

Thesis Dissertation

**GAMING HEADPHONES FOR PEOPLE WITH HEARING  
IMPAIRMENTS**

**Ioannis Strouthos**

**UNIVERSITY OF CYPRUS**



**COMPUTER SCIENCE DEPARTMENT**

**May 2024**

**UNIVERSITY OF CYPRUS**  
**COMPUTER SCIENCE DEPARTMENT**

**GAMING HEADPHONES FOR PEOPLE WITH HEARING IMPAIRMENTS**

**Ioannis Strouthos**

Research Supervisor  
Dr. Costas Pattichis

This Thesis is Submitted in Partial Fulfillment of the Requirements for the Degree of  
Bachelor of Computer Science at the University of Cyprus

May 2024

# Acknowledgments

Firstly, I would like to extend my deepest gratitude to my supervisor, Costas Pattichis, for his invaluable oversight and support throughout my dissertation journey. His expertise and insights ensured I would adhere to my project's original framework that I had planned. Additionally, I am profoundly thankful for my family's unwavering support and love, which has been my foundation and inspiration. Their belief in me has made all the difference.

# Abstract

This dissertation introduces an innovative software framework designed to revolutionize the gaming audio experience for hearing-impaired users and shorten the competitive advantage people with typical hearing capabilities have over them through sophisticated audio manipulation. Through the use of advanced equalization (EQ) techniques and cutting-edge frequency-shifting strategies, this software can customize audio output to align with individual hearing profiles. This effectively bridges the gap between the auditory capabilities of users and the demands of high-fidelity sound environments. Central to this endeavor is the utilization of audiograms to tailor sound experiences, ensuring that adjustments in sound frequency and intensity are precise and user-specific. This approach not only significantly enhances accessibility for the hearing impaired by redefining how sound is experienced in interactive applications but also sets a new benchmark for personalized audio technology. Moreover, the project explores the implications of such tailored audio experiences on user engagement and immersion in digital environments, promising a future where auditory accessibility is seamlessly integrated into the broader landscape of interactive media and entertainment.

## Contents

Acknowledgments .....	3
Abstract .....	4
Chapter 1 .....	7
Introduction.....	7
1.1 Motivation.....	7
1.2 Objective and Contribution.....	8
1.3 Structure of the Thesis .....	8
Chapter 2.....	10
Literature Review .....	10
2.1 Existing Technologies.....	10
2.2 Advances and Limitations .....	10
2.3 Gaps this research aims to address .....	11
Chapter 3.....	12
Theoretical Background.....	12
3.1 Auditory science, the nature of hearing loss .....	12
3.2 Audiogram Utilization in Audio Customization.....	12
3.3 Frequency Lowering Techniques and Their Application .....	13
3.4 Advanced EQ Techniques for Hearing Impairment .....	14
Chapter 4.....	15
Software Design, Architecture, and Implementation.....	15
4.1 System Architecture Overview .....	15
4.2 Audio Processing Algorithms .....	16
4.3 Audio quality parameters and their significance .....	17
4.4 Integration of Audiograms into Audio Processing .....	19
4.5 Using ports and virtual cables to start audio stream .....	20
4.6 How the live audio stream is used to apply EQ adjustments.....	21
4.7 User Interface Design with PyQt5 .....	21
4.8 UI Components and Layout.....	22
4.9 UX Best Practices and Accessibility Standards.....	23
4.10 Libraries Used in Development .....	24
Chapter 5.....	27
Testing and Evaluation .....	27
5.1 Methodology and Results .....	27
5.2 Consequences of Misconfiguration .....	38
Chapter 6.....	40
Accessibility and User Diversity .....	40
6.1 User Profiles and Scenarios of Hearing Impairments.....	40
6.2 How the software can help said User Profiles .....	42
Chapter 7.....	44
Legal and Ethical Considerations .....	44
7.1 GDPR Compliance .....	44
7.2 Global Compliance .....	45
Chapter 8.....	46
Discussion.....	46
8.1 Interpretation of Results.....	46
8.2 Analysis of User Interactions with software through forms .....	47
8.3 Assumptions to factors that may affect system evaluation from users.....	51
Chapter 9.....	52

Conclusion and Future Work.....	52
9.1 Summary of Findings.....	52
9.2 Recommendations for Future Research.....	52
Bibliography .....	54
Appendices.....	58

# Chapter 1

## Introduction

---

1.1 Motivation	7
1.2 Objective and Contribution	8
1.3 Structure of the Thesis	8

---

### 1.1 Motivation

The motivation for this dissertation stems from a deeply personal and underserved need within the audio technology landscape. People with hearing impairments, a minority group, often find themselves overlooked by companies aiming for mass-market appeal, leaving their specific auditory requirements unaddressed. My own experiences as someone with hearing impairment have highlighted this gap. Even when I purchase the most immersive and rich soundscapes offered by the latest headphone technology and put them on in hopes of an enriched hearing experience, the outcome is always the same. I remain confined to what any standard headset offers without the nuances of advanced sound engineering. And while I watch my friends wear those same headsets, close their eyes, and watch their excitement from clear sound quality and immersion from noises coming from all around them, I can't help but feel frustrated and wish that I and everyone else in my place could have this experience. This discrepancy not only underscores a significant disparity in technological accessibility but also emphasizes a broader societal oversight toward inclusivity in the audio experience. The drive behind this research is not just professional but deeply personal, fueled by years of being unable to fully experience music and sounds in the way those with ordinary hearing can. It's a call for change, advocating for a world where audio technology does not just cater to the majority but is inclusive, enhancing the auditory experience for everyone,

regardless of their hearing capabilities. This thesis is an endeavor to bridge this divide, offering solutions that personalize the audio experience for hearing-impaired individuals, bringing them closer to the sonic richness available to their hearing counterparts while also expanding the auditory possibilities for all users.

## **1.2 Objective and Contribution**

The objective of this dissertation is to enhance the audio experience for individuals with hearing impairments through the development of advanced audio processing software. This software primarily aims to accommodate the specific auditory requirements of the hearing impaired by employing customizable audio enhancements such as equalization and frequency shifting while at the same time improving audio quality for all users. The innovative contribution of this research lies in its dual approach: first, by integrating audiogram data directly into the audio output through tailored equalization profiles, it personalizes sound delivery according to individual hearing capabilities. Second, it utilizes frequency-shifting techniques to make certain sound frequencies more accessible to those with specific hearing loss profiles. This is particularly notable in addressing the often-overlooked needs of the hearing impaired within the gaming community, proposing a solution that adjusts game audio outputs to be inclusive without diminishing the quality of the gaming experience. This work not only fills a significant gap in adaptive audio technologies but also sets a precedent for future advancements in audio accessibility, particularly within multimedia environments.

## **1.3 Structure of the Thesis**

The structure of the thesis is meticulously organized to guide the reader through a comprehensive exploration of audio processing enhancements tailored for individuals with hearing impairments. The current chapter (Chapter 1) introduces the research, framing the motivation, objectives, and contributions of the study. Chapter 2 delves into a detailed Literature Review, identifying existing technologies, recent advancements, and the limitations they present in the field of audio processing for the hearing impaired.

Chapter 3 presents the Theoretical Background necessary to understand auditory science, audiogram utilization, and the application of advanced audio processing techniques. This sets the stage for Chapter 4, which describes the Software Design,



Architecture, and Implementation, detailing the system architecture, the specific audio processing algorithms used, and the integration of user interfaces designed with accessibility in mind.

Chapter 5 discusses testing and evaluation. Chapter 6 focuses on Accessibility and User Diversity, discussing various user profiles and scenarios to highlight how the software meets diverse needs. Chapter 7 addresses legal and Ethical Considerations, covering compliance with GDPR.

The Discussion in Chapter 8 interprets the results and discusses their implications for user engagement. Finally, Chapter 9 concludes the thesis with a Summary of Findings and offers Recommendations for Future Research, suggesting directions for further enhancing the accessibility and functionality of audio processing technologies.

# Chapter 2

## Literature Review

---

2.1 Existing Technologies	10
2.2 Current advances and their limitations	10
2.3 Gaps this research aims to address	11

---

### 2.1 Existing Technologies

In the domain of audio processing for the hearing impaired, two primary technologies have emerged as front-runners: hearing aids [1] and headphones equipped with EQ software. Hearing aids have traditionally focused on amplifying environmental sounds using various forms of signal-processing technologies to enhance speech clarity and reduce background noise. Today's hearing aids are more advanced, using digital processing to adjust the sound frequency and amplitude based on an individual's specific hearing loss profile. On the other hand, headphones with EQ software represent a more consumer-oriented approach, offering users the ability to simply change the gain of sound frequencies through equalization settings.

### 2.2 Advances and Limitations

Recent advancements in audio technology for the hearing impaired have significantly leveraged frequency shifting [2]. Unlike traditional methods that primarily focus on amplifying certain frequencies, frequency shifting alters the pitch of sounds, making them more accessible within a user's hearing range. This approach can be particularly beneficial for those with significant high-frequency hearing loss, where simply boosting the volume of higher frequencies may not suffice.

## **2.3 Gaps this research aims to address**

Despite the advancements in technology aiding those with hearing impairments, the gaming sector still largely neglects accessibility for this demographic. Most gaming audio solutions focus on enhancing the experience for users with typical hearing. This research aims to bridge this gap by developing software that not only adjusts audio output through customized EQ profiles based on individual audiograms but also incorporates frequency shifting to better cater to the specific hearing needs of users. By doing so, the project seeks to make gaming and other multimedia experiences more inclusive, ensuring that individuals with hearing impairments can enjoy audio content fully and without compromise.

# Chapter 3

## Theoretical Background

---

3.1 Auditory science, the nature of hearing loss	12
3.2 Audiogram Utilization in Audio Customization	12
3.3 Frequency Lowering Techniques and Their Application	13
3.4 Advanced EQ Techniques for Hearing Impairment	14

---

### 3.1 Auditory science, the nature of hearing loss

Auditory science explores the process by which sound is perceived by humans. Sound waves travel through the ear canal to vibrate the tympanic membrane. These vibrations are amplified by the ossicles in the middle ear and transmitted to the cochlea in the inner ear. Hair cells within the cochlea convert these vibrations into electrical signals that the brain interprets as sound. This complex auditory process enables individuals to distinguish between different frequencies and intensities of sounds, forming the basis of hearing [3,4].

Hearing impairment can disrupt this intricate process and is categorized mainly into conductive and sensorineural hearing loss. Conductive hearing loss occurs when there is a blockage or malformation in the outer or middle ear, while sensorineural loss results from damage to the cochlea's hair cells or the auditory nerve. This type of hearing loss not only reduces the overall loudness of sounds but also affects the clarity of speech, particularly in noisy settings [5,6].

### 3.2 Audiogram Utilization in Audio Customization

Audiograms play a critical role in customizing audio experiences for individuals with hearing impairments by mapping out the specific frequencies and intensities that a person can or cannot hear. This detailed hearing profile allows audiologists and audio engineers to tailor auditory devices, ensuring that sound amplification and clarity are adjusted according to individual needs. Such customization is vital because it not only compensates for the frequencies that are lost but also enhances the overall sound quality, making speech and other sounds more discernible and less strenuous to comprehend.

The importance of personalized audio settings goes beyond mere amplification, impacting overall accessibility and inclusion for the hearing impaired. Studies have shown that tailored audio solutions, such as cochlear implants and advanced hearing aids that are adjusted based on detailed audiograms, significantly improve the quality of life. These personalized settings help users better engage with their environments, facilitating clearer communication and greater participation in social activities. This not only enhances the auditory experience but also supports psychological well-being and independence [5].

### **3.3 Frequency Lowering Techniques and Their Application**

Frequency-lowering techniques, particularly frequency shifting, are crucial in the management of hearing loss, especially for individuals unable to hear high-frequency sounds. Frequency shifting involves altering higher frequencies into lower ones that are within an individual's audible range. This method shifts the audio down, making high-frequency sounds, such as certain speech sounds, more accessible and distinguishable to the listener [6].

The effectiveness of frequency shifting depends on several factors: the user's degree of hearing loss, their cognitive abilities, and their acclimatization to the new auditory inputs. The degree of hearing loss determines how much frequency shifting is necessary; more severe high-frequency losses may require greater shifts. Cognitive abilities are also critical, as individuals must cognitively adapt to interpreting sounds that are now presented at different frequencies than they are used to. Additionally, the

acclimatization period can vary significantly among users, with some adapting quickly while others may take longer to adjust to the altered soundscape. Tailoring the amount and type of frequency shifting to each user's specific profile ensures optimal understanding and comfort [7,8,9].

### **3.4 Advanced EQ Techniques for Hearing Impairment**

Advanced equalization (EQ) techniques are integral to enhancing audio output for individuals with hearing impairments. These techniques are designed to precisely adjust audio frequencies to compensate for specific hearing loss patterns, thereby improving speech intelligibility and overall sound quality. By using EQ settings, the software can boost or attenuate certain frequencies to match an individual's unique audiogram, thus creating a personalized listening experience that aligns with their auditory capabilities [10].

The customization capabilities of the software allow for the implementation of EQ settings that adapt to real-time audio environments simulated in games. This means that whether in a quiet room or a noisy outdoor environment, the software's adjusted settings optimize audio clarity and detail for the user as they are for their profile. This adaptability is crucial for users with hearing loss, as it provides them with the flexibility to interact seamlessly in various auditory environments without losing sound quality or intelligibility. The technology behind these advanced EQ techniques involves not only simple gain adjustments but also complex algorithms that manage how sounds are processed and delivered, ensuring that users experience enhanced audio that is both clear and contextually appropriate.

# Chapter 4

## Software Design, Architecture, and Implementation

---

4.1 System Architecture Overview	15
4.2 Audio Processing Algorithms	16
4.3 Audio quality parameters and their significance	17
4.4 Integration of Audiograms into Audio Processing	19
4.5 Using ports and virtual cables to start audio stream	20
4.6 How the live audio stream is used to apply EQ adjustments	21
4.7 User Interface Design with PyQt5	21
4.8 UI Components and Layout	22
4.9 UX Best Practices and Accessibility Standards	23
4.10 Libraries Used in Development	24

---

### 4.1 System Architecture Overview

The software architecture for the audio enhancement project, is built on a modular and integrative framework, facilitating seamless interaction between various components that handle different stages of audio processing. The project is structured within a designated workspace, containing directories for python and C++ source code, compiled programs, and resources like images, ensuring organized access and systematic development.

In this thesis, the ideal software User Interface (UI) will be discussed at various points to provide a structured view of how the system should work. At the core of the system, the software initialization and user interface should be handled by a main python program, which will set up the primary window and its interactive elements using

PyQt5. This script is crucial for delivering the initial user interaction point with the tutorial window. Adjacent to this, the Python file with the main UI structure will be responsible for rendering the user-friendly page that follows common UI/UX practices. The page shall host sub-pages like the audiogram input interface, a critical component for capturing user-specific hearing profiles, and the EQ profile page. The user's input in the audiogram page directly influences the customization of audio output, feeding into the subsequent processing stages.

The Python script with the sound conversion logic will act as a bridge between the user interface and the backend audio processing algorithms implemented in the C++ program. The C++ program leverages the PortAudio library for real-time audio streaming and FFTW for fast Fourier transformations to adjust the audio signal in accordance with the user-defined EQ settings derived from the audiograms. This setup supports dynamic audio customization based on real-time input.

