



UNIVERSITY OF NAIROBI

DEPARTMENT OF COMPUTING AND INFORMATICS

MEDTALK:

AN END-TO-END AUDIO ENCRYPTED TELECONSULTATION SYSTEM

BY

ABUOR ISABELLA MERCY

**THIS PROJECT REPORT HAS BEEN SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS OF THE DEGREE OF BACHELOR OF SCIENCE IN
COMPUTERSCIENCE OF THE UNIVERSITY OF NAIROBI.**

AUGUST 2023

DECLARATION

I hereby declare that the project report entitled “Medtalk: an end-to-end audio encrypted teleconsultation system” written and submitted to the Department of Computing, Faculty of Science and technology, University of Nairobi, for the award of a Degree of Bachelor of Science is an outcome of my own efforts and a record of an original work under the guidance of Prof. Elisha Toyne Opiyo, it contains no material which has been accepted for the award of any other degree in this or other University. I hereby certify that the work presented in this report is my own and that work performed by others is appropriately cited.

Signature: _____

Date: _____

Student: Abuor Isabella Mercy

Registration number: P15/136964/2019

This project report has been submitted in partial fulfillment of the requirements of the Degree of Bachelor of Science in Computer Science of the University of Nairobi with approval as University Supervisor.

Signature: _____

Date: _____

Supervisor: Prof. Elisha T. O. Opiyo

Department of Computing and informatics

University of Nairobi

ACKNOWLEDGEMENT

This project is a product of a long-lasting process, and I would like to thank everyone who has been part of this process. Firstly, I would like to thank God, the Almighty for His showers and blessings throughout the development and the completion of the project. I would like to express my special thanks of gratitude to my supervisor Professor Elisha Opiyo who guided me through the whole process of creating this project and for giving me an opportunity to work under his guidance. I would also like to thank the various healthcare professionals who took time out of their work to help me understand healthcare communications. My deepest appreciation goes to my beloved family for their love, prayers, and sacrifices, for supporting me financially, and for always being there, providing constant encouragement to be able to achieve the making and completion of this project. At times it was tough, and I felt like giving up but I would look back and draw strength from each of you. Thank you.

DEDICATION

I dedicate this to my lovely and supportive parents Brigitte Ang'anya and Philip Abuor.

ABSTRACT

In the era of digital transformation, secure online communication has become paramount in the healthcare industry due to the emergence of telemedical solutions. Audio communication plays a vital role in facilitating effective collaboration among healthcare professionals and improving communication between patients. However, ensuring the confidentiality, integrity, and authenticity of audio data remains a significant challenge. The existing approaches to online communication systems are focused on partial encryption or relied on third-party tools and are not developed with the healthcare sector in mind. This paper presents an end-to-end audio encrypted teleconsultation system designed specifically for the healthcare sector to address these security concerns. The proposed system main achievement is development of a streamlined system that seamlessly integrates secure audio encryption into the teleconsultation workflow. Additionally, authentication mechanisms are implemented to verify the identity of the sender, preventing any potential impersonation attacks. The system provides a robust solution to protect sensitive patient information and enable secure collaboration among healthcare professionals. Implementation of this system can lead to enhanced privacy, data integrity, and trust in audio communications, ultimately improving the overall quality of healthcare services.

Keywords: Communication, End-to-End Encryption, Audio Encryption, Teleconsultation.

TABLE OF CONTENTS

1	CHAPTER ONE: INTRODUCTION.....	11
1.1	GENERAL BACKGROUND	11
1.2	PROBLEM STATEMENT	12
1.3	OBJECTIVES	12
1.3.1	Project Objectives	12
1.3.2	Research Objectives.....	12
1.4	PROBLEM JUSTIFICATION.....	13
1.5	SCOPE	13
1.6	ASSUMPTIONS	13
2	CHAPTER TWO: LITERATURE REVIEW	14
2.1	CHAPTER OVERVIEW	14
2.2	CRYPTOGRAPHY.....	14
2.3	TRADITIONAL CRYPTOGRAPHY.....	14
2.3.1	Caesar Shift Cipher	14
2.3.2	Enigma Machine	15
2.4	MODERN CRYPTOGRAPHY	16
2.4.1	Hash Functions.....	16
2.4.2	Asymmetric Encryption Algorithms.....	16
2.4.3	Symmetric Encryption Algorithms	18
2.5	REVIEW OF RELATED SYSTEMS	23
2.5.1	Encryption and Decryption of Audio Signal Based on RSA Algorithm	23
2.5.2	RF Secure Coded Communication System.....	24
2.5.3	WhatsApp	25
2.5.4	Google Meet.....	25

2.6	THE PROPOSED SOLUTION.....	25
3	CHAPTER THREE: METHODOLOGY	27
3.1	CHAPTER OVERVIEW	27
3.2	RESEARCH METHODOLOGY.....	27
3.3	SYSTEM DEVELOPMENT METHODOLOGY	27
3.3.1	Requirements Planning.....	28
3.3.2	User Design.....	28
3.3.3	Construction.....	28
3.3.4	Cutover.....	28
3.4	RESOURCES.....	29
3.5	GANTT CHART.....	30
4	CHAPTER FOUR: SYSTEM ANALYSIS AND DESIGN	31
4.1	CHAPTER OVERVIEW	31
4.2	SYSTEM ANALYSIS	31
4.2.1	Requirements Gathering	31
4.2.2	Requirements Specifications.....	32
4.2.3	Feasibility Study	34
4.3	SYSTEM DESIGN	35
4.3.1	Components of the Designed System	35
4.3.2	High Level Design	35
4.3.3	Low Level Design.....	36
4.3.4	Flowcharts.....	37
4.3.5	User-Interface Design	39
5	CHAPTER FIVE: SYSTEM IMPLEMENTATION	41
5.1	Chapter Overview	41

5.2	Programming Tools, Techniques and Technologies	41
5.3	Implementation Results.....	42
5.3.1	Audio Chat Development	42
5.3.2	Encryption Implementation	42
5.4	Software Testing	43
5.4.1	Unit Testing	43
5.4.2	Integration Testing	43
5.4.3	System Testing.....	44
5.4.4	Test Cases	44
6	CHAPTER SIX: CONCLUSIONS AND RECOMMENDATIONS.....	47
6.1	Chapter Overview	47
6.2	Achievements	47
6.3	Challenges and limitations	48
6.4	Conclusions	48
6.5	Recommendation for further work	49
	REFERENCES.....	50
	APPENDICES	52
	APPENDIX A: User Manual	52
	APPENDIX B: Sample Code.....	57

