes

if cmsg[-1] == 10: break # (break out of while clause)

# end of while clause

msg = cmsg.decode("utf-8)")

# end of if clause

return msg.rstrip("\n")

# Function to receive short or long message from client. Long messages can fill buffer many times.

def receive\_from\_client():

MTU = 8192; cmsg=b""; #MTU is buffer size, cmsg is for message as byte-string

while True:

msgs = Clientsocket.recv(MTU) # Receive up to MTU bytes

cmsg += msgs; # Build up message in bytestring (b"..") form

if len(msgs) < MTU :

msg=cmsg.decode("utf-8"); #Decode from bytestring to normal string

break

return msg.rstrip("\n")

# Function for converting between signed integer array & CSV string (myExample6)

def csvString\_to\_intList (csvS):

# Converts csvS to List of signed integers

strList = csvS.split(",")

A = [ ] # empty list

for i in range(0,len(strList)):

A.append(int(strList[i]))

return A

# end of function

# Create a TCP/IP (IPv4) socket object & bind it to an IP address & port number:

S = socket.socket(socket.AF\_INET,socket.SOCK\_STREAM)

S.bind(('localhost',1234)) # IP address is 'localhost'(127.0.0.1) & port is 1234

# Block & wait for incoming connection. Can have queue of up to up to 5 connect requests:

S.listen(5) ; # Make S a listening socket capable of holding a queue of up to 5 clients.

print("Waiting for a client to connect")

Clientsocket, address = S.accept() # Blocks until there is a client

# When client is accepted, we get a new socket object 'Clientsocket'

# and a 'tuple' holding the address of the new client:

print("Clientsocket = ",Clientsocket)

print(f"connection from {address} has been established")

Clientsocket.send(bytes("Welcome to the server\n","utf-8"))

msg = ""

while (msg[0:4] != "QUIT"):

msg = alt\_receive\_from\_client()

if len(msg)<100 :

print("From client:", msg)

else: print("From client: ",msg[0:300]," ...")

time.sleep(3);

if msg[0:8] == "REGISTER" :

clientuserName = msg[9:].rstrip() #rstrip removes spaces at end

Clientsocket.send(bytes( "Your userName is now registered with the server\n","utf-8"))

elif msg[0:8] == "PASSWORD":

password=msg[9:].rstrip()

Clientsocket.send(bytes( "Your password is now registered with the server\n","utf-8"))

elif msg[0:6] == "UPLOAD":

CSVstring=msg[7:].rstrip()

Clientsocket.send(bytes( "Your recording has been uploaded the server\n","utf-8"))

Arec = csvString\_to\_intList (CSVstring) ; Fs=16000;

print("Play copy of recording received by Server to check it is correct:\n")

from IPython import display as Dp

Dp.display(Dp.Audio(Arec,rate=Fs))

elif msg[0:4] == "QUIT":

Clientsocket.send(bytes("You will be disconnected from the server\n","utf-8"))

print("Disconnecting client:") ; Clientsocket.close()

else:

Clientsocket.send(bytes("You are still connected to the server\n","utf-8"))

print("Closing server socket"); S.close()

print("Goodbye")

# References

developers, a. (2021, August 02). *android developers*. Retrieved from android developers: https://developer.android.com/studio/write/image-asset-studio

developers, a. (2021, February 24). *android developers*. Retrieved from android developers: https://developer.android.com/topic/performance/threads

seanxiaoxiao. (2012, October 24). *stackoverflow*. Retrieved from stackoverflow: https://stackoverflow.com/questions/13039846/what-do-the-bytes-in-a-wav-file-represent