Together, these components form a cohesive architecture that supports the project's goal of enhancing auditory experiences for individuals with hearing impairments, emphasizing flexibility, user customization, and real-time processing capabilities.

## **4.2 Audio Processing Algorithms**

The audio processing algorithms central to this software focus on enhancing clarity and intelligibility for hearing-impaired users, leveraging techniques like equalization (EQ) and frequency lowering. These techniques are intricately designed to address the unique auditory challenges faced by individuals with hearing loss, allowing for personalized audio enhancement based on user-specific audiograms.

Equalization (EQ) is instrumental in modifying the audio signal to better suit each user's hearing profile. It adjusts the amplitude of specific frequency bands to compensate for hearing deficiencies at those frequencies. By amplifying sounds that are difficult to hear and reducing the dominance of others, EQ settings ensure that the audio output is optimized for the listener's individual needs. The EQ adjustments are directly informed



by detailed audiograms provided by the users, making these adjustments finely tuned and highly personalized.

Frequency lowering targets users with significant high-frequency hearing loss, a common issue among the hearing impaired. This technique shifts high-frequency sounds to a lower frequency range where the user's hearing is typically better preserved. The key goal is to transform high-pitched sounds into lower ones that are more easily detected and interpreted by the user. By doing this, the technology helps to reduce the challenges posed by high-frequency sound cues. This approach continually adjusts the incoming audio to ensure optimal clarity and understanding.

### **4.3 Audio quality parameters and their significance**

Audio quality parameters are essential metrics that define the characteristics and performance of digital audio systems. These parameters—frames per buffer, sample rate, bit depth, channels, bitrate, and suggested latency—directly influence the clarity, fidelity, and responsiveness of audio output. They are crucial in various applications, especially in environments like gaming, where precise and immersive audio can significantly enhance the user experience. Each parameter plays a specific role in shaping how sound is processed and perceived, making their understanding and optimization key to achieving high-quality audio.

In digital audio, a sample represents a single data point in a sound wave captured over time. These samples are the building blocks of digital sound, defining the amplitude (or loudness) of the audio signal at a specific moment. Each sample contains bits of data that indicate the sound wave's characteristics at that very instant. The quality and accuracy of these samples are determined by parameters such as bit depth and sample rate, which dictate the dynamic range and frequency resolution of the audio output.

Channels in audio refer to the individual paths in which audio signals are processed and transmitted. Each channel carries a separate audio stream; common configurations include mono (1 channel), stereo (2 channels), and various surround sound setups (e.g., 5.1, which has 6 channels). In gaming and multimedia environments, channels are

crucial for delivering spatial audio cues that enhance the realism of the scene and provide directional information to the user, improving both immersion and interactive responsiveness.

Frames per buffer define the quantity of audio samples per channel that the system processes in one batch. This parameter directly affects audio processing efficiency and latency—the time delay experienced between input and output. Lower buffer sizes can reduce latency, offering more immediate feedback, which is critical in real-time applications like gaming or live performances. However, smaller buffers require more CPU resources as they increase the frequency of data processing cycles, potentially impacting system performance.

Sample rate is a critical audio quality parameter that measures how often samples of the audio signal are taken per second, expressed in hertz (Hz). Common sample rates include 44.1 kHz, used in CDs, and 48 kHz, typical in professional audio settings. The sample rate determines the frequency range that can be accurately reproduced; according to the Nyquist theorem, this is up to half the sample rate. Thus, a higher sample rate allows for the reproduction of higher sound frequencies, enhancing the audio detail and clarity crucial for high-fidelity applications. However, for 44.1kHz, the maximum reproducible frequency is 22 kHz, which exceeds the upper limit of human hearing, typically around 20 kHz. This means the most optimal choice for both responsive audio processing and pure sound quality is 44.1 kHz.

Bit depth refers to the number of bits used to encode each audio sample, directly impacting the audio signal's dynamic range—the difference between the softest and loudest sounds that can be accurately captured. Higher bit depths, such as 24 or 32 bits, provide a greater dynamic range, leading to finer distinctions in sound intensity. This enhances the audio detail and reduces distortion and noise, which is particularly beneficial in environments requiring precise sound reproduction, such as in music production or when delivering a highly immersive gaming experience.

Bitrate is a measure of the amount of data processed per unit of time in an audio stream, typically expressed in bits per second (bps). It is calculated by multiplying the sample

rate, bit depth, and the number of audio channels. For instance, a configuration using a 44.1 kHz sample rate, a 24-bit depth, and stereo channels would have a bitrate of approximately 2,116,800 bps. High bitrate is essential for maintaining sound quality, especially in lossless audio formats where higher bitrates allow for more detailed and complex soundscapes without compression artifacts. This is particularly significant in applications where audio quality can enhance user experience, such as in high-end audio playback or gaming environments.

Suggested latency refers to the recommended delay time for audio processing to balance between immediate responsiveness and buffer underruns. Low latency is crucial in interactive environments like gaming or live audio workstations where immediate auditory feedback is necessary for performance synchronicity and gameplay dynamics. It affects the buffer size settings in audio devices; lower latency settings require smaller buffer sizes to minimize the delay between sound production and its playback. However, excessively low latency might overload the CPU and cause glitches in audio output, while higher latency settings, although more stable, could disrupt user interaction by introducing noticeable delays.

#### **4.4 Integration of Audiograms into Audio Processing**

In the software, audiograms are integrated to tailor audio output specifically for users with varying hearing profiles. This integration begins by capturing the audiogram data, which details the hearing thresholds across different frequencies for an individual. These thresholds illustrate the minimum sound levels at which a person can detect sounds across a spectrum of frequencies, providing a distinct auditory profile that can significantly vary from person to person.

The audiogram data is crucial because it directly informs the equalization (EQ) settings applied during audio processing. By using these audiograms, the software adjusts frequencies specifically to compensate for hearing deficiencies identified in the audiogram results. For instance, if the audiogram indicates a significant loss at higher frequencies, the software can boost these frequencies, ensuring sounds in this range are more perceivable to the user. It can also further attenuate frequencies that the user hears

too well to make the frequency ranges the user is deficient in pop out. Such integration of audiograms into audio processing bridges the gap between medical data and consumer electronics, providing a seamless and enhanced listening experience that, as mentioned before, adjusts to the needs of each individual.

#### **4.5 Using ports and virtual cables to start audio stream**

In the developed software system, the audio streaming process involves the use of ports and virtual cables, which are crucial for routing audio signals within the computer environment. This setup allows for the real-time processing of audio data before it reaches the user's output device, such as speakers or headphones. The system utilizes virtual audio cables, which act as virtual sound devices. These devices can receive audio input from one application and send it to another as if they were physically connected by an audio cable. This method is particularly beneficial for applications where audio data needs to be processed or monitored by different software simultaneously without the need for external hardware.

To facilitate audio streaming, specific software ports are designated for capturing and redirecting audio streams through these virtual cables. The integration involves configuring the software to recognize and utilize a virtual audio driver that simulates the presence of physical audio interfaces. This configuration is managed within the system settings, where the virtual cables are set as the default audio input and output devices. By doing this, all audio signals generated by the computer (such as system sounds, media player outputs, and game audio) are routed through these virtual cables. More specifically, the “Virtual Audio Cable Input” is set as the default system output. Then, in the software, the “Virtual Audio Cable Output” is used to get the sounds that are produced by system applications like games. Then these two devices along with the output device where the audio will be directed, the audio quality parameters, and the EQ customizations are parsed as input to another program.

The mentioned above input is parsed by a local port in the computer from the Python program to the other executable C++ program. This effectively makes the Python program act as a server, sending parameters and live user input to the C++ program,

which acts as a client and listens to the server. The C++ software intercepts these signals, allowing for real-time audio processing based on the EQ settings determined by the user's audiogram data. After processing, the audio is sent back through the virtual cables to the output device (for example, gaming headphones). This process is nearly instantaneous, ensuring there is minimal latency between the original audio output and the enhanced audio reaching the user's ears.

#### **4.6 How the live audio stream is used to apply EQ adjustments**

The live audio stream within the software system is utilized to apply equalization (EQ) adjustments in real-time. This dynamic process begins with the capture of audio data through the configured virtual cables and ports, as previously outlined. The program then leverages digital signal processing algorithms that modify the audio signal based on the adjustments determined to be suitable for the user from the audiogram. The adjustments involve altering various frequency bands' gain levels, effectively reshaping the audio spectrum to enhance clarity and intelligibility. This is done by converting the audio samples from the time domain to the frequency domain using FFT, and applying the changes to the frequency bands/bin there before converting it back to the time domain with IFFT. The processed audio is then routed back to the output device through the virtual cables.

#### **4.7 User Interface Design with PyQt5**

The user interface of the software should be developed using PyQt5 [11], a powerful set of Python bindings for Qt5 that enables rapid prototyping and the creation of feature-rich applications. PyQt5's extensive widget toolkit allows for the efficient design of complex user interfaces, which in this project range from a tutorial page to the main application page and specialized sub-pages like the audiogram. The interface structure should be organized using horizontal and vertical box layouts to systematically arrange interface elements such as labels, buttons, and frames, enhancing the user's interaction flow. This arrangement facilitates easy navigation and manipulation of audio settings, with interactive elements like labels and buttons that allow users to adjust audio parameters dynamically. The cross-platform nature of PyQt5 ensures that the application maintains consistent functionality and appearance across different operating

systems, broadening its accessibility and usability without requiring significant modifications to the underlying codebase.

#### **4.8 UI Components and Layout**

The system must be designed using the best UI practices so that the user can have a seamless experience navigating and interacting with the application. To achieve this, an extensive search for the best HCI practices was conducted.

It is imperative that the design model and the user model [10] be indistinguishable. The project aims to achieve this by utilizing standardized mental models [11]. Mental models are the user's framework for a system or interface based on past experiences, perceptions, and understanding of similar systems. What makes a good mental model are interface metaphors, visual affordance, visual mapping, constraints, the causality of interaction, and instructions that augment visuals [12]. Breaking down each of the elements a good mental model consists of, we have interface metaphors, which means incorporating items from everyday life into the system's User Interface to facilitate familiarity. An example from the ideal system's interface metaphor is the main widget's sidebar, which can show the audiogram sub-page as a soundwave or equalizer sub-page, with volume knobs that can slide up and down. Visual affordance is when a button looks pressable, like the "Add frequency" button on the audiogram page, which invites you to click it. Visual mapping is the intuitive mapping of actions, like sliding up or down the equalizer knobs. Constraints prevent the user from executing unwanted actions; for example, a user can have a maximum number of 15 frequencies in the audiogram graph. Causality of interaction is the feedback the user gets from the system, ensuring them that they have executed an action. For example, clicking the button to save the audiogram must show a message indicating that the audiogram has been successfully saved. Instructions that augment visuals are icons with labels that clarify their actions, and it should be seen on the main widget sidebar where the EQ sub-page has the text "Equalizer" spelled next to the icon to guarantee the user understands their function.

Furthermore, Nielsen's usability heuristics [13] were used as general principles to create a more accessible, user-friendly, and intuitive User Interface. These heuristics match the system and the natural world by using words familiar to the user and having explanations wherever necessary, so the user can easily understand the system rather than writing internal project-related jargon. The user must have control and freedom by integrating functionalities to undo the progress, like canceling an audiogram creation. Another Nielsen principle describes the significance of the user's ability to recognize parts of the system instead of recalling them as a priority. Humans can recognize items much faster than they can recall [14], so whenever necessary, icons or pictures should be used alongside text to ensure that capability is met. Lastly, another heuristic imperative to the project is the creation of an aesthetic and minimalistic design for the UI.

#### **4.9 UX Best Practices and Accessibility Standards**

The design of the software must meticulously adhere to UX best practices and accessibility standards to ensure it is both functional and user-friendly, particularly for individuals with varying degrees of hearing impairment. Key design elements include clear and accessible controls for audio adjustments, such as sliders for volume and frequency, which are designed with ample size to facilitate easy interaction, especially for those who might struggle with fine motor skills. Visual feedback must be emphasized, with indicators that show changes in audio settings or levels, providing users who cannot hear adjustments with visual confirmation of changes. Carefully structured navigation menus and intuitively placed interface elements reduce cognitive load and enhance usability, making the software accessible to users with diverse abilities (from tech-savvy to novice technology users). Expert controls should notably be placed in the software, for example, a skip tutorial button.

By adhering to these principles, the software not only aligns with general accessibility guidelines but also addresses the specific needs of its target user group, enhancing the auditory experience and promoting ease of use in everyday scenarios. These considerations must be visually supported within the UI, such as in the tutorial and the specific UI components displayed, which demonstrate the practical application of these

UX strategies, significantly aiding users in effectively navigating and utilizing the software.

#### **4.10 Libraries Used in Development**

The development of the software leverages various libraries and frameworks that significantly enhance its functionality and user experience. In the Python program, the main libraries used are subprocess, threading, sounddevice, socket and json. The socket selects a socket in the local system to connect with the C++ Program. The sounddevice library initiates the stream for the audio and the C++ executable. The threading and subprocess are used to have a thread handle the C++ program, and another listen to its outputs. Lastly, the json library is used to parse the audio quality parameters and other useful information to the C++ program.

For the ideal User Interface Python, the two main libraries to be used are the “os” library and the “PyQt5”. Without going into much detail on PyQt since it was already explained in the dissertation, it should be noted that it can work alongside the “os” libraries to succeed in parts on the goal of creating the UI. PyQt5 is a useful tool for creating user interfaces quickly. More specifically, for the purpose of studying its libraries, its modules, such as QtWidgets, QtGui, and QtCore, are used to manage GUI elements of an application, from basic windows to complex layouts and controls. It works in harmony with the “os” library to set up paths that belong in the project from which components like images can be accessed to be integrated into the UI design.

In the project, PortAudio plays a crucial role by providing a robust, cross-platform API for managing real-time audio input and output, which is essential for processing and manipulating audio streams effectively (I/O). This library is employed to handle the audio data flow between the software and the physical audio hardware, ensuring low-latency transmission and high fidelity of the audio signals, which are critical for real-time audio applications such as this project that focuses on enhancing audio accessibility and experience for individuals with hearing impairments.



PortAudio is particularly advantageous for such projects due to its ability to operate seamlessly across different operating systems, which means the software can be adapted for various platforms without altering the core audio handling logic. It supports a wide range of audio formats and systems, making it versatile for handling the diverse audio processing needs of different users. In this project, PortAudio interfaces with FFTW (Fastest Fourier Transform in the West) for spectral analysis, necessary for the dynamic equalization features that adjust audio frequencies based on the user's audiogram results. This integration allows the software to modify the sound output by increasing or decreasing the magnitude of specific frequency bands, making sounds clearer and more distinguishable for users with specific hearing loss profiles.

Furthermore, good practice in using PortAudio involves careful management of audio streams and buffer configurations to minimize latency and ensure synchronization of audio output with real-time adjustments. This is evidenced in the software's setup where `framesPerBuffer` and sample rate are configured to balance CPU load and real-time processing requirements efficiently. Error handling is meticulously implemented to ensure that any issues during audio initialization or processing are caught and managed effectively, preventing the application from crashing and providing debugging information for maintenance and updates.

