

# National College of Ireland

BSC In Computing

Software Development

2023/2024

Ruby Rapple Kehoe

X20382263

X20382263@student.ncirl.ie



## Voice Training Application Technical Report

## Contents

Executive Summary.....	3
1.0 Introduction .....	3
1.1. Background.....	3
1.2. Aims .....	4
1.3. Technology .....	4
1.4. Structure.....	5
2.0 System.....	5
2.1. Requirements .....	5
2.1.1. Functional Requirements .....	5
2.1.1.1. Requirement 1: Voice Training Practice.....	6
2.1.1.1.2. Description & Priority.....	6
2.1.1.1.3. Use Case: Voice Training Practice / Voice Input .....	6
2.1.1. Requirement 2: Pie chart of user's overall progress.....	8
2.1.2. Description & Priority.....	8
2.1.3. Use Case .....	8
2.1.3.1. Requirement 3: Contact Speech Therapist .....	10
2.1.3.1.2. Description & Priority.....	10
2.1.3.1.3. Use Case .....	10
2.1.3.4. Requirement 4: Journal.....	12
2.1.3.4.5. Description & Priority.....	12
2.1.3.4.6. Use Case: Journal & Rating .....	12
2.1.3.7. Requirement 5: Registration and Login .....	14
2.1.3.7.8. Description & Priority.....	14
2.1.3.7.9. Use Case .....	14
2.1.4. Data Requirements .....	15
2.1.5. User Requirements .....	15
2.1.6. Environmental Requirements .....	16
2.1.7. Usability Requirements .....	16
2.2. Design & Architecture .....	17
2.3. Implementation.....	17
2.4. Graphical User Interface (GUI) .....	29
2.5. Testing .....	31
3.0 Conclusions .....	43
4.0 Further Development or Research .....	43
5.0 Poster .....	45

6.0	References .....	45
7.0	Appendices.....	46
7.1.	Project Proposal .....	46
1.0	Objectives .....	48
2.0	Background .....	49
3.0	State of the Art .....	49
4.0	Technical Approach.....	50
5.0	Technical Details .....	50
6.0	Special Resources Required .....	51
7.0	Project Plan .....	51
8.0	Testing.....	53
7.2.	Ethics Approval Application (only if required).....	53
8.1.	Reflective Journals.....	53

## Executive Summary

My aims for this technical report are to document the different requirements and development strategies of my Voice Training Application project, to give a brief summary of what the application is, how it should function, the target audience. The application is for trans / non-binary people to change their voice to more of their own liking by practicing different sounds.

This report will cover my own motivations for the pursuing the project, the aims of the project hope to achieve, the tools that I will use to achieve those aims and all the requirements that are needed to meet for the project to achieve the aims in a professional manor. This report covers technologies and algorithms during the development cycle including GitHub, Android Studio, Google Firebase, jTransforms and AndroidPlot. The application will be thoroughly tested during and after development using testing libraries and manual means.

After the development cycle is finished the application will follow a thorough test plan and will be document in this report.

## 1.0 Introduction

### 1.1. Background

This project is rather personally to me as I am trans women myself, I remember the moment my voice dropped when I was younger and it was such a daunting experience, my voice dropped when I was at a school friend's birthday party and all the other kids would not stop pointing it out since then my voice was one of the main sources of my own depression when growing up, I know I am not the only person with this kind of experience which is why I went to set on helping with the same struggle as my own.

That story and the feelings gotten worse along with a recent experience with speech therapist has motivated me into developing an application to help those like me who might be in as bad of a place I was in during my childhood years.

### 1.2. Aims

The end all be all goal of the project is to create an application for android that allows a person who seeks to have a voice that is more in line with their own identity, 'voice training' the process that people who seek to change their voice is often carried out by transgender and non-binary individuals the reasoning why people decide to train their voices vary e.g to help with passing, improve mental health are the main reasoning, this process can possibly take a number of years and has plenty of factors to account for that vary from person to person.

To make the process of voice training more streamline the application will need to meet certain objects, those objectives being as follows:

- Use gathered knowledge from own voice training experience.
- Track progress of the user's voice.
- Display the progress in an understandable manor.
- Explain the information that is being displayed e.g frequency ranges for masculine and feminine sounding voices.

Reaching these objectives should help streamline the practice of voice training making the users goal more obtainable.

### 1.3. Technology

In pursuit of the completion of my project goals taking consideration of what technology should or should not be used can be a difficult thing to figure out, my train of thought of what would be best the best approach would to use a mix of technology I already know how to use alongside technology that I do not know how to use.

There are many technologies that are a part of the voice training application which I will talk about in detail but I believe it is best to start with one that I have the most experience with, that being the language I have being using over the past few years over various different modules which being the programming language Java which I have had a good past few years of practice at my time at NCI as previously mentioned. Java is well known for its robustness and versatility which has been proven on multiple occasions which means been seen as fairly reliable. Java is an object-oriented language which may help if the scale of my project were to grow over time.

Since the plan is to have the Voice Training Application run on an android device, I believe the best development environment for the project would be Android Studio as it one of the most popular choices for android developers which means there is a lot of tutorials and documentation that can help me master using it, learning to use Android Studio [1] will enable me to create GUI screens for the user to interact using their android device as well as providing a development environment to write Java methods needed for the project's success.

In actual development environments in work places not all functionalities are developed in house, some companies decide that an API could be used in place of having their own developers write a similar function the same is true for my own project, the main way other voice training apps show results to the user is by the use of a graph that shows the users vocal frequency and info regarding where the frequency falls into e.g masculine or feminine. In my

own research I have come across an API that should assist me in creating my own graph using vocal inputs, this API is called 'Android Plot'. [2]

To get voice input from the user there are quite a few packages in Android that can be used to get vocal input using the microphone on the users android device as well as using that input to get the frequency of their voice and other information of it such as how loud it is in decibels, the package that enables the application to get voice input is called 'Media' which can be found under "android.media" [3], in order to calculate the frequency of the voice inputs I'll be using a library I had come across in my own research called 'jTransforms' [4], this library uses an algorithm called Fast Fourier Transforms [5] on the vocal input in order to calculate the frequency.

In terms of a backend, I was initially stuck on what to use as some SQL hosting services that I had read good reviews about on Reddit were no longer free so I had consulted my supervisor Frances Sheridan about this and she suggested that I use Google Firebase. By implementing Firebase into my project, I am able to save user data and user credentials and encrypt the passwords via Firebase Authenticate which also comes with registration and login methods. Firebase is very much essential to the applications success as the application needs to keep track of the user's voice training progress over time. [6]

#### 1.4. Structure

This high-level report is structured into four main sections: Introduction, System, Conclusion and Further Development & Research. The most comprehensive section is the System section, as it goes into detail about the functional and non-functional requirements the application should meet, Design & Architecture of the application, Implementation, Graphical User Interface and Testing & Evaluation. Aspects of the project will be presented both through text and visual representation to deliver a comprehensive understanding of the applications logic and appearance.

## 2.0 System

### 2.1. Requirements

#### 2.1.1. Functional Requirements

The Below list is the set of requirements for the voice training application, the order of the requirements is also their priority i.e the top of the list is the most important and the bottom of the list is the least important in terms of actually having an application, the Registration and Login functionality is very important but the application still needs its features developed for it to function to some capacity.

- Voice Training Practice / Voice Input | Use Case 1:  
The application must be able to take in voice inputs so that the application can work out the frequency of the user's voice, this is by far the most important aspect of the application.
- Pie chart of user's overall progress | Use Case 2:  
To keep track of the users progress the application needs away of showing this to the user, through the use of a pie chart the application will be apply to display the users progress in an easy-to-read manor this functions priority is medium due to the work not being as heavy as voice input mainly.
- Contact Speech Therapist | Use Case 3:  
No other voice training application has this feature which makes this application

stand out, this feature is basically a button that when tapped would a user to a wait list to speak with a speech therapist speech therapy can be very useful in terms of trying to achieve one’s desired voice, this functionality does have a low priority due to level of work I do believe it work take to complete.

- Journal | Use Case 4:  
Another feature that is unique to my application is the idea of having a journal the use can use to talk about how they feel about their voice and or progress they have made so far, to keep
- Registration & Login | Use Case 5



Figure 1: Use Case Diagram of application

#### 2.1.1.1. Requirement 1: Voice Training Practice

#### 2.1.1.2. Description & Priority

So, for transgender and non-binary people who seek to change how their voice sounds so that it may sound more in line with their identity, this functionality is the most important of the applications feature as it is the core of the project without it there is no project.

#### 2.1.1.3. Use Case: Voice Training Practice / Voice Input

Sequence number: RUC-1

RUC – Requirement Use Case

#### Scope

The scope of the “Voice Training Practice” functionality is to cover the process which a user (the actor) would use this functionality of the application (the system), the user’s goal is to train their voice so that it may sound closer to want it to sound and or to sound closer to their gender identity.

## Description

This use case describes the process of how a user will use the “Voice Training Functionality” of the application so that may practice different sounds to help get their voice to a stage that they are more comfortable with.

## Use Case Diagram



## Flow Description

### Precondition

The application is in an idle state waiting for the users input.

### Activation

This use case starts when the voice training fragment is loaded.

### Main flow

1. The system identifies that the voice training fragment has loaded.

2. The user taps the explanation button to listen to a recording of the developer's explanation of how to explain.
3. The user taps the start recording button, 15 second timer will start till the application stops recording.
4. The user makes the sound to practice.
5. The first round ends after 15 seconds. (See A1)
6. Repeat steps 3 to 5 two more times.
7. The user taps the view results button to load the graph showing the frequencies over time for the session. (See E1)
8. The system goes into a wait state for further input.

#### **Alternate flow**

A1: User cancels their practice

5. The first round ends after 15 seconds.
6. The user presses the back button to go back to the home screen.

#### **Exceptional flow**

E1: Voice input is read as null

7. The user taps the view results button to load the graph showing the frequencies over time for the session.
8. The graph read all values as 0.

#### **Termination**

The user goes back to the home screen.

#### **Post condition**

The system goes into a wait state ready for the users next interaction.

**List further functional requirements here, using the same structure as for Requirement1.**

#### [2.1.1. Requirement 2: Pie chart of user's overall progress](#)

Sequence number: RUC-2

#### [2.1.2. Description & Priority](#)

The idea of the pie chart which uses the Android Plot API to show the users overall progress with their voice so it would their voice has been come more feminine or masculine over time depending on the voice's frequency over time, this is a very important part of the project as we need to show to the user that their training is paying and may get the user to keep with their training meaning they will use the application more often.

#### [2.1.3. Use Case](#)

##### **Scope**

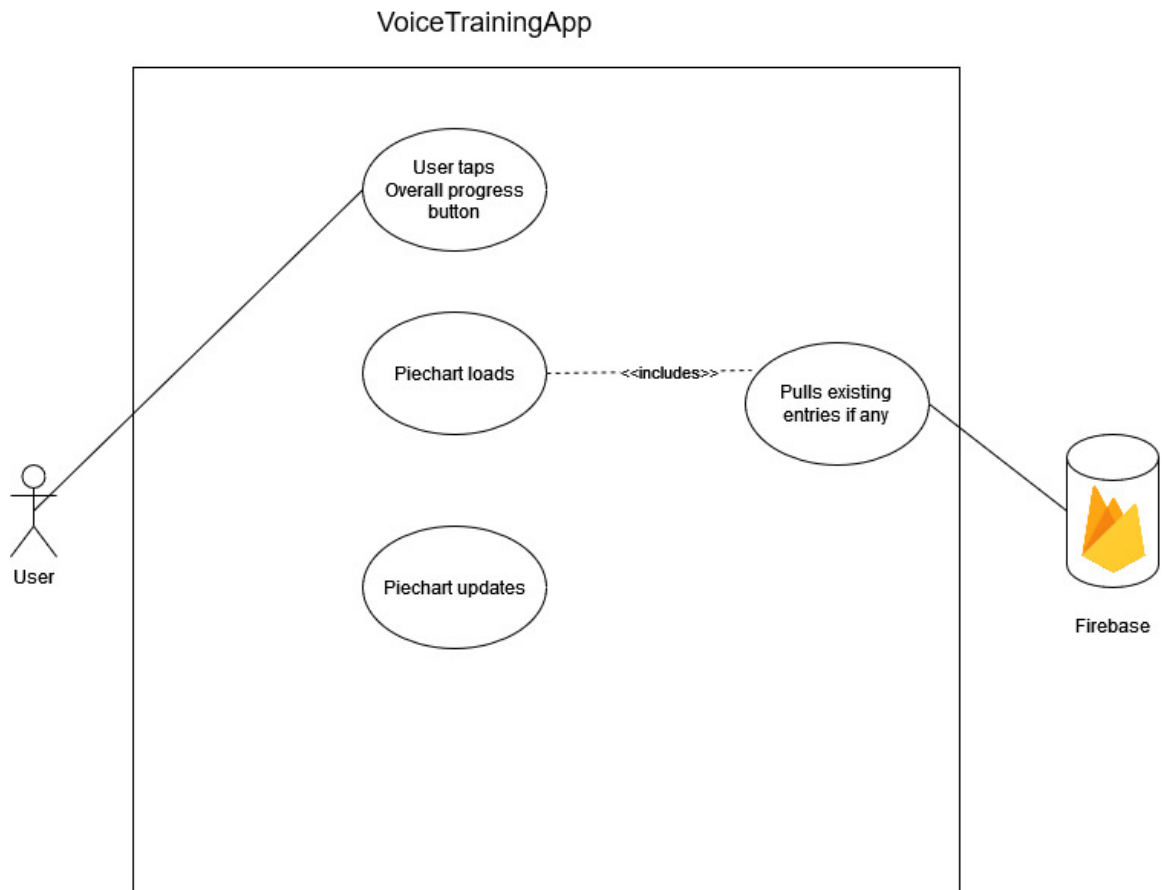
The scope of the "Pie chart of user's overall progress" functionality is to cover the process which a user (the actor) would use this functionality of the application (the system), the user's goal is to gain an idea of where their voice is currently at so that they can determine if their training is paying off. so that it may sound closer to want it to sound and or to sound closer to their gender identity.

##### **Description**

This use case describes how the graph that displays the users overall progress updates as the application is used more.



## Use Case Diagram



### Flow Description

#### Precondition

The system has record of the user having done voice training before.

#### Activation

This use case starts when a user taps the overall progress icon on the main screen.

#### Main flow

1. The system loads the overall progress activity and loads the piechart.
2. During loading, a Firebase query is done to pull all the frequencies associated with the users email address. (see A2 and E2)
3. The frequencies are sorted into masculine, feminine, androgynous values.
4. The pie chart is updated with the masculine, feminine, androgynous values show the percentage of each.

#### Alternate flow

- A2: User does has not done any previous voice training sessions.
2. During loading, a Firebase query is done to pull all the frequencies associated with the users email address.
  3. Firebase cannot find any frequencies associated with the users email address.
  4. The pie chart title updates suggesting to do voice training sessions and coming back and the pie chart is updated with place holder values.

