Hear Together

Design Document v. 3

Team Number: 48
Client/Adviser: Mat Wymore

Team Members:
Jessie Rutledge - Communicator/Full Stack Developer
Andrew Peterson - Backend Developer
Paul Licata - Full Stack/QA Developer
Richard Smith - Frontend Developer
Roger Ferguson - Test Engineer

sdmay20-48@iastate.edu
http://sdmay20-48.sd.ece.iastate.edu/

Executive Summary

Engineering Standards and Design Practices

Practices used:

- ➢ DRY principle
- ➢ Defensive Coding
- ➢ Regular Refactoring
- ➢ Simple Design
- ➢ Version Control Systems

Engineering Standards considered:

- ➢ IEEE 1448a-1996 - Software Life Cycle Processes
- ➢ IEEE 1233-1996 - IEEE Guide for Developing System Requirements Specifications

Summary of Requirements

- ➢ Android application running on version 7.0 or later
- ➢ “Session” system in which users connect to a single instance
- ➢ Supports members speaking into headset/earbuds with microphone
- ➢ Delay between speaking and hearing is <= 1 second
- ➢ Users can connect to other users
- ➢ Cloud storage capabilities via Firebase

Applicable Courses from Iowa State University Curriculum

Jessie Rutledge:
- ➢ Com S 388: Introduced the fundamentals of android studio development. Experience with working on larger android projects in a team using git. Worked with many of the concepts used and to be used in Hear Together
- ➢ SE 309: Experience in designing projects. Provided ideas relating to how to organize code structure in a way that allows for expansion in the codebase without much issue. Practical use of MVC within a java project.

John Ferguson:
- ➢ SE 309: Worked in a team on a project learning how to use tools like Git
- ➢ SE 319: Learned of the Software Development Process
- ➢ Com S 227: Programming in an object oriented environment

Andrew Peterson:
- ➢ SE 309: Worked on a software project in a team environment. Learned how to use build tools like Maven and Gradle.

- ➢ SE 329: Learned how to manage risk in a project. Wrote project documentation analyzing risk. Managed the software development life cycle up to project completion.

Paul Licata:

- ➢ Com S 227, 228 - Learned about various complex data structures and learned how to program with an object oriented language.
- ➢ Com S 309 - Worked in a group to make an android application, first hands on experience with Android Studio.
- ➢ Engl 314 - Learned the basics of technical writing and wrote comprehensive design documents for companies.

New Skills and Knowledge Acquired

- ➢ Cloud experience using Google’s Firebase
- ➢ Android Studio Development
- ➢ Connections over WiFi P2P
- ➢ Sound processing
- ➢ Android Framework Knowledge
- ➢ Experience with Java libraries

Table of Contents

List of Figures/Tables

List of Figures

Figure		Page
1	Design Plan showing interaction between parts of the system.	9
2	UI Mockup of planned visual design of application.	10
3	Flowchart of typical use case in application.	11
4	Flow Diagram of process when running tests.	19

List of Tables

Table		Page
1	Roadmap for the Hear Together project.	15
2	Breakdown of reasoning for task length	16

1 Introduction

1.1 Acknowledgement

Mathew Wymore, our client, will contribute assistance in the form of technical advice, and user feedback to the project.

1.2 Problem and Project Statement

During meetings, it is important to be able to understand each person in the conversation without needing to stop and clarify what a person said. However, when there is loud background noise, or the participants in the conversation have some level of hearing loss, it can result in frustration or even technical misunderstandings.

This project seeks to ease the difficulties faced by those who are hard of hearing in group settings. The planned product would be an application that can be downloaded to an Android device. The device would send a stream of audio data to the listeners phone to be transformed into sound that is intelligible for the listener to understand. By the end of the project, we hope to have a working application that will reduce the number of sound related disruptions during meetings.

1.3 Operational Environment

The software should be run in an indoor environment with a minimal level of background sound polluting the environment. On top of this, the hardware microphone must be outside in the open, free to capture noise. Hardware capability to withstand the elements is dependent on the user's device and is beyond the scope of the project.

1.4 Requirements

- ➢ Android application running on version 7.0 or later
- ➢ "Session" system in which users connect to a single instance
- ➢ Supports members speaking into headset/earbuds with microphone
- ➢ Delay between speaking and hearing is <= 1 second
- ➢ Users can connect to other users
- ➢ Cloud storage capabilities via Firebase

1.5 Intended Users and Uses

The intended user base are people with an impaired but still somewhat working level of hearing. More specifically, people who can hear but because of circumstances, cannot hear as well as your average person. These users are intended to use the product to amplify or modify the volume of their meetings with other people in order to be fully engaged without further accommodation. Another intended use of the application is to provide a low cost solution to some hearing issues because of a reliance on just an android device and earbuds/headphones with a microphone on them, which many people already own. There is also the secondary consideration of non-hearing imparied users who are utilizing the app to increase their ability to communicate with those who are impaired.