LIST OF FIGURES

Figure 1: Showing Caesar shift cipher example.	15
Figure 2: Showing Asymmetric Cryptography method process	17
Figure 3: Showing Symmetric Cryptography method process	18
Figure 4: Showing the summary of the 3DES Encryption and Decryption Process	19
Figure 5: Summary of AES Encryption and Decryption Process.....	23
Figure 6: Block Diagram of encryption and decryption of audio signal based on RSA algorithm.....	24
Figure 7: Conceptual Model of the proposed system.	26
Figure 8: Showing Summary of the RAD methodology Process	27
Figure 9: Gantt Chart	30
Figure 10: Evidence of the system analysis discussion.	32
Figure 11: Block Diagram of the designed E2E audio encryption System prototype.	35
Figure 12: Sequence diagram of the E2E audio encryption System prototype.....	36
Figure 13: Showing the Transmission process of the audio message	37
Figure 14: Showing the Receiver Side flowchart	38
Figure 15: UI Design for the Dashboard Page.....	39
Figure 16: UI Design for the Meetings Page.	39
Figure 17: UI design of the GUI Call Page.	40
Figure 18: WebRTC functionality	42
Figure 19: Final Login page	52
Figure 20: Final Signup page	52
Figure 21: Google Login Popup	53
Figure 22: Picture of the Dashboard	53
Figure 23: Picture of the create meeting form.	54
Figure 24: Picture of My Meetings page.....	54
Figure 25: Picture of the Edit Functionality	55
Figure 26: Picture of the call homepage.....	56
Figure 27: Picture of the Call page room.	56
Figure 28: Sample Code for the WebRTC.....	57
Figure 29: Sample of the Server-side code.....	57
Figure 30: Sample code of the AES encryption.....	58

LIST OF ABBREVIATIONS

AES Advanced Encryption Standard

DES Data Encryption Standard

3DES Triple Data Encryption Algorithm

E2EE End to End Encryption

N/A Non-Applicable

RAD Rapid Application Development

RF Radio Frequency

RSA Rivest-Shamir-Adleman encryption

WebRTC Web Real time Communications

1 CHAPTER ONE: INTRODUCTION

1.1 GENERAL BACKGROUND

Keeping information secure has been of paramount importance since the earliest forms of communication, such as handwritten letters and telegraph messages, especially in sensitive fields such as government, military, and healthcare industries.

End-to-end encryption (E2EE) is a method of secure communication that allows two or more users to exchange information in a way that ensures only the intended recipients can access and read the information. This is achieved by encrypting the data at the sender's device and decrypting it at the receiver's device, with no intermediary able to access or intercept the information. (lutkevich, 2021)

E2EE has become increasingly important in the healthcare industry today as it is rapidly embracing digital technologies to enhance patient care, improve operational efficiency, and enable seamless communication among healthcare professionals. With the increasing reliance on audio communication for consultations, diagnostics, and collaborative decision-making, ensuring the security of audio data has become a critical concern. Protecting patient confidentiality and preventing unauthorized access to sensitive medical information are paramount in maintaining the trust and integrity of healthcare systems.

End-to-end encryption is typically implemented using a combination of cryptographic keys and algorithms, which generate unique codes that are used to encrypt and decrypt messages. Encryption uses sophisticated algorithms to ensure that the encrypted data cannot be deciphered without the proper encryption key. Advanced encryption algorithms, such as Advanced Encryption Standard (AES), are commonly used to secure files.

Implementing an end-to-end audio encrypted communication system in healthcare can have significant benefits. It not only safeguards patient privacy and protects sensitive medical information but also enhances the efficiency and effectiveness of healthcare services. Secure audio communication enables healthcare professionals to collaborate seamlessly, share expertise, and make informed decisions without concerns about unauthorized access or privacy breaches.

1.2 PROBLEM STATEMENT

In Kenya, with the increase of telemedical solutions, the healthcare industry faces a critical challenge in securing audio communications to protect patient confidentiality and prevent unauthorized access to sensitive medical information. This leaves the audio communication channels open to potential security breaches, which could lead to financial losses, breaches of privacy, and even national security threats. Beyond financial losses, healthcare organizations that suffer a data breach may have to pay for credit monitoring for consumers whose information has been exposed, therefore leading to consumer trust degradation and damage to reputation or brand. Additionally, many current solutions are not specifically designed to meet the unique needs of the healthcare industry.

1.3 OBJECTIVES

The main objective of the project is to build an end-to-end audio encrypted teleconsultation system through a wireless system.

1.3.1 Project Objectives

1. To prepare system requirements analysis specification for E2EE audio communication systems.
2. To design a communication system that securely sends and receives audio data wirelessly.
3. To ensure the audio data is sent and received through radio frequency.
4. To incorporate a highly secure algorithm for audio transmission and reception at low costs.
5. To test and evaluate the proposed system.
6. To write documentation on the proposed system.

1.3.2 Research Objectives

1. To investigate the challenges that practitioners face using teleconsultation or videoconferencing services for online consultation.
2. To identify the different types of common modern cryptographic algorithms that are appropriate for audio communication.
3. To review the literature on the performance of the common modern cryptographic algorithms mentioned in 1, in terms of speed, memory usage, and energy consumption.

1.4 PROBLEM JUSTIFICATION

The need and development of an end-to-end audio encryption teleconsultation system in Kenya that can provide robust and comprehensive audio encryption would be advantageous to the healthcare industry in ensuring that audio data is secure and cannot be intercepted or tampered with by unauthorized persons. The system would help in the outside world in that it would reduce the risk of data infringement by increasing the difficult levels of hacking to ensure maximum security of information which would provide peace of mind for businesses and individuals who need to communicate sensitive information through audio channels.

Additionally, such a system would be affordable, scalable, and that is specifically designed for the Kenyan healthcare industry to meet the unique needs. The development of an end-to-end audio encryption teleconsultation system in Kenya would not only improve the security of audio data, but also ensure to democratize healthcare by affording every Kenyan the privilege of professional medical guidance, regardless of their physical location.

1.5 SCOPE

The boundaries of the study can be said to be the fact that this system mainly focuses on the creation of an end-to-end audio encrypted teleconsultation system specially designed for the healthcare industry and will be determined successful if the system involves the sender side and the receiver side. The system also incorporates AES mode of encryption during the transmission. On the receiving end the system is integrated with AES mode of decryption and one can only hear the message if they are authorized to, then the transmitted message is played.

1.6 ASSUMPTIONS

The following assumptions are held as true in the development of this system:

1. The communication channel is free to noise.
2. It is assumed that there is a reliable and reasonably stable internet connectivity infrastructure available.
3. Healthcare professionals will readily provide information needed for development of the system.

2 CHAPTER TWO: LITERATURE REVIEW

2.1 CHAPTER OVERVIEW

This chapter discusses the basis of the study, the traditional and modern methods of cryptography an overview of previous similar projects related to secure audio communication systems and the proposed solution.

2.2 CRYPTOGRAPHY

Cryptography is a method of protecting information and communications through the use of codes, so that only those for whom the information is intended can read and process it.(Richards, 2021) The term is derived from the Greek word kryptos, which means hidden. It is closely associated to encryption, which is the act of scrambling ordinary text into what's known as ciphertext and then back again upon arrival which is known as plaintext.

2.3 TRADITIONAL CRYPTOGRAPHY

The art of cryptography is born along with the art of writing. As civilizations evolved, human beings got organized into tribes, groups, and kingdoms. This led to the emergence of ideas such as power, battles, supremacy, and politics. These ideas further fueled the natural need of people to communicate secretly with selective recipients which in turn ensured the continuous evolution of cryptography as well.