#### Exceptional flow

- E2: Cannot connect to Firebase

2. During loading, a Firebase query is done to pull all the frequencies associated with the users email address.
3. Firebase query does not return anything.
4. The pie chart title updates suggesting to do voice training sessions and coming back and the pie chart is updated with place holder values.

**Termination**

The system terminates when the user goes back to the main screen.

**Post condition**

The system goes into a wait state ready for the users next interaction.

#### 2.1.3.1. [Requirement 3: Contact Speech Therapist](#)

Sequence number: RUC-3

#### 2.1.3.2. [Description & Priority](#)

This use case goes through how a user would get in touch with a speech therapist so that the user would have someone experienced to console about their issues with their voice. Overall, it is the third most important requirement due it being a unique functionality in comparison to competitors.

#### 2.1.3.3. [Use Case](#)

Sequence number: RUC-3

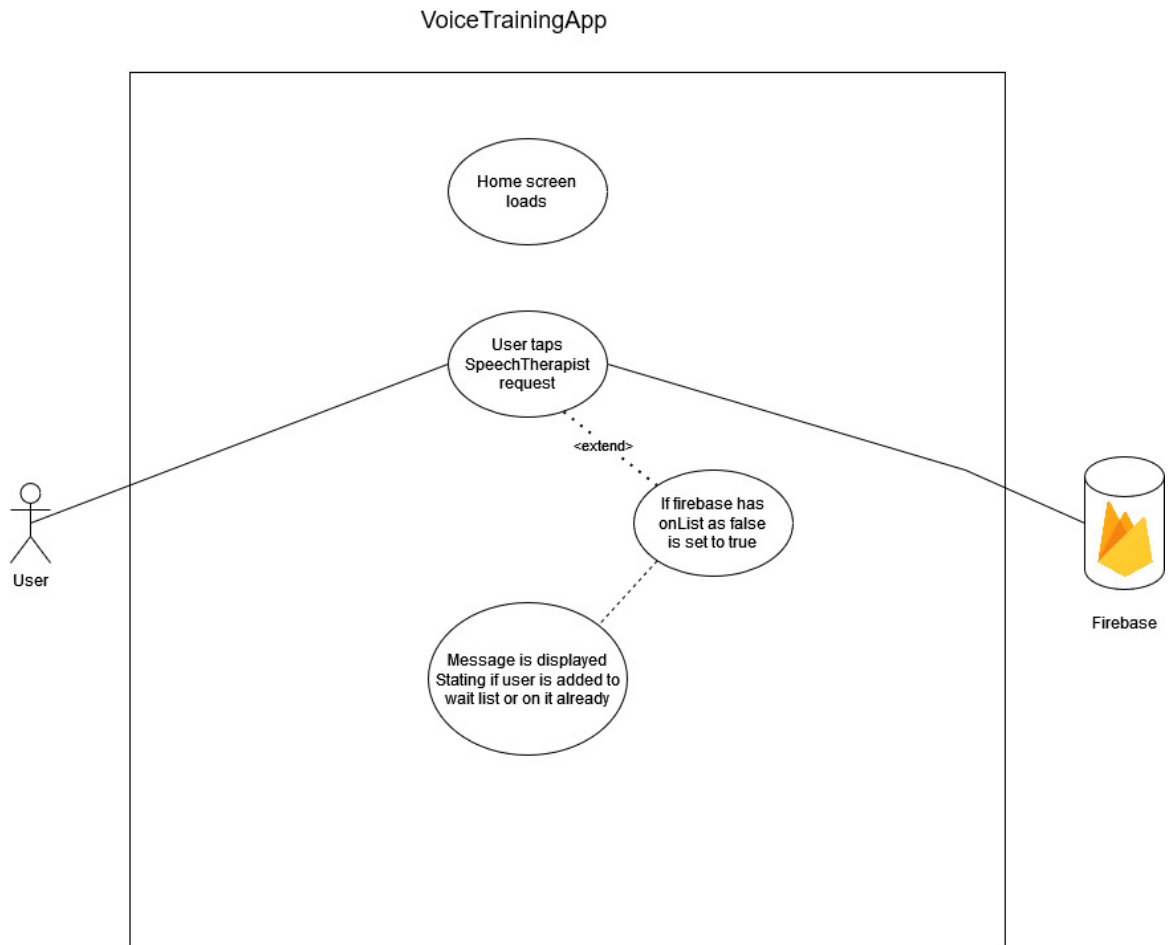
**Scope**

The scope of this use case is to demonstrate how a user (the actor) would get in touch with a speech therapist using the application.

**Description**

This use case describes the process of the Contact Speech Therapist function.

## Use Case Diagram



### Flow Description

#### Precondition

The system is idle waiting for user interaction.

#### Activation

This use case starts when the applications home screen has loaded.

#### Main flow

1. The system identifies the user has tapped the email icon.
2. The system query's Firebase to check if the onList Boolean is true indicating that the user is already on the wait list to see a speech therapist. (see A3).
3. Message is displayed stating they have been added to the list.

#### Alternate flow

##### A3:

2. The system query's Firebase to check if the onList Boolean is true indicating that the user is already on the wait list to see a speech therapist. (see A3).
3. Message is displayed stating they are on the list already.

#### Exceptional flow

N/A

#### Termination

The system goes into a wait state after the email is sent.

#### Post condition

The system goes into a idle state waiting for next user interaction.

#### 2.1.3.4. Requirement 4: Journal

Sequence number: RUC-4

#### 2.1.3.5. Description & Priority

The journal functional will allow users to talk about their own voice and reflect on how they feel, this may lead to further discovery of what the user likes and or dislikes about their voice or even think of inspiration of whom they may want to sound like. In terms of priority it is the second least important requirement as it is not required for the applications main function.

#### 2.1.3.6. Use Case: Journal & Rating

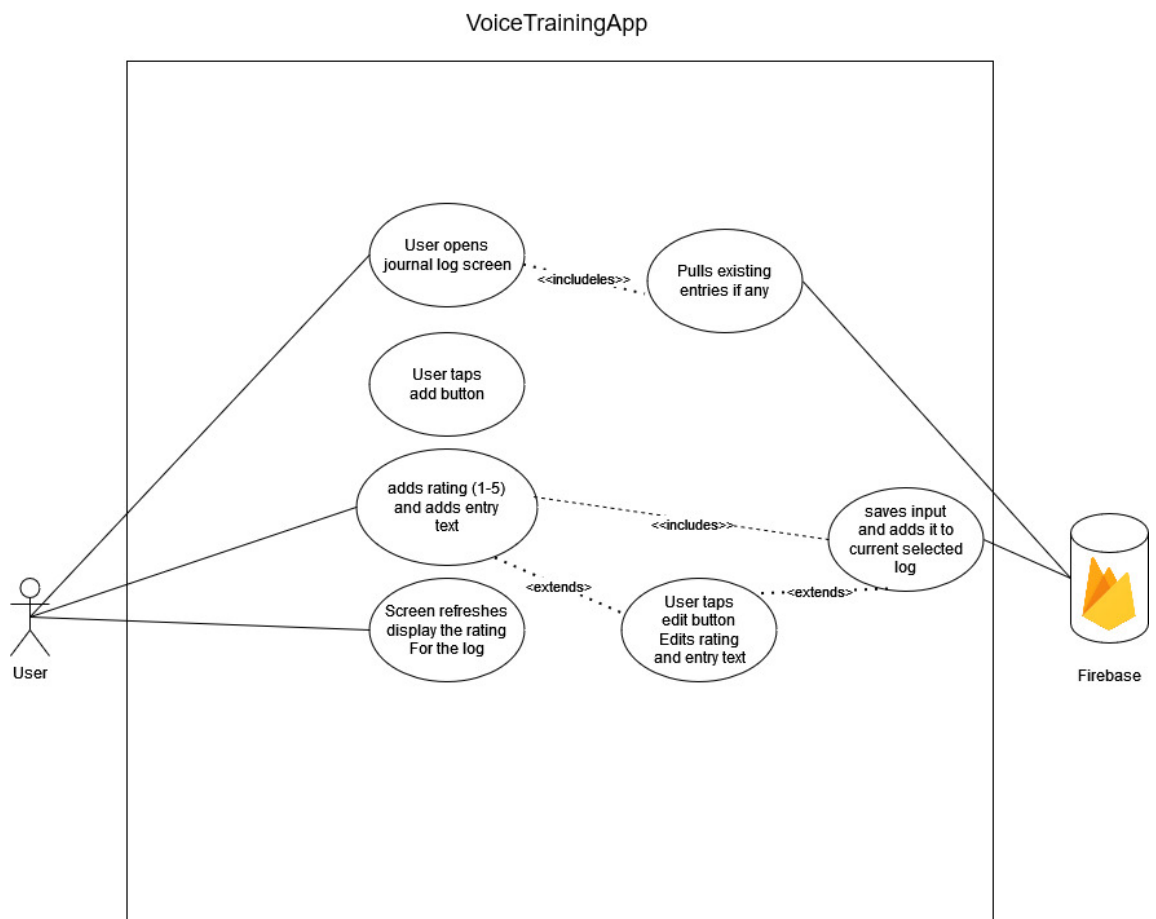
##### Scope

The scope of this use case is to demonstrate how the user would use the journal function.

##### Description

This use case describes the process of which a user takes to add a new entry to their journal.

##### Use Case Diagram



##### Flow Description

##### Precondition

The system is in an idle state.

### **Activation**

This use enters the journal screen.

### **Main flow**

1. The system loads and displays existing entries. (see A4)
2. The user taps the add button.
3. The user taps the entry input field, bringing up the systems keyboard and enters text. (See E5)
4. The user taps the rating input, bringing up the systems keyboard and chooses a number between 1 – 5.
5. The user taps the submit button
6. The system saves the result to Firebase.
7. The system takes the user back to the journal page displaying the new entry.

### **Alternate flow**

A4: User edits exiting entry

1. The system loads and couldn't find existing entries.
2. The user taps the add button.
3. The user taps the entry input field, bringing up the systems keyboard and enters text.
4. The user taps the rating input, bringing up the systems keyboard and chooses a number between 1 – 5.
5. The user taps the submit button
6. The system saves the result to Firebase.
7. The system takes the user back to the journal page displaying the new entry.
8. The user taps the new entry and taken to the entry edit page, the user adds more text and changes the rating to another valid input.
9. The system saves the result to Firebase.
10. The system takes the user back to the journal page displaying the new entry.

### **Exceptional flow**

E4: Inputs invalid

4. The user taps the entry input field, bringing up the systems keyboard and enters text over 250 characters.
5. The user taps the rating input, bringing up the systems keyboard and chooses a number not between 1 – 5.
6. The user taps the submit button.
7. The system gives respective error message about the inputs.
8. The user either reduces the amount of text and changes the rating input to be between 1 -5.
9. The user taps the submit button.
10. The system saves to Firebase.
11. The system takes the user back to the journal page displaying the new entry.

### **Termination**

The system adds the new entry.

### Post condition

The system refreshes the diary log screen showing the new entry.

### 2.1.3.7. Requirement 5: Registration and Login

Sequence number: RUC-5

### 2.1.3.8. Description & Priority

A description of the requirement and its priority. Describes how essential this requirement is to the overall system.

The registration and login functionalities are very important as they are used to create user accounts and create instances in order for saved to be kept upon reopening the app.

### 2.1.3.9. Use Case

Sequence number: RUC-5

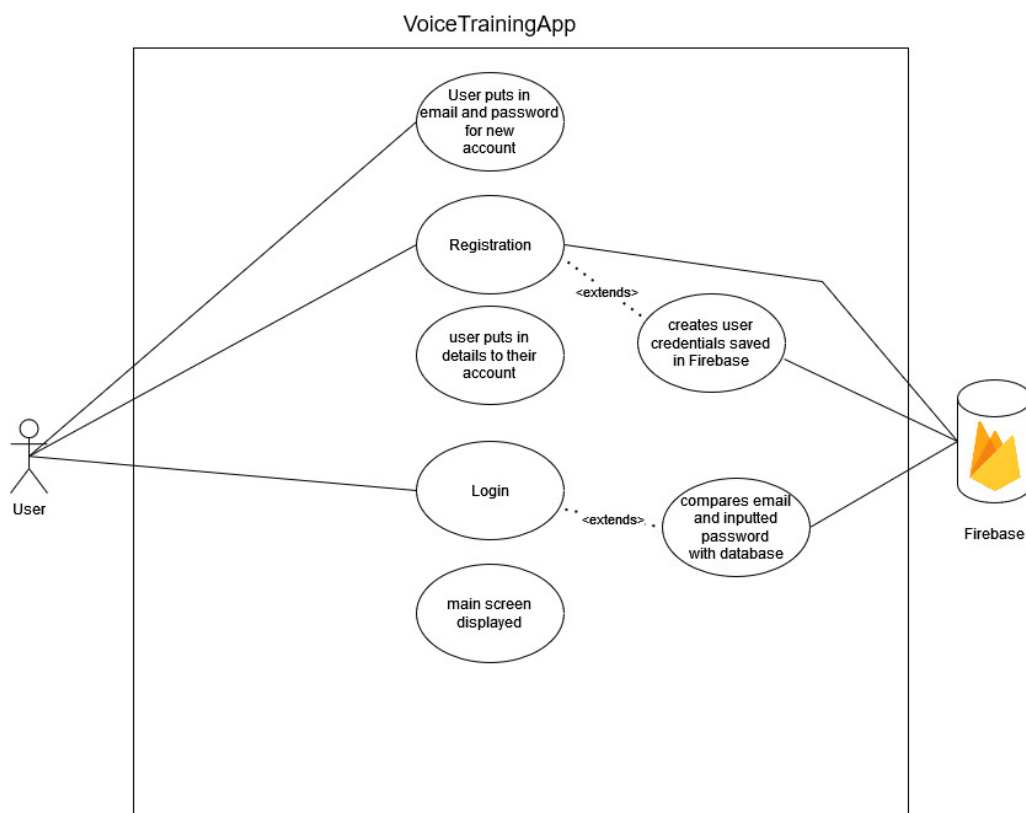
### Scope

The scope of this function is to cover the process where a user (the actor) would use this functionality to create an account and login into the account.

### Description

This use case describes the process of how a user will use the “Voice Training Functionality” of the application so that may practice different sounds to help get their voice to a stage that they are more comfortable with.

### Use Case Diagram



### Flow Description

### Precondition

The system is in initialisation mode at the registration screen.

### **Activation**

This use case starts when the user

### **Main flow**

1. The system identifies that the user tapped the rating icon. (See A5)
2. The system presents the numbers one through five, one meaning feeling the worst five meaning the best.
3. The system saves the rating (See E5)
4. The system refreshes the screen showing that rating has appeared beside the entry title.

### **Alternate flow**

A5: User cancels the rating

1. The system identifies that the user tapped the rating icon.
2. The user decides that they do not want to add a rating
3. The user taps the rating icon again to cancel.
4. The use case continues at position 1 of the main flow

### **Exceptional flow**

E5: Rating is not saved

1. The system identifies that the user tapped the rating icon.
2. The system presents the numbers one through five, one meaning feeling the worst five meaning the best.
3. The system does not save the rating, due to an exception.
4. The system refreshes the screen showing that rating has not appeared beside the entry title.
5. The use case continues at position 1 of the main flow

### **Termination**

The system presents the user with the main screen.