1.6 Assumptions and Limitations

Assumptions:

- ➢ The product will be used indoors and the users will be stationary
- ➢ The users will be in close proximity of each other
- ➢ There will be a single user connected to a "host" in a session
- ➢ The end product will be available to all devices that can run it
- ➢ A minimalist UI will be used for ease of use
- ➢ Java will be used to code the application

Limitations:

- ➢ The system must operate on Android
- ➢ The system must use phone compatible headphones
- ➢ Is not designed for users that are not hard of hearing or not communicating with someone who is hard of hearing
- ➢ Must be completed by May 2020
- ➢ The system must have as little delay as possible

1.7 Expected End Product and Deliverables

The end product is an Android smartphone application that the user may install and then open to be greeted by the user interface. The user interface will be clear enough for the user to be able to do a basic interaction of creating or joining a session in which they may engage in interaction with assisted hearing. The assisted hearing will take the form of a microphone input directly sending sound information into the android device and outputting modified sound as output to a set of earbuds or headphones.

Deliverable 1: User interface

There will be a user interface that allows for ease of navigation for an average user. This user interface should show how to initiate a session within the application. The user interface should also template some inactive features such as sound adjustment for an individual.

Deliverable 2: Sound "engine" with basic P2P connectivity

This is more of a demo than a deliverable due to the mainly backend nature of this task. The software should be able to take in an input in the form of data that represents sound, and be able to output a modified version of that sound. The ability to input sound from foreign android devices (mocked for this deliverable) should be completed and the classes used to handle this interaction should be developed. Phones should be able to connect to each other on a very basic level but not much should be done with that connectivity besides proof of concept. Engine should be able to process audio into component frequencies, adjust the intensity of each frequency per user input, then recombine.

Deliverable 3: Phone to phone interaction v2

Phone to phone interactivity should be expanded upon, as well as host to server/database interactivity should be finished. The connection handlers should be able to take in real data from other android devices and funnel them into the sound engine, which spits out the resulting sound and delivers it back to the devices. Interactivity and verification on the server should also be complete, should the user opt to store data on the database. Considering this is the final thing to be worked on, the application should be completed outside of some bugs that may arise from unification of features. Those bugs are intended to be fixed on final release shortly after this deliverable.

1.8 Adjusted End Product and Deliverables

Due to factors such as limited educational resources, coronavirus quarantine, as well as personal issues, a consultation with our client has resulted in the following adjusted deliverables for the product.

Deliverable 1: Single Connection

The application will be able to support a single connection between two devices transferring data via WiFi Direct. One phone will act as the “host” of the session and be treated as a server performing all processing, while the other phone acts as a client connecting to the host.

Deliverable 2: Sound Modification

A proven proof of concept ability to modify sound within the frequency domain as a result of some user defined setting will be completed and commented out within the project. The work done here will be used as a stepping stone for future groups to expand upon in the future.

2. Specifications and Analysis

2.1 Proposed Design

The current proposed design is an application that takes in the sound from its own microphone, and uses the android accessibility tools to isolate the sound. Then, it will send its sound stream to a connected phone, in a leader-follower setup. Finally, we will use the fourier transform to act as a sort of bias on the sound data within the frequency domain according to the users settings. This will fulfill the requirement related to individual members volumes, and should ideally stay under a delay of 1 second as required. This process will not occur all the time, but only when a session between users of the application is set up, to allow for the requirement related to the “Session” system.