As for the `fftw3` library used (mentioned before as "Fastest Fourier Transform in the West"), it is essential for the application of the DSP (Digital Signal Processing) tasks within the project. FFTW3 excels in computing discrete Fourier transforms (DFT), which is a cornerstone for analyzing the frequency components of digital signals. In this audio processing application, FFTW3 transforms time-domain audio data into the frequency domain, allowing the software to apply frequency-specific adjustments effectively. This capability is pivotal for implementing the equalization strategies based on user-specific audiograms, where certain frequency ranges are amplified or attenuated to match the user's hearing profile. Moreover, FFTW3's efficiency and speed in executing complex transformations ensure that these DSP tasks do not introduce significant latency, which is critical for maintaining the real-time processing requirements of the audio stream demanded in gaming headphones. This makes FFTW3 an invaluable tool for enhancing the clarity and intelligibility of sound for users with

hearing impairments, directly contributing to the project's goal of accessible audio communication.

The json library from nlohmann, also known as 'JSON for Modern C++,' plays a pivotal role in the configuration and serialization of audio settings within this project, facilitating dynamic adjustments based on user preferences or specific audiogram data. The app is parsed JSON-formatted strings from the Python program (Acting as a server through a socket) that represent various audio settings. Additionally, libraries like cstring, cmath, iostream and string were used for basic coding.

In the development of our audio streaming application, critical libraries such as the socket library in Python and the Boost.Asio library in C++ play essential roles. As mentioned in section 4.5 of the dissertation, the Python socket library establishes a server to manage communication, sending audio parameters and commands to the C++ client application, which uses Boost.Asio to handle these network communications efficiently. Additionally, Python's "time" library is crucial for pacing the interactions between the Python server and the C++ client, ensuring the C++ client server has had enough time to start the server and awaits files to be sent (for synchronous data transmission).

The integration of these libraries and their harmonic interaction ensures that the software can efficiently manage complex audio processing tasks while maintaining a responsive and intuitive user interface.

# Chapter 5

## Testing and Evaluation

---

5.1 Methodology and Results	27
5.2 Consequences of Misconfiguration	38

---

### 5.1 Methodology and Results

When it comes to testing, there is a route of operations to execute along the program's flow to get a clear view of what happens and the validity of the results. The program begins by executing Python code, which connects to the C++ program through a specified port. The Python application has a toggle for a test mode, which significantly impacts the operational mode of the C++ backend by activating additional debugging outputs that trace computations and data handling steps. Since the C++ program's executable is called using Python, we do not have a direct view of the debugging information from the terminal, so a second thread is used in the Python program to continuously listen for output from the C++ program.

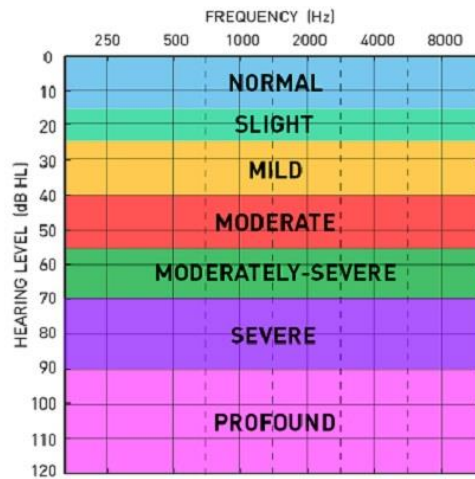
The test mode flag is passed from the Python interface to the C++ executable, allowing the latter to differentiate between normal operation and testing scenarios. Before delving deeper into the C++ processing, the Python script performs preliminary checks and setups. It queries the connected audio devices using the PortAudio library, which provides a cross-platform API for handling audio input and output. The script retrieves details about available audio devices, including whether they support input, output, or both. Additionally, it shows how many channels each device has available in its respective input and output sections. A couple of the output devices when testing the

validity of the PortAudio library are seen below, but the realistic output of a computer might contain a long list of outputs like the one in the appendices, which shows the actual output of the computer testing the software.

Device Number	Device Name	Driver Type	Channels In	Channels Out
2	CABLE-A Output (VB-Audio Cable	MME	8	0
10	Speakers (High Definition Audio	MME	0	2

Note that the two selected devices seen above, are the input and output devices used for the entire testing process seen.

Next up is the method for calculating the EQ settings to be adjusted. It is important to know that the audio industry usually applies gains and attenuations to frequency bands of  $\pm 12$  dB to make significant changes to the frequency balance of a track without making the adjustments too drastic. This way the targeted band can be changed while at the same time keeping the sound polished and the hardware cost of headphones and their power consumption low [12]. The algorithm utilized takes the audiogram data given by the user and iterates through each frequency a doctor may have tested the user on. Each frequency has a corresponding dB mark for how much gain is required by the user to hear said frequency [13]. Taking the dB threshold of each frequency, we add gain to sound depending on how close they are to 70dB. When on 70dB, we apply the max gain and set that frequency band for frequency shifting.



*Figure 1.0 An audiogram graph depicting the thresholds on which frequencies at that required level, change the severity profile of hearing loss [14].*

An example of audiogram input used for testing is the following:

Frequency (Hz)	Threshold (dB)
250	25
500	45
1000	45
2000	71
4000	74
8000	69
16000	85
20000	100

*Figure 1.1 shows the point in an audiogram where each frequency has its respective dB threshold for it to start being audible to a user.*

When observing the audiogram above, we can see that there are four frequency ranges that reach severe hearing loss (above 70dB). The frequency ranges 2000-4000, 4000-8000, and 16000-20000 have a threshold of more than 70dB for their respective frequencies to be audible. Taking the above audiogram, the program creates the

frequency adjustments to be parsed on the C++ program. When putting the above audiogram to the test, the frequency adjustments created are the following:

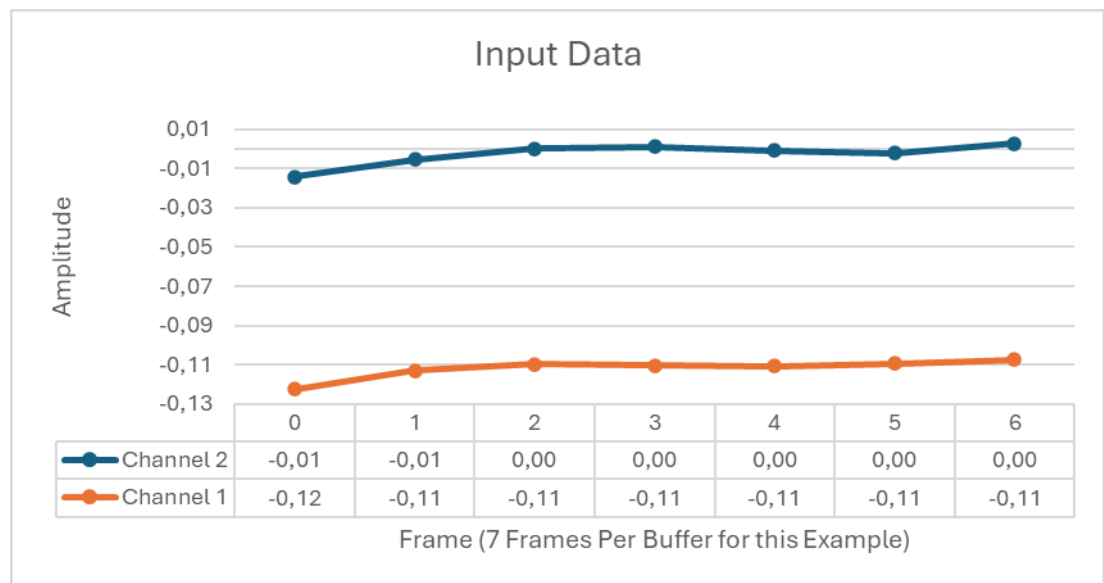
Frequency Range (Hz)	Adjustment Gain (dB)
250-500	4.29
500-1000	7.71
1000-2000	8.57
2000-4000	12
4000-8000	12
8000-16000	11.83
16000-20000	12

*Figure 1.2 shows a table with frequency ranges and their dB gain to be adapted after the algorithm was applied to Figure 1.1*

From the above table, we can see that the adjustments in gain are proportional to how close each frequency range was to 70dB. If it was above 70dB to hear, then it received the maximum dB boost. The goal is for the user to better hear frequencies that require a higher volume.

Once the above settings are figured out, the .json file sent to the C++ program sets some global parameters and structures for the C++ program to utilize when changing the audio live. The program then initiates the live audio stream, where the input samples are constantly fed to the program, which then alters them. The test case being studied sets the Virtual Audio Cable's input channels to two for stereo audio, which is the same number of output channels as the output device. The sample rate of the audio is 44100Hz for CD-quality audio since, according to the Nyquist Theorem will allow accurate representation of audio frequencies up to 22050 Hz, which covers the human hearing range, which is approximately from 20 Hz to 20000 Hz [15]. Note that the Nyquist theorem states that the sample rate must be at least twice the highest frequency to avoid aliasing from appearing in the audio playback. The Fast Fourier Transform converts a signal from the time domain to the frequency domain. In the time domain, the audio signal is represented as a series of amplitude values over time. The FFT processes these time-domain samples and produces a frequency-domain representation,

where the signal is decomposed into its constituent frequencies or frequency bins. This test case has a frame per buffer of 7, meaning that each buffer has 7 frames, or 14 samples (7 samples per channel). Once the audio signal is in the frequency domain, each bin represents a specific frequency component of the original signal. The magnitude of each bin indicates the amplitude of the corresponding frequency. Gain adjustments are then applied to each frequency bin. The gain for a bin is determined based on predefined frequency adjustment parameters, which can be derived from an audiogram. Studying the initial input buffer for this use case scenario, we have the below values:



*Figure 2.0 shows the graph of an input buffer with 7 frames (2 channels\* 7 frames = 14 samples). The orange line and markers represent the amplitude of each sample of channel one, and the blue line and markers represent the amplitude of each sample of channel two.*

In the graph above, there are 7 samples for each channel, and the amplitude of each sample is seen on the y-axis. The values are near 0 and have both positive and negative values. This is because the value of amplitude can range from -1 to 1.

Next up, we have the alteration on the gain of each frequency bin. When performing a Fast Fourier Transform (FFT) on a set of 7 frames (samples), the resulting frequency spectrum contains  $N/2+1$  unique frequency bins, where  $N$  is the number of input frames. This is due to the symmetrical properties of the FFT for real-valued input signals. In this

case, with  $N=7$ , we obtain 4 frequency bins. These bins correspond to the following frequencies: 0 Hz (DC component),  $1/7 \times 44100 \approx 6300$  Hz,  $2/7 \times 44100 \approx 12600$  Hz, and  $3/7 \times 44100 \approx 18900$  Hz. The absence of bins for intermediate frequencies such as 3150 Hz, 9450 Hz, and 15750 Hz is a direct consequence of the limited number of input frames, resulting in a lower frequency resolution. Knowing this, we have the following graph:

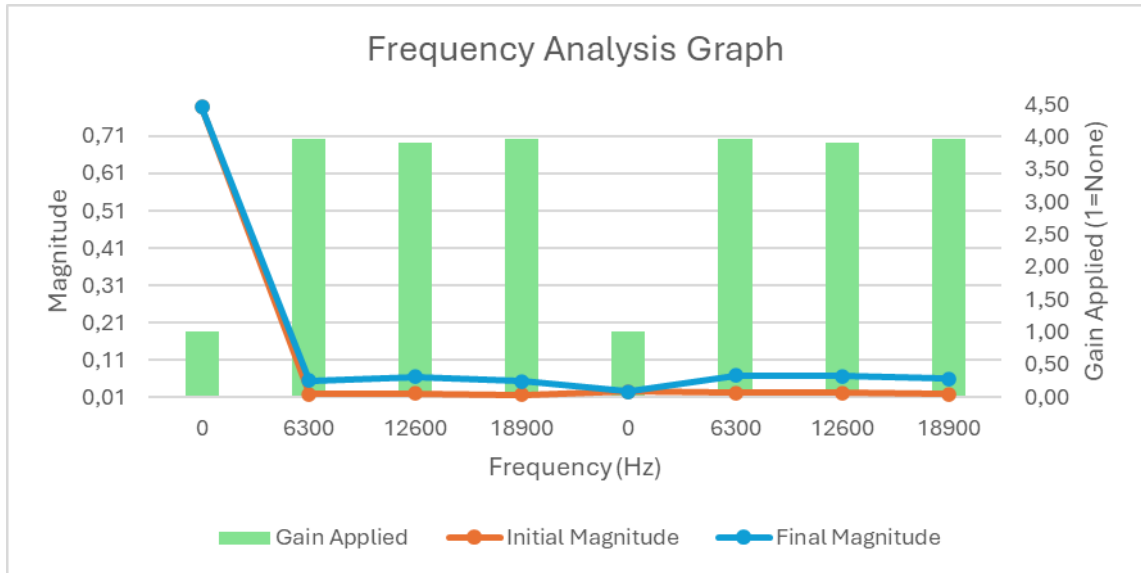
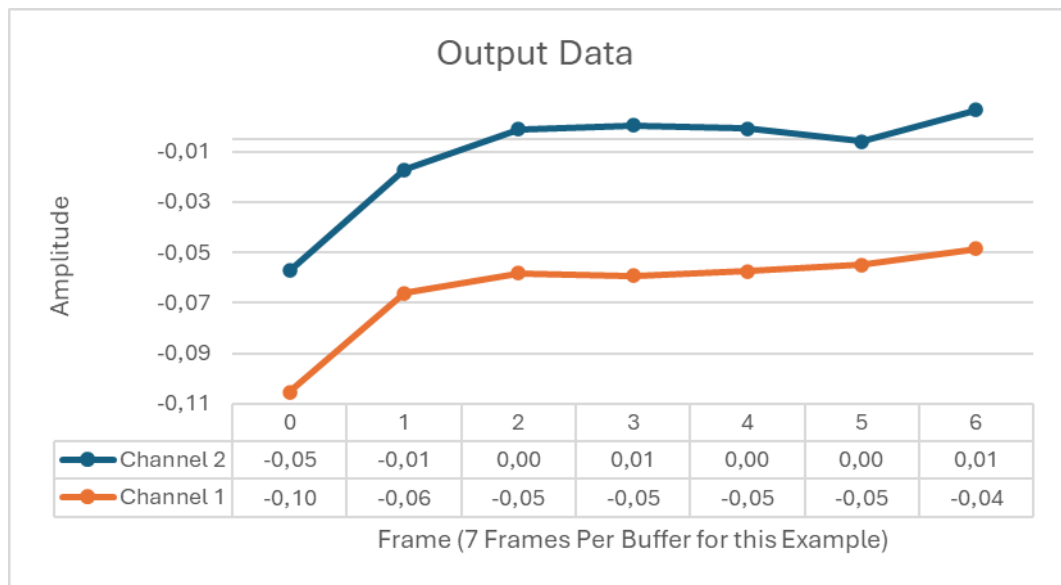


Figure 2.1 shows the frequency analysis graph on a buffer size of 7 frames.

In the graph, we can see the magnitude of each frequency bin on each of the two channels. On the x-axis, we can see the 4 frequency bins of the 1<sup>st</sup> channel, followed by the 4 frequency bins of the 2<sup>nd</sup> channel. We have a second y-axis showing the gain being applied to each frequency bin. The gain is calculated by the formula  $\text{Gain} = 10^{(\text{Gain in dB}/20)}$ , where “Gain in dB” is the dB boost determined after studying the audiogram to be the correct to apply in a specific frequency range. For example, when looking at the frequency bin at 6300, if we recall the adjusted EQ, a dB boost of 12 dB was found appropriate between 4000-8000 Hz as the audiogram in that range required 74 dB to be audible. Therefore, in that case, we have a linear gain of  $10^{(12/20)} \approx 3.981$ , which is reflected on the graph for 6300 on both channels. Once that gain is applied, it’s visible that the magnitude increases in each frequency bin where a gain of above 1 was applied. In the case of frequency 0, it is apparent that the magnitude does not increase, and that is because the gain is 1, which means no dB boost is to be given.