### **Post condition**

The system goes into a wait idle state waiting for the next user interaction.

## **2.1.4. Data Requirements**

This section will cover the Data requirements needed for the application to run properly the user's device should ideally have at least a gigabyte of storage for the application, as the application itself is fairly light but the application stores the audio files local which would build up over time as the user continually uses the application, in my testing of the prototype each audio file that I had created through the application was around ten kilobytes each, ensuring if the device has at least a gigabyte of storage for the application there should be a lot of room for voice training outputs.

## **2.1.5. User Requirements**

The user requirements of the voice training application describe the needs/expectations user must meet say that they may be able to operate the voice training application to its fullest potential.

1. After a day of total usage, experienced users should have the acquired experience needed to use all functions of the voice training application.
2. Experienced users must be able to navigate the application with ease.
3. Experienced users must stick to a consistent schedule in order to achieve breakthroughs from practice.
4. Experienced users should be able to practice at anytime and anywhere.
5. Experienced users must have the option to get in contact with a speech therapist.

#### 2.1.6. Environmental Requirements

The voice training application is built within Android version 8.1.1 and runs on Java 11, since these are older versions of Android and Java the vast majority of devices will be able to run the application without fear of any of the libraries or plugins failing. Additionally, the application will need to access the users' Android devices' microphone in order to function for the voice training practice function as intended, the application will.

#### 2.1.7. Usability Requirements

After observing other voice training applications that are available on the Google play store (which act as my competitors) I have come up with the following Usability Requirements.

1. The GUI must be clear and concise so that a new user can understand what everything on screen means without much explanation.
2. The application must preserve / remember the users' previous results.
3. The application must ask the user for permission to use their Android device's microphone.
4. The user must be able to tell if they are making progress based off the application's functions.
5. The application must not have a significant strain on the device's resources (battery, CPU,



## 2.2. Design & Architecture

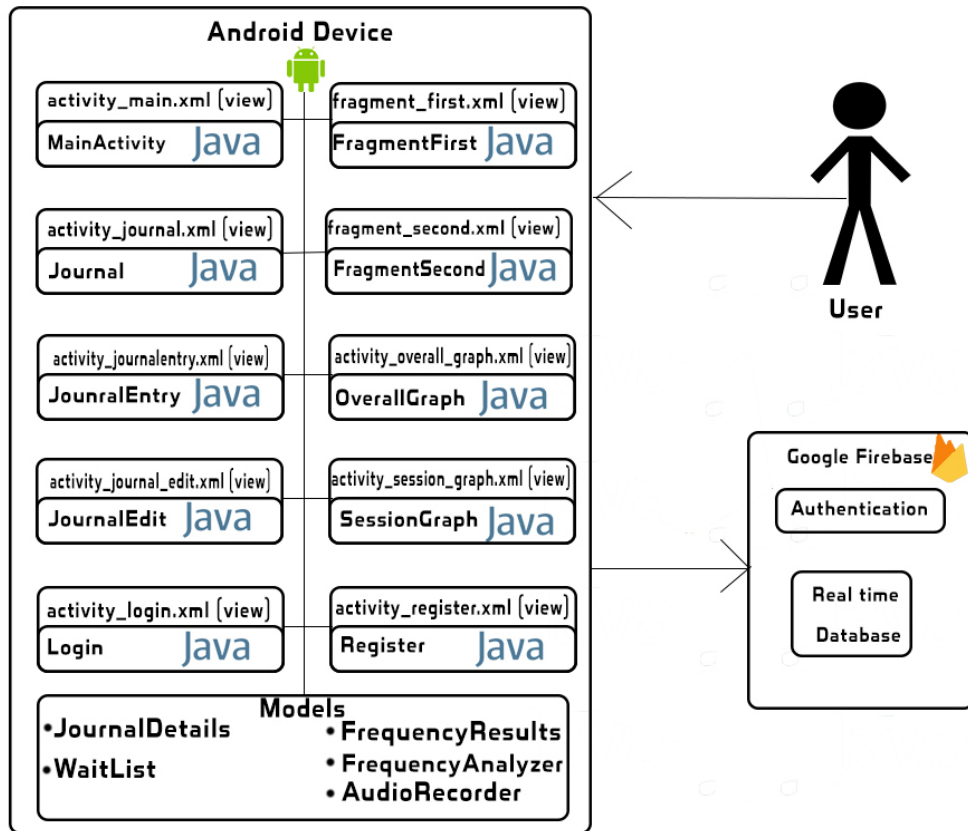


Figure x: Architecture Diagram

In today's worlds it is expected that any sort of web application or mobile application make use of a database and server-side monitoring, the Voice Training Application follows this standard. For the application to work as intended successfully the right architecture and design must be set in stone.

The Voice Training Application is designed to work with a client-server relationship, the Android application being the client and Google Firebase acting as the server. The Android application is built with a Model-View-Controller (MVC) architecture in mind, the models used to `JournalDetails.java` is used to create details object when dealing with Journal entry's to submitted to Firebase or an existing entry on Firebase to be updated, `WaitList.java` is used to updated the wait list in Firebase that the user is indeed on the list, `AudioRecorder.java`, `Frequency Analyzer` and `FrequencyResults.java` work in tandem, `AudioRecorder` takes in the audio input from the Android devices microphone and saves it to a .wav, `FrequencyAnalyzer` takes the file and puts it through a algorithm to work out the frequencies to use in graphs and `FrequencyResults` saving those as an object in Firebase.

Overall, The Voice Training App is designed to where every function is built around accessing Firebase to some compacity either sending data or receiving data which allows for clear connectivity between the different functions that interact.

## 2.3. Implementation

This section covers the development and implementation of the

```
dependencies { this: DependencyHandlerScope
    implementation("androidx.appcompat:appcompat:1.6.1")
    implementation("com.google.android.material:material:1.8.0")
    implementation("androidx.constraintlayout:constraintlayout:2.1.4")
    implementation("androidx.navigation:navigation-fragment:2.5.3")
    implementation("androidx.navigation:navigation-ui:2.5.3")
    implementation("com.android.support:support-annotations:28.0.0")

    //jTransforms
    implementation("com.github.wendykierp:JTransforms:3.1")

    // Firebase SDK
    implementation(platform("com.google.firebase:firebase-bom:32.8.1"))
    implementation("com.google.firebase:firebase-database-ktx")
    implementation("com.google.firebase:firebase-storage-ktx")
    implementation("com.google.firebase:firebase-auth-ktx")
    implementation("com.google.firebase:firebase-analytics")
    // Firebase UI Library
    implementation("com.firebaseui:firebase-ui-auth:8.0.2")
    implementation("com.firebaseui:firebase-ui-database:8.0.2")

    //Android Plot
    implementation("com.androidplot:androidplot-core:1.5.10")
    implementation("androidx.annotation:annotation:1.7.1")
    implementation("androidx.lifecycle:lifecycle-livedata-ktx:2.7.0")
    implementation("androidx.lifecycle:lifecycle-viewmodel-ktx:2.7.0")
    implementation("com.google.firebase:firebase-auth:22.3.1")

    //Testing
    testImplementation("junit:junit:4.13.2")
    androidTestImplementation("androidx.test.ext:junit:1.1.5")
    androidTestImplementation("androidx.test.espresso:espresso-core:3.4.0")
    androidTestImplementation("androidx.test:runner:1.4.0")
    androidTestImplementation("androidx.test:rules:1.4.0")
}
```

Above dependencies are used in the project, the first six dependencies came by default when making the project, jTransforms was used to graph frequency data for each voice training session and the users overall progress using the app via a plot graph and a pie chart. Firebase SDK dependencies enable the use of Firebase in the application and Firebase UI Library allows me to query data from the database and use it in my application. For testing the junit and espresso are used for unit and integration testing respectively.

## AudioRecorder:

```
public void startRecording() {
    prepareFile(); // get the filepath ready to write to
    if (audioRecord == null) { // If there no instance of audio record create one with the above settings
        audioRecord = new AudioRecord(MediaRecorder.AudioSource.MIC, SAMPLE_RATE, CHANNEL_CONFIG, AUDIO_FORMAT, BUFFER_SIZE);
    }
    audioRecord.startRecording(); // start recording
    isRecording = true; // set boolean to true so we can end the recording
    new Thread(new AudioRecordRunnable()).start(); // new thread so theres no issue when taking in data and writing it.
}

//end recording
2 usages  RubyK01 *
public void stopRecording() {
    if (audioRecord != null) { // if theres a currently a instance kill it
        isRecording = false; //
        audioRecord.stop(); // to stop recording
        audioRecord.release(); // release the resources taken up by recording e.g memory
        audioRecord = null; // kill the instance
    }
}

1 usage  RubyK01 *
private void prepareFile() { // method to set the output directory,
    // https://stackoverflow.com/a/36695883
    // https://stackoverflow.com/a/46657146
    // From the above I was able to get the downloads folder and learn how to create a folder within the downloads folder
    // and write a file to that folder
    File outputDir = new File(Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_DOWNLOADS), child: "audioOutput");
    File testFilePath = new File(Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_DOWNLOADS), child: "audioOutput");
    if (!outputDir.exists()) { // Here I check if directory I have set exists "downloads/audioOutput" and if it doesnt create it
        outputDir.mkdirs();
        Log.e(tag: "AudioRecorder prepareFile", msg: "Failed to create directory");
        return;
    }
    outputFile = new File(outputDir, child: System.currentTimeMillis() + ".wav"); // I set the file name to the current
    testFile = new File(testFilePath, child: "50hztest.wav");
}

1 usage  RubyK01 *
public String getAudioFilePath() { // Return the file path of the audio so that I can pass the file through the frequencyAnalyzer class
    if (outputFile != null) { // if they file hasnt been created yet return null
        return outputFile.getAbsolutePath();
    }
    return null;
}

//Separate thread for writing the audio file
1 usage  RubyK01 *
private class AudioRecordRunnable implements Runnable {

    // https://www.tabnine.com/code/java/methods/java.io.ByteArrayOutputStream/flush
    RubyK01 *
    @Override
    public void run() {
        byte[] audioData = new byte[BUFFER_SIZE]; // byte buffer to hold audio data from the mic
        try (FileOutputStream fos = new FileOutputStream(outputFile)) { // Writing raw audio to the outputFile
            while (isRecording) { // While recording read data from the audioRecord instance and write to the buffer
                int read = audioRecord.read(audioData, offsetInBytes: 0, audioData.length);
                if (read > 0) { // To make sure i do capture audio i check if the bytes coming in are greater than 0
                    fos.write(audioData, off: 0, read); // only the bytes greater than 0 get read so theres no unneeded data being saved
                }
            }
            fos.flush(); // flush to make sure data in the buffer is written
        } catch (IOException e) { // catch any errors and print it
            System.out.println("AudioRecorder error: " + e);
        }
    }
}
```

In the AudioRecorder class the when the startRecording method is called it calls the prepareFile() method which sets the filepath of the wav file that will be written to during the recording, an instance of audioRecord is also created to access the microphone, a thread is created to handle the process of written whats coming in from the microphone to a wav file.

The prepareFile method uses a File type variable to the filepath of where the wav file will be created, to ensure the file name is random I use the .currentTimeMillis method to have a random name which the current time in milliseconds.

The Thread AudioRecordRunnable is used to read data coming in from the microphone and output it to the output file path, this is done by reading every byte and only saving the ones that are more than zero, lastly the output streamed is flushed saving all the bytes currently in the buffer to the file.

### FrequencyAnalyzer:

2 usages RubyK01 \*

```
public List<Double> calculateFrequency(File audioFile) { // Calculation takes in the new audio file that was saved
    List<Double> frequencies = new ArrayList<>(); // array list to hold the frequencies results
    final int sampleRate = 44100; // Sample rate in Hz
    final int bytesPerSample = 2; // 16-bit audio
    final int channelCount = 1; // Mono audio
    final int bytesPerSecond = sampleRate * bytesPerSample * channelCount;

    // Fast Fourier Transforms
    // https://stackoverflow.com/a/29570220 - Where I learned to load the file into a byte buffer
    // I wasn't able to recreate it with AudioFileInput so I used FileInputStream instead
    try (FileInputStream fis = new FileInputStream(audioFile)) { // try getting the file that will be used to calculate the frequencies
        byte[] buffer = new byte[bytesPerSecond]; // byte buffer to go through the file

        // I order the bytes from WAV file by little endian as WAV files themselves use this method of storing bytes.
        // Little Endian stores bytes by going smallest to largest

        // https://vi-control.net/community/threads/need-technical-advice-with-riff-wave.137584/
        // "All WAV files are RIFF files, and all RIFF files are little-endian by default.
        // Little-endian means the least significant byte is the first byte. (Big-endian means the most significant byte is first
        // - but you probably guessed that)
        // A WAV file consists of two data "chunks", the header and the data.
        while (fis.read(buffer) != -1) { // read the buffer till theres nothing else
            double[] audioAsDouble = new double[buffer.length / 2];
            ByteBuffer byteBuffer = ByteBuffer.wrap(buffer);
            byteBuffer.order(ByteOrder.LITTLE_ENDIAN);
            for (int i = 0; i < audioAsDouble.length; i++) {
                audioAsDouble[i] = byteBuffer.getShort();
            }

            // https://stackoverflow.com/a/11868383
            // Creating the Fast Fourier Transforms Object from jTransforms that takes in arraylist holding bytes
            DoubleFFT_1D fft = new DoubleFFT_1D(audioAsDouble.length);
```

```

// https://stackoverflow.com/a/12209744
// Since I am using real audio I am using realFoward function
fft.realForward(audioAsDouble);

// https://stackoverflow.com/a/7675171
// From the above I used as a basis on how to get the result of multiple frequencies
// The post covers one frequency but I want one for every second of the recording
double maxMagnitude = 0;
int maxIndex = 1;
// start at 1 to ignore DC component, the DC component is 0Hz which we do not care for
// https://dsp.stackexchange.com/a/12973
for (int i = 1; i < audioAsDouble.length / 2; i++) {
    double re = audioAsDouble[2 * i]; // work real number
    double im = audioAsDouble[2 * i + 1]; // work out imaginary number
    double magnitude = Math.sqrt(re * re + im * im); // workout the magnitude
    if (magnitude > maxMagnitude) {
        maxMagnitude = magnitude;
        maxIndex = i;
    }
}

double frequency = maxIndex * (sampleRate / (double) audioAsDouble.length);
frequencies.add(frequency);
System.out.println(frequencies);
}
} catch (IOException e) {
    System.err.println("Error processing audio file: " + e.getMessage());
    e.printStackTrace();
}
return frequencies; // return the arraylist

```

The FrequencyAnalyzer class was defiently the hardest to implement as most of knowledge of the maths even works comes from Stack Overflow posts, the class takes in the filepath created from the end of a voice training session. The sample rate is set 44,100 Hz due that to being the standard for modern day recordings []