2.2 Design Analysis

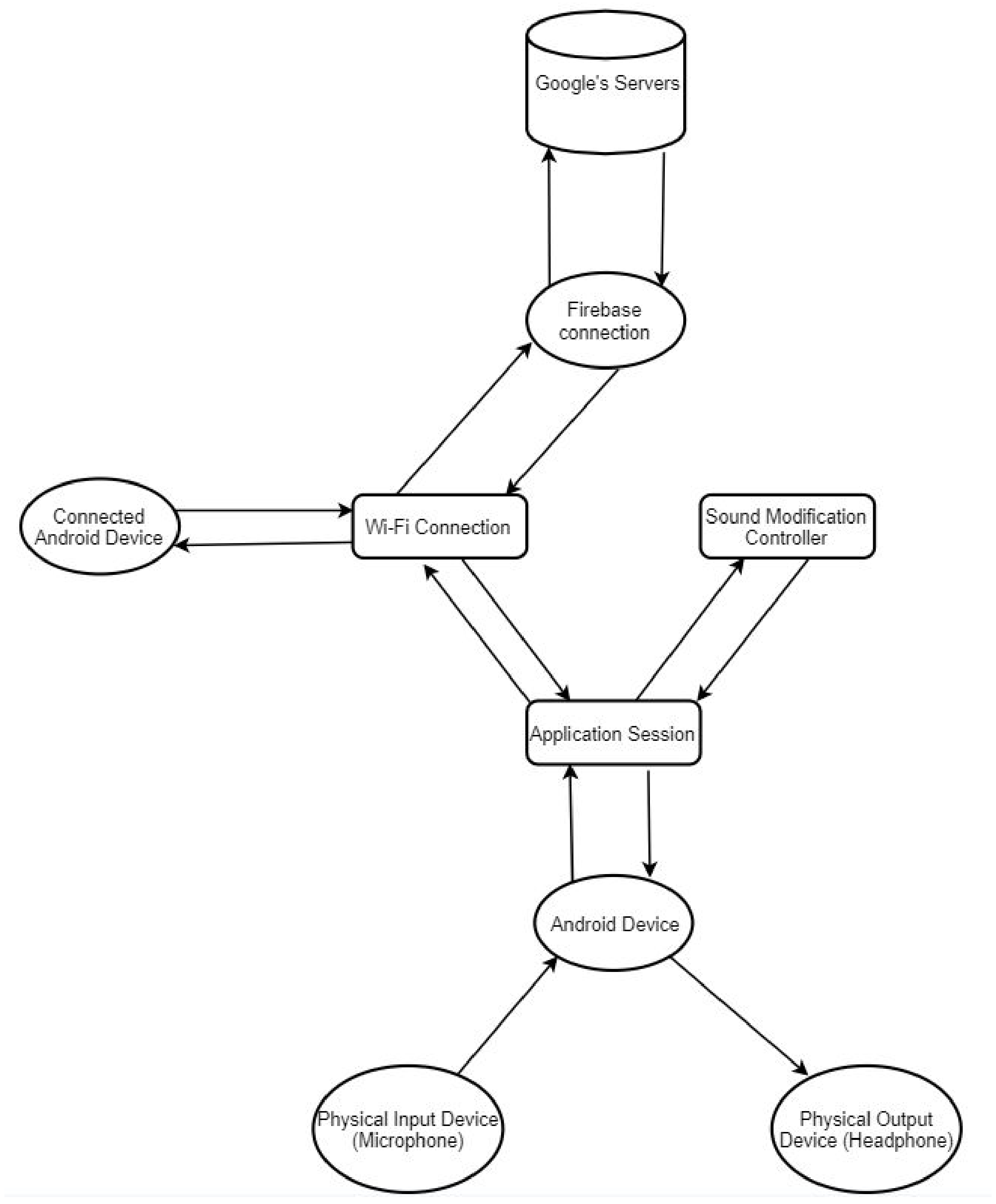
The proposed solution is an Android application that establishes a session that “contains” the conversation a group of people will have, and this session will provide modifications to the volume and overall sound of the conversation. This has a strength in that this is a very concrete and manageable Java object that is easy to visualize without even having done work on it yet. On top of that, being able to represent interaction as a singleton allows the group to manipulate it more freely as we can compartmentalize it and do concrete work on the data for it. A weakness of this design approach would be the opt-in nature of it, unless someone actually has the app, they can’t just jump into the conversation. Another consequence of the Android based solution is the Android platform itself, or more specifically only being on the Android platform; it would require additional work to add support to other platforms.

Audio processing will be done by two systems. The first is Android voice recognition, used to filter background noise from the conversation. This program is available and accessible, but may not be as effective as other systems on the market. The second is an audio processor; this system uses a discrete Fourier Transform to separate the signal into component frequencies, then adjust the volume of each frequency based on the user’s settings. This will be using the JTransforms library, as it offers both a (somewhat) quick means of having an initial implementation while offering room for the team to implement a more efficient “sparse” FFT algorithm later in development.

2.3 Development Process

We intend to do an agile approach for the project. We are using trello as a scrum board and our weekly meetings are every Wednesday at 9:30 AM for one hour. People will be assigned tasks and are to reflect on their past work for the week during the meeting. This meeting is paramount to our organization as it is our only regularly scheduled in-person meeting. A Sunday group work meeting is also established as a guaranteed means of work being done in an environment in which communication should be easy and low effort. In case of communication being lacking, the weekly agile meeting is intended to promote honesty on progress and allow us to combat issues proactively in the project.