As time progressed more sophisticated methods of securing information were invented. These methods include:

2.3.1 Caesar Shift Cipher

This method relies on shifting the letters of a message by an agreed number, for instance three, the recipient of this message would then shift the letters back by the same number and obtain the original message. It was used to protect messages of military significance.

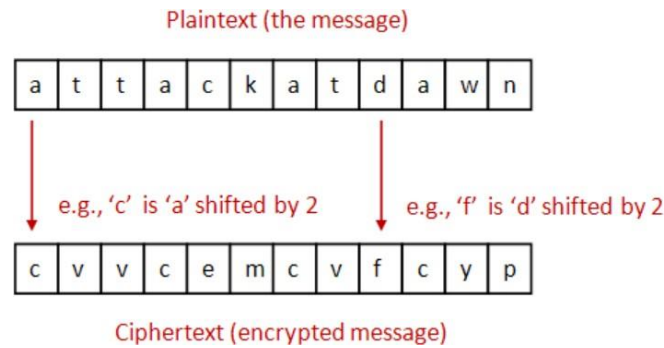


Figure 1: Showing Caesar shift cipher example.

An example is, the letter A becomes the letter C. Therefore, the word “security” will be written as “ugewtkva”. The problem with this cipher is that the encryption method is fixed; there is no key. Thus, anyone learning how Caesar encrypted his messages would be able to decrypt effortlessly. A version of this encryption method, namely, ROT-13 where instead of shifting 3 letters, 13 letters are shifted is still in use to this date. However, it does not provide any cryptographic security; it is used merely to ensure that the text is unintelligible unless the reader of a message consciously chooses to decrypt it. An application is submitting movie spoilers. (Ravi, 2017)

2.3.2 Enigma Machine

An Enigma machine is made up of several parts including a keyboard, a lamp board, rotors, and internal electronic circuitry. Some machines, such as the ones used by the military, have additional features such as a plug board. Each time a letter was mapped to another, the entire encoding scheme changed. After each button pressed, the rotors move and repressing that same button routes current along a different path to a different revealed letter.

So, for the first press of a key, one encoding is generated, and when the second key is pressed, another encoding is generated, and so on. This greatly increases the number of possible encoding configurations. Each time a key is pressed on an Enigma machine, the rotors turn, and the code changes. This means that the message “AA” could be encoded as something like “TU” even though the same key was pressed twice.

It was mainly used by the Germans during the World War II to transmit coded messages. It deemed unbreakable until Alan Turing and other researchers exploited a few weaknesses in the

implementation of the Enigma code and gained access to German codebooks which allowed them to design a machine called a Bombe machine, which helped to crack the most challenging versions of Enigma. With the fall of the enigma machine, advancement of wireless communication and the internet, modern cryptography methods were invented. (Karleigh, W and Dean, 2016)

2.4 MODERN CRYPTOGRAPHY

From the previous section, cryptography was historically more of an art than a science. Schemes were designed in an adhoc manner and evaluated based on their perceived complexity or cleverness. Over the past several decades, cryptography has developed into more of a science. Schemes are now developed and analyzed in a more systematic manner, with the ultimate goal being to give rigorous proof that a given construction is secure.

Modern cryptography is defined as the study of mathematical techniques for securing digital information, systems, and distributed computations against adversarial attacks. (Katz and Lindell, 2015)

There are three types of modern cryptography methods, these include:

2.4.1 Hash Functions

This algorithm makes no use of any keys. A hash function is a mathematical function or algorithm that simply takes a variable number of characters (called a “message”) and converts it into a string with a fixed number of characters (called a hash value or simply, a hash). The act of hashing is, therefore, running an input into a formula that converts it into an output message of fixed length. No matter how many characters long the input is, the output will always be the same in terms of the number of hexadecimal (letters and numbers) characters.

2.4.2 Asymmetric Encryption Algorithms

Asymmetric encryption, also known as public-key encryption, is a type of encryption that uses two different keys, a public key and a private key. The public key can be given to anyone, trusted or not, while the private key must be kept secret.

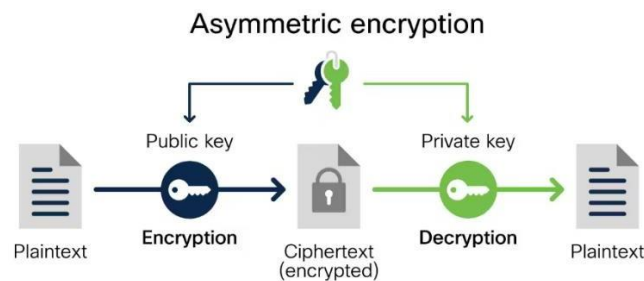


Figure 2: Showing Asymmetric Cryptography method process.

An example of asymmetric encryption algorithm in use today:

2.4.2.1 Rivest-Shamir-Adleman (RSA) Algorithm

The Rivest-Shamir-Adleman (RSA) algorithm is a widely used public-key encryption algorithm developed by Ron Rivest, Adi Shamir, and Leonard Adleman in 1977. RSA is based on the mathematical problem of factoring large numbers, which is believed to be computationally infeasible for sufficiently large numbers.

RSA involves the use of two keys: a public key for encryption and a private key for decryption. The public key can be shared freely, while the private key must be kept secret.

The encryption process works as follows:

The plaintext message is first represented as a number, typically using a standardized encoding such as ASCII or Unicode. The public key, which consists of a modulus n and an exponent e , is used to encrypt the message. The ciphertext is calculated as $C = M^e \pmod{n}$, where M is the plaintext message and e is the public exponent.

The resulting ciphertext can be sent over an insecure channel without fear of interception, as it can only be decrypted with the corresponding private key.

The decryption process works as follows:

The ciphertext is decrypted using the private key, which consists of the same modulus n and a different exponent d . The private key is calculated in such a way that d is the modular inverse of e (i.e., $e \cdot d = 1 \pmod{\phi(n)}$, where $\phi(n)$ is the Euler totient function of n).

The plaintext message is calculated as $M = C^d \pmod n$.

The strength of RSA comes from the difficulty of factoring large numbers. To break RSA encryption, an attacker would need to factor the modulus n into its two prime factors, p and q , which is believed to be infeasible for sufficiently large primes. As a result, the security of RSA depends on the size of the modulus n , which is typically chosen to be several hundred or thousand bits long. (Puneet, 2020)

2.4.3 Symmetric Encryption Algorithms

Symmetric encryption is a type of encryption that uses the same key for both encryption and decryption of data. This means that the sender and the recipient of the data share the same secret key that is used to encrypt and decrypt the data.

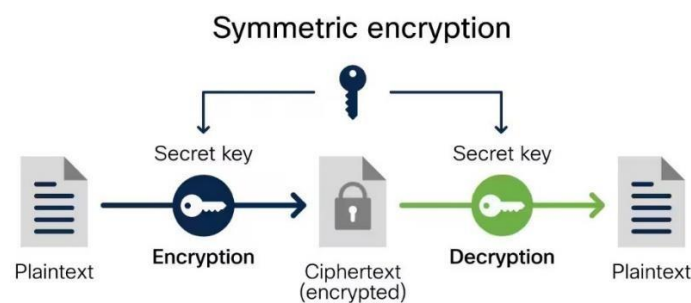


Figure 3: Showing Symmetric Cryptography method process.