## Journal:

```
1 usage
static ArrayList<String> notes = new ArrayList<>();
2 usages
static ArrayAdapter arrayAdapter;
1 usage
FirebaseDatabase db = FirebaseDatabase.getInstance(); // connects instance to firebase
2 usages
DatabaseReference dbRef; // database reference
4 usages
FirebaseUser user; // variable containing user details e.g email
no usages
String entryText, id, idp1, idp2, rating, date, expectedId;

± RubyK01
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_journal);

    ListView listView = findViewById(R.id.listView); // Initializing the listview where the entrys will be displayed
    Button backBtn = findViewById(R.id.backButton);
    Button addBtn = findViewById(R.id.addBtn);
    TextView noEntry = findViewById(R.id.noEntryText);
    noEntry.setVisibility(View.GONE);
    ArrayList<String>entryList = new ArrayList<>();
    ArrayList<String>dateList = new ArrayList<>();
    ArrayList<String>ratingList = new ArrayList<>();
    ArrayList<String>idList = new ArrayList<>();
    ArrayList<String>combinedList = new ArrayList<>();

    try {
        user = FirebaseAuth.getInstance().getCurrentUser();
        if (user != null) {
            dbRef = db.getReference().child(user.getEmail().replace( oldChar: '.', newChar: ','));
        } else {
            // User is not logged in, redirect to Login screen
            Toast.makeText( context: this, text: "User not logged in. Redirecting to login screen.", Toast.LENGTH_LONG).show();
            startActivity(new Intent( packageContext: this, Login.class));
            finish();
            return; // Important to stop further execution in this case
        }
    } catch (Exception e) {
        Toast.makeText( context: this, text: "Failed to initialize database reference: " + e.getMessage(), Toast.LENGTH_LONG).show();
        // Optionally handle other initialization steps or close the activity
    }

    // Getting all the journal entry's from firebase that belong to the logged in user by checking if the entry belongs to their email
    Query query = dbRef.orderByChild( path: "email").equalTo(user.getEmail());

    ± RubyK01
    query.addListenerForSingleValueEvent(new ValueEventListener() {
        ± RubyK01
        @Override
        public void onDataChange(@NonNull DataSnapshot snapshot) {
            if (snapshot.exists()) {
                for (DataSnapshot child : snapshot.getChildren()) {
                    String entryId = child.getKey(); // Get the actual key for each journal entry
                    String entryText = child.child( path: "entryText").getValue(String.class);
                    String date = child.child( path: "date").getValue(String.class);
                    String rating = child.child( path: "rating").getValue(String.class);

                    entryList.add(entryText);
                    dateList.add(date);
                    ratingList.add(rating);
                    idList.add(entryId);

                    String entryTextShortened = entryText.length() > 8 ? entryText.substring(0, 8) + "...": entryText;
                    String combinedEntry = entryTextShortened + " - Date: " + date + " - Rating: " + rating;
                    combinedList.add(combinedEntry);
                }
            }
        }
    });
}
```

```

    }
    if (!combinedList.isEmpty()) {
        ArrayAdapter<String> arrayAdapter = new ArrayAdapter<>(context: Journal.this, android.R.layout.simple_list_item_1, combinedList);
        listView.setAdapter(arrayAdapter);
    }
} else {
    noEntry.setVisibility(View.VISIBLE);
    listView.setVisibility(View.GONE);
}
}

```

The Journal page gets the users email by checking the instance, if the user has existing journals then they are all pulled in order and into an arrayList and the moved into a arrayAdapter, the geeksforgeeks layout I had used for the GUI had the arrayAdapter connected to the list view where all the entires would appear setup already so I simply pulled all the entries from Firebase and sorted them into it. If the user has not made any entry before than a message stating so is displayed on the screen.

### JournalEntry:

```

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

    // UI components
    Button submitBtn = findViewById(R.id.submitBtn);
    Button cancelBtn = findViewById(R.id.cancelBtn);
    EditText ratingValue = findViewById(R.id.rating);
    EditText entryValue = findViewById(R.id.editText);

    // Initialize journal details object
    details = new JournalDetails();
    //Get current date in dd-mm-yyyy format
    SimpleDateFormat dateFormat = new SimpleDateFormat( pattern: "dd-MM-yyyy");
    date = dateFormat.format(new Date());

    // Making sure there is a user logged in so we can get the as a reference
    try {
        user = FirebaseAuth.getInstance().getCurrentUser();
        if (user != null) {
            dbRef = db.getReference().child(user.getEmail().replace( oldChar: '.', newChar: ','));
        } else {
            // User is not logged in, redirect to Login screen
            Toast.makeText( context: this, text: "User not logged in. Redirecting to login screen.", Toast.LENGTH_LONG).show()
            startActivity(new Intent( packageContext: this, Login.class));
            finish();
            return;
        }
    }

    dbRef.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot snapshot) {
            if(snapshot.exists()){
                entryID = (int) snapshot.getChildrenCount();//check for the number of existing entries there are for the user so i can assign the correct id for
                System.out.println("currently "+entryID+" exists for "+user.getEmail().toString());
            }
            else if(!snapshot.exists()){
                entryID = 0;
            }
        }
    })

    @Override
    public void onCancelled(@NonNull DatabaseError error) {
        System.out.println("Could not connect to firebase :c");
    }
}

```

```

@Override
public void onClick(View v) {
    if (user != null) { //Checking if the user is signed in, user should only have been able to get here if they are logged in so double checking to be sa
        // https://stackoverflow.com/questions/2841550/what-does-d-mean-in-a-regular-expression
        // https://www.javatpoint.com/java-regex
        // Here I used the matches method from regex to see if the ratingValue contains a number since in the app it says it has to be a number between 1
        if (ratingValue.getText().toString().matches(regex: "\\d+")) {
            if (ratingValue.getText().toString().contains("1") || ratingValue.getText().toString().contains("2") || ratingValue.getText().toString().contains("3") || ratingValue.getText().toString().contains("4") || ratingValue.getText().toString().contains("5")) {
                email = user.getEmail().toString();
                details.setEmail(email.toString());
                details.setDate(date.toString());
                details.setRating(ratingValue.getText().toString());
                details.setEntryText(entryValue.getText().toString());
                idp1 = user.getEmail().toString();
                idp2 = String.valueOf(entryID);
                //The child would look something like test@test.com0
                id = idp1 + idp2;
                details.setId(id);

                // if the entry is over 250 characters it is invalid
                if (details.getEntryText().length() > 250) {
                    Toast.makeText(context: JournalEntry.this, text: "Entrys cannot be greater than 250 character!", Toast.LENGTH_SHORT).show();
                }
                // if the entry has no text it is not valid
                else if (details.getEntryText().length() == 0) {
                    Toast.makeText(context: JournalEntry.this, text: "Entrys must have text!", Toast.LENGTH_SHORT).show();
                }
                else {
                    dbRef.child(pathString: String.valueOf(entryID + 1) + "Journal").setValue(details);
                    Toast.makeText(context: JournalEntry.this, text: "Journal Entry Updated!", Toast.LENGTH_SHORT).show();
                    Intent journalPage = new Intent(getApplicationContext(), Journal.class);
                    startActivity(journalPage);
                    finish();
                }
            }
            else { //If the rating is not a number between 1 and 5 it is not valid
                Toast.makeText(context: JournalEntry.this, text: "Rating must be a number between 1 - 5!", Toast.LENGTH_SHORT).show();
                return;
            }
        }
    }
}

```

When the journal entry page loads a new instance of the journalDetails class is created this used later to push an object made up of the journal entry ID, the users email, the date the entry was made, the entries text and the rating value. Firebase then grabs the users email address replacing the "." With "," due to firebase not liking the "." Being used for a name of a child, this also checks if there is a user logged in as that should be impossible but it's just to make sure, a Firebase query is ran to check how many existing entries in the journal the user has already, the journal ID is incremented by one so I need to make sure that the ID is set to avoid overwriting an entry. A valid journal entry has a rating between one and five and has 250 characters of text or less anything outside of that logic is rejected with the appropriate error message, if entry is valid it is pushed to Firebase.



## OverallGraph:

```
query.addValueEventListener(new ValueEventListener() {  
    ⚡ RubyK01 *  
    @Override  
    public void onDataChange(@NonNull DataSnapshot snapshot) {  
        if (snapshot.exists()) {  
            int mascCount = 0;  
            int femCount = 0;  
            int androCount = 0;  
            ArrayList<Long> frequencies = new ArrayList<>();  
            // https://stackoverflow.com/a/40251242  
            // I learned how to retrieve arrayList from firebase from the above  
            ⚡ RubyK01  
            GenericTypeIndicator<ArrayList<Long>> x = new GenericTypeIndicator<ArrayList<Long>>() {  
            };  
            if (user != null) { //Check if a user somehow got here without logging in  
                // https://stackoverflow.com/a/40367515  
                // From the above I learned how to loop through each child in the database  
                for (DataSnapshot freqSnapshot : snapshot.getChildren()) {  
                    ArrayList<Long> firstSoundResults = freqSnapshot.child( path: "firstSoundResults").getValue(x);  
                    System.out.println("First Array: "+firstSoundResults);  
                    ArrayList<Long> secondSoundResults = freqSnapshot.child( path: "secondSoundResults").getValue(x);  
                    System.out.println("Second Array: "+secondSoundResults);  
                    ArrayList<Long> thirdSoundResults = freqSnapshot.child( path: "thirdSoundResults").getValue(x);  
                    System.out.println("Third Array: "+thirdSoundResults);  
  
                    if (firstSoundResults != null) frequencies.addAll(firstSoundResults);  
                    if (secondSoundResults != null) frequencies.addAll(secondSoundResults);  
                    if (thirdSoundResults != null) frequencies.addAll(thirdSoundResults);  
                    System.out.println("Combined Array: "+frequencies);  
                }  
            }  
        }  
    }  
});
```

```

for (int i = 0; i < frequencies.size(); i++) {
    if(frequencies.get(i) >= 85 && frequencies.get(i) <= 175) {
        mascCount++;
    }
    else if(frequencies.get(i) >= 147 && frequencies.get(i) <= 294) {
        femCount++;
    }
    else{
        androCount++;
    }
}

private void updateChart(int mascCount, int femCount, int androCount, int total){
    PieChart HzChart = findViewById(R.id.pie_chart);
    TextView chartTitle = findViewById(R.id.chart_title);
    HzChart.getBackgroundPaint().setColor(Color.WHITE);
    HzChart.clear();

    if(total>0){
        int mascPercentage = 100 * mascCount / total;
        int femPercentage = 100 * femCount / total;
        int androPercentage = 100 * androCount / total;
        //Chart segments
        //I have the titles left blank as the text within the segments really small
        //There are labels at the bottom of the screen for the user to read
        Segment masculine = new Segment( title: "", mascPercentage);
        Segment feminine = new Segment( title: "", femPercentage);
        Segment androgynous = new Segment( title: "", androPercentage);

        //Chart formatters
        SegmentFormatter mascFormatter = new SegmentFormatter(Color.parseColor( colorString: "#3F51B5"),Color.BLACK);
        SegmentFormatter femFormatter = new SegmentFormatter(Color.parseColor( colorString: "#FF4081"),Color.BLACK);
        SegmentFormatter androFormatter = new SegmentFormatter(Color.parseColor( colorString: "#4CAF50"),Color.BLACK);

        HzChart.addSegment(masculine, mascFormatter);
        HzChart.addSegment(feminine, femFormatter);
        HzChart.addSegment(androgynous, androFormatter);

        HzChart.redraw();

        chartTitle.setText("Overall Progress");
        mascText.setText("Masculine: \n"+mascPercentage+"%");
        femText.setText("Feminine: \n"+femPercentage+"%");
        androText.setText("Androgynous: \n"+androPercentage+"%");
    }
    else{
        chartTitle.setText("Overall Progress \n Please do voice training to show progress");
        mascText.setText("Masculine: 0%");
        femText.setText("Feminine: 0%");
        androText.setText("Androgynous: 0%");
    }
}
}

```

The OverallGraph class is straight forward to understand, when the activity is started a Firebase query is ran to grab all the frequency result objects saved under the users email in Firebase, the frequencies are saved in three arraylists in the objects due to their being three sessions of voice training, these are sorted into long type array lists temporally before they are all combined into one arraylist.

The new arraylist has all its values checked against the IF to organise the values into masculine, feminine or androgynous by adding one to corresponding int variables, In the updateChart method the percentage of the voices features is the calculated by multiply the three int

variables by 100 and dividing by the total number of frequency values gained from the initial query. The chart now displays the what percentage of the three characters (masculine feminine and androgynous).

### SessionGraph:

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_session_graph);
    //https://www.geeksforgeeks.org/how-to-get-extra-data-from-intent-in-android/
    //Above link is where i learned how to get data from other activities.
    //grabbing the Hz data from second fragment
    ArrayList<Double> firstSoundResults = (ArrayList<Double>) getIntent().getSerializableExtra(name: "firstSoundResults");
    ArrayList<Double> secondSoundResults = (ArrayList<Double>) getIntent().getSerializableExtra(name: "secondSoundResults");
    ArrayList<Double> thirdSoundResults = (ArrayList<Double>) getIntent().getSerializableExtra(name: "thirdSoundResults");
    //A list to hold time data, since the .wav files are always 15 seconds the list will always be 0-15
    ArrayList<Double> seconds = new ArrayList<>(Arrays.asList(0.0,1.0,2.0,3.0,4.0,5.0,6.0,7.0,8.0,9.0,10.0,11.0,12.0,13.0,14.0,15.0));

    //http://halfho.github.io/androidolot/docs/xuolot.html
```

```

LineAndPointFormatter firstResultFormatter = new LineAndPointFormatter(Color.RED, vertexColor: null, Color.argb( alpha: 50, red: 255, green: 0, blue: 0), plf: null);
firstResultFormatter.getVertexPaint().setColor(Color.RED); // Hide the vertex markers
firstResultFormatter.setFillPaint(new Paint());
firstResultFormatter.setFillPaint().setColor(Color.argb( alpha: 50, red: 255, green: 0, blue: 0)); // Semi-transparent fill

LineAndPointFormatter secondResultFormatter = new LineAndPointFormatter(Color.BLUE, vertexColor: null, Color.argb( alpha: 50, red: 0, green: 255, blue: 0), plf: null);
secondResultFormatter.getVertexPaint().setColor(Color.BLUE);
secondResultFormatter.setFillPaint(new Paint());
secondResultFormatter.setFillPaint().setColor(Color.argb( alpha: 50, red: 0, green: 255, blue: 0));

LineAndPointFormatter thirdResultFormatter = new LineAndPointFormatter(Color.GREEN, vertexColor: null, Color.argb( alpha: 50, red: 0, green: 0, blue: 255), plf: null);
thirdResultFormatter.getVertexPaint().setColor(Color.GREEN);
thirdResultFormatter.setFillPaint(new Paint());
thirdResultFormatter.setFillPaint().setColor(Color.argb( alpha: 50, red: 0, green: 0, blue: 255));