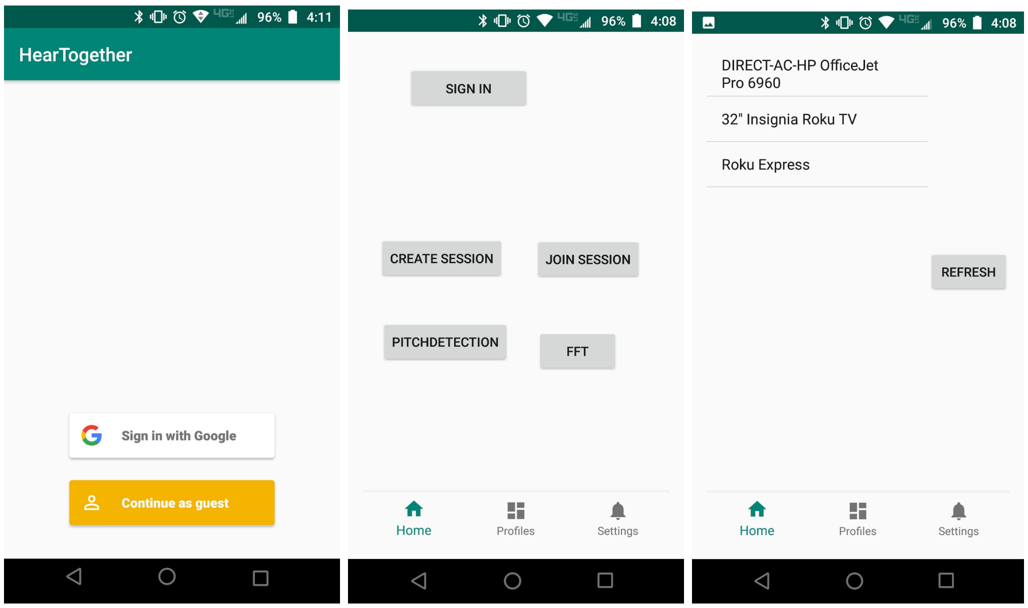
2.4 Design Plan



(Figure 1)

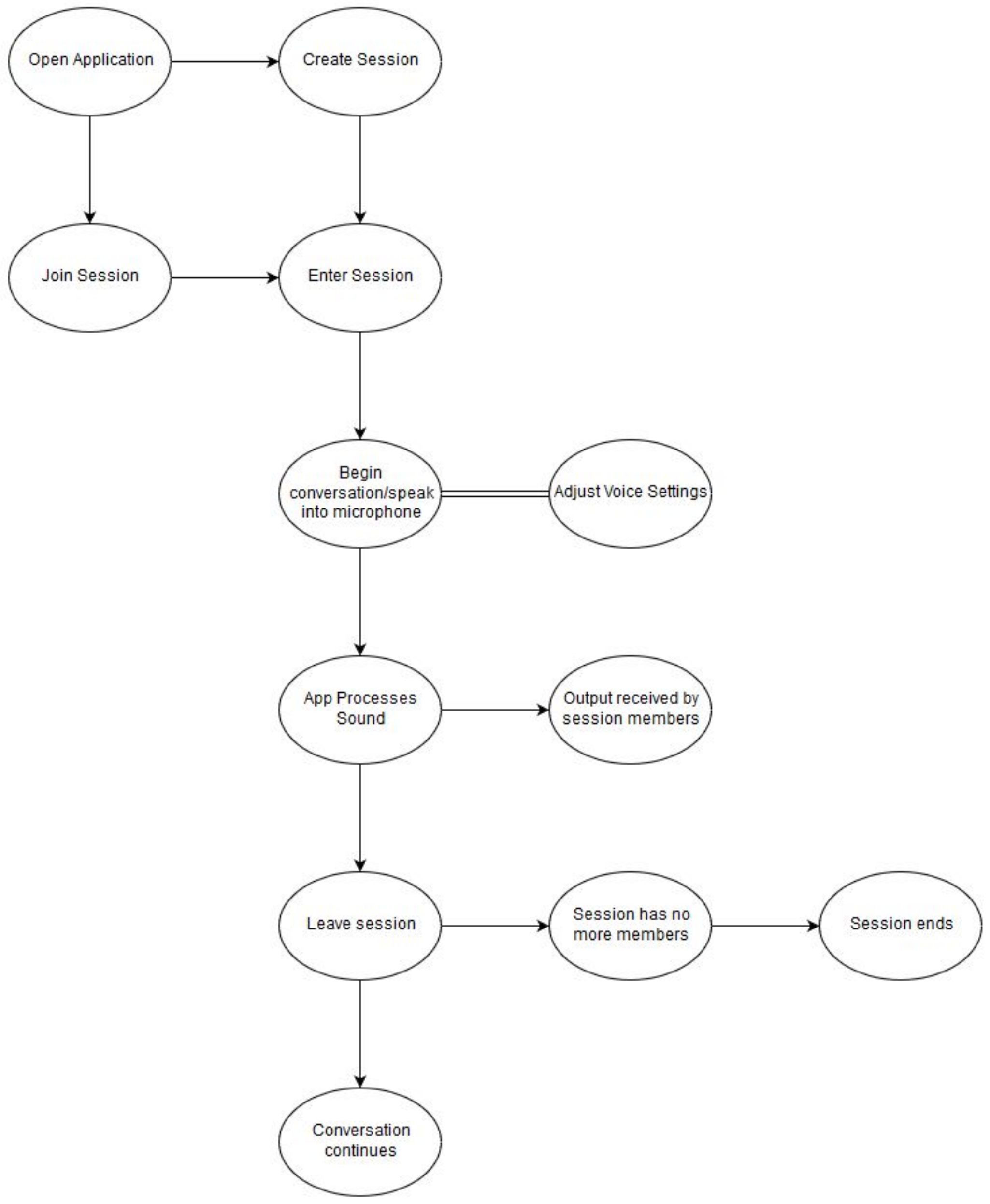
Our design makes use of the idea that concerns should be separated. Our application takes in input from a device, and funnels that information into the connection threads. These threads are separate for client, server, sending, and receiving information. From here the conversation data can be stored if desired, or sent directly to the sound modification controller. This controller will then send data back to the handler where it will distribute information back to the devices to be output in an analog format on headphones.