There are several symmetric encryption algorithms in use today, including:

2.4.3.1 Data Encryption Standard (DES)

The Data Encryption Standard (DES) is a symmetric encryption algorithm that operates on 64-bit blocks of data and uses a 56-bit key to encrypt and decrypt the data. The key is used to generate 16 subkeys, which are used in a series of substitution and permutation operations known as the Feistel network. In each round of the Feistel network, the data is divided into two 32-bit halves, and one half is subjected to the substitution and permutation operations while the other half remains unchanged. The two halves are then swapped, and the process is repeated for a total of 16 rounds. (Javatpoint, no date)

The strength of DES comes from the complexity of the Feistel network and the number of rounds it uses. However, the small key size of DES means that it is vulnerable to brute-force attacks, where an attacker tries all possible combinations of keys until the correct one is found. This vulnerability led to the development of the Triple DES (3DES) algorithm, which applies the DES algorithm three times with three different keys, increasing the effective key length to 168 bits.

2.4.3.2 Triple DES (3DES)

Triple DES (3DES) is a variant of the Data Encryption Standard (DES) encryption algorithm that provides a higher level of security by applying the DES algorithm three times with three different keys. 3DES is also known as TDEA (Triple Data Encryption Algorithm).

In 3DES, the data is first encrypted with one key, decrypted with a second key, and then encrypted again with a third key. This provides a much larger effective key length of 168 bits, making it much more secure than the original 56-bit DES. The three keys used in 3DES can either be all different or two of the keys can be the same. (Javatpoint, no date)

The following figure shows the encryption and decryption using 3DES:

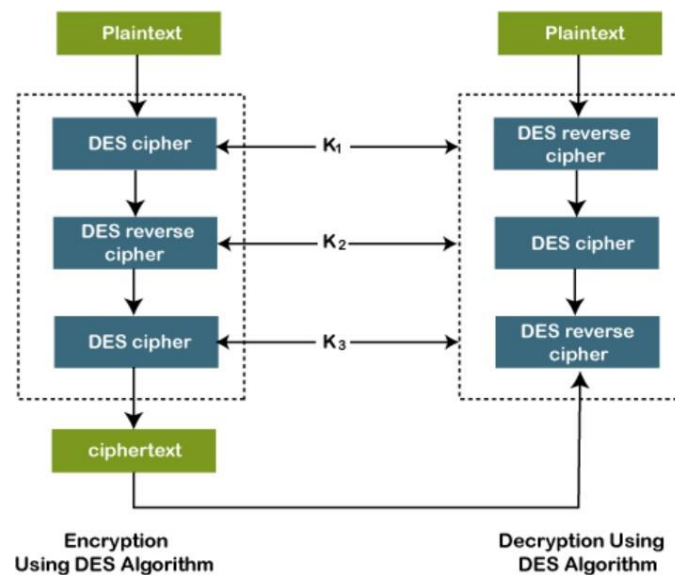


Figure 4: Showing the summary of the 3DES Encryption and Decryption Process.

3DES supports two different modes of operation: ECB (Electronic Codebook) and CBC (Cipher Block Chaining). ECB mode works by dividing the data into 64-bit blocks and encrypting each block independently using the same key. This can result in security weaknesses, as identical plaintext blocks will produce identical ciphertext blocks. CBC mode, on the other hand, works by XORing each plaintext block with the previous ciphertext block before encryption. This makes it more secure than ECB mode, as it ensures that identical plaintext blocks produce different ciphertext blocks.

One of the benefits of 3DES is that it can be used with existing DES hardware and software implementations, making it a popular choice for organizations that need to upgrade their security without replacing their entire encryption infrastructure. However, its main disadvantage is its relatively slow performance compared to more modern encryption algorithms such as the Advanced Encryption Standard (AES).

2.4.3.3 Blowfish

Blowfish is a symmetric-key block cipher that was designed by Bruce Schneier in 1993. It is widely used for encryption in various applications, including SSL/TLS protocols, VPNs, and file encryption. Blowfish has a variable key length, which can range from 32 to 448 bits. It works by dividing the plaintext into 64-bit blocks and encrypting each block independently using a series of rounds. The number of rounds depends on the key length and can range from 16 to 72 rounds.

The encryption process works as follows:

The key is first expanded into a set of subkeys using a key schedule algorithm. The subkeys are used in each round of encryption to generate a different encryption key for each round. The plaintext message is then divided into 64-bit blocks and XORed with an initialization vector (IV) before encryption. Each block is then encrypted using a series of rounds that involve several operations, including a substitution step, a permutation step, and a key mixing step. After all blocks have been encrypted, the ciphertext is produced.

The decryption process works in reverse:

The key schedule algorithm is used to generate the same subkeys that were used for encryption. The ciphertext message is divided into 64-bit blocks and XORed with the IV before decryption. Each block is then decrypted using the same series of rounds that were used for encryption, but

with the subkeys used in reverse order. After all blocks have been decrypted, the plaintext message is produced.

Blowfish is known for its speed and efficiency, as it can encrypt and decrypt data quickly even with large key sizes. It is also relatively resistant to attacks such as differential cryptanalysis and linear cryptanalysis. However, it is vulnerable to certain types of attacks, such as brute-force attacks if the key size is too small or if the same key is used repeatedly.

To address these vulnerabilities, Blowfish has been largely replaced by more modern encryption algorithms such as the Advanced Encryption Standard (AES). However, Blowfish is still widely used in legacy systems and applications, and it remains an important algorithm in the history of cryptography.

2.4.3.4 Advanced Encryption Standard (AES)

The AES algorithm (also known as the Rijndael algorithm) is a symmetrical block cipher algorithm that takes plain text in blocks of 128 bits and converts them to ciphertext using keys of 128, 192, and 256 bits. Since the AES algorithm is considered secure, it is in the worldwide standard. (Vardhaman and Ivanov, 2023)

How does it Work?

For Example, let's take a 128- bit block

1. The plaintext is XORed (which means a key is added) and the Key is expounded.
2. Byte Substitution (Sub-Bytes)

Instead of having a long string of bits, AES likes to arrange them in grid (4x4 Grid for 128-bits)

b0	b4	b8	b12
b1	b5	b9	b13
b2	b6	b10	b14
b3	b7	b11	b15

3. Shift rows

Each of the four rows is shifted to the left. The first Row remains the same, the second is shifted by one position, the third rows is shifted by two positions and the fourth is shifted three positions. Hence the grid would look like this.

b0	b4	b8	b12
b5	b9	b13	b1
b10	b14	b2	b6
b15	3b	b7	b11

4. Mix columns.

Each column of fours bytes is now transformed using a matrix.