//Adding plots to graph as default view
plot.addSeries(firstResults,firstResultFormatter);
plot.addSeries(secondResults,secondResultFormatter);
plot.addSeries(thirdResults,thirdResultFormatter);
//setting the title of the graph
plot.setTitle("Pitch Over Time");
//Labeling the x axis
plot.setDomainLabel("Time (seconds)");
//Labeling the y axis
plot.setRangeLabel("Pitch (Hz)");
//https://www.tabnine.com/code/java/methods/com.androidplot.xy.XYPlot/setDomainBoundaries
//Using Domain/Range Boundaries I can have the x & y axis go from 0 to 15 in seconds
//and 0 from 255 in hz
plot.setDomainBoundaries( lowerBoundary: 0, upperBoundary: 15, BoundaryMode.FIXED);//Set y values start to end
plot.setDomainStep(StepMode.INCREMENT_BY_VAL, value: 3);// how much y increases
plot.setRangeBoundaries( lowerBoundary: 85, upperBoundary: 255, BoundaryMode.FIXED);//Set x values start to end
plot.setRangeStep(StepMode.INCREMENT_BY_VAL, value: 25);//how much x increases
plot.redraw();

```

```
Button viewButton = findViewById(R.id.viewButton);
```

```
    RubyK01
```

```
viewButton.setOnClickListener(new View.OnClickListener() {
```

```
    RubyK01
```

```
@Override
```

```
public void onClick(View view) {
```

```
    //Every time the view button is tapped viewCount variable has a 1 added
```

```
    //The resulting number decides on which plot is visible
```

```
    viewCount = viewCount + 1;
```

```
    plot.clear();
```

```
    if(viewCount == 1){//Tap once for just the first result
```

```
        plot.addSeries(firstResults,firstResultFormatter);
```

```
    }
```

```
    else if(viewCount == 2){//Tap twice for just the second result
```

```
        plot.addSeries(secondResults,secondResultFormatter);
```

```
    }
```

```
    else if(viewCount == 3){//Tap three times for just the third result
```

```
        plot.addSeries(thirdResults,thirdResultFormatter);
```

```
    }
```

```
    else{//Tapping a fourth time will reset the view to showing all three plots at once
```

```
        plot.addSeries(firstResults,firstResultFormatter);
```

```
        plot.addSeries(secondResults,secondResultFormatter);
```

```
        plot.addSeries(thirdResults,thirdResultFormatter);
```

```
        viewCount = 0;
```

```
    }
```

```
    plot.redraw();
```

```
    }
```

```
});
```

The SessionGraph class pulls the from the SecondFragment class during when the SessionGraph activity is starting or in order words when the “View Result” button is tapped, the lines getIntent().getSerializableExtra is the recieving end to the putExtra in SecondFragment, is basically like

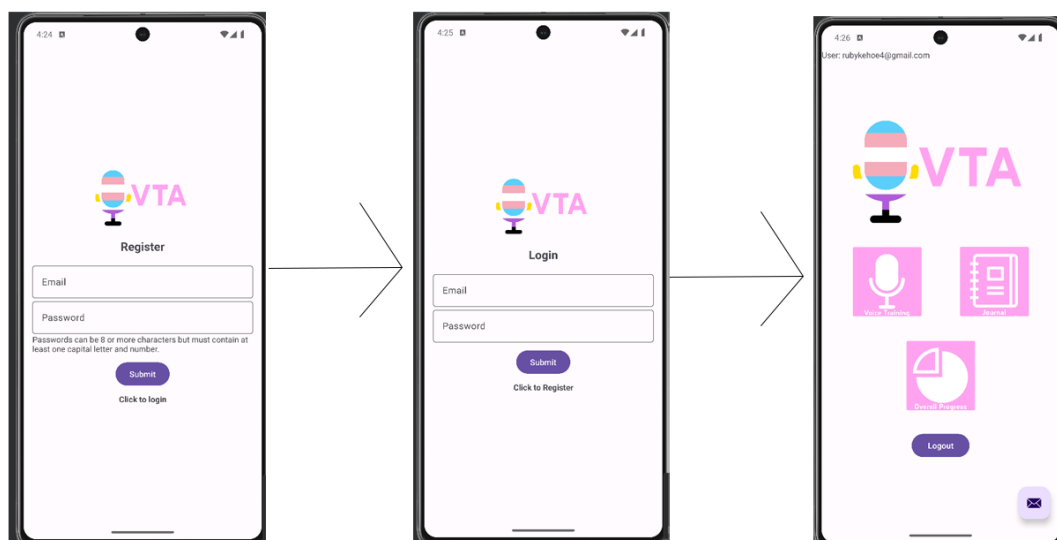
copying and pasting but between instances, I then use formatters from the AndroidPlot to choose what colors the plots for each frequency array list will be and link the formatters to the array lists through a series basically I'm one this array list to partner with a colour, I then set the title of the graph as well as the x and y axis names by using plot.set methods, next I set the starting and end values for each axis and how much it increases by at every point e.g x axis starts at zero and ends at fifteen but goes up in threes.

The view button allows the user to filter which set of frequencies they are looking at on the graph, there are four options, the default one which is all of them, the second being the first set of frequencies and so fourth, I achieve this by having an int variable increases by one each time the button is tapped, if it is tapped twice the second set of frequencies will be plotted by itself, the graph is reset at every four taps to the default view.

## 2.4. Graphical User Interface (GUI)

The GUI section of this report is meant to demonstrate each of the screens that make up the front end of the Voice Training Application.

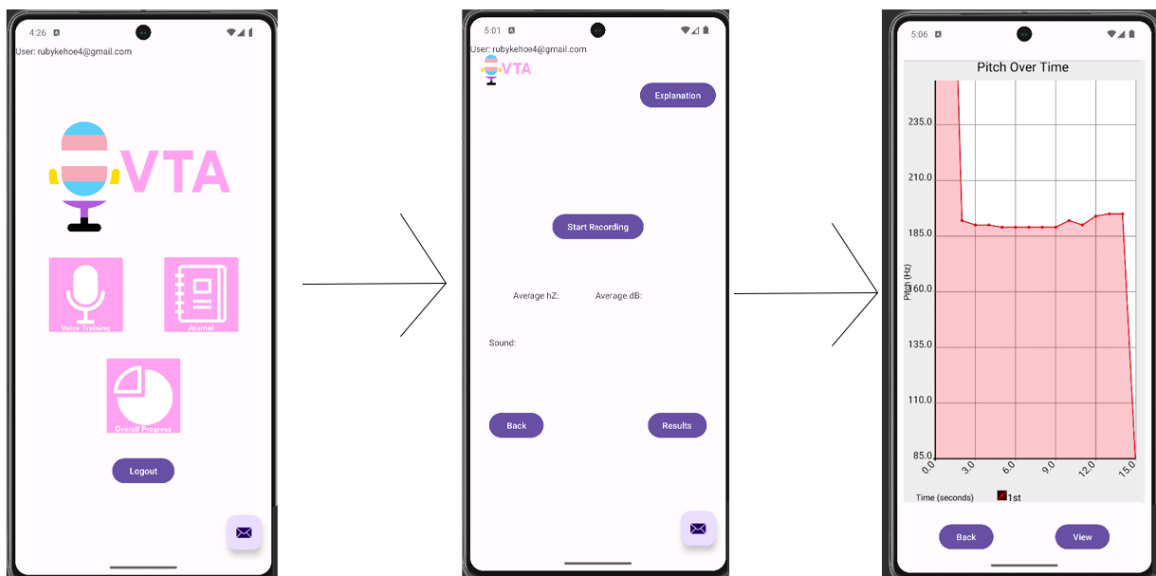
For a user who has only opened the app for the first time they will start at the register screen and create an account with their email and a unique password, once the account is created, they will be brought the login screen to input their new account details and lastly the user logs in and is taken to the home screen.



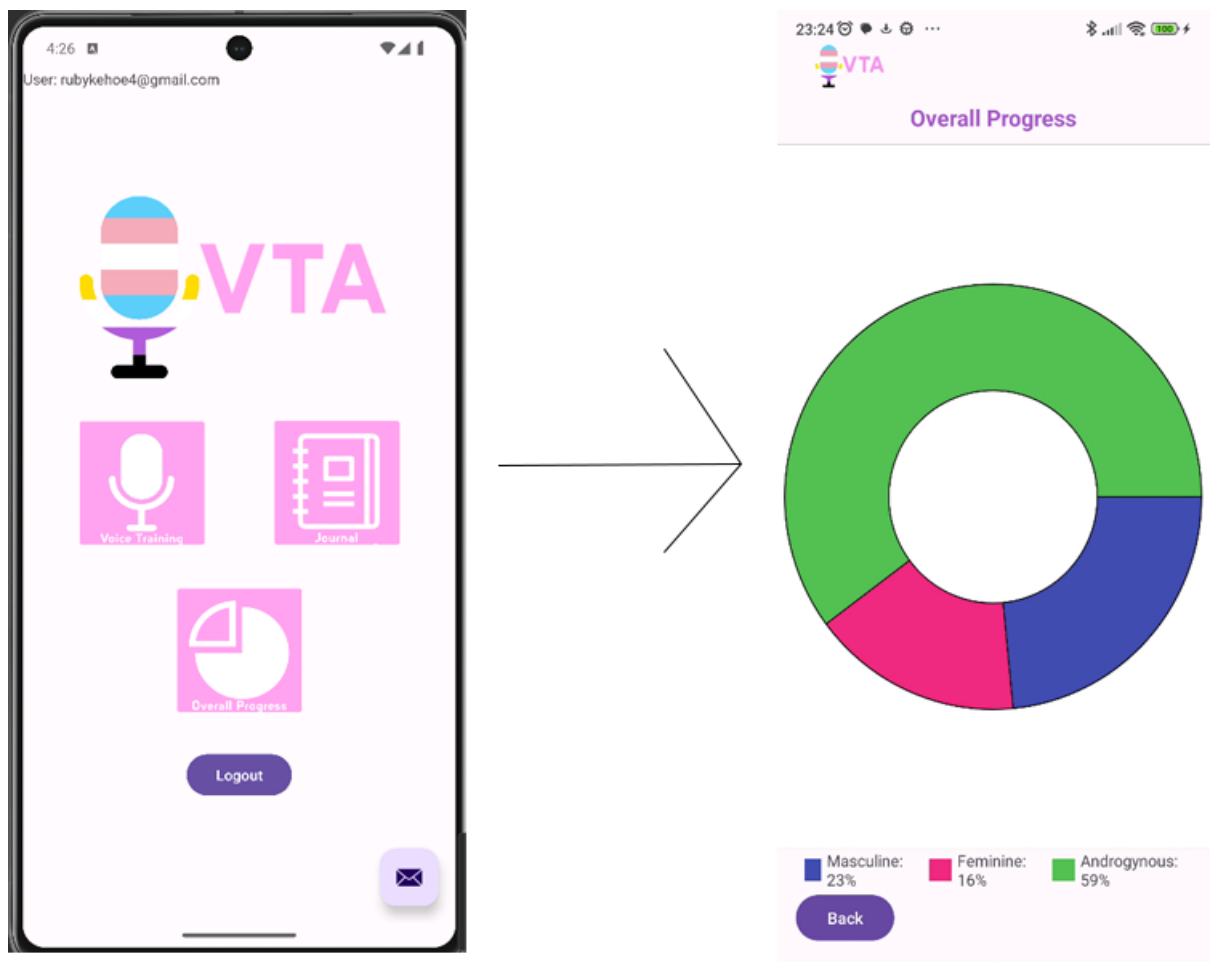
*GUI 1: Account creation and login flow: Register screen – Login screen – Home screen*

Note: A user who was logged when closing the app will still be logged when the open the app again, this will make the flow skip straight to the home screen.

The flow for the applications main feature is as follows, the user starts on the home screen already logged into their account, they then tap the voice training icon which is the pink square with the microphone



GUI 2: Voice training: Home screen – Voice Training screen – Session Graph screen



GUI 3: Overall Progress: Home screen to Overall Progress pie chart



GUI 4: Journal & Add/Edit entry: Home – Journal – Journal Add/Edit

## 2.5. Testing

The test plan for the Voice Training Application is to use various techniques which I have learned from the Software Quality and Testing module from my second year of the course which covers White Box and Block Box testing techniques, White Box techniques including Unit tests and Integration tests and Black Box including Decision Table and Boundary Value Analysis. SMART which is a testing framework that I have learned about over the course of the project's development, SMART is an acronym standing for Specific, Measurable, Achievable, Relevant and Time-bound, SMART is often used for project management.

### Unit Testing:

```

no usages  RubyK01
public class JournalEntryTest {
    5 usages
    private EntryLogic validator = new EntryLogic();

    no usages  RubyK01
    @Test
    public void ratingMustBeBetween1And5() {
        assertEquals( expected: "Rating must be a number between 1 - 5!", validator.validateEntry( rating: "0", entryText: "Valid text"));
        assertEquals( expected: "Rating must be a number between 1 - 5!", validator.validateEntry( rating: "6", entryText: "Valid text"));
    }

    no usages  RubyK01
    @Test
    public void entryCannotBeTooLong() {
        String a = "a";
        String bunchOfA = "";
        for (int i = 0; i < 252; i++) {
            bunchOfA = bunchOfA + a;
        }
        assertEquals( expected: "Entries cannot be greater than 250 characters!", validator.validateEntry( rating: "5", bunchOfA));
    }

    no usages  RubyK01
    @Test
    public void entryMustHaveText() {
        assertEquals( expected: "Entries must have text!", validator.validateEntry( rating: "5", entryText: ""));
    }

    no usages  RubyK01
    @Test
    public void validEntry() {
        assertEquals( expected: "Valid", validator.validateEntry( rating: "5", entryText: "This is a valid entry"));
    }
}

```



```

1 package com.example.voicetrainingapp;
2
3 public class EntryLogic {
4     @
5     public String validateEntry(String rating, String entryText) {
6         if (!rating.matches(regex: "[1-5]")) {
7             return "Rating must be a number between 1 - 5!";
8         } else if (entryText.length() > 250) {
9             return "Entries cannot be greater than 250 characters!";
10        } else if (entryText.isEmpty()) {
11            return "Entries must have text!";
12        } else {
13            return "Valid";
14        }
15    }
16 }
17
18 import ...
19
20 no usages RubyK01
21 @RunWith(JUnit4.class)
22 public class LoginTest {
23     1 usage
24     String DB_EMAIL = "example@example.com";
25     1 usage
26     String DB_PASSWORD = "Password1";
27
28     // Methods to validate credentials
29     2 usages RubyK01
30     private boolean validateEmail(String inputEmail) { return DB_EMAIL.equals(inputEmail); }
31
32     2 usages RubyK01
33     private boolean validatePassword(String inputPassword) {
34         return DB_PASSWORD.equals(inputPassword);
35     }
36
37     no usages RubyK01
38     @Test
39     public void invalidDetails() {
40         assertTrue( message: "The input email should match the expected email.",
41             validateEmail( inputEmail: "example@example.com"));
42         assertFalse( message: "The input password should not match the expected password.",
43             validatePassword( inputPassword: "password"));
44     }
45
46     no usages RubyK01
47     @Test
48     public void validDetails() {
49         assertTrue( message: "The input email should match the expected email.",
50             validateEmail( inputEmail: "example@example.com"));
51         assertTrue( message: "The input password should match the expected password.",
52             validatePassword( inputPassword: "Password1"));
53     }
54 }

```