Below are images of the current interface:



(Figure 2)

The UI is clean and simple with the main navigation points (we will call them “tabs”) being on the bottom of the display. The first tab shows the actual options about the session system itself, as well as some profiling information, which is essentially a preset for the options set in other tabs. The second tab details options relating to the sound itself. The third tab shows the user some basic data regarding the output based in settings set in the second tab. The final tab is more relevant to the final deliverable in which extra features beyond our customer’s scope (but has been approved for a stretch goal) is made, currently this is only planned to pertain to storing conversations on a database.



(Figure 3)

A figure detailing the average use case of entering and exiting a session is displayed above. The user is greeted with two initial options which lead to entering the session state. From here, there is generally only a linear UI pertaining to the act of having the actual conversation up until the point the conversation ends. As you may notice, a session does not end until the members leave the session, what is not outlined in this diagram is the edge case in which the person who started the session leaves the ongoing conversation, although we are considering that outside of the scope of the *generalized* use case, it is still something we will handle by ending the conversation outright on that as well.

3. Statement of Work

3.1 Previous Work And Literature

Current products for the hearing impaired mainly center around the hearing aid. Most existing products for android connect a device to a hearing aid, with development allowing direct connections instead of through an intermediary. Our application will instead send audio to the user’s headphones or earbuds, making this product suitable for persons who require occasional hearing assistance but not constant hearing aid use. Additionally, our project networks several devices together to improve the quality of collected audio, whereas other products simply use the input from one device.

3.2 Technology Considerations

Android and android studio offers us a powerful and robust platform to develop that has over the years added useful functions and classes that will offer a lot of productivity efficiencies to our work. Android studio, as well as android devices obtained through the university, are also free resources, this is the primary reason we have an estimated monetary cost of zero for the entire project. Unfortunately, Android as a platform has drawbacks with the type of project we are doing in that sound modification will likely have a latency issue. This is a drawback in comparison to a focus on iOS as our research into the subject has shown that the iOS platform is better in this regard. We have opted to tackle this weakness head on and have a criteria of a maximum one second delay for the modified sound to be audible.

3.3 Task Decomposition

The tasks, in no particular order, are:

- ➢ Android activity development
- ➢ Sound Processing Development
- ➢ Advanced UI Features
- ➢ Communication Handling
- ➢ Server/Backend Setup
- ➢ Backend/Frontend Connectivity
- ➢ Phone to phone connectivity
- ➢ Sound Processing Finishing Touches
- ➢ UI Finishing Touches
- ➢ Final Bug-fixing

The task decomposition is mostly flat, allowing us to work on all parts of the application at once, except for those which are finishing touches on another task. These include the “Advanced UI features” task, the “Sound Processing Finishing Touches” task, the “UI Finishing Touches” task, and the “Final Bug-fixing” task. There is a degree that all of these tasks will require some understanding of android before completion, and therefore task “Android activity development” is parallel with nearly all these tasks.

3.4 Possible Risks And Risk Management

Risks are entirely relating to knowledge of areas of the project. Many of the project members have not done any android development in the past, nor have many of us processed sound in a program either. There is also a concern with being accurate on the adjustments needed to sound, as well as a possible roadblocks with attempting to have a low latency device on the Android operating system.

To manage these risks, we will mostly rely on trying to reduce the risk as much as possible. We can do that through quickly iterating to build prototypes to understand the problem space as much as possible. We set up spikes to quickly learn about a subject so that we can reduce the risk. FInally, we intend to use prebuilt libraries to avoid having to take on risk whenever possible.