Lastly, below is a final graph depicting the output data to be given to the buffer:



*Figure 2.2 shows the output data of both channels with their respective 7 samples along with their amplitude after DSP was applied on it.*

After the Digital Signal Processing, the algorithm was applied to it, and the graph shows the output to be given to the buffer after Inverse FFT was applied to revert it back to the time domain. This output is then directly played back using C++'s PortAudio with the adjusted gains and frequency shifting.

It should be noted that due to the debugging and logging limits of the program with live audio playback, the way all the above tests were conducted was by observing the limited 7-frame buffer with the terminal output seen in the figure below:

```

Applying DSP Logic Test
Initial Input Buffer:
Frame 0: Channel 1: -0.122559 Channel 2: -0.0142517
Frame 1: Channel 1: -0.112823 Channel 2: -0.00540161
Frame 2: Channel 1: -0.109833 Channel 2: -3.05176e-05
Frame 3: Channel 1: -0.110382 Channel 2: 0.000976562
Frame 4: Channel 1: -0.11084 Channel 2: -0.000946045
Frame 5: Channel 1: -0.109497 Channel 2: -0.00213623
Frame 6: Channel 1: -0.1073 Channel 2: 0.00256348
Freq: 0 Hz, Initial Mag: 0.783234, Gain applied: 1, Final Mag: 0.783234
Freq: 6300 Hz, Initial Mag: 0.0125015, Gain applied: 3.98107, Final Mag: 0.049
Frequency 6300 Hz shifted to 5800 Hz (index 0)
Freq: 12600 Hz, Initial Mag: 0.0149818, Gain applied: 3.90327, Final Mag: 0.05
Freq: 18900 Hz, Initial Mag: 0.0118783, Gain applied: 3.98107, Final Mag: 0.04
Frequency 18900 Hz shifted to 18400 Hz (index 2)
Freq: 0 Hz, Initial Mag: 0.0192261, Gain applied: 1, Final Mag: 0.0192261
Freq: 6300 Hz, Initial Mag: 0.0159209, Gain applied: 3.98107, Final Mag: 0.063
Frequency 6300 Hz shifted to 5800 Hz (index 0)
Freq: 12600 Hz, Initial Mag: 0.0154718, Gain applied: 3.90327, Final Mag: 0.06
Freq: 18900 Hz, Initial Mag: 0.0134456, Gain applied: 3.98107, Final Mag: 0.05
Frequency 18900 Hz shifted to 18400 Hz (index 2)
464
Frame 4: Channel 1: -0.052443 Channel 2: 0.00413201
Frame 5: Channel 1: -0.0499827 Channel 2: -0.00100237
Frame 6: Channel 1: -0.0435251 Channel 2: 0.0115224
DSP Logic Test Completed

```

Figure 2.3 shows the program's terminal output when run in test mode for a buffer with a size of 7 frames.

In the terminal above, we can see the deconstructed samples of each frame, the frequencies studied in the frequency analysis graph and their magnitudes, as well as the frequencies on which they shift to if qualifies for the shift as per the audiogram restrictions for shifting mentioned before.

Investigating the program's logic further, we can look at what goes on in the magnitude spectrum, also known as the frequency spectrum. Since the nature of the program did not allow us to do so before, we mimicked the program's behavior in an "offline" setting where we read a local audio file to further study the logic of the software.

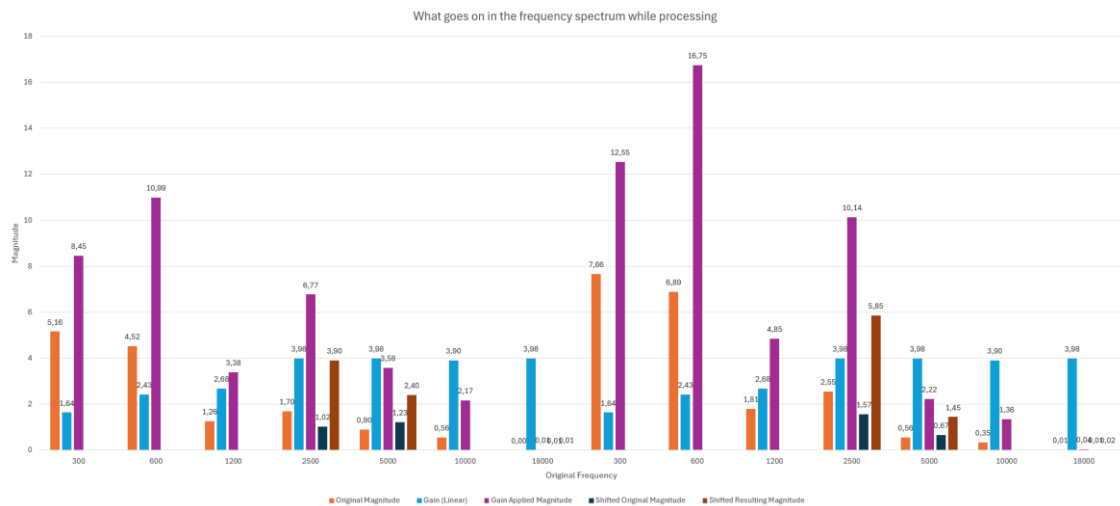
Sampling a few frequencies from a specific input buffer reading an audio file, we get the below numbers to explore:

Frequency	Original Magnitude	Gain (dB)	Gain (Linear)	Gain Applied Magnitude	Shifted Frequency	Shifted Original Magnitude	Shifted Resulting Magnitude
300	5,16	4,29	1,64	8,45			
600	4,52	7,71	2,43	10,99			
1200	1,26	8,57	2,68	3,38			
2500	1,70	12,00	3,98	6,77	2000	1,02	3,90
5000	0,90	12,00	3,98	3,58	4500	1,23	2,40
10000	0,56	11,83	3,90	2,17			
18000	0,00	12,00	3,98	0,01	17500	0,01	0,01
300	7,66	4,29	1,64	12,55			
600	6,89	7,71	2,43	16,75			
1200	1,81	8,57	2,68	4,85			
2500	2,55	12,00	3,98	10,14	2000	1,57	5,85
5000	0,56	12,00	3,98	2,22	4500	0,67	1,45
10000	0,35	11,83	3,90	1,36			
18000	0,01	12,00	3,98	0,04	17500	0,01	0,02

*Figure 3.0 shows the frequency spectrum data at specific frequencies sampled from an audio file while being processed in the software.*

The above table takes a few frequencies sampled from an audio file's input buffer. In the method that processes the magnitude spectrum, the frequency column shows the frequencies sampled from the magnitude spectrum. For each frequency, we have its original magnitude and the corresponding gain in dB depending on how close it is to 70dB, with the same logic we have seen before. To the right of the dB gain column, we have the linear gain corresponding to each frequency, and if we multiply the Original Magnitude value of each frequency by its Linear Gain, we get Gain Applied Magnitude, which is the magnitude we end up with once the gain is applied. Working with the same audiogram as the one in Figure 1.1, we can see if a frequency is within the ranges that apply for a frequency shift. The specific frequencies that are within those ranges are 2500, 5000, and 18000, as the user requires more than 70dB of gain to hear said frequencies. Since only those frequencies are up for shifting, they are the only ones that have corresponding entries in the column "Shifted Frequency", where 500 Hz is

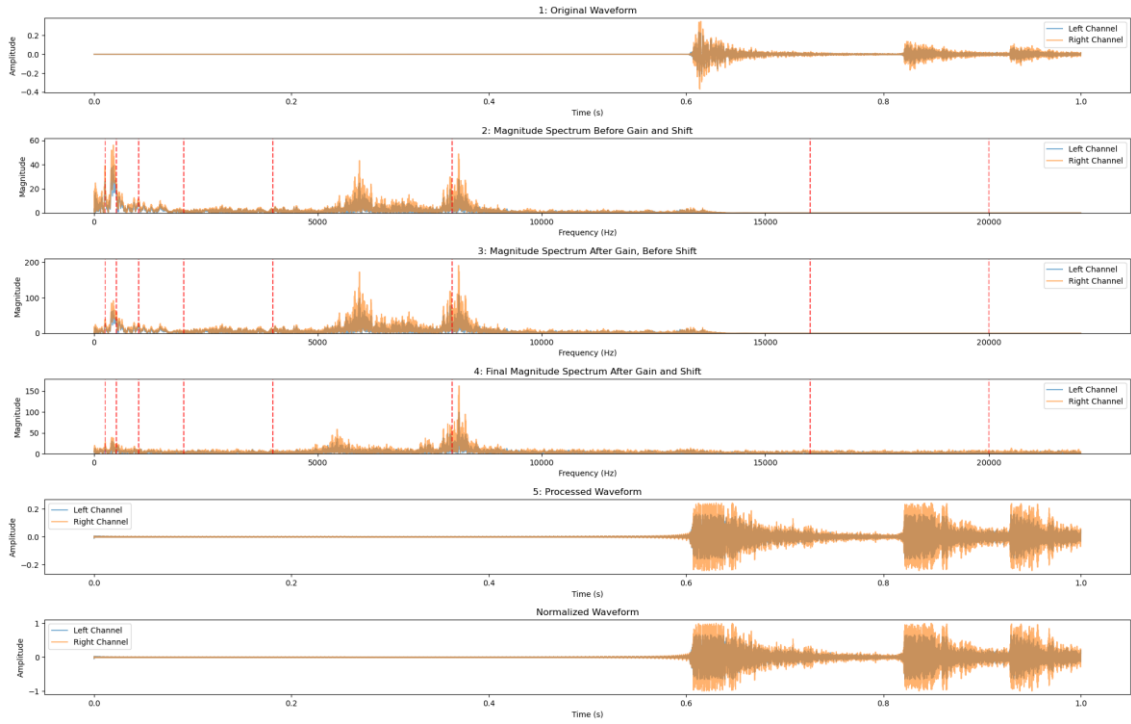
subtracted from each original frequency. Now that the destination frequency is found, the “Gain Applied Magnitude,” we figured that the original frequency is averaged with the “Shifted Original Magnitude,” which contains the magnitude that was already in the “Shifted Frequency.” The value from this calculation is put in the “Shifted Resulting Magnitude” column. How the values are multiplied or multiplied and then averaged can visually be seen in the graph below.



*Figure 3.1 is a graph depicting the sampled frequency spectrum data from the table in Figure 3.0 at specific frequencies (x-axis) sampled from an audio file while being processed in the software.*

The above graph shows the frequencies on the x-axis, with the left half of the frequency values representing the left channel and the right half representing the right channel. The bar values are displayed right above them to further aid in the visualization of the multiplication and averaging of values.

Looking at how the actual waveforms and magnitude spectrums as they appear visually, we get the figure below:



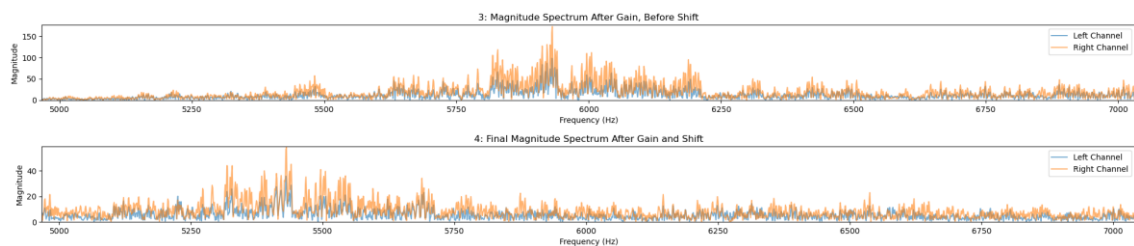
*Figure 3.1 shows the evolution of the waveform of the sampled audio file with a duration of 1 second.*

Each graph shows both the left and right channels of the audio waveform, with the left channel in transparent blue and the right channel in transparent orange on top of the blue waveform. The first graph shows the original sampled audio file waveform with a duration of 1 second and a sample rate of 44100 Hz, meaning there are 2.646.000 samples visible in each waveform graph.

All the magnitude spectrums have frequencies on the x-axis with ranges of 0 to 20000, which is the range we care about, as explained in previous sections of the dissertation. graph. If we take a look at the graph with the “Magnitude Spectrum Before Gain and Shift” (graph 2), we can see the original magnitude spectrum has magnitudes with values up to 60.

Looking at the “Magnitude Spectrum After Gain, Before Shift” (graph 3), we can see that all the magnitude values across the spectrum have relatively the same dominance as graph 2, but with a much larger magnitude on specific values. This is because the linear gain is multiplied on each frequency bin, as explained in Figure 3.1. It is even more apparent if the y-axis is looked at, which reaches values of magnitude up to 200.

Advancing to the graph “Final Magnitude Spectrum After Gain and Shift” (graph 4), we can see that the y-axis values drop to a max of around 150, but that is not because of the logic of the processing dropping them, but rather a result of the frequency shift on this very specific sample from the audio file. In fact, this can be supported when observing the graph around the frequency values of 5000 to 7000. That area was considerably dominant in magnitude values around 5750-6200, whereas after the shift, we can visually see how those values are “shifted down” by 500 Hz. The picture below zooms in on the value ranges pointed out:



When looking at the “Processed Waveform” (graph 5), it’s apparent that the shape of the graph reminds us of the shape of graph 1, but with further boosted amplitudes due to the gain boost produced as an indirect result of the audiogram values. If a few slight changes can be seen that are not explained by the gain, they are not artifacts, but rather the result of the frequency shifting.

Finally, when looking at the very last waveform “Normalized Waveform”, it is seemingly identical to graph 5, and that is the goal since the only thing we want to change are the amplitude of the waveform so that it ends up being normalized between the values 1 to -1, as this is the range waveform should be between.

## 5.2 Consequences of Misconfiguration

While the application is currently carefully configured, future adjustments and additions to the software pose a threat of misconfiguration that can have significant and far-reaching consequences, affecting both the performance and possibly the user’s health. One of the primary impacts is on the operational efficiency and functionality of the software. Incorrect settings may lead to suboptimal performance, where the software

does not utilize system resources effectively. This can result in slower processing times, increased latency, and a general decrease in the responsiveness of the application. For instance, misconfigured memory allocation settings could cause the software to either overuse or underuse available memory, leading to either system crashes or inefficient processing.

Furthermore, as mentioned, misconfiguration of the EQ gain can have severe consequences, particularly in terms of auditory health for both users with normal hearing and those with hearing impairments. Incorrectly configured EQ settings that amplify sound frequencies beyond safe listening levels can lead to prolonged exposure to loud sounds, significantly increasing the risk of noise-induced hearing loss [16]. For users with normal hearing, excessive gain can cause discomfort, tinnitus, and permanent damage to the auditory system. For hearing-impaired users, the risks are even more pronounced; over-amplification can exacerbate existing hearing conditions and potentially accelerate the degradation of their hearing abilities. Therefore, ensuring accurate and appropriate EQ gain settings in the software's evolution is crucial to prevent further auditory damage and to promote safe listening practices.

If there is a spike due to a miscalculation on the processed waveform, the normalized waveform will take that spike as the borders with a maximum of 1 and a minimum of -1, thus breaking the entire sample buffer by shrinking the rest of the amplitudes.