```
package com.example.voicetrainingapp;
```

```
import java.util.regex.Pattern;
```

2 usages RubyK01

```
public class RegisterLogic {
```

5 usages RubyK01

```
    public String validDetails(String email, String password) {
```

```
        Pattern textPattern = Pattern.compile(regex: "^(?=.*[a-z])(?=.*[A-Z])(?=.*\\d).+$");
```

```
        if (email.isEmpty()) {
```

```
            return "Email required.";
```

```
        }
```

```
        else if (!email.contains("@") || !email.contains(".")) {
```

```
            return "Invalid email format.";
```

```
        }
```

```
        else if (password.isEmpty()) {
```

```
            return "Password required.";
```

```
        }
```

```
        else if (!textPattern.matcher(password).matches() || password.length() < 8) {
```

```
            return "Password not valid.";
```

```
        }
```

```
        else {
```

```
            return "Valid";
```

```
        }
```

```
    }
```

```
}
```

```

import ...

no usages  RubyK01
public class RegisterTest {

    5 usages
    private RegisterLogic RegisterTestLogic = new RegisterLogic();

    no usages  RubyK01
    @Test
    public void emailsIsRequired() {
        assertEquals( expected: "Email required.", RegisterTestLogic.validDetails( email: "", password: "Password1!"));
    }

    no usages  RubyK01
    @Test
    public void invalidEmailFormat() {
        assertEquals( expected: "Invalid email format.", RegisterTestLogic.validDetails( email: "example.com", password: "Password1!"));
    }

    no usages  RubyK01
    @Test
    public void passwordIsRequired() {
        assertEquals( expected: "Password required.", RegisterTestLogic.validDetails( email: "example@example.com", password: ""));
    }

    no usages  RubyK01
    @Test
    public void passwordNotValid() {
        assertEquals( expected: "Password not valid.", RegisterTestLogic.validDetails( email: "example@example.com", password: "pass"));
    }

    no usages  RubyK01
    @Test
    public void validDetails() {
        assertEquals( expected: "Valid", RegisterTestLogic.validDetails( email: "example@example.com", password: "Password1!"));
    }
}

```

All the unit tests run as intended, they all compare a result to an IF statement in replacement of using pretending a button is pressed but they show that the logic of the tested classes work without fail.

### Integration Testing:

```

import ...
// https://stackoverflow.com/questions/28390574/checking-toast-message-in-android-espresso
3 usages RubyK01
@SuppressLint("deprecation", "unchecked")
public class ToastMatcher extends TypeSafeMatcher<Root> {

    RubyK01
    @Override
    public void describeTo(Description description) { description.appendText("is toast"); }

    no usages RubyK01
    @ public static Matcher<Root> isToast() { return new ToastMatcher(); }

    RubyK01
    @Override
    public boolean matchesSafely(Root root) {
        int type = root.getWindowLayoutParams().get().type;
        if (type == WindowManager.LayoutParams.TYPE_TOAST) {
            IBinder windowToken = root.getDecorView().getWindowToken();
            IBinder appToken = root.getDecorView().getApplicationWindowToken();
            if (windowToken == appToken) {
                // windowToken == appToken means this window isn't contained by any other windows.
                // if it was a window for an activity, it would have TYPE_BASE_APPLICATION.
                return true;
            }
        }
        return false;
    }
}

no usages RubyK01
@Before
public void setup() {
    activityScenarioRule.getScenario().onActivity(activity -> decorView = activity.getWindow().getDecorView());
}

no usages RubyK01
@After
public void tearDown() {
    // Intents.release();
}

no usages RubyK01
@Test
public void testElementsDisplayed() {
    onView(withId(R.id.email)).check(matches(isDisplayed()));
    onView(withId(R.id.password)).check(matches(isDisplayed()));
    onView(withId(R.id.btnLogin)).check(matches(isDisplayed()));
    onView(withId(R.id.progressBar)).check(matches(not(isDisplayed())));
    onView(withId(R.id.loginNow)).check(matches(isDisplayed()));
}