3.5 Project Proposed Milestones, Adjusted Milestones, and Evaluation Criteria

In no particular order the proposed initial milestones include:

- ➢ UI Completion
    - ○ Client will use device and give input or recommendations for any changes until UI is satisfactory
- ➢ Sound modification controller produces any workable result
    - ○ The controller needs to be able to take in a sound, and output a modified sound that resembles the desired output enough in order be able to further build upon project components relying on the controller’s output.
- ➢ Sound modification controller produces satisfactory and ideal result
    - ○ The controller must be able to output the desired modified sound in a state deemed as satisfactory enough to provide an average user.
- ➢ Server backend accessibility is established
    - ○ Firebase is added and used within the project for cloud features
- ➢ User connectivity between other users is established
    - ○ WiFi Direct connectivity between two devices is shown by application
    - ○ Multiple devices can connect with no established limit on the number of connections
- ➢ Session system shows reliability to relay information to users

    - Activity from the backend logic can be initiated from within the session controller or activity
    - Result from logic can be sent back to the session controller
    - Session controller can service multiple (an array of) users within it

As development progressed, difficulties have let to a consultation with the client and the following adjusted milestones are established:

- ➢ Single connection over WiFi Direct
    - The application is able to support a single connection between two devices transferring data
- ➢ Sound Modification
    - WiFi P2P connectivity between two devices is shown by application
    - The application has a proven proof of concept ability to modify sound within the frequency domain as a result of some sort of user defined setting

3.6 Project Tracking Procedures

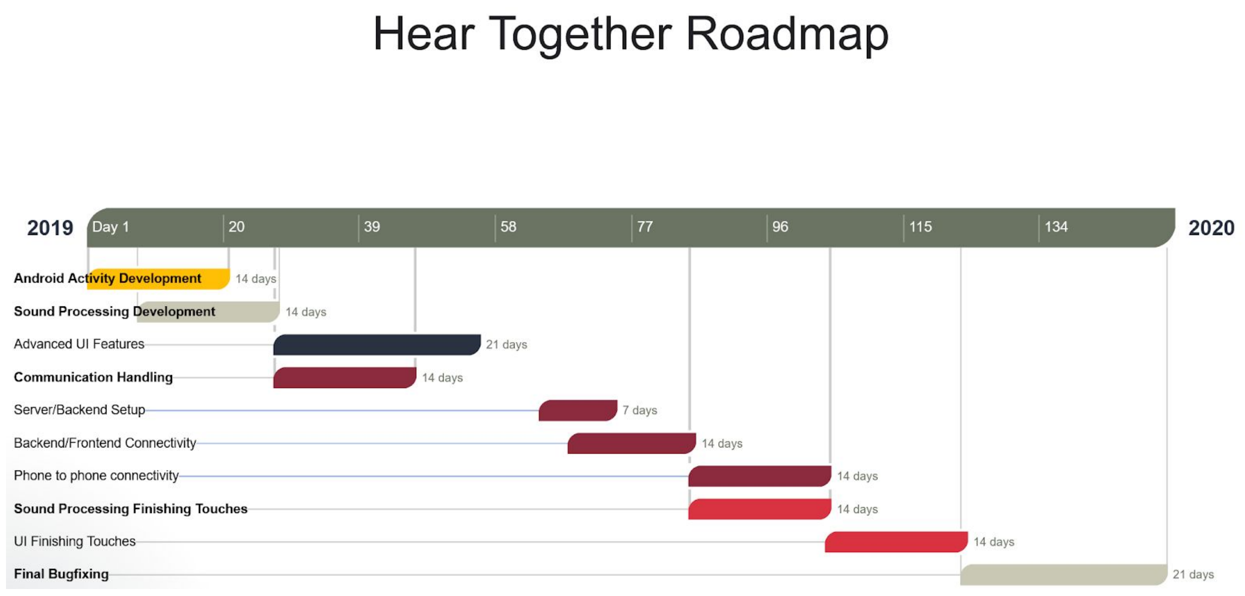
Communication is to be done on Discord. GroupMe was initially used and intended for informal or very short term planning, while Trello was also intended to be used for very formalized and long term planning. Discord in initial stages was to be used as an interim or communication platform to cooperate on the actual work itself. We believed that these platforms are specialized for the uses we intend to use them for, which is why the division in communication was done in this way. The division caused issues for the group as it was too decentralized. Eventually, the group decided it was best to rely primarily on Discord for all communication needs while relying on GitLab for a high level view of tasks. Weekly scrums were more heavily emphasized and held on Wednesday mornings at 9:30 AM.