# Chapter 6

## Accessibility and User Diversity

---

6.1 User Profiles and Scenarios of Hearing Impairments	40
6.2 How the software can help said User Profiles	42

---

### 6.1 User Profiles and Scenarios of Hearing Impairments

Sensorineural hearing loss (SNHL) [17] is the most common type of permanent hearing impairment. It occurs due to damage to the hair cells in the cochlea (a part of the inner ear) or to the nerve pathways from the inner ear to the brain. Common causes include aging, exposure to loud noise, genetics, head trauma, and diseases. SNHL typically results in a reduction of sound level or the ability to hear faint sounds. It can also affect the clarity of speech, especially in noisy environments, making it difficult to distinguish high-frequency sounds like 's' or 'th'. This type of hearing loss is generally irreversible.

Age-related hearing loss, or presbycusis [18], is a type of sensorineural hearing loss that occurs as part of the natural aging process. It falls under the category of disorders. It gradually affects both ears and is most pronounced at higher frequencies. This condition often makes it challenging for older adults to hear and understand speech, particularly when there is background noise. Presbycusis is cumulative and influenced by a combination of genetic, health, and environmental factors. It is one of the most common conditions affecting elderly adults.

Noise-induced hearing loss (NIHL) [19] is another form of sensorineural hearing loss that results from exposure to excessively loud sounds, either from a single loud event or repeated exposure to high levels of noise. This exposure damages the hair cells in the



cochlea, leading to hearing impairment. NIHL can be sudden or gradual, affecting one or both ears, and it can be temporary or permanent. Common sources include industrial noise, loud music, and firearms. Preventive measures, such as using hearing protection, are crucial to avoiding this type of hearing loss.

Sudden sensorineural hearing loss (SSHL), commonly known as sudden deafness, is an abrupt loss of hearing, typically in one ear. It can occur instantly or over a span of several days [20]. SSHL is considered a medical emergency and requires prompt treatment for the best chance of recovery. The cause of sudden deafness is often unknown, but it can be associated with viral infections, head injuries, autoimmune diseases, and circulation problems. Approximately half of those affected recover spontaneously, usually within one to two weeks of onset.

Tinnitus [21] is commonly described as ringing in the ears, but it can also manifest as buzzing, hissing, whistling, swooshing, or clicking sounds that are not present in the external environment. Tinnitus is not a disease itself but rather a symptom of underlying conditions, such as age-related hearing loss, ear injury, or a circulatory system disorder. The discomfort from tinnitus can vary from slightly annoying to severely debilitating, affecting daily activities and quality of life. It can be present in one or both ears and may be constant or come and go.

Auditory Neuropathy Spectrum Disorder (ANSD) [22,23] is a rare form of hearing loss that occurs when sound enters the ear normally, but because of damage to the inner ear or the auditory nerve, the transmission of sound signals to the brain is impaired. People with ANSD can experience a range of hearing difficulties, from mild to severe, and may have difficulty understanding speech clearly, especially in noisy environments. ANSD can occur in individuals of any age, from infants to the elderly, and its causes can include genetic factors, conditions that reduce oxygen supply to the brain, and certain infections.

Congenital hearing loss [24] refers to hearing loss that is present at birth. It can be caused by genetic factors, infections during pregnancy, complications during birth, or

other conditions like prematurity. Congenital hearing loss might be detected through newborn hearing screening programs and can range from mild to profound.

## **6.2 How the software can help said User Profiles**

Finding ways the software can mitigate the effects of the types of hearing loss mentioned, we examine ways the EQ and frequency shifting can help, in the same way that people fitted with hearing aids are helped by them.

Digital signal processing techniques in hearing aids have significantly improved hearing rehabilitation in sensorineural hearing loss patients [25].

New treatment options for profound unilateral sensorineural hearing loss include bone-anchored hearing aids, transcranial hearing aids, and cochlear implants [26]. Adults with mild sensorineural hearing loss (MSNHL) benefit from hearing aids, despite limited and dated studies [2].

Presbycusis (Age-related hearing loss) [18] is a progressive bilateral loss of hearing for high tones in the aged, with no medical therapy effective in treating it; auditory training and hearing aids are crucial for rehabilitation.

Children with auditory neuropathy spectrum disorder (ANSD) can achieve functional speech outcomes similar to those with mild-to-severe sensorineural hearing loss when fitted with hearing aids [23].

The use of hearing aids alone provides significant benefits with respect to alleviating the effects of tinnitus [27].

When congenital hearing loss involves a conductive component, such as malformations of the ear structures that can block sound transmission, hearing aids or similar devices can indeed help by amplifying sound to bypass the obstruction [28]. The article on fully implantable hearing aids in patients with congenital auricular atresia would be relevant in this context if the congenital hearing loss includes a conductive component,

especially in cases where the anatomy of the ear canal or middle ear is affected, as is the case with auricular atresia.

Noise-induced hearing loss leads to disability, and hearing aids can be beneficial even at a low degree of hearing loss, as they can serve as therapeutic instruments [29].

# Chapter 7

## Legal and Ethical Considerations

---

7.1 GDPR Compliance	44
7.2 Global Compliance	45

---

### 7.1 GDPR Compliance

Adherence to the General Data Protection Regulation (GDPR) [30] is paramount in the development of audio enhancement technology, particularly when it involves the processing and storage of personal data such as audiograms. GDPR sets the benchmark for data protection laws in the European Union and impacts any entity that handles the data of EU citizens, regardless of the company's location.

Firstly, consent is a critical aspect of GDPR. The software will need to ensure that users are fully informed about what data is collected, why it is collected, and how it will be used. A clear and concise consent form is provided at the point of data collection, which users must agree to before any personal data is processed. This consent can be withdrawn by the users at any time, reflecting the GDPR's stipulations on user consent management. Secondly, the principle of data minimization must be rigorously applied. The software will only collect data necessary for its function, which in this case includes audiograms and user preferences for sound settings. No extraneous data will be collected, ensuring compliance with GDPR's requirements for relevance and limited data collection.

Furthermore, the system will need to implement robust security measures to protect the data from unauthorized access, alteration, and loss. This includes the use of encryption for data at rest and in transit, regular security audits, and secure coding practices. Lastly,

the software and its future database will need to be designed to support data subject rights outlined in GDPR, including the right to access, the right to rectification, the right to erasure, and the right to data portability. Users should be able to easily access their data, request corrections, or even delete their information from the servers, all through simple interfaces within the application.

By integrating these GDPR principles directly into the software's architecture and operational procedures, the project upholds the highest standards of data protection, respects user privacy, and enhances trust in our technology.

## **7.2 Global Compliance**

Global compliance in the development of audio enhancement technology entails understanding and adhering to a variety of international laws and standards that regulate privacy, data protection, and accessibility. While the General Data Protection Regulation (GDPR) sets a rigorous framework for handling personal data within the European Union, similar principles are echoed in other regulatory environments such as the California Consumer Privacy Act (CCPA) [31] in the United States, the Personal Information Protection and Electronic Documents Act (PIPEDA) in Canada [32], and the General Data Protection Law (LGPD) [33] in Brazil. Each of these regulations mandates strict guidelines on the consent of data subjects, the minimization of data collection, the security of the data stored, and the rights of individuals to access or delete their personal information. Ensuring compliance across these diverse legal landscapes necessitates a flexible system architecture that can be configured to meet specific local requirements, such as data residency provisions that require data to be stored within a particular jurisdiction.

# Chapter 8

## Discussion

---

8.1 Interpretation of Results	46
8.2 Analysis of User Interactions with software through forms	47
8.3 Assumptions to factors that may affect system evaluation from users	51

---

### 8.1 Interpretation of Results

The primary objective of this project was to develop a DSP algorithm capable of dynamically adjusting audio frequencies to enhance the listening experience for users with varying degrees of hearing impairment. The analysis of the results obtained from the methodology demonstrates that the project has successfully achieved its goals, meeting the predefined criteria for gain adjustments and frequency shifting.

The Python interface seamlessly interacts with the C++ backend to manage the audio processing pipeline, including preliminary checks and device configuration using the PortAudio library. The debugging outputs activated during test mode provided detailed insights into the program's operation through a secondary thread due to limitations in direct terminal access.

The EQ settings calculated from the audiogram data adhered to industry standards, applying gains and attenuations within a range of  $\pm 12$  dB. This ensured significant but non-drastic changes to the frequency balance. The resulting frequency adjustments, generated from the algorithm and sent to the C++ program, reflect a meticulous adaptation of the user's hearing profile.

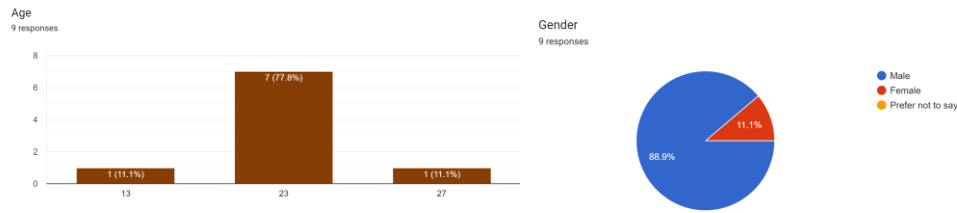
The analysis of how samples are adjusted included examining an initial input buffer of 7 frames, performing FFT to transform the audio into the frequency domain, applying gain adjustments, and reverting the processed data back to the time domain through Inverse FFT. This comprehensive approach verified that the gain adjustments were applied correctly.

The results from the frequency spectrum analysis of the sampled audio file provided further validation of the DSP algorithm's effectiveness. The sampled frequencies, their original magnitudes, applied gains, and resulting magnitudes were consistent with the expected outcomes based on the input audiogram. The process of frequency shifting was accurately implemented, as seen in the example of 2500 Hz being shifted to 2000 Hz with the magnitudes averaged appropriately. This ensured that the frequency shifts did not introduce distortions or artifacts, maintaining a high-quality audio output. Lastly, the rigorous analysis of the waveforms and magnitude spectrums from samples taken directly from an audio file is paramount to visually connecting the logic of the program with the expected results.

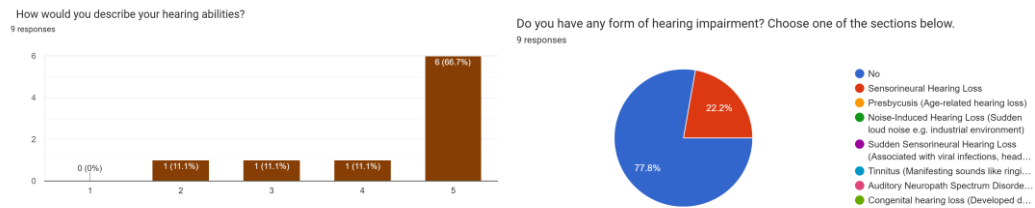
## **8.2 Analysis of User Interactions with software through forms**

A questionnaire was conducted using the Google Forms service. There were four sections of questions: the first was for general information, the second was for hearing-impaired users, the third was for “Experience with Audio Equipment,” and the fourth (“Software”) was for evaluating the users on the system. Due to the extensive number of questions, not all the graphs will appear in the dissertation, but everything will be available in the appendices to support the conclusion of the software assessment. While displaying the results, this section will investigate the cause for some of them as well by framing assumptions and the logic behind them.

In the first section, nine individuals provided their answers, two of whom were hearing-impaired users.

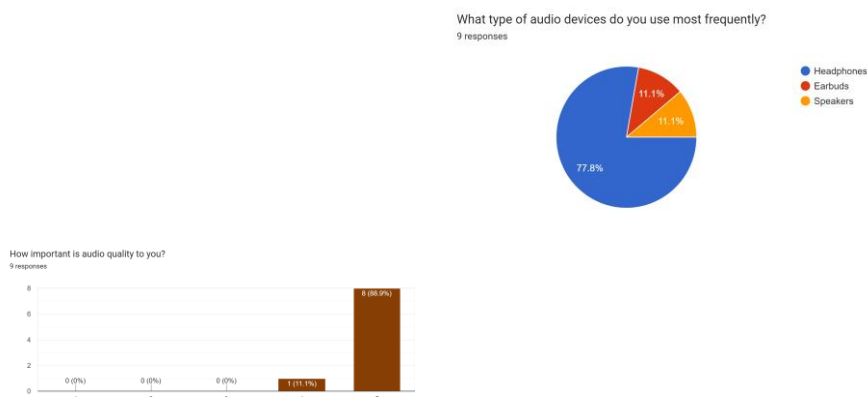


Based on the images provided, it is apparent that the demographic primarily consists of young males.



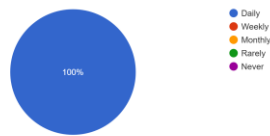
Most responders describe their hearing as well to excellent, while the impaired users describe them as bad to mediocre, as one of them has profound hearing loss and the other moderate.

In the second section, we see that both the responders with hearing impairments state that they suffer from Sensorineural Hearing Loss. One of them uses a hearing aid weekly, and the other does not use one at all. Both have been impaired for over 10 years, and state that enhancing their hearing is very important to them. Additionally, one impaired user answered “Few” to the question “Were you able to make out different sounds, voices, or extra tones that you normally weren't able to hear?” and the other answered “Yes.”





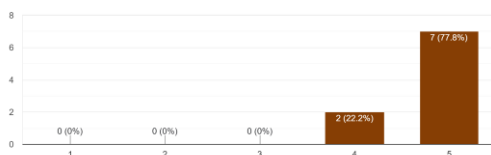
How often do you use audio streaming services?  
9 responses



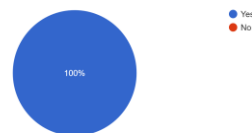
Moving on to the third section, “Experience with Audio Equipment,” we see that users utilize headphones for audio streaming. We also see that 100% of them use audio streaming services and that think audio quality is critical.

The last section (section 4) showcases the user assessment of the software.

How important do you think the software as a concept is for providing more accessibility in the field of gaming for people with hearing impairments?  
9 responses

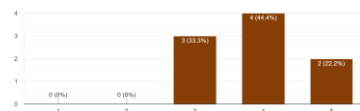


Were you able to clearly differentiate sound coming from the left and right channels for the stereo sound the software produces (left channel=left ear/speaker, right channel=right ear/speaker).  
9 responses

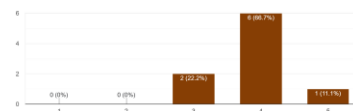


Most of them think the software is imperative for hearing-impaired user accessibility in gaming, and a few think it is important. All of them were able to distinguish between the left and right channels of incoming audio to the speaker.