5. Add Round Key

6. XOR is now used on the 128 bits.

This is now repeated until the last round where It just outputs the ciphertexts. The number of rounds depends on if its 128-bit,192-bit or 256-bit key. For 128-bits its 10 rounds, For 192-bits its 12 rounds, For 256-bits its 14 rounds.

For the Decryption Part is just the inverse of the encryption:

1. Inverse add round key
2. Inverse Shift rows
3. Inverse byte substitution
4. Inverse add round key
5. Inverse mix columns
6. Inverse shift rows
7. Inverse byte substitution
8. Inverse add round Key (x 9 or x 11 or x 13, depending on whether the key is 128,192 or 256-bits (Simplelearn, 2020) (Aes Algorithm Steps - Google Search, n.d.)

This can be summarized with the flowchart below:

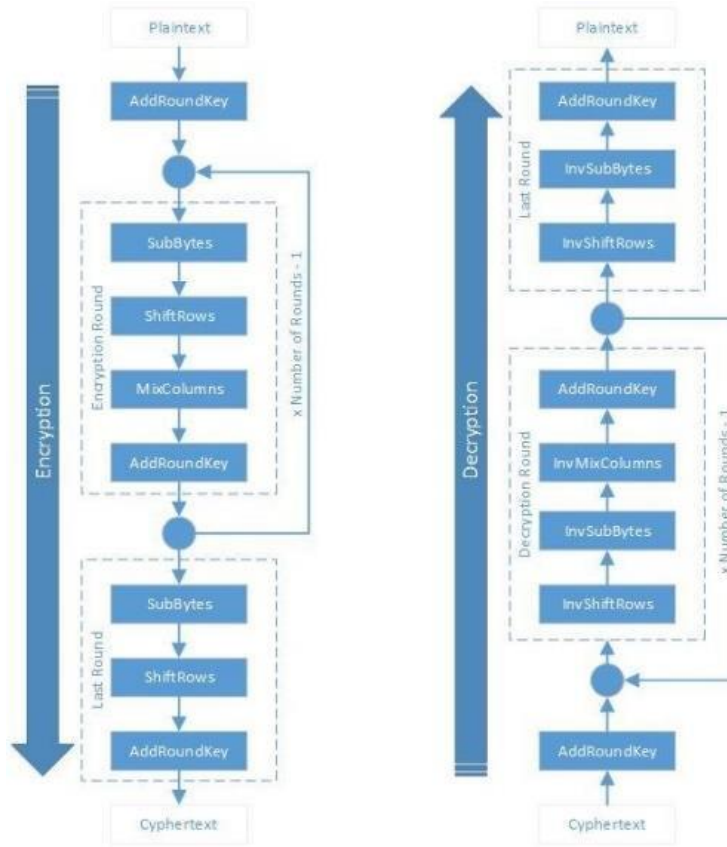


Figure 5: Summary of AES Encryption and Decryption Process.

2.5 REVIEW OF RELATED SYSTEMS

There are already existing systems that are similar to the E2E audio encrypted teleconsultation systems which are:

2.5.1 Encryption and Decryption of Audio Signal Based on RSA Algorithm

The system consisted of two stages: encryption and decryption of the audio signal using RSA algorithm. Public and private keys are generated previously and then the public key is used to encrypt the acquired speech or audio samples at the transmitter. The ciphered or encrypted audio samples are sent to the receiver sequentially through a communication channel who will decrypt each sample by employing the private key. For simplicity, it is assumed that the transmission or communication channel is ideal or free of noise. (Fahmy, 2018) The block diagram of the system is shown below.

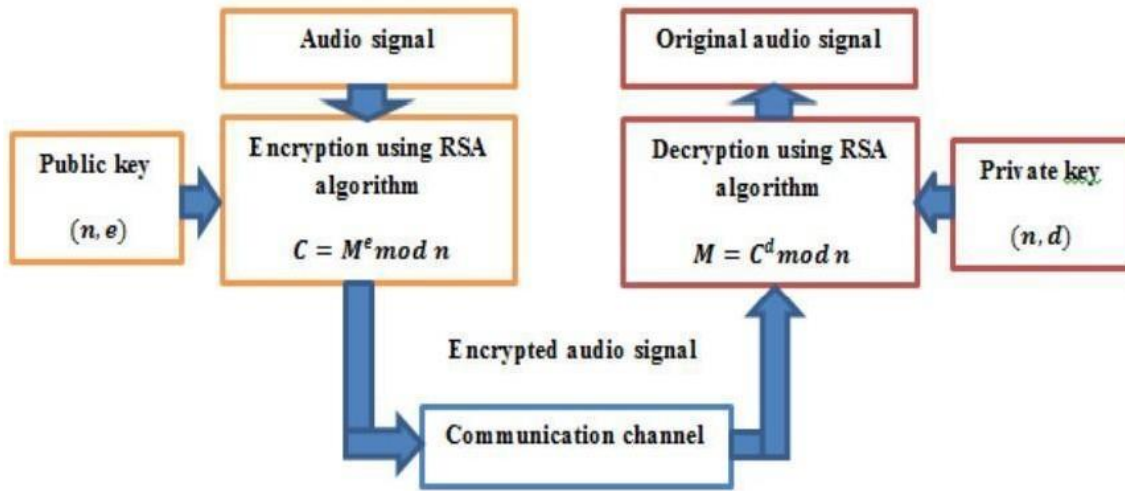


Figure 6: Block Diagram of encryption and decryption of audio signal based on RSA algorithm.

2.5.1.1 Disadvantages

The solution of the existing system above is not satisfactory. The method of encryption used in this system is asymmetric encryption which is slow. This is because the keys are longer, and the server needs to calculate two different keys for encryption and decryption. In addition, one cannot use asymmetric encryption where there is a huge quantity of data involved; otherwise, the servers get exhausted and become slow. Hence becomes a time-consuming process and limits the size of the data.

2.5.2 RF Secure Coded Communication System

This system was designed to ensure maximum level of secrecy when transmitting text data over RF from one device to another Securely using Advanced Encryption Standard (AES) algorithm. The User enters a message through the Graphical User Interface (GUI) from the PC through a computer keyboard and the Message will be encrypted using AES algorithm and transmitted using RF to the Receiver. On the receiver side, the message will be decrypted, and they will be alerted that there is a new message displayed on the LCD, then they will be prompted to put a password and if and only if the password is correct then the transmitted Message will be displayed on the screen.(Abuor, 2021)

The disadvantage for this system is that it only focuses on the alphanumeric characters which is limiting.

2.5.3 WhatsApp

A mobile, desktop and web instant messaging application that allows users to securely send text, voice, and video messages, as well as make voice and video calls, over the internet. It uses end to end encryption to ensure that messages and calls stay between the sender and the receiver. No one else can read or listen to them, not even WhatsApp. Beyond end-to-end encryption, additional layers of protection are added to all conversations, like two-factor authentication, disappearing messages and privacy control. (*WhatsApp*, no date)

The disadvantages of this system are that it is not professional, in that WhatsApp uses personal numbers which is not appropriate and can cause confusion in trying to differentiate work and personal calls. In addition, it cannot be easily integrated with other system like Electronic health records, calendar for automated reminders.