3.7 Expected Results and Validation

If the product functions as intended, a person using the application will be able to effectively connect to another user. On top of this, a proven ability to take in sound with the microphone and do some level of operation on it within the frequency domain should be established. WiFi Direct will be implemented in such a way in that expansion and/or refactoring of the sound processing should have minimal impact on connections due to abstracting the data sent over the WiFi connection.

4. Initial Project Timeline, Estimated Resources, and Challenges

4.1 Project Timeline



(Table 1)

The initial timeline proposition is outlined above. Although there are concrete days but down and the schedule seems very definite, the reality is this is more of a guideline. The team plans to take an agile approach to development. All items outlined above are purposely done in multiples of seven and in parallel, as we plan to have the team divided when possible, in order to work on different components in an efficient manner when dependencies allow for it. This timeline was being proposed because it establishes a separation of concerns and reinforces the idea that different people will be able to work on different items at the same time. Not only that, but we believed that having everyone work on one component at a time would be counterproductive and lead to many merge conflicts in the code.

As time went on it became apparent that the proposed timeline was not working due to the lack of experience of the group. As such the group shifted to a deliverable-centric timeline, in which every scrum meeting something was expected to be accomplished and by certain dates (usually planned as weeks gone by during scrum) certain aspects of the project would be completed to a bare minimum state. Agile, at least in the initial stages, was planned to revolve around a weekly meeting at 11 AM on Friday (to be adjusted next semester for schedule accommodations). The agile meeting is intended to promote honesty in the group about progress so that issues can be identified before significant delays are made. Over time this was found to not work for us well and the group shifted to a 9:30 AM meeting on Wednesdays with the client.

4.2 Feasibility Assessment

The task the product will perform is straightforward to understand. The application will transmit audio to other devices in the conversations, and process audio from the other devices before outputting through the user's audio device. No new technologies need to be developed for this project, as hearing aids, audio processing, and wireless communication are well-established technologies. However, in spite of this fact, a lack of resources for how to use some of these technologies did add a level of constraint to development, so while the project was still feasible, what was seen as possible to complete over certain periods of time became more limited.

4.3 Personnel Effort Requirements

The initial plan for personal effort requirements is provided below:

Android Activity Development	14 days is assigned as a generous estimate and will involve all members due to it being a mix of everyone getting involved and used to android development in general. This task is one of two tasks in which the foundation for the rest of the project will be built upon. The reality is the UI is probably a bit over one week of work but there is a level of research and development on the group's part still needed that can only be done while doing work. Richard, as the member with the most android experience, is expected to do a lot of the initial work while the rest of the team catches up in understanding of the framework.
Sound Processing Development	This task is the other initial task in which the foundations of the project will be built upon, but is more black box in nature than the overall Android development. This is due to us having a definite input and output for this module in the form of actual data we can work with. There is a level of Android code needed to start, the 14 days assigned to this task is taking place one week into the android activity development. This is expected to be enough time for a team of 2 or 3 to be able to figure out how to initially set up sound working in the Android OS and to come up with sound modifying logic that can produce an output applicable to the features of the application. There will be a one week period in which the team working on the initial android activity task is expected to chip into smaller tasks after the core logic for the sound is figured out.

Advanced UI Features	This task is expected to be done in parallel with the communication handling. There are some level of UI features that cannot be touched upon in the android activity development task due to a reliance on needing data from the result of the sound processing development task. Some of these include applying adjustments to the frequency or volume of the sound. This task adds options for which the sound controller made in the previous task can act upon, and thus is an extension of the first two tasks, justifying doing it after the initial development of the application. A total of twenty-one days, to be split over an estimated four overall “mini-tasks” is expected for each tab in the UI. This gives an average of just over five days per tab, allowing what we estimate to be just enough time for this task as there is a level of logic needed to be implemented by the team for each user option made.
Communication Handling	This task is expected to be done in parallel with the advanced ui features. The crux of this task is to lay out the foundation for the application to allow connection between other android devices, as well as room for potentially connecting to the database. Following establishing connection, a foundation of logic (but definitely not a complete implementation) for passing data over these connections is expected to be complete. The means of communication between phones is planned to be Wi-Fi P2P via tools provided by the android framework while any database/server connectivity should be done via http. A total of fourteen days is given in order to learn and apply the knowledge needed between around two to three members. Upon finishing this task, members who have worked on this task should switch over to assist in finishing advanced ui features that pertain to connectivity, which is why the advanced ui features are given an extra week over this task.
Server/Backend Setup	The operation of any needs provided by a server will be primarily done using Google’s Firebase. Firebase is to be used and imported in a more mature stage of the project where the application has shown capability of producing data from a sound input provided by the surroundings. We implemented our app with Firebase’s Real-time Database which is hosted on a cloud and is a NoSQL database. It allows us to store and sync JSON data, real-time.