no usages RubyK01
@Test
public void testSuccessfulLogin() {
    onView(withId(R.id.email)).perform(typeText(stringToBeTyped: "example@example.com"), closeSoftKeyboard());
    onView(withId(R.id.password)).perform(typeText(stringToBeTyped: "Password1"), closeSoftKeyboard());
    onView(withId(R.id.btnLogin)).perform(click());
    isToastMessageDisplayed("Logged in!"); // Directly use the expected toast message
}

```

The above should work but it does not from what I understand it is something to do with a Android version but it is too late at the time of writing to fix it, the video I have linked in the Login Integration test follows my logic so it should work.

## Black Box Testing:

Class: Register.java

Technique: Decision Table Testing

What is Decision Table Testing? It is a Black Box testing technique

Testing context:

This decision table cover the logic of registering an account, an inputted email is considered valid if "@" and "." are in the input as for the password an inputted password is valid if it contains at least one capital letter, one number and is eight or more characters in length. If the email and password meet those requirements the validDetails Boolean will be set to true and the account will be created.

Test Case	Email	Email valid?	Password	Password length	Password Valid?	Result
1	null	false	null	0	false	"Email Required" validDetails = false
2	"example"	false	null	0	false	"Invalid email format" validDetails = false
3	"user@example.com"	true	null	0	false	"Password required" validDetails = false
4	"user@example.com"	true	Password1	9	true	validDetails = true
5	"user@example.com"	true	Pass	4	false	"Password not valid" validDetails = false
6	"user@example.com"	true	Pass1	5	false	"Password not valid" validDetails = false

7	"user@example.com"	true	123	3	false	"Password not valid" validDetails = false
8	example@email	false	Password1	9	True	"Invalid email format" validDetails = false
9	example.com	False	Password1	9	true	"Invalid email format" validDetails = false

Technique: Boundary Value Analysis

Test Cases	Test Values For Email
1	example
2	example@email
3	example.com
4	example@email.com

Invalid Test Cases	Valid Test Cases	Invalid Test Cases
1	4	2, 3

Test Cases	Test Values For Password
1	Password1
2	Pass

3	pass
4	Pass12
5	pass12
6	12
7	ThisPasswordIs32CharactersLong
8	thispasswordhasmorethaneightcharacters
9	ThisPasswordHasMoreThanEightCharacters
10	123456789

Invalid Test Cases	Valid Test Cases	Invalid Test Cases
2, 3, 4, 5, 6	1, 7	8, 9, 10

Class: Login.java

Technique: Decision Table Testing

Test Case	Email empty	Password empty	Authentication Successful	Result
1	True	False	False	"Email required"
2	False	True	False	Password required"
3	False	False	True	"Logged in!"
4	False	False	False	"Invalid details"

Class: JournalEntry.java / JournalEdit.java

Technique: Decision Table Testing

Test Case	Rating numeric input	Rating input between 1 - 5	Entry text > 250 characters	Entry text =< 0	Result
1	False	N/A	False	False	"Rating must be a number between 1 -5)"
2	True	False	True	False	"Rating must be a number between 1 -5)"
3	True	True	True	False	"Entry cannot be greater than 250 characters"
4	True	True	False	True	"Entries must have text"
5	True	True	False	False	"Journal entry added/updated"

Class: JournalEntry.java / JournalEdit.java

Technique: Boundary Value Analysis

Context: A-Z and a-z meaning upper and lower case alphabet respectively. +x plus infinity

Invalid Inputs	Valid Inputs	Invalid Inputs
A-Z, a-z, 0	1, 2, 3, 4, 5	6,7,8,9,10+x

Context: Numbers = number of characters in the entry

Invalid Inputs	Valid Inputs	Invalid Inputs
0	1 - 250	251+x

SMART Testing:



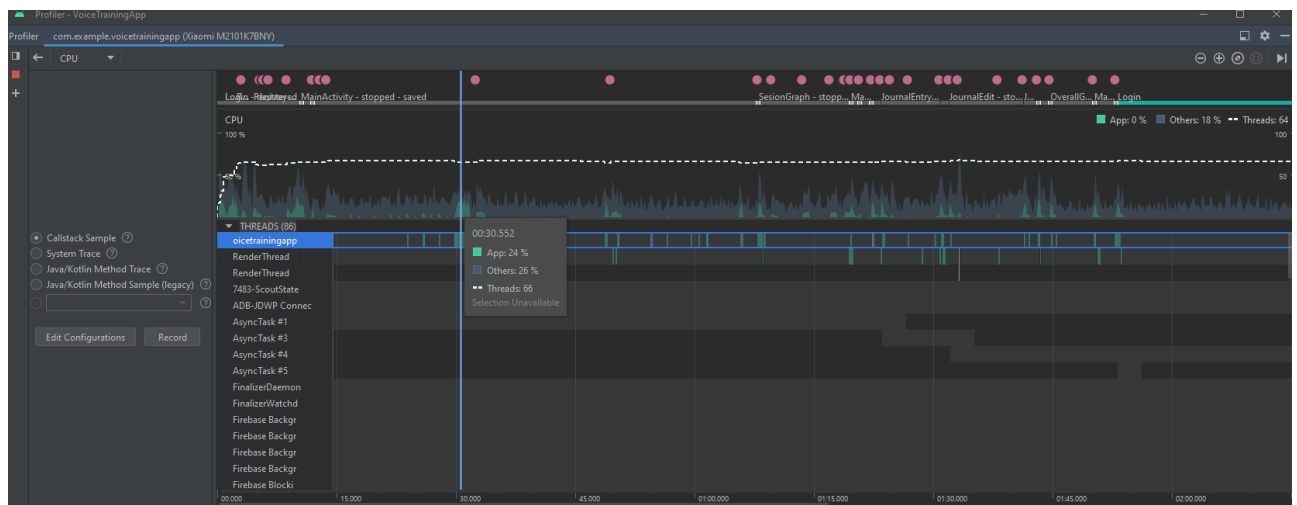
Conducting SMART testing on the Voice Training App meant I was able to ensure that the app adhered to the SMART frameworks requirements that being Specific, Measurable, Achievable, Relevant and Time-bound. The SMART testing framework is a highly effective approach to validate the quality and reliability of the Voice Training App. []

#### Specific:

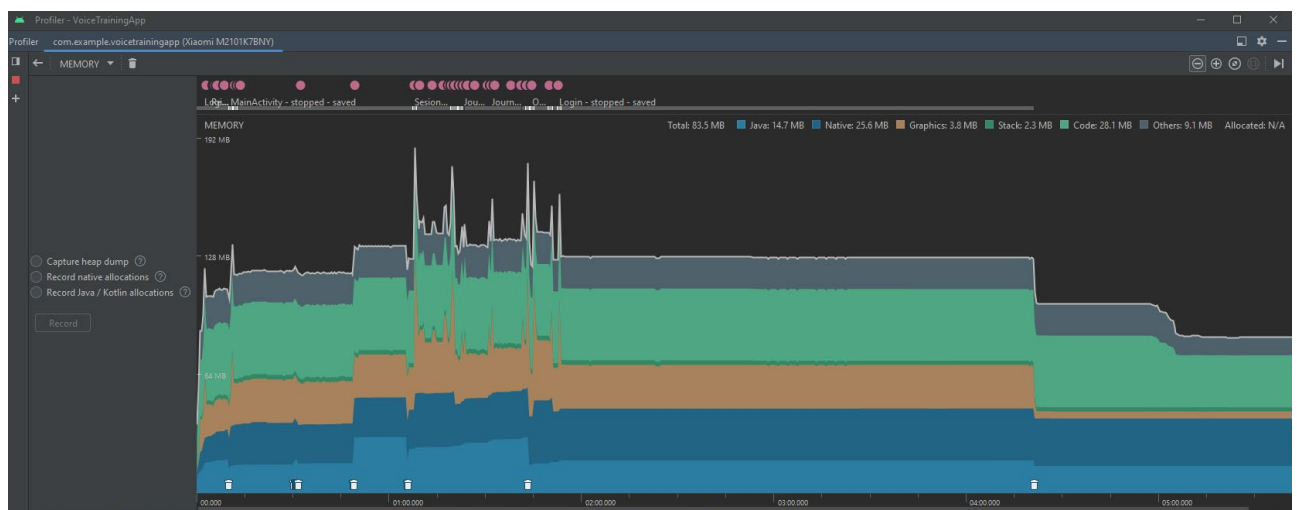
Testing included the physical testing using my own device (Xiaomi Note 10 S) where I had tested each feature of the application (account registration, login, voice training, viewing training session results, journal entry, editing of a journal entry, viewing overall progress and contacting a speech therapist). The goal of testing on a physical device was to ensure that every feature had met the objectives and scopes set out for them.

#### Measurable:

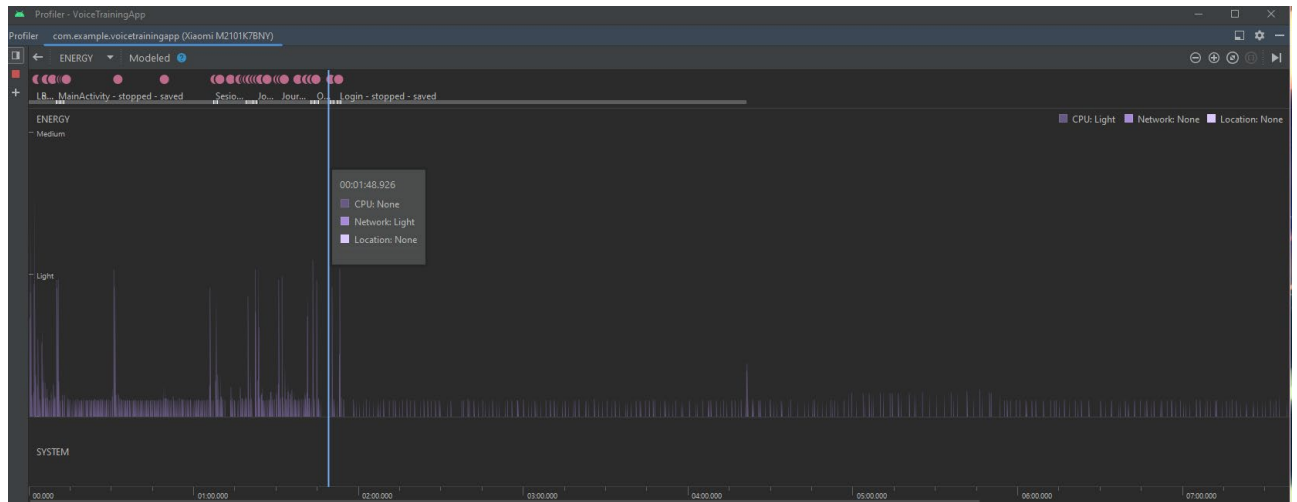
Using the Profiler tool in Android Studio I was able to generate performance data for the Voice Training Application. These tests gathered metrics of the usage of the device's resources such as CPU, memory and power.



Voice Training App figure 1: Performance CPU Test



Voice Training App figure 2: Performance Memory Test



*Voice Training App figure 3: Performance Energy Test*

Looking at the above Voice Training App figure 1 we can see that the CPU usage is fairly light for the most part with some spikes around voice training and viewing the results for that training session, there also a spike in usage around the other features such as when the journal page is loaded, when a new entry is added and edited and when the overall progress is checked. The spikes still come under as light as seen in figure x so the application is not putting a massive strain on the devices CPU with the highest usage being 24%.

From Voice Training App figure 2 we can see that the applications memory usage is fairly consistent most notable when the results from a voice training session is displayed but in total the application only uses 83.5 MB of the system memory out of the total 6 GB so overall the application does not have a heavy impact on the devices memory.

Lastly in Voice Training App figure 3 we can see that there are spikes in energy usage particular when a new activity is opened and the previous one closes, for instances around voice training this could be due wav files being created and Fast Fourier Transform being performed on said files to calculate the frequencies, but according to the chart the energy usage is light meaning the application is not using much of the device's battery.

#### Achievable:

Making use of my own device to test the Voice Training App made the testing process much more efficient as each feature of the application could be tested in real time over the course of development this made it easier to compare the actual result with the expected result and make any adjustments that might be needed and be able to fix any errors that appeared during testing.

#### Relevant:

The main goal of the testing was to verify that the application met the objectives and adhered to the Software Specifications Requirements.

#### Time-bound:

Manually testing the application throughout the course of development had made fixing errors and finalizing features way easier. Having created a timeline for projects development I was able to set deadlines for when certain features would be finalized and allocate time for testing.

### 3.0 Conclusions

The Voice Training Application, was developed and designed for the transgender to help people who have a desire to sound how they feel they should sound or in better words help them sound the same way they do in their head. The application was designed in a user-friendly manor as the GUI is simple to understand for any anyone one regardless of how tech savvy, they may be which might be a reason some users decide to give the application a chance.

As with any execution of an idea there is always something that is done well and something that could still be improved on. I believe the design being straightforward in terms of design can be an advantage as anyone can pick it up with little explanation but the same could be said about most voice training apps, what stands out with what I have developed are a couple key features those being the journal and contacting a speech therapist. The journal feature giving a user a space where they can think to themselves about how their feelings, their progress or even think of a person they may like to sound like or even find an aspect of their voice that they do enjoy upon reflection. Speech therapy is a path that might suit some people as having someone to guide you through procedures can be much easier than going at it alone, from my own personal experience seeing a speech therapist did help find parts of my voice that I did like as well as being able practice methods to help change the aspects I do not like and I believe this application can help me do so.

In terms of what the application is lacking, as it stands the user is not able to see the results of previous voice training sessions on a plotted graph in the current state the application only lets the user see the plotted graph after a session but once they go off that screen, they won't be able to go back to that instance of the graph, however the user is still able to gauge their overall progress by the pie chart that updates in real time with the end of each session. Secondly, the application takes in audio and saves it as wav file in a folder within the devices downloads folder but the application does not have a method for deleting the files as it stands which could lead to build of recordings taking up storage space on the device, the user can however delete the folder outside of the application without resulting in any issues with the application.

Giving the uniqueness of the advantages of the application and the unusual drawbacks it has proved the application to be both distinctive in its approach with features that are not only innovative but also address the specific needs of the target audience.

### 4.0 Further Development or Research

The Voice Training Application whilst already having a unique approach in terms of its features in comparison to apps of a similar calibre, could have definitely benefited from having more time for development to implement a way to delete the recording files the application creates and allowing users to revisits the results screen for their previous training sessions, this could be done by having another activity setup that pulls the frequencies from Firebase using the date they were recorded on by the user in fact the date does currently get push with the frequencies already so this would be the most realistic feat that could be done if extra time was given.

In terms further developing the applications with ideas that were not considered during the original development cycle the journal function could be expanded further in a couple of ways such as allowing the user to attach a recording of their voice to have right with the entries text so when the user is reflecting in the future, they can get an even clearer picture of how far they have gone, another thought that I have of the speech therapist with the users consent can see certain

journal entries so that they can plan out their next appointment with the user and set training plans for the user, additionally having a function to manage appointments with the speech therapist within the application itself would be more beneficial to the user and therapist as they wouldn't have to move between applications to see when their next appointment with a user and seeing the users training plan and journal.

These ideas for potential future updates while not mind boggling would allow the application to go far beyond the initial goals but ultimately would lead the application to become a much more valuable resource for the transgender community.

## 5.0 Poster



### Voice Training App

Ruby Rappale Kehoe, BSc (Hons) in Computing



#### Overview

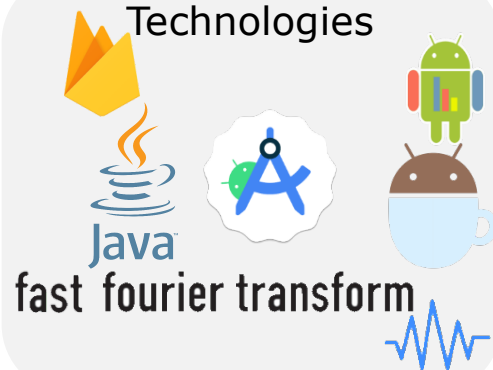
VTA your ultimate voice training companion! Enhance your vocal journey with comprehensive voice training exercises. Keep track of your thoughts, experiences, and milestones with your own journal, allowing for personal reflection and growth. Monitor your progress through real time graph updates. And whenever you need expert guidance, reach out directly to our certified speech therapists for personalized advice!



#### Motivation

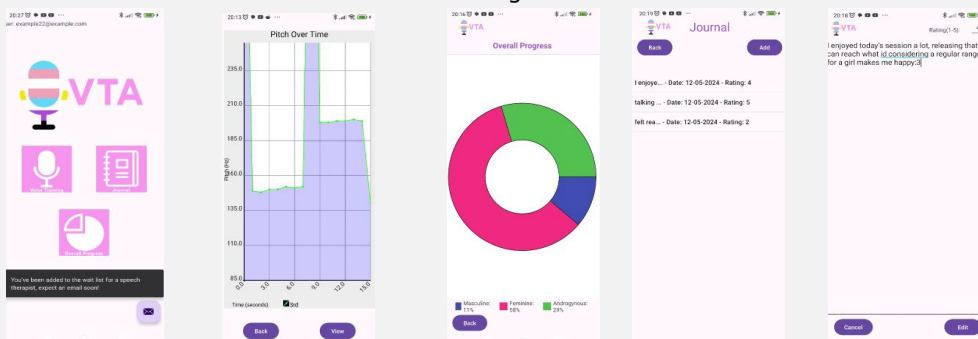
As a trans person, I struggle with my voice and sought speech therapy to learn vocal exercises. I realized a voice training app focused on these exercises could significantly improve my progress, boost mental health, and help others like me.

#### Technologies



### How it Works.

- Reach out
- Voice Training
- Track Progress
- Keep track of your well being



## 6.0 References

[1] "Android Studio," [Online]. Available: <https://developer.android.com/> [Accessed at 24/03/2024]

[2] *Android.media : android developers* (no date) *Android Developers*. Available at: <https://developer.android.com/reference/android/media/package-summary> [Accessed: 23 March 2024]

[3] *Androidplot*. Available at: <http://androidplot.com/> [Accessed: 24 March 2024]

[4] Wendykierp (no date) *WENDYKIERP/jtransforms: Jtransforms is the first, open source, multithreaded FFT library written in pure java., GitHub*. Available at: <https://github.com/wendykierp/JTransforms> [Accessed: 24 March 2024]

[5] *Fast fourier transform* (2024a) *Wikipedia*. Available at: [https://en.wikipedia.org/wiki/Fast\\_Fourier\\_transform](https://en.wikipedia.org/wiki/Fast_Fourier_transform) [Accessed: 24 March 2024]

[6] "Google's Firebase," [Online]. Available: <https://firebase.google.com/> [Accessed at 24/03/2024]

Adobe (no date) *Sample rates and audio sampling: A guide for beginners | adobe*. Available at: <https://www.adobe.com/uk/creativecloud/video/discover/audio-sampling.html> [Accessed: 08 May 2024]

[7] Wikipedia, "SMART criteria," [Online]. Available: [https://en.wikipedia.org/wiki/SMART\\_criteria](https://en.wikipedia.org/wiki/SMART_criteria) [Accessed at 24/04/2024]

## 7.0 Appendices

### 7.1. Project Proposal

# National College of Ireland

## Project Proposal Voice Training App 19/10/2023

BSC In Computing  
Software Development  
2023/2024  
Ruby Rapple Kehoe  
X20382263  
X20382263@student.ncirl.ie

### Contents

1.0	Objectives .....	48
2.0	Background .....	49

3.0	State of the Art.....	49
4.0	Technical Approach.....	50
5.0	Technical Details .....	50
6.0	Special Resources Required .....	51
7.0	Project Plan .....	51
8.0	Testing.....	53

## 1.0 Objectives

The end all be all goal of the project is to create an application for android that allows a person who seeks to have a voice that is more in line with their own identity, 'voice training' the process that people who seek to change their voice is often carried out by transgender and non-binary individuals the reasoning why people decide to train their voices vary e.g to help with passing, improve mental health are the main reasoning, this process can possibly take a number of years and has plenty of factors to account for that vary from person to person.



To make the process of voice training more streamline the application will need to meet certain objects, those objectives being as follows:

- Use gathered knowledge from own voice training experience.
- Track progress of the user's voice.
- Display the progress in an understandable manor.
- Explain the information that is being displayed e.g frequency ranges for masculine and feminine sounding voices.

Reaching these objectives should help streamline the practice of voice training making the users goal more obtainable.

## 2.0 Background

This project is rather personally to me as I am trans women myself, I remember the moment my voice dropped when I was younger and it was such a daunting experience, my voice dropped when I was at a school friend's birthday party and all the other kids would not stop pointing it out since then my voice was one of the main sources of my own depression when growing up, I know I am not the only person with this kind of experience which is why I went to set on helping with the same struggle as my own.

How I plan meeting my objectives is by using knowledge from my own training and my experience with speech therapist Ruth Duffy, to track the users voice progression I plan on using a graph to show the frequency of the user's voice over time which in term would the users progress in an understandable manor.

## 3.0 State of the Art

From my own experience of voice training, I have encountered quite a few applications that already exist, the one I have the most experience in using is called 'Voice Pitch Analyzer' how this application gathers the user's frequency over time involves recording the user read off paragraphs that appear on screen and then graphing it, the application I seek to develop differs in the way I want to collect the data, instead of having the user read off text I plan on having the user perform vocal exercises that I have been practicing myself which I had gotten from a speech therapist I had seen which may give better results in the long, I'd imagine most applications in this area aren't made without the consolation of a speech therapist which may help my application stand out in comparison to other application.

There are other applications that I have attempted to use as well such as 'Voice Tools' the main issue I had found with this application is that it does not save your progress in the application but instead you can send yourself an email of your last sessions results within the application, this seems rather unconventional, in my opinion so having a way of saving the results within the application would make the user experience must better and the third similar application I have found is called 'Vocal Image' this application has a variety of features not just aimed towards the trans community but to people who want to get better at singing as well however the main grip I have with this application is that it is locked behind a pay wall asking users to subscribe to an annual payment in order to use the app, I believe making my app free to use or even being free with ads is a much is much better alternative and allows for a much bigger user base as not everyone can afford to paying for subscriptions.

So for features that none of the applications that I have discussed is that having a feature where a user could get in contact with a speech therapist would be very beneficial to the user after my

own experience with talking to a speech therapist I do believe it could benefit potential users which may even be a reason for potential users to check out my application and lastly I believe a diary feature could be beneficial to the users mental health as they note their feelings that could want to discuss with the speech therapist at a point in the future or even provide the opportunity to reflect on themselves which may result in finding features of users voice that they do like.

## 4.0 Technical Approach

During my work placement at Revenue, I gotten to experience working within an agile development environment, having worked in multiple sprints I understand the benefits of adopting the agile development methodology, with this experience in mind I believe it is best for me to approach this project following the agile development methodology.