2.5.4 Google Meet

Google Meet is a video conferencing platform developed by Google. It allows users to conduct virtual meetings, conferences, and webinars with participants from around the world. Google Meet was designed to facilitate online collaboration, communication, and remote work, making it particularly useful for businesses, educational institutions, and individuals.

The disadvantage of this system is that it is not purpose-built. It does not have features necessary for use in the medical field and does not meet the demands of the health industry and its

2.6 THE PROPOSED SOLUTION

The proposed system fills in the existing gaps by incorporating symmetric encryption and decryption by using Advanced Encryption method. The Web based system enables the user to speak and send voice messages through a microphone which will then be processed and delivered to the receiver end wirelessly through web real time transmission (WebRTC). This system has a secret code attached to the transmission. The audio data will also then be encrypted using the Advanced Encryption standard (AES) method. At the receiving end, the receiver is sent a unique code in which they can only hear the decrypted voice message if authenticated.

Why AES? AES algorithm Standard was chosen due to the following advantages:

1. This robust security algorithm can be implemented in both hardware and software.
2. It is resilient against hacking attempts, due to its higher-length key sizes (128, 192, and 256 bits).
3. It is an open-source solution. Since AES is royalty-free, it remains highly accessible for both the private and public sectors.
4. AES is the most common security protocol used for a wide variety of applications today such as wireless communication, financial transactions, e-business, encrypted data storage among others. (Neha, 2020)

Below is the conceptual diagram of the proposed system:

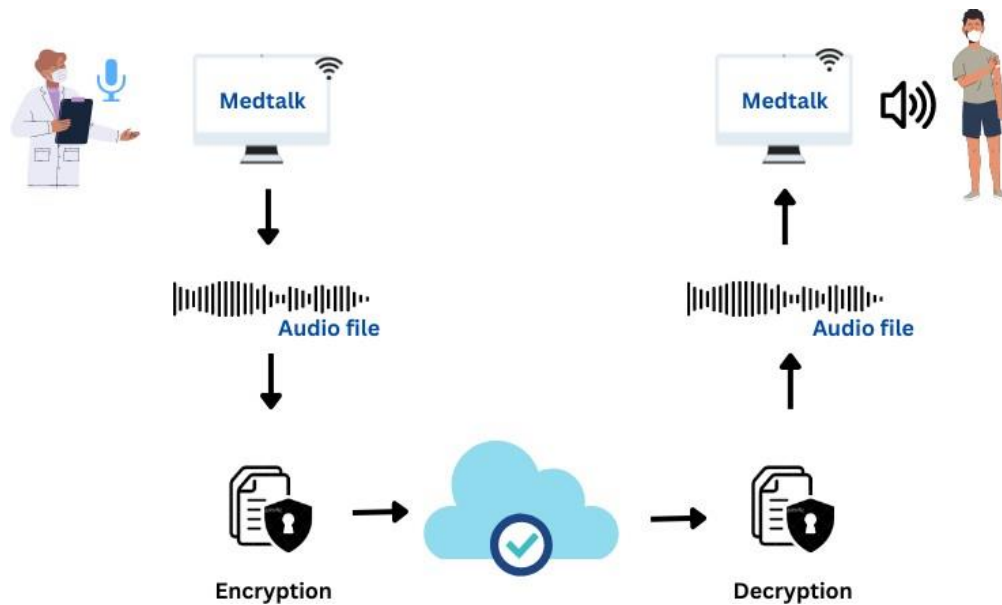


Figure 7: Conceptual Model of the proposed system.

3 CHAPTER THREE: METHODOLOGY

3.1 CHAPTER OVERVIEW

This chapter outlines the systematic approach that will be used in the execution of the project to achieve the project and research objectives. It provides details on data collection, and data analysis techniques used to address the project objectives. The approach used in improving efficiency in transmitting secure audio messages is Rapid Application Development (RAD) methodology.

3.2 RESEARCH METHODOLOGY

The research methodology used in achieving the research objectives is quantitative research methodology, in which, is an approach used to gather, analyze, and interpret numerical data to investigate relationships, patterns, and trends in a systematic and objective manner. It involves the collection of data through structured surveys, experiments, or observations, followed by statistical analysis to draw objective conclusions. The key characteristics and elements of quantitative research methodology include research design, data collection, data analysis and objectivity.

3.3 SYSTEM DEVELOPMENT METHODOLOGY

Rapid Application Development (RAD) describes a software development process based on prototyping and iterative development. It focuses and emphasizes on speed, efficiency, and continuous feedback in the development process to deliver a working product as quickly as possible. There are four steps in the RAD model which are Requirements definitions, Prototype, Construction and Deployment. The following demonstrates a diagrammatic representation of the model.

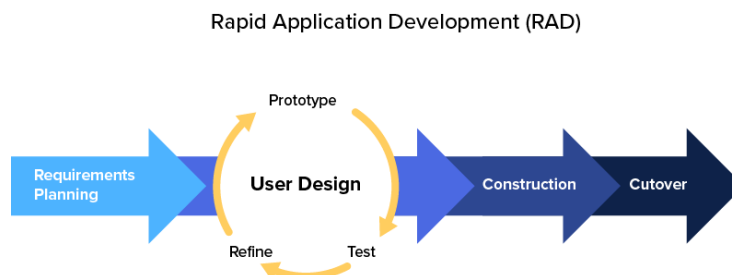


Figure 8: Showing Summary of the RAD methodology Process.

3.3.1 Requirements Planning

This is the initial stage of the software development process, where the requirements necessary hardware and software requirements for the software application are researched, gathered, documented, analyzed, and prioritized. The project schedule is also created, and the cost of the resources is also analyzed. The users would be interviewed to get information on how the security of audio transmitted messages can be improved.

3.3.2 User Design

The user design phase in RAD is a crucial part of the development process. In this stage, the user interface and user experience of the application are designed. The user design phase typically involves the prototype design based on the requirements. Block diagrams, flowcharts, circuit design layouts are designed. The Graphical user interface appearance is also designed during this stage to show the layout of the application, the placement of elements, and the flow of the user interface.

These prototypes are to be then showcased to the supervisor and other members for review and to gather feedback and identify any issues or areas for improvement. This feedback is used to refine the design of the user interface. Based on the feedback, the user interface design is refined and updated. This process may be repeated several times until the design is finalized and of satisfactory.

3.3.3 Construction

During this stage, application coding, system testing, and unit integration occurs, converting prototype and beta systems into a working model. This phase can also be repeated supporting changes, or new components and alterations. The working model goes through a series of testing to ensure that it meets the project objectives.

3.3.4 Cutover

The final stage is the cutover stage where intensive scale testing, final customizations, and final technical documentation is to be done. Spending time debugging the application and running final updates and maintenance tasks before final presentation.