Backend and Frontend Connectivity	Firebase Realtime Database provides us with an API that would allow us a level of backend service and scalability far beyond anything this application needs, and gives us functionality like analytics, databases, messaging and crash reporting so we can move quickly and focus on our users.
Phone to phone connectivity	As a core part of the application, inter-device connectivity will require some time to implement and test. As such, two weeks provides sufficient room to develop an appropriate system, and should be done in parallel with the finalization of audio processing to maximize performance.
Sound Processing's Finishing Touches	After other functionality has been implemented, changes to audio processing will likely be necessary, and optimization will be critical to proper performance of the product. Two weeks provides time for testing and refinement of the sound processing system. At the end of this period, an implementation of a fast fourier transform is expected to be completed and capable of applying a user-supplied bias on data from their application settings.
UI Finishing Touches	The last full task, finishing the UI requires knowing the full functionality of the application, which is still subject to change. This work is likely to require a large amount of experimentation to find a design comfortable for the user, so two weeks have been allocated.
Final Bug Fixing and Integration Changes	Given that many factors are currently unknown, finalization and debugging could take significant amounts of time or none at all. Given the estimates of other tasks and the project timeframe, three weeks provides adequate room for emergencies, unforeseen issues, final testing, and potential addition of features.

(Table 2)

All tasks were set to a multiple of seven days in order to align with the weekly meeting schedule.

Over time the group adopted a more kanban-like approach in which things that had to be completed were decided as things were being completed. This worked much better for the group's lack of experience as completion of tasks was simply too difficult for individuals needing training in most of the subject matter being worked with in the project.

4.4 Other Resource Requirements

All members require:

- ➢ Computers capable of running the Android Studio software
- ➢ A physical Android device (at least 2 for WiFi Direct) to compile software onto
- ➢ An internet connection reliable enough to push and pull code onto
- ➢ A set of earbuds or headphones with a microphone that may be plugged into the Android device

4.5 Financial Requirements

There are no financial requirements to the project as any unowned devices needed are provided by the school for students.

5. Testing and Implementation

5.1 Interface Specifications

There is a heavy reliance on logging and manual testing due to the nature of the application and the frameworks it relies on. Automated testing is not feasible for the stage of the product being worked on due to frequent switching of technologies, let alone the nature of these technologies themselves are difficult to test via automation given the group's resources.

5.2 Hardware and software

Hardware will be our personal laptop computers (both Mac and Windows) in which we will write the code on and the software will be the Android Studio IDE with integrated JUnit tests.

The laptop computers are absolutely necessary to develop and fully implement this Android Application, all of the computers are above or up to par with Android Studio's dependencies and or hardware requirements for running the integrated debugger.

5.3 Functional Testing

There was an initial plan to use JUnit and System testing, which would be the Android Studio's integrated debugger that comes integrated with the Android Studio IDE for both Mac and Windows. JUnit testing was abandoned but reliance on the debugger was still paramount to testing the pro

5.4 Non-Functional Testing

We shall use Androids “Profiler” which is an extension of the Android Studio IDE that monitors/inspects GPU rendering speed and overdraw, along with identifying CPU hot spots.

5.5 Process

Android Studio lets us test any Android application function that we want to test via the debugger provided by the software. On top of this, specialized logging functions and abilities to create messages for developers allows for some conveniences when debugging.

5.6 Results

Some of us have no experience with Android Studio while some of us have done multiple basic projects in Android Studio. Given this fact, implementation did prove to be somewhat of a challenge, as well as debugging. By the end of the project the members have a much greater understanding of how to work with Android Studio and Android itself as a platform and what to pay attention to when debugging the code.

6. Closing Material

6.1 Conclusion

The application in its current state is designed as a launching platform for expanding and adding to the responsibilities of both being a hearing aid and conversation platform. WiFi has been implemented in a way to accommodate for both changes in project specification as well as expansion of the “Session”’s responsibilities. The sound management implementation can bring the sound into and out of the frequency domain and the application can take in sound with varying quality (as high a sample rate as the device can support). Google’s firebase is our flexible, scalable database. It keeps our user data in sync across the app through real time listeners regardless of network latency or Internet connectivity.

6.2 References

Accessible Audio: Android Hearing Aid Support and the Audio Framework (Google I/O'19)

Accessible Audio: Android Support and the Audio Framework

Fast Fourier Transform

Wolfram Mathworld: Fast Fourier Transform

Understanding the FFT Algorithm

Pythonic Perambulations: Understanding the FFT Algorithm

Get started with Cloud Firestore

Firebase: Firestore dependencies and set up

Authenticate Using Google Sign-In on Android

Firebase: User sign-in authentication

Our advisor:

M. Wymore, 2019

6.3 Appendices