As I am a part of the target audience, I'd like to believe I have a good sense of what most transgender people would like to have in a voice training app in addition I have also discussed what kind of features should be included from the perspective of a speech therapist.

Once I have requirements identified I'll be breaking them into stories, these stories will be organised into three different priorities those being low, medium and high, these priorities will be used as a basis as to identify which stories need to be completed sooner rather than later.

To keep track of milestones my development schedule will follow a schedule similar to Revenue as that is what I am most used to, Revenue works in two-week sprints before the start of a sprint the team would meet deciding how many of the stories in the back log would be moved into the next active sprint. I believe if I setup a Trello board, create a back log, every couple of weeks I will assess myself in terms of how many of the active stories have been completed against how many remain unfinished and need to be brought into the next sprint.

## 5.0 Technical Details

For the upcoming project I wanted to use a platform that would give me the option to use Java, this led me to Android Studio which is an IDE used for Android App development, it gives developers the option to use Java and Kotlin which is exactly what I need with this in mind I have decided to use Java as the implementation language for my project because it is the language, I am most familiar with as the majority of the course work, I have done here at NCI has been with Java throughout the years. By using a language, I am already quite proficient in it should enable me to be more efficient in my work and should hopefully make debugging a much easier process, to me using Java is the most ideal choice for this project as it just makes the most sense to continue using rather than using a language, I have less exposure to or even tackling a new language all together, since this will be an android application, I have decided to use the Android Studio IDE since it has its own GUI designer similar to which of the NetBeans IDE.

Since the project is based in Java as that is the programming language that I am most familiar with, I wanted to find a suitable library to enable the ability to take vocal inputs. I came across a library in Android called 'mediaRecorder' which seems fairly promising as the vocal inputs is a big part of the project as I can't do anything without it but downside to the library it can only give audio inputs so I need another library to process the vocal data, upon my research I found a library called 'jTransforms' which is a library that uses an algorithm called 'fast Fourier transforms' or FFT for short, this algorithm should hopefully give me the means to get the frequency's from the inputs.

Since I want the user to have a clear understanding of their own progress, I need a way of displaying the frequency data that application has collected, I had stumbled across another library that may have the solution I have been looking for, the library in question is called 'Android Plot', this library should allow to create graphs which I can plot the gathered frequency's.

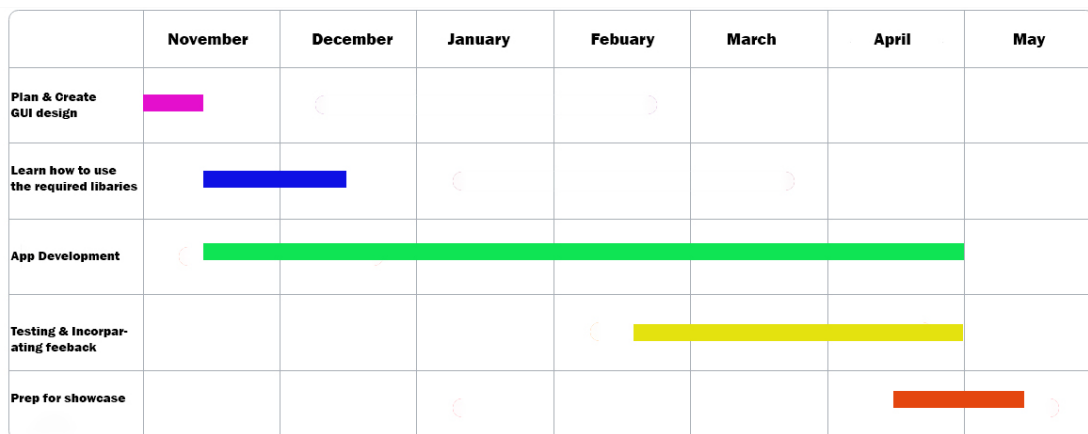
For storing the results

In summary, I have chosen Java to be my implementation language due to my familiarity of the language after all my experience with my work at NCI which may help me in the process of debugging, since vocal input is one of the most important requirements for my project, I am using the Android package called Media to enable the ability to take in the vocal data, since this package does not have method of getting the frequency from its input I had to find another library that had the ability to do so the library 'jTransforms' using to perform Fast Fourier Transform on the input to work out the frequency and lastly to ensure users have a clear understanding of the progress made by them, I intend to take full advantage of the 'Android Plot' library which help me create graphs and plot frequency data upon.

## 6.0 Special Resources Required

Purely for testing purposes, I had an idea that it would be interesting to compare the test results between a mid-range or even high-range microphone with equipment such as a pop filter to ensure that the audio is more clean and crisp sounding in comparison to using the average smart phone microphone which is more realistic for test scenarios, I wouldn't call it required per say but I'm really so curious as to how the frequency results will differ between the two, in terms of hardware that is actually need to take on the project at the time of writing I believe I may have everything needed.

## 7.0 Project Plan



Figure

1: Overall Gantt Chart.

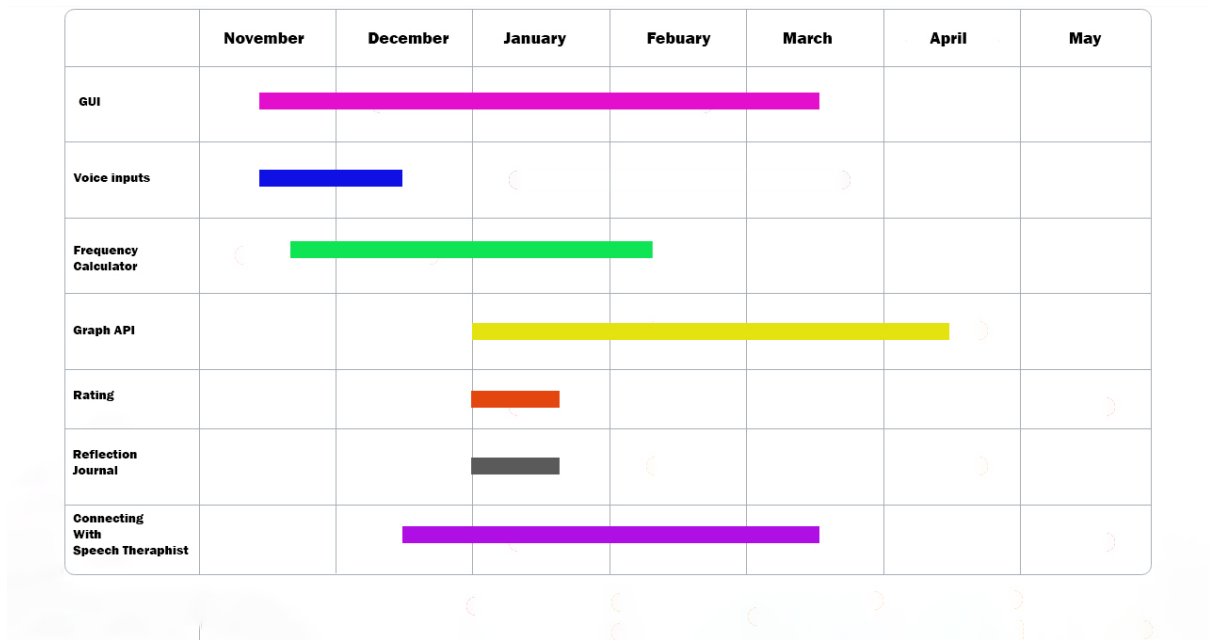


Figure 2: Dev Gantt Chart

In the Overall Gantt chart, I have made estimates on how long different process of the project may take for me to complete, this chart is subject to change as the accuracy may change over the next few weeks or months. I will now go through each of the different sections in the Gantt chart to further explain the timelines.

### Plan & Create GUI Design:

This part is fairly straight forward, every application is expected to have a robust and or dynamic GUI design that is easy for users to use, for coming up with the design I will look at similar applications and try improve on aspects of designs in my own, the only difficult part would be creating the GUI in android studio I have not much experience with.

### Learn how to use the required libraires:

Since this will be my first using these libraries in an application, I won't be an all-knowing expert on using them, this time will be dedicated to creating 'dummy' applications so that I can get a good grasp on how to use them most efficiently in the actual project.

### App Development:

After getting comfortable with the libraries that I have chosen for the project the development of the application will have begun, this part of the project will obviously take the longest out of all the aspects of the project. During this I will be creating all the different methods that are required for the application to function.

### Testing & Incorporating feedback:

To ensure that the application is working as it is intended to, I will have setup different methods of testing it, this will include the following, white box testing, automated testing and manual testing done by myself and participants, I will go into more detail in section 8.0 Testing.

During development of any project feedback from peers should always taken into consideration, my project is no different, any and all feedback on my project will be taken into account to ensure that the application is the best of my work thus far.

### Prep for showcase:

Lastly at this stage I will be making sure that everything related to the project is flawless and prepare for any questions regarding it.

## 8.0 Testing

Testing is one of the most important aspects of any application but sometimes it can be really underappreciated, during my work placement at Revenue I was primarily testing different submissions to the API that was being developed at the time, I believe that my work as a tester in a professional environment will give me an edge in testing my application as well as having completed the testing module during my 2<sup>nd</sup> year at NCI.

So far my strategy for testing my application is as follows, I plan on carrying out different white box testing methods such as unit tests, there are a plethora of different testing libraires available for Java applications such as 'JUnit' which is actually the default testing library for android studio but I would also need to take into consideration the unit testing of the GUI which I haven't had much exposure too, from my research I gathered that the testing library 'Expresso' is by many developers for testing the GUI of an android based application which makes me believe it would be a good choice to go with for testing my application.

In the regard of manual testing the application I plan on participating myself as well asking other people in the trans community to take part in manual testing the application after I fill out the appropriate ethics paper work.

Given the different testing strategies I plan on taking full advantage during my time working on this project I believe that the first actual release of my project should fairly competent in its own regard and completely bug / issue free.

### 7.2. Ethics Approval Application (only if required)

N/A

### 8.1. Reflective Journals

---

#### Supervision & Reflection Template

---

<b>Student Name</b>	Ruby Rapple Kehoe
<b>Student Number</b>	X20382263
<b>Course</b>	BSC In Computing 4: Software Dev
<b>Supervisor</b>	Frances Sheridan

Month: October

**What?**

Reflect on what has happened in your project this month?

So far this month I have been thinking about how to make my project more unique, based on the feedback from my project pitch video.

**So What?**

Consider what that meant for your project progress. What were your successes? What challenges still remain?

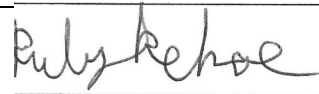
I have gone back and forth with my speech therapist Ruth Duffy about a feature she suggested, in my voice training she got me to download an app called 'Perfect Piano', the goal of using this in the training to try make an 'oooo' type of sound at the same pitch of certain keys.

**Now What?**

What can you do to address outstanding challenges?

I want to talk to my supervisor to see what she thinks about this feature before implementing any plans to develop the feature while I think of other ideas to make my project more unique.

**Student Signature**



---

Supervision & Reflection Template

---

<b>Student Name</b>	Ruby Rapple Kehoe
<b>Student Number</b>	X20382263
<b>Course</b>	BSC In Computing 4: Software Dev
<b>Supervisor</b>	Frances Sheridan

Month: November

**What?**

Reflect on what has happened in your project this month?

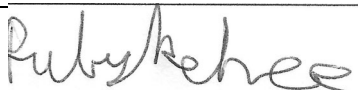
In November after getting some feedback on my project pitch I had done more research for ways I could make the project stand out more and what other features could be implemented into the project. I also had queries about using a database for my project, I had trouble with using a database service called 'Planet Scale' but my supervisor recommended that I look into 'Google firebase'.

**So What?**

Consider what that meant for your project progress. What were your successes? What challenges still remain?

I have decided on that I will add rating feature to the project that will allow the user to rate how they feel on their own voice and also include how loud the user's is in decibels.



<p><b>Now What?</b></p> <p>What can you do to address outstanding challenges?</p> <p>I will be planning on how I can actually get the result of the decibels and I plan to learn how to use Google Firebase</p>	
<p><b>Student Signature</b></p>	

### Supervision & Reflection Template

<b>Student Name</b>	Ruby Rapple Kehoe
<b>Student Number</b>	X20382263
<b>Course</b>	BSC In Computing 4: Software Dev
<b>Supervisor</b>	Frances Sheridan

**Month: December**

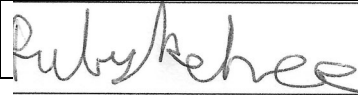
<p><b>What?</b></p> <p>As the mid-point deadline approached, I had encountered an issue that may have been a blocker for the project, Android Studio lets me boot my application within its own emulator, the issue I was having was I couldn't get the prompt from emulator to request permission for accessing the microphone in order to record the user's voice.</p>
<p><b>So What?</b></p>

If I am not able to get the prompt asking for permission to use the microphone, I would not be able to demo anything for the mid-point and would also block development on my application's other features.

**Now What?**

To prevent this roadblock from stopping me reaching the mid-point deadline I had to do some research using Stack Overflow and other sources which I have referenced in my code to figure how to have the prompt appear when testing my code.

**Student Signature**



**Supervision & Reflection Template**

<b>Student Name</b>	Ruby Rapple Kehoe
<b>Student Number</b>	X20382263
<b>Course</b>	BSC In Computing 4: Software Dev
<b>Supervisor</b>	Frances Sheridan

**Month: January**

**What?**

This month I had encountered another issue regarding how I have setup the different screens of my application, they I have my application screens setup is through the use of fragments which let me reuse assets from other screens, which in theory is useful but the issue that is occurring is that I cannot hide visuals such as buttons from other screens on the screen I am currently working on.

**So What?**

This could potentially lead to a roadblock for the GUI development as if I cannot hide assets that are not relevant to certain screens could end up confusing end users.

**Now What?**

Right now, I have been working to resolve this issue which I have a few options of doing, I can either figure out how to hide the assets or start the GUI from scratch, I would rather not do the latter as I would rather use as much time as possible on main functionalities so right now I'm trying to learn more about fragments.

Student Signature

Ruby Kehoe

### Supervision & Reflection Template

Student Name	Ruby Rapple Kehoe
Student Number	X20382263
Course	BSC In Computing 4: Software Development
Supervisor	Frances Sheridan

### Month: February

#### What?

Reflect on what has happened in your project this month?

During this time, I was able to get the Frequency calculation to start working, to ensure that I would be getting a consistent sample size for the calculation I made changes to how the application records audio, when the start recording button is pressed, a timer will appear that starts at fifteen and will countdown to zero, when the timer reaches zero the recording will have stopped and a mp3 file of the recording will have been created, doing this should allow for me consistent results.

#### So What?

Consider what that meant for your project progress. What were your successes? What challenges still remain?

While I do believe what I have accomplished is a feat of itself but as previously discussed with Frances I am going to incorporate other metrics such as decibels I plan on trying to figure out how I can work out the users dB from the obtained frequency .

**Now What?**

What can you do to address outstanding challenges?

I plan on trying to either take my existing byte array in frequency analyser class and see if I can get the dB value based of that or try to see if its possibly to work out the dB value based on the resulting frequency value.

Student Signature

**Supervision & Reflection Template**

Student Name	Ruby Rapple Kehoe
Student Number	X20382263
Course	BSC In Computing 4: Software Development
Supervisor	Frances Sheridan

**Month: March**

**What?**

Reflect on what has happened in your project this month?

This past month has been somewhat productive but there has been a minor set back that I am still confused on at the moment, firstly I was able to come with a much simpler way of calculating the decibel, instead of having another method to calculate the decibel from the byte array made up of the mp3 file I can simply take the resulting frequencies and throw them into this formula  $dB = 20 * \log(\text{value1}/\text{Value2})$ , using the frequencies as the value I can a get an accurate reading of how loud the users voice is. There is however an issue I am now facing, at some point during development I had made a change to my code that caused it to only work in my android studio instance, attempting to use the application in emulators such as bluestacks will result in the frequency to result as -1 and the dB to result as NaN.

**So What?**

Consider what that meant for your project progress. What were your successes? What challenges still remain?

Considering the decibel calculator was reduced from being its own method to literally two lines of code was a great success and big-time saver of trying to debug it is a big deal in my own opinion but the issue of the basically the main feature not working outside of android studios own emulator worries me as it is a pretty big blocker in terms of the grand scheme of things.

**Now What?**

What can you do to address outstanding challenges?

I am going to thinker around with older apks of the project I have saved on one of my drives to see what is causing the mic input to not work outside of android studios emulator and compare it to the current version to see a change I had made may have caused the issue without me realising it.

Student Signature

**Supervision & Reflection Template**

Student Name	Ruby Rapple Kehoe
Student Number	X20382263
Course	BSHC 4: Software Development
Supervisor	Frances Sheridan

## Month: April

### What?

Reflect on what has happened in your project this month?

So after finally getting the frequency calculator sorted out, I was able to move onto other functions of my project that were defiantly behind, during this month I was able to work on Google Firebase setup, account creation, login system, make the app remember if the user is logged in after the app has been closed and the journal feature. During these tasks there were some blockers that came up that stalled me for a day or two on each.

### So What?

Consider what that meant for your project progress. What were your successes? What challenges still remain?

As mentioned before there were some blockers, I'll talk about them in order that I had encountered them. Setting up the Firebase real time database was fairly easy same goes for connecting my project to the database, following tutorials on how to setup account authentication was straight forward enough till I had encountered an issue once the user had logged in, once the user logs in everything would appear to be fine, but when closing the app and attempting to reopen it I would be met with a pop up saying that it had crashed when trying to launch, after spending a couple of hours going through both the login and registration classes I found that I set the app to launch on a fragment instead of an activity if the user was already logged upon launching the app which caused the crashing issue. The other issue I had encountered was setting up the design of the page where a user would make a new entry to their journal, the design is fairly simple, a small input field to give a rating between one and five at the top with a larger input field below that for the user to type how they feel alongside a cancel and submit button, the issue was is that the larger input field was not clickable with the current layout scheme I was using so I ended up switching from constraint to linear layout, currently speaking the Journal function is about 80% complete, the GUI for it is finished, the user can submit a new entry with a rating and have it saved to the Firebase database I am currently having a bit of trouble getting all the entries being displayed on the main journal screen.

### Now What?

What can you do to address outstanding challenges?

With the majority of the coding challenges out of the way, once I am able to get information from the database back onto the app it is basically time to start the test plan which would cover system, integration and unit testing.

Student Signature	Ruby Kehoe