3.4 RESOURCES

This section focuses on all the necessary items that are required to make the project successful. The types of resources to be used are carefully outlined below:

Hardware Resources

1. Operating system - Ubuntu 20.04, window, macos
2. 2.70GHz processor speed.

Software Resources

1. Visual Studio Code IDE that will be used during the coding of the application.
2. Node Package Manager
3. Google Chrome browser.
4. Firebase as a database.
5. WebRTC for implementing real time communications.
6. Agora sdk.
7. Creatly for implementing all the designs/drawings
8. GitHub for backing up the project and management of the project.
9. Microsoft word and PowerPoint are used for documentation and presentation respectively.

Programming Tools and Technologies

1. JavaScript, TypeScript, HTML 5, CSS would be used for the implementation of the system.
2. Node.js and Express.js. for backend development
3. Libraries - Crypto.js, mail.js to send emails to the users

3.5 GANTT CHART

The following is a visual representation of the project schedule that outlines the timeline of various tasks and activities involved in developing the system:

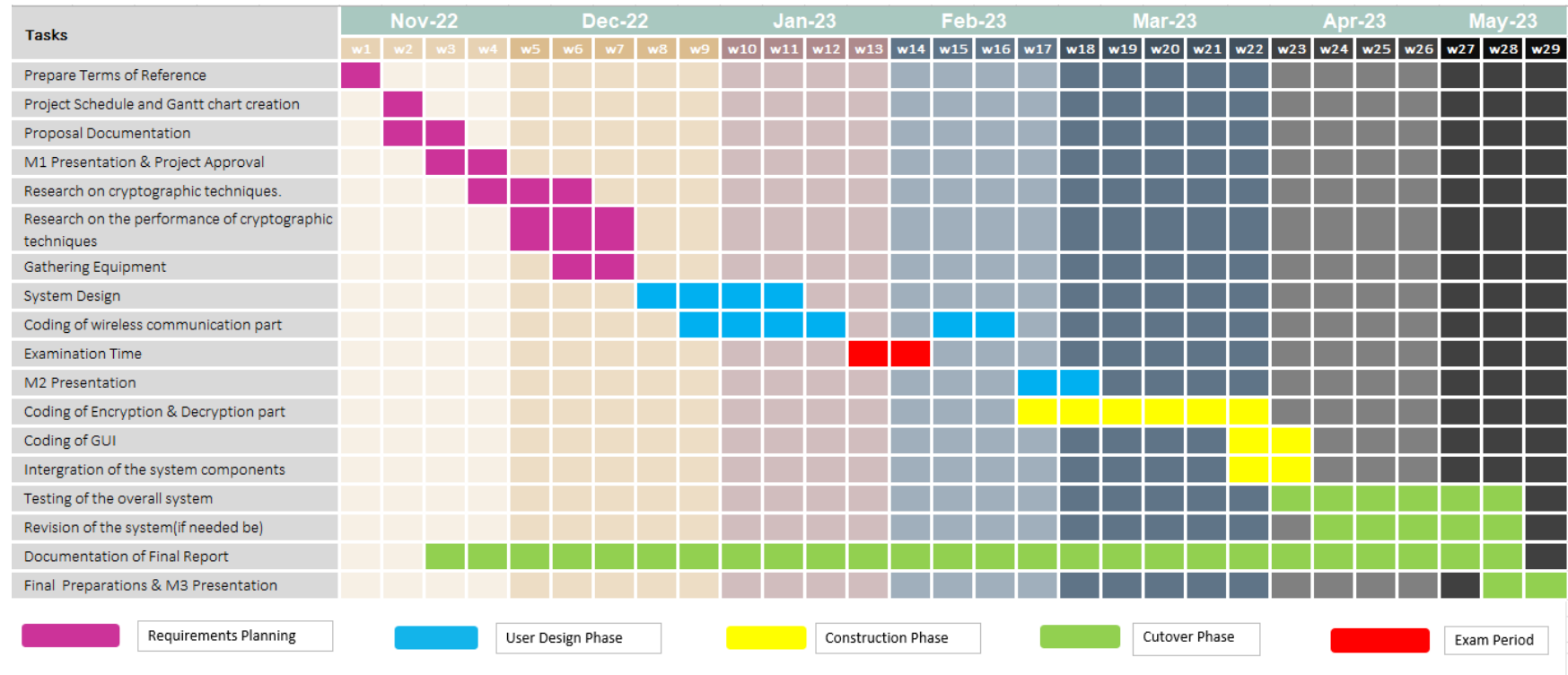


Figure 9: Gantt Chart

4 CHAPTER FOUR: SYSTEM ANALYSIS AND DESIGN

4.1 CHAPTER OVERVIEW

This chapter captures the requirements necessary for the effective functioning of the system, the tools used, and outlines a detailed design of the system. It will also analyze and evaluate the system to determine whether it meets the user's requirements and objectives of the study.

4.2 SYSTEM ANALYSIS

4.2.1 Requirements Gathering

Informal interviews were conducted in Aga Khan University Hospital and Kijabe National Hospital, within the Country. The interviews with the users of the proposed system allowed the researcher to identify the requirements and necessary features that should be included to the system. Content analysis was carried out to identify some of the key questions that included the nature of the expected graphical user interface, audio security needs, and features needed within the system.

A survey approach was used to understand the encryption algorithms methods that have been adopted in carrying out secure audio transmission, their advantages, and disadvantages. Some of these such as Advanced Encryption Standard (AES), RSA Encryption, Hybrid Encryption Algorithms were adopted for comparison. This was a key step in the research since it made it easier to identify an appropriate algorithm for audio encryption depending on the level of security required, the processing power available, and the desired speed of encryption and decryption.

A document analysis was used where reviewing existing documentation, such as user manuals, technical specifications, and business requirements, was used to further identify and compare the different encryption algorithms methods used.

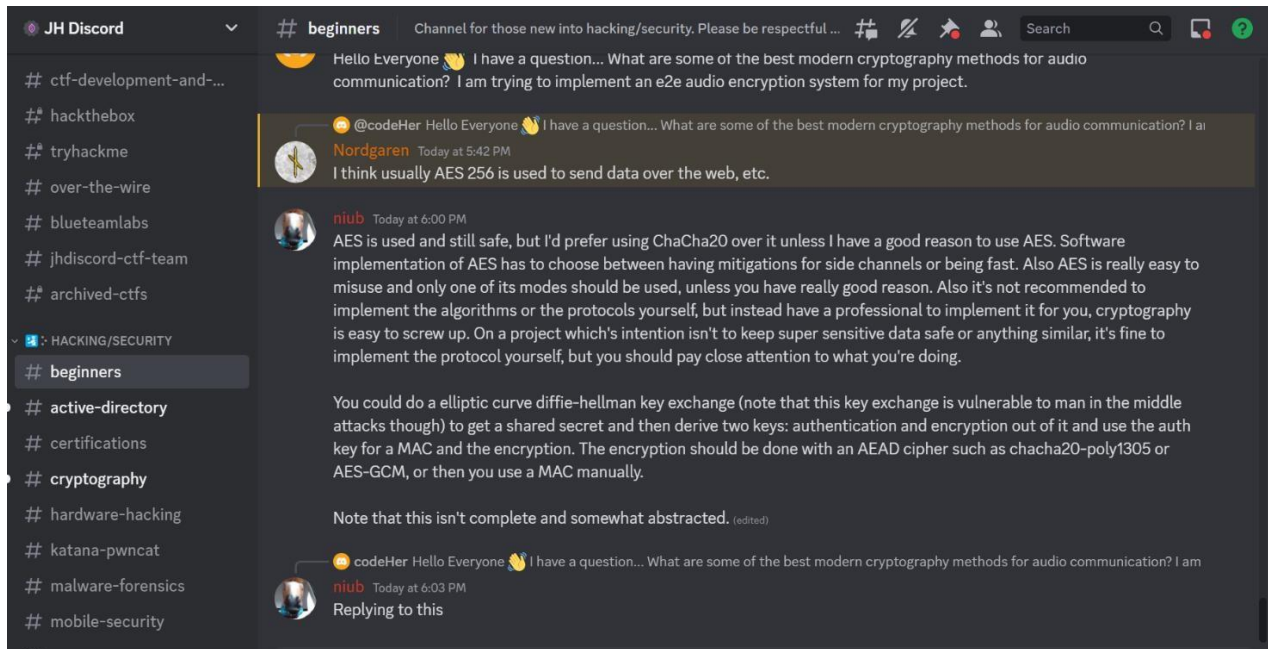


Figure 10: Evidence of the system analysis discussion.

4.2.2 Requirements Specifications

The target users of this system are healthcare professionals. The proposed System architecture is comprised of two main components, the transmitter (the practitioner), which is responsible for transmitting encrypted audio message to the receiver, who is responsible for receiving the encrypted audio message and decrypting it. Both components are to be implemented using the WebRTC and programmed using JavaScript on the Visual Studio IDE. The end-to-end encryption algorithm to be used to ensure maximum level of security of data would be Advanced encryption Standard (AES) because of its capability to be fast, efficient, and its implementation on a wide range of hardware and software platforms.

All these components help in the generation of an End-to-End audio encryption system that ensures secure communication among two parties which is the general objective of the study. The system comprises of both functional and non-functional requirements. The non-functional requirements are a set of criteria used to evaluate a system's specific operation. Therefore, a software system's quality attribute is defined by a non-functional need. A functional requirement on the other hand, defines a system or one of its components and it specifies the tasks that the program or system must complete.

4.2.2.1 *Functional Requirements*

The following are Functional requirements:

1. **Consultation meeting creation:** The system should allow users to create consultation meeting rooms.
2. **Audio Conferencing:** The system should allow users to connect to meetings using their computer's microphone.
3. **Secure key generation:** The system must be able to generate secure encryption keys that can be used to encrypt and decrypt audio data.
4. **Audio encryption and decryption:** The system must be able to encrypt and decrypt audio data using the encryption keys, so that it can be securely transmitted between parties.
5. **Authentication:** The system must provide secure authentication of users, to ensure that only authorized parties are able to access and use the system.
6. **Compliance with encryption standards:** The system must comply with industry-standard encryption algorithms and protocols, to ensure that it provides a high level of security.

4.2.2.2 *Non- Functional Requirements*

The following are non-functional requirements:

1. **Security:** The system must provide a high level of security, including protection against unauthorized access, interception, and eavesdropping.
2. **Performance:** The system must perform efficiently and reliably, with minimal latency and delay, and minimal impact on the quality of the audio data.
3. **Scalability:** The system must be able to scale up or down to meet the needs of different users and environments, without compromising its security or performance.
4. **Usability:** The system must be easy to use, with clear instructions and guidance for users.
5. **Reliability:** The system must be reliable and available, with minimal downtime or interruption in service.
6. **Maintainability:** The system must be easy to maintain and update, with minimal disruption to service.

By meeting these functional and non-functional requirements, an end-to-end audio encryption system can provide a high level of security for the communication of audio data, performance, and usability, and ensure that it is scalable, reliable, and easy to maintain and update.

4.2.3 Feasibility Study

Analysis the information gathered during the requirement's gathering phase was used determine the feasibility of the proposed system. The following were the feasibility studies that were done:

4.2.3.1 Technical Feasibility

A technical feasibility was carried out and it was evident that the project was feasible with minimum risk as outlined below:

Table 1: Technical Feasibility summary

Technology required	Current availability	Risk	Action
Audio Encryption Algorithms	Available	None	N/A
Encryption and WebRTC libraries	Available (Open source)	None	N/A
Programming languages and Environments	Available (JavaScript, Java, HTML/CSS and Visual Studio IDE)	None	N/A
Storage and Processing power	Available	Managing processing power and storage	Optimizing the system's processing and storage capabilities to ensure that the encryption and decryption process does not significantly affect system performance.

4.2.3.2 Economic Feasibility

There were no significant implications on cost since all the required technologies were open source.

4.3 SYSTEM DESIGN

This section describes the overall system design. The design phase transforms the requirements gathered during the requirements planning phase into a computer system design, which details what the system is required to do.

4.3.1 Components of the Designed System

The developed prototype is comprised of two main components, the transmitter (the sender) which is responsible for transmitting the audio message and the receiver which is responsible for receiving the audio message. Both the transmitter and receiver comprise of audio devices (speaker, microphone), Network infrastructure, Security and Encryption.

4.3.2 High Level Design

The following diagram illustrates the overall high-level block diagram of the End-to-End audio encryption teleconsultation system.

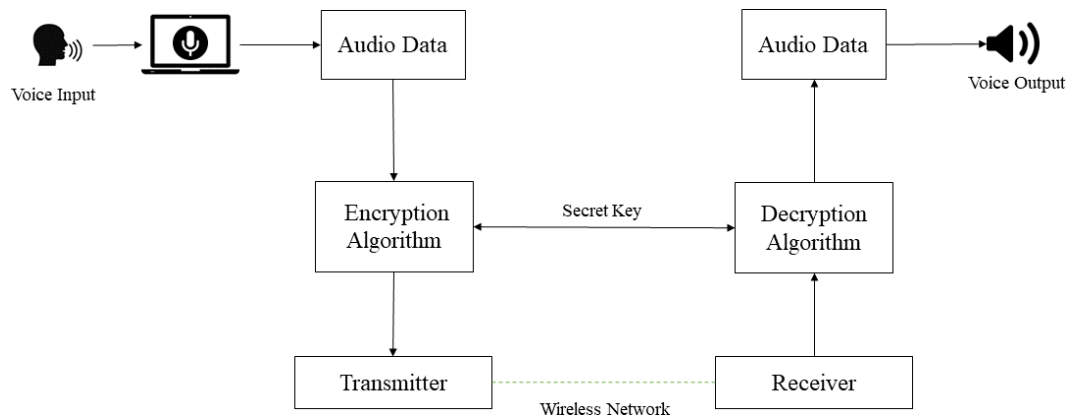


Figure 11: Block Diagram of the designed E2E audio encryption System prototype.

4.3.3 Low Level Design

The following diagram illustrates the overall low-level block diagram of the End-to-End audio encryption teleconsultation system.