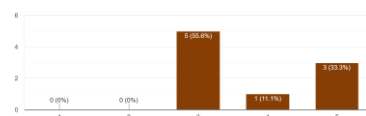
To what degree were you able to make out new sounds that you previously could not when listening to instrumental music? Music tested: Lost Woods - The Legend of Zelda: Ocarina of Time  
9 responses



To what degree were you able to make out new sounds that you previously could not when listening to rock music? Music tested: Queen - Bohemian Rhapsody  
9 responses



To what degree were you able to make out new sounds that you previously could not when listening to synth-pop music? Music tested: The Weeknd - Save Your Tears  
9 responses

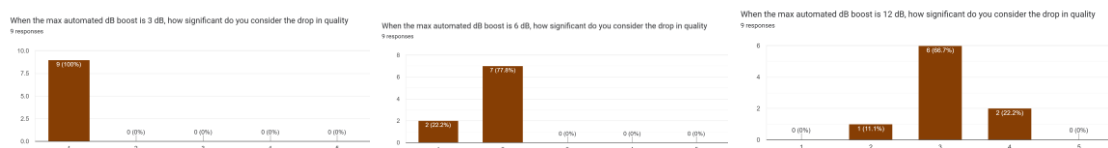


The software was tested on 3 separate music categories: instrumental, rock, and synth-pop (the original music was heard before the software-altered version). When it comes to instrumental music, with the specific one consisting of a few instruments making a simple melody, most of them found the software advantageous in recognizing sound cues they did not before. For rock music, Queen’s Bohemian Rhapsody was chosen with lots of vocals of all pitches along with the rock music. They also were able to recognize new sounds they could not hear before, probably due to the diverse nature of

the song. Lastly, The Weeknd's Save Your Tears was used for synth-pop, where most people were not able to make out any new sounds, and it is assumably due to the chaotic background produced in synth-wave music, which disallows the distinction of specific sound cues that might be happening at the same time. This assumption is made with the same logic that people with hearing impairments have trouble hearing noises when there is intense background music.

It is vital to mention that the answers of the people with hearing impairments who answered the questionnaire didn't differentiate from the average answer given by people with normal hearing in the above music tests and why this happened. People who are hard of hearing are normally unable to hear sounds played in the original music from the above three soundtracks due to the physical nature of their hearing loss. However, when the music is altered, hearing-impaired users can hear the sound they formerly could not because of the enhancement of certain frequencies and the shifting of frequencies they could not hear no matter how much the music received gains. Therefore suggesting that the software successfully worked to their advantage in hearing music. At the same time, people with normal hearing report the same outcome, not because they benefited in the same way as hearing-impaired, but because they experienced the altercations of the original music they **already** recognized to the new-sounding version with the frequency shifting at work. This observation is crucial since the hearing-impaired users suggest they can hear new sound cues, and the normal-hearing users confirm their claim.

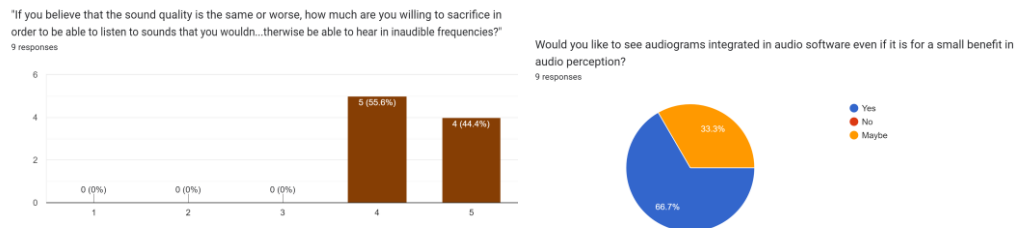
Moving on to the dB boost given to frequencies, the people who took the test heard the original music before analyzing it at different max boosts across the magnitude spectrum, and we got the results below:



We see that all users noticed no change in the audio output when boosting to 3dB, most noticed a small drop in quality at 6dB, and most of them were confident there was a drop in quality at 12dB, but not too significant. This test was conducted a second time

on the 3dB to remove any placebo factor that might have played a role the first time around in the 3dB test after hearing the original soundtrack.

Additionally, 67% of users noticed improvements in audio clarity. Most stated that they could hear the bass, treble, vocals, and background instruments well. Almost all of them did not notice any delay in the audio playback when taking a closer look at the live audio stream.



Most of the users stated they were willing to sacrifice audio quality for the sake of hearing new sound cues they otherwise would not be able to, with the hearing-impaired users strongly advocating for these changes in the industry of gaming headphones.

### 8.3 Assumptions to factors that may affect system evaluation from users

Analyzing the form results, it is not farfetched to assume that the acclimatization period of any user to the software to the point where it becomes beneficial to them may be long and vary from user to user. This is because the same is observed for hearing-aid users, as seen in section 3.3. Therefore, it is also not a stretch to assume that the rating on the experience and the usefulness the users have on the software may gradually increase in favor of the software with further use.

It is also safe to assume that any minuscule latency perceived by the user between the time the audio is generated from an application to the time it reaches the user's ears is an immediate effect of the Virtual Cables. This is because they are used to route the sound and act as a middleman for the sound configuration, a process that would otherwise be skipped in a scenario where headphones are being developed, and their driver is in the arsenal.

# Chapter 9

## Conclusion and Future Work

---

9.1 Summary of Findings	52
9.2 Recommendations for Future Research	53

---

### 9.1 Summary of Findings

The research successfully demonstrated the effectiveness of a digital signal processing (DSP) system designed to improve live audio for gaming headphones. The main goal was to create software that can adjust audio in real time, focusing on frequency ranges important for people with hearing impairments. The implementation of this software confirmed its ability to dynamically adjust and process live audio signals, greatly improving the auditory experience for users. The findings show that the software effectively applies customized equalization and frequency shifts, enhancing the clarity and quality of audio. Additionally, the dissertation explains that the system can assist individuals with various degrees of hearing loss, making it a versatile tool for improving auditory capabilities. Furthermore, the research has provided a comprehensive schema for an ideal software user interface (UI), which emphasizes accessibility and ease of use, ensuring that users can intuitively navigate and customize their audio settings. Overall, when combining all the findings together, the software effectively and efficiently provides a tool for hearing-impaired individuals to gain immersion and a competitive edge in the world of gaming.

### 9.2 Recommendations for Future Research

Future research should explore several avenues to further enhance the capabilities and applications of the developed DSP software. One area of interest is the integration of machine learning algorithms to adaptively refine audio processing based on user preferences and real-time feedback. Additionally, expanding the software's compatibility with a broader range of audio devices, including various gaming consoles, mobile platforms, and virtual reality systems, would increase its accessibility and utility. Investigating the psychological and physiological impacts of prolonged use of the DSP system, particularly concerning auditory fatigue and cognitive load, would provide valuable insights into its long-term effectiveness and user well-being. Furthermore, collaboration with audiologists to refine the frequency adjustment algorithms could lead to more precise and personalized audio enhancements. Rigorous clinical trials on both normal-hearing and hard-of-hearing patients should be conducted to validate the effectiveness of the software across all parameter combinations. The software's optimization is also imperative when functioning across various computer hardware architecture and their embedded operating systems. Finally, exploring the potential of this software in educational and professional settings, where clear audio is critical, could open new avenues for its application. By addressing these areas, future research can build on the current findings to create even more sophisticated and user-friendly audio enhancement tools.

# Bibliography

- [1] [W. Penn, "Fundamentals of hearing-aid design" Electr. Eng., vol. 63, pp. 1-1, Oct. 1944.](#)
- [2] [C. E. Johnson, J. Danhauer, B. B. Ellis, and A. Jilla, "Hearing Aid Benefit in Patients with Mild Sensorineural Hearing Loss: A Systematic Review" Journal of the American Academy of Audiology, vol. 27, no. 4, pp. 293-310, Apr. 2016, doi: 10.3766/jaaa.14076.](#)
- [3] [A. Oxenham, "How We Hear: The Perception and Neural Coding of Sound" Annu. Rev. Psychol., vol. 69, pp. 27-50, Jan. 4, 2018.](#)
- [4] [A. Hudspeth and M. Konishi, "Auditory neuroscience: development, transduction, and integration" Proc. Natl. Acad. Sci. USA, vol. 97, no. 22, Oct. 24, 2000.](#)
- [5] [S. Barden, C. Simon, and N. Jones, "Hearing Loss" InnovAiT, vol. 3, no. 11, Nov. 2010.](#)
- [6] [A. Kral and G. M. O'Donoghue, "Profound Deafness in Childhood" N. Engl. J. Med., vol. 363, no. 15, pp. 1438-1450, Oct. 2010.](#)
- [5] [T. Y. C. Ching and H. Dillon, "Major findings of the LOCHI study on children at 3 years of age and implications for audiological management" Int. J. Audiol., vol. 52, sup2, pp. S65-S68, Dec. 2013.](#)
- [6] [Q. Wang, R. Liang, S. Rahardja et al., "Piecewise-Linear Frequency Shifting Algorithm for Frequency Resolution Enhancement in Digital Hearing Aids" Appl. Sci., vol. 7, no. 4, art. 335, Mar. 2017.](#)
- [7] [T. Yasuno, S. Ibata, N. Kawai, H. Sasaki, "Hearing Aid Adaptation for Deaf Children" Audiol. Jpn., vol. 15, no. 4, pp. 291-302, 1972.](#)

- [8] [D. Brooks, "The time course of adaptation to hearing aid use" Br. J. Audiol., vol. 30, no. 1, pp. xxx-xxx, 1996.](#)
- [9] [S. Arlinger, T. Lunner, B. Lyxell, and M. Pichora-Fuller, "The emergence of cognitive hearing science," Scand. J. Psychol., vol. 50, no. 5, Oct. 2009.](#)
- [10] [J. G. W. Bernstein and A. Oxenham, "The relationship between frequency selectivity and pitch discrimination: sensorineural hearing loss" J. Acoust. Soc. Am., vol. 120, no. 6, pp. 3929–3945, 2006.](#)
- [11] [PyQT5 Software documentation.](#)
- [12] [J.A. Lima, A. Petraglia, "On designing OTA-C graphic-equalizers with MOSFET-triode transconductors," ISCAS 2001. The 2001 IEEE International Symposium on Circuits and Systems \(Cat. No.01CH37196\), May 6, 2001.](#)
- [13] [E. Fowler, "Interpretation of audiograms," Arch Otolaryngol., vol. 12, pp. 760-768, 1930. doi:10.1001/archotol.1930.03570010864006.](#)
- [14] <https://www.babyhearing.org/what-is-an-audiogram>
- [15] <https://www.sciencedirect.com/topics/engineering/nyquist-theorem#:~:text=The%20Nyquist%20theorem%20specifies%20that,equal%20to%20twice%20per%20cycle>.
- [16] [M. Śliwińska-Kowalska, K. Zaborowski, "WHO Environmental Noise Guidelines for the European Region: A Systematic Review on Environmental Noise and Permanent Hearing Loss and Tinnitus," Int. J. Environ. Res. Public Health, vol. 14, pp. 1265, 2017.](#)
- [17] [R. Salvi, D. Henderson, R. Hamernik, and one other author, "Neural correlates of sensorineural hearing loss" Ear and Hearing, vol. 4, no. 3, pp. 115-129, May 1983.](#)

- [18] [R. Lehman and A. L. Miller, "PRESBYCUSIS" Journal of the American Geriatrics Society, vol. 18, no. 6, June 1970.](#)
- [19] [E. Daniel, "Noise and hearing loss: a review," The Journal of School Health, vol. 77, no. 5, May 2007.](#)
- [20] [B. Schreiber, C. Ågrup, D. Haskard, L. Luxon, "Sudden sensorineural hearing loss" The Lancet, vol. 375, no. 9721, pp. 1203-1211, Apr. 2010.](#)
- [21] [J. Eggermont, L. Roberts, "The neuroscience of tinnitus" Trends Neurosci., Nov. 2004.](#)
- [22] [K. Kaga, "Auditory nerve disease and auditory neuropathy spectrum disorders" Auris Nasus Larynx, vol. 2, Feb. 2016.](#)
- [23] [E. Walker, R. McCreery, M. Spratford, and P. Roush, "Children with Auditory Neuropathy Spectrum Disorder Fitted with Hearing Aids Applying the American Academy of Audiology Pediatric Amplification Guideline: Current Practice and Outcomes" J. Am. Acad. Audiol., Mar. 1, 2016.](#)
- [24] [A. M. H. Korver, R. J. H. Smith, G. Van Camp, M. R. Schleiss, M. A. K. Bitner-Glindzicz, L. R. Lustig, S.-i. Usami, and A. N. Boudewyns, "Congenital hearing loss" Nature Reviews Disease Primers, vol. 3, Art. no. 16094, 2017.](#)
- [25] [J. Chung, "Rehabilitation of Sensorineural Hearing Loss: Hearing Aid" Hanyang Medical Reviews, vol. 35, no. 2, pp. 97-102, May 2015.](#)
- [26] [C. E. Bishop and T. Eby, "The current status of audiologic rehabilitation for profound unilateral sensorineural hearing loss" The Laryngoscope, vol. 120, no. 3, Mar. 2010.](#)
- [27] [J. Henry, M. T. Frederick, S. Sell, et al., "Validation of a Novel Combination Hearing Aid and Tinnitus Therapy Device" Ear and Hearing 36\(1\):p 42-52., Jan. 2015.](#)



[28] R. Siegert, S. Mattheis, J. Kasic, "Fully Implantable Hearing Aids in Patients With Congenital Auricular Atresia" Laryngoscope, vol. 117, no. 2, Feb. 2007.

[29] C. Giordano, M. Garzaro, J. Nadalin, G. Pecorari, R. Boggero, P. Argentero, R. Albera, "Noise-induced hearing loss and hearing aids requirement" Acta Otorhinolaryngologica Italica : organo ufficiale della Societa italiana di otorinolaringologia e chirurgia cervico-facciale, vol. 28, no. 4, pp. 200-205, Aug. 2008.

[30] <https://gdpr.eu/what-is-gdpr/#:~:text=The%20General%20Data%20Protection%20Regulation,to%20people%20in%20the%20EU.>

[31] <https://oag.ca.gov/privacy/ccpa>

[32] [https://www.priv.gc.ca/en/privacy-topics/privacy-laws-in-canada/the-personal-information-protection-and-electronic-documents-act-pipeda/pipeda\\_brief/](https://www.priv.gc.ca/en/privacy-topics/privacy-laws-in-canada/the-personal-information-protection-and-electronic-documents-act-pipeda/pipeda_brief/)

[33] <https://iapp.org/resources/article/brazilian-data-protection-law-lgpd-english-translation/>

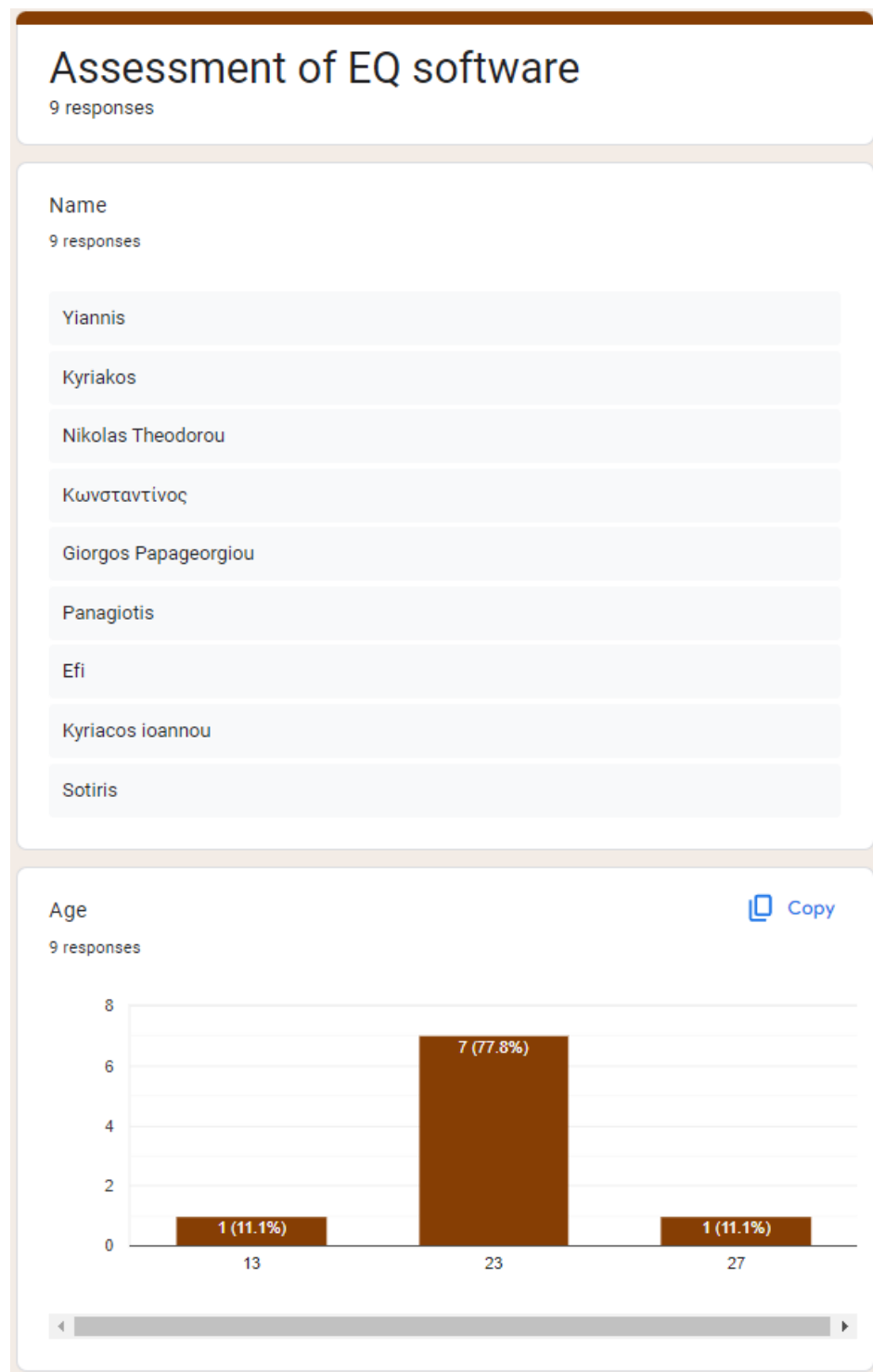
# Appendices

Device Number	Device Name	Driver Type	Channels In	Channels Out
0	Microsoft Sound Mapper - Input	MME	2	0
1	Headset Microphone (3- DualSens	MME	2	0
2	CABLE-A Output (VB-Audio Cable	MME	8	0
3	Microphone (Razer Seiren Mini)	MME	1	0
4	Microphone (Steam Streaming Mic	MME	8	0
5	CABLE-B Output (VB-Audio Cable	MME	8	0
6	CABLE Output (VB-Audio Virtual	MME	8	0
7	Microsoft Sound Mapper - Output	MME	0	2
8	CABLE-A Input (VB-Audio Cable A	MME	0	8
9	Speakers (3- DualSense Wireless	MME	0	4
10	Speakers (High Definition Audio	MME	0	2
11	CABLE-B Output (VB-Audio CABLE-B)	Windows WDM-KS	8	0
12	Speakers (VB-Audio CABLE-B)	Windows WDM-KS	0	8
13	Headphones ()	Windows WDM-KS	0	8
14	Line Out ()	Windows WDM-KS	0	2
15	Stream ()	Windows WDM-KS	2	0
16	Line Out ()	Windows WDM-KS	0	2
17	Microphone ()	Windows WDM-KS	2	0
18	Output ()	Windows WDM-KS	0	2
19	Headphones ()	Windows WDM-KS	0	8
20	Headphones ()	Windows WDM-KS	0	8
21	Speakers (HD Audio Speaker)	Windows WDM-KS	0	2
22	SPDIF Out (HD Audio SPDIF out)	Windows WDM-KS	0	2
23	Speakers ()	Windows WDM-KS	0	2
24	CABLE Output (VB-Audio Point)	Windows WDM-KS	8	0
25	Speakers (VB-Audio Point)	Windows WDM-KS	0	8
26	Microphone ()	Windows WDM-KS	1	0
27	Headset Earphone ()	Windows WDM-KS	0	1
28	Headset Microphone ()	Windows WDM-KS	1	0
29	Microphone (Steam Streaming Microphone Wave)	Windows WDM-KS	8	0
30	Speakers (Steam Streaming Microphone Wave)	Windows WDM-KS	0	8
31	Input (Steam Streaming Speakers Wave)	Windows WDM-KS	8	0
32	Speakers (Steam Streaming Speakers Wave)	Windows WDM-KS	0	8
33	Input ()	Windows WDM-KS	8	0
34	Speakers ()	Windows WDM-KS	0	8
35	CABLE-A Output (VB-Audio CABLE-A)	Windows WDM-KS	8	0
36	Speakers (VB-Audio CABLE-A)	Windows WDM-KS	0	8
37	Speakers (DualSense Wireless Controller)	Windows WDM-KS	0	4
38	Headset Microphone (DualSense Wireless	Windows WDM-KS	2	0

	Controller)			
39	Microphone (Razer Seiren Mini)	Windows WDM-KS	1	0


Output of PortAudio's "query\_devices()" to display the system's available input and output devices.

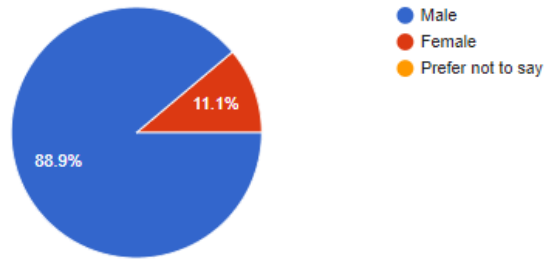
## Questionnaire on system evaluation assessment:



### Gender

9 responses

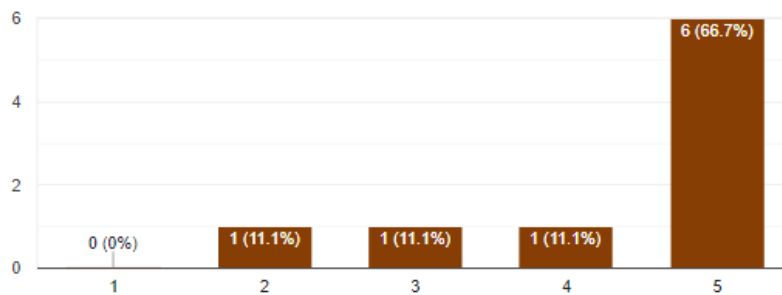
 Copy



### How would you describe your hearing abilities?

9 responses

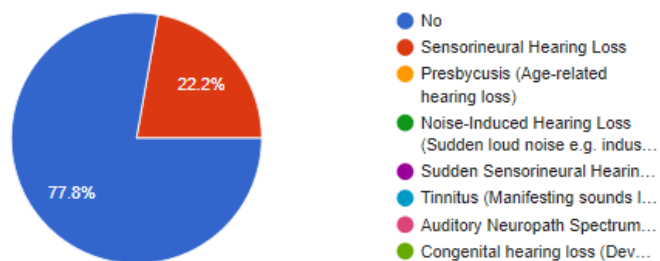
 Copy



### Do you have any form of hearing impairment? Choose one of the sections below.

9 responses

 Copy

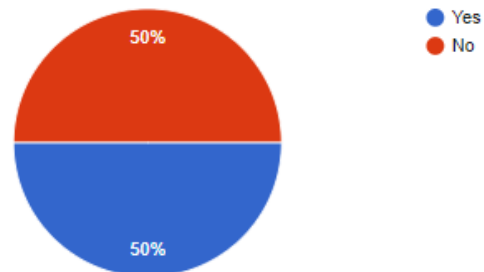


This section is for users with hearing impairment

Did you ever use hearing aids or any other hearing assistive devices?

 Copy

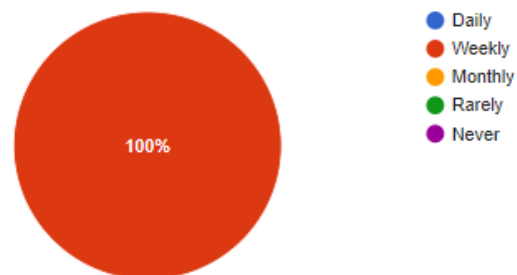
2 responses




If you use hearing aids or any other hearing assistive devices, how often do you do so?

 Copy

1 response



How long have you been experiencing hearing impairment?

 Copy

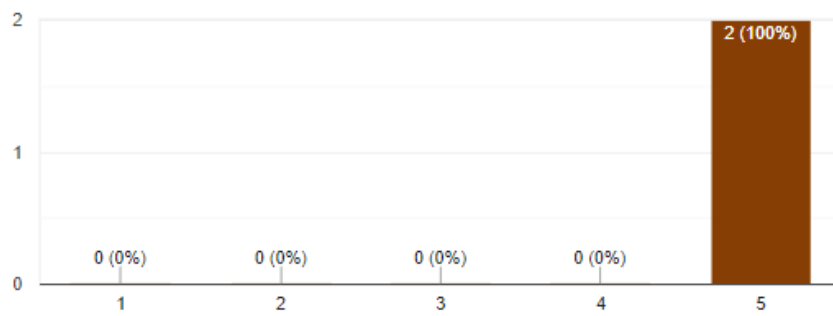
2 responses



How important is it for you to enhance your hearing capabilities?

[Copy](#)

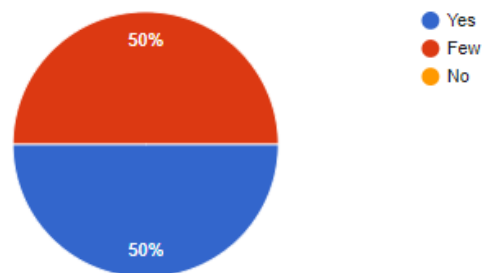
2 responses



Were you able to make out different sounds, voices or extra tones that you normally weren't able to hear?

[Copy](#)

2 responses

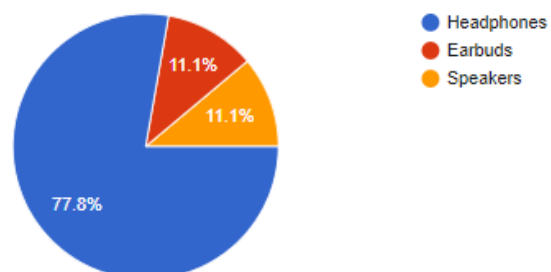


### Experience with Audio Equipment

What type of audio devices do you use most frequently?

[Copy](#)

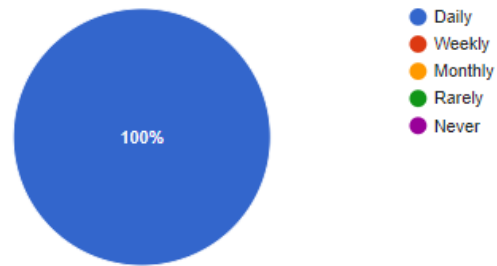
9 responses



How often do you use audio streaming services?

 Copy

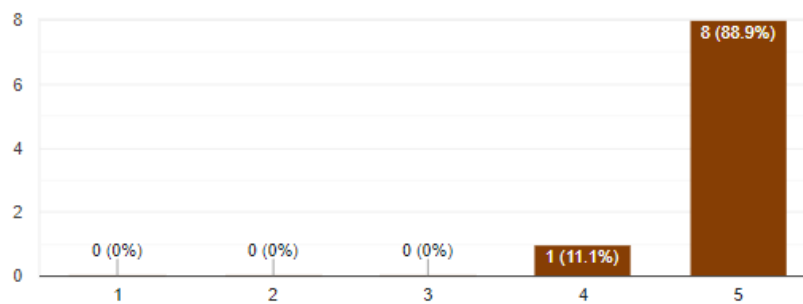
9 responses



How important is audio quality to you?

 Copy

9 responses

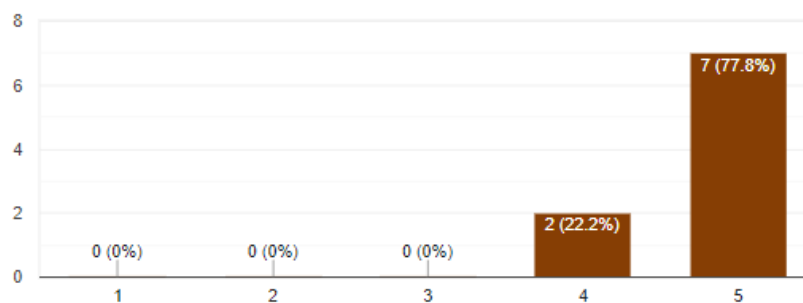


## Software

How important do you think the software as a concept is for providing more accessibility in the field of gaming for people with hearing impairments?


 Copy

9 responses

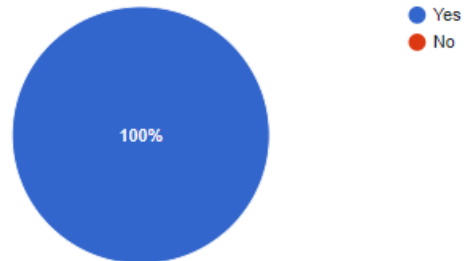




Were you able to clearly differentiate sound coming from the left and right channels for the stereo sound the software produces (left channel=left ear/speaker, right channel=right ear/speaker).

 Copy

9 responses

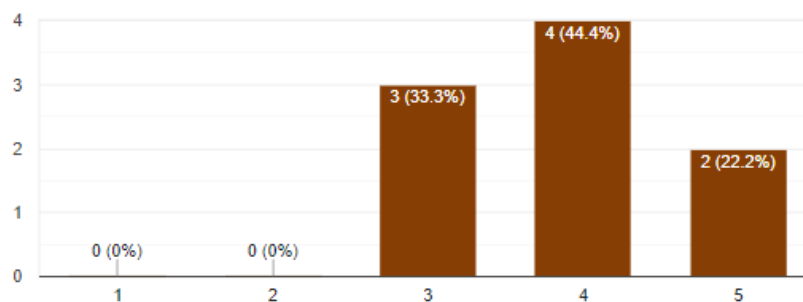


To what degree were you able to make out new sounds that you previously could not when listening to instrumental music?


 Copy

Music tested: Lost Woods - The Legend of Zelda: Ocarina Of Time

9 responses

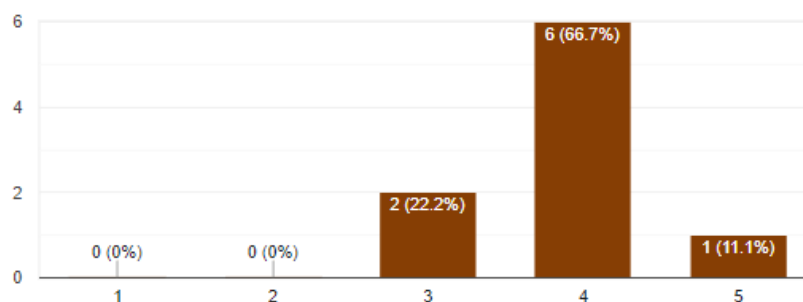


To what degree were you able to make out new sounds that you previously could not when listening to rock music?

 Copy


Music tested: Queen – Bohemian Rhapsody

9 responses

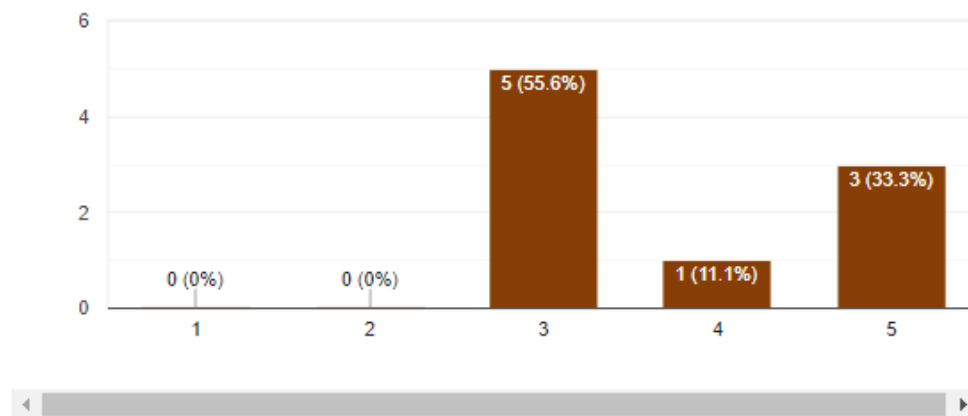





To what degree were you able to make out new sounds that you previously could not when listening to synth-pop music?  
Music tested: The Weeknd - Save Your Tears

 Copy

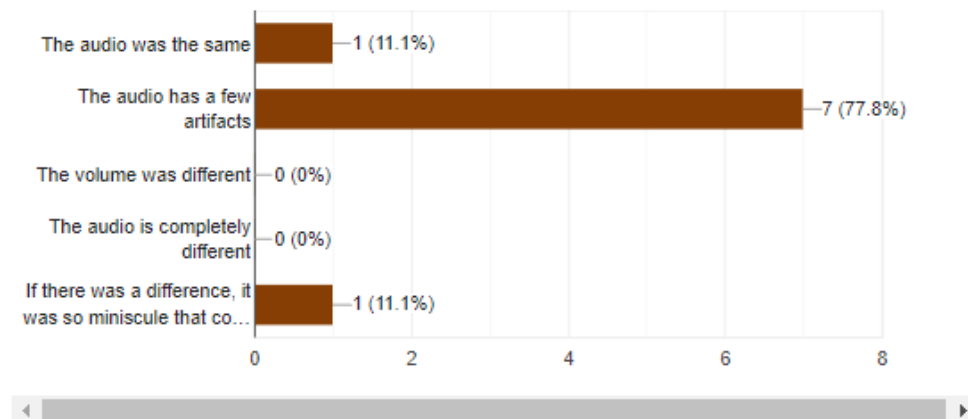
9 responses



When the software played the live audio without any adjustments to the original audio, did you notice any difference in the audio, like audio artifacts or changes in the volume? Answer if you are sure what the original audio is supposed to sound like.

 Copy

9 responses



If you found the sound was distorted/unpleasant/distorted/had artifacts, etc., describe those sounds

2 responses

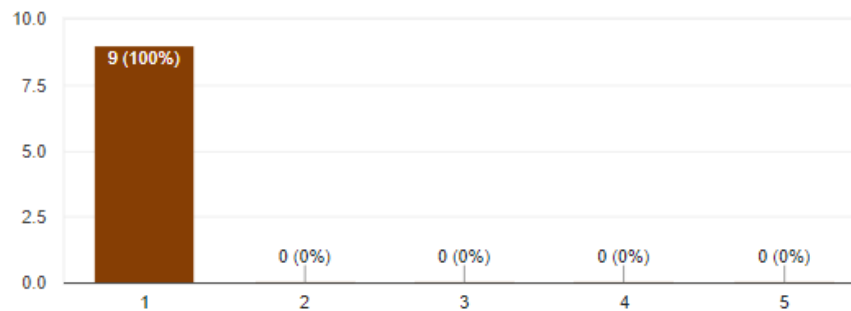
Very subtle static

a few artifacts on the first and third song

When the max automated dB boost is 3 dB, how significant do you consider the drop in quality

[Copy](#)

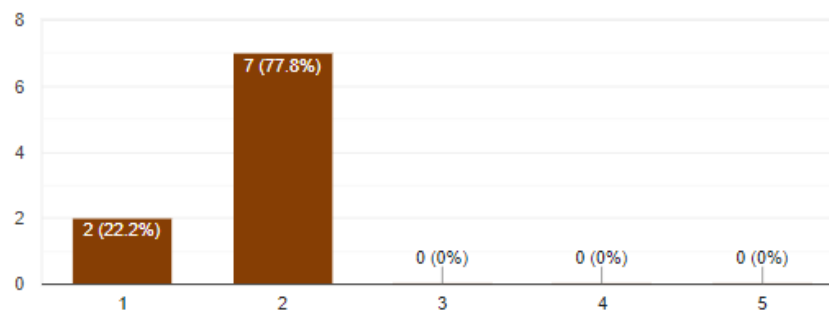
9 responses



When the max automated dB boost is 6 dB, how significant do you consider the drop in quality

[Copy](#)

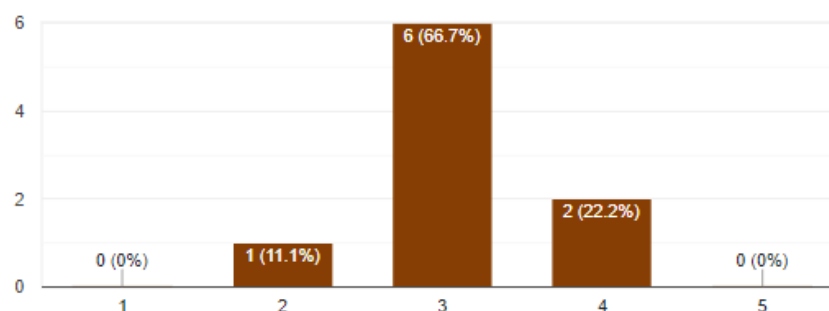
9 responses



When the max automated dB boost is 12 dB, how significant do you consider the drop in quality

[Copy](#)

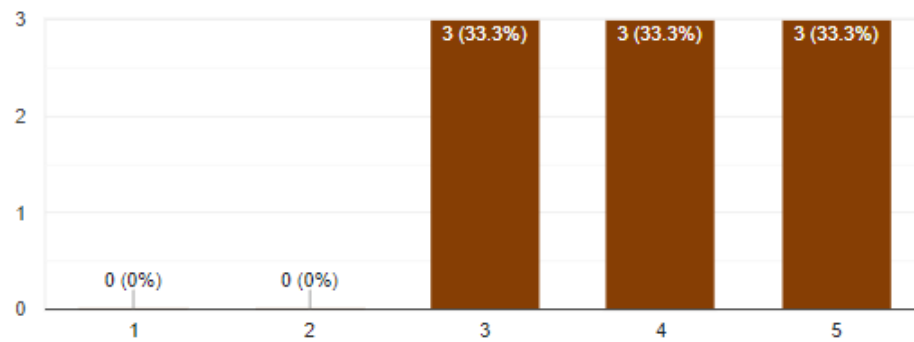
9 responses



How clear was the audio you listened to?

 Copy

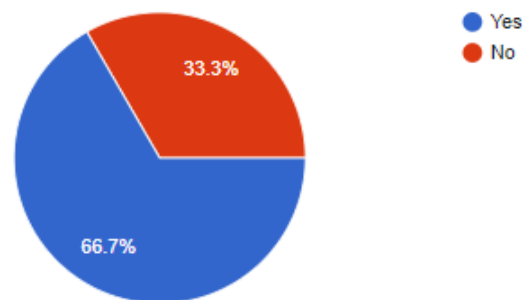
9 responses



Did you notice any improvements in audio clarity?

 Copy

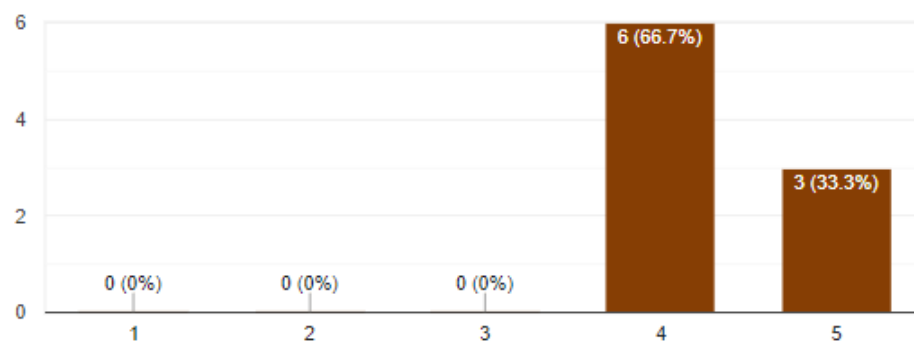
9 responses



How well could you hear the bass and treble?

 Copy

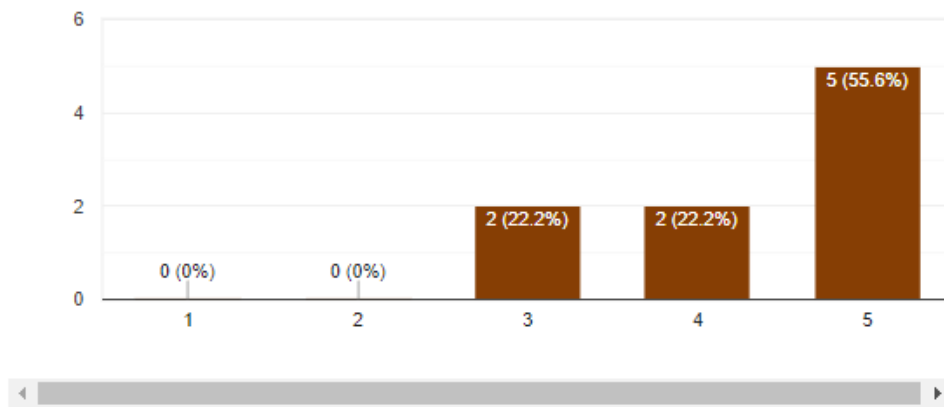
9 responses




How well could you hear the vocals?

 Copy

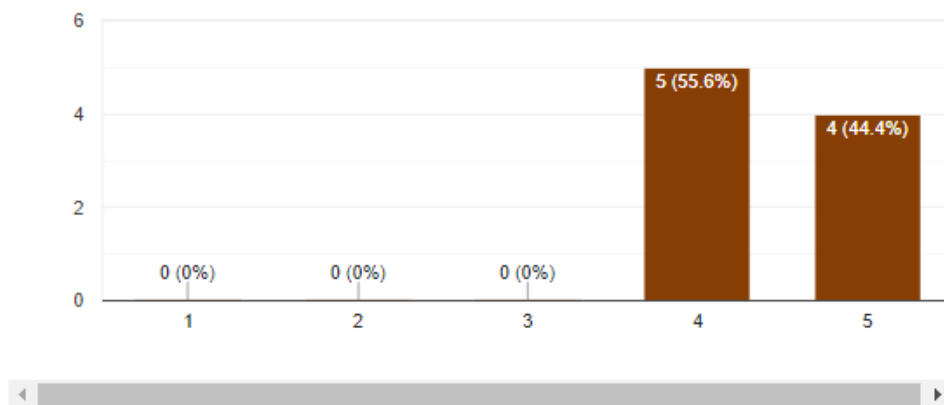
9 responses



How well could you hear background instruments or sounds?

 Copy

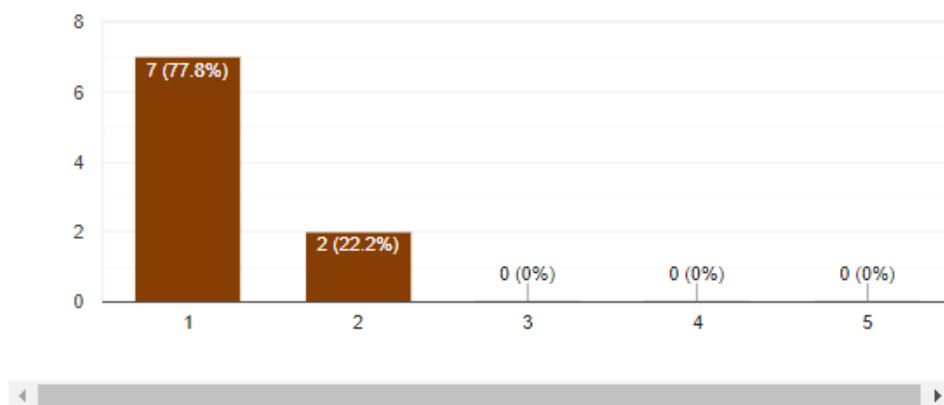
9 responses



Was there any noticeable delay in the audio stream?

 Copy

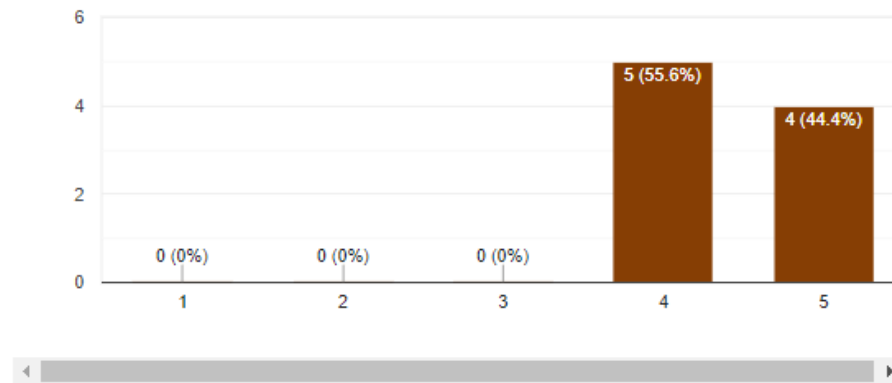
9 responses




"If you believe that the sound quality is the same or worse, how much are you willing to sacrifice in order to be able to listen to sounds that you wouldn't otherwise be able to hear in inaudible frequencies?"

 Copy

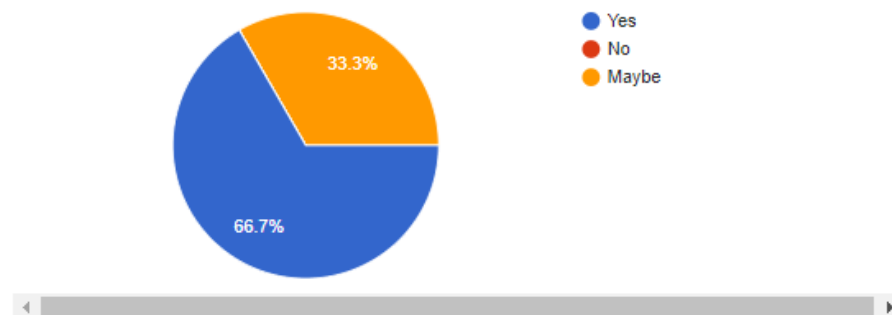
9 responses



Would you like to see audiograms integrated in audio software even if it is for a small benefit in audio perception?

 Copy

9 responses



Do you have any suggestions for improving the audio processing specifically for hearing-impaired users?

1 response

Not sure

Any additional comments or feedback regarding the altered audio stream for hearing-impaired users?

1 response

It helps a lot for people with hearing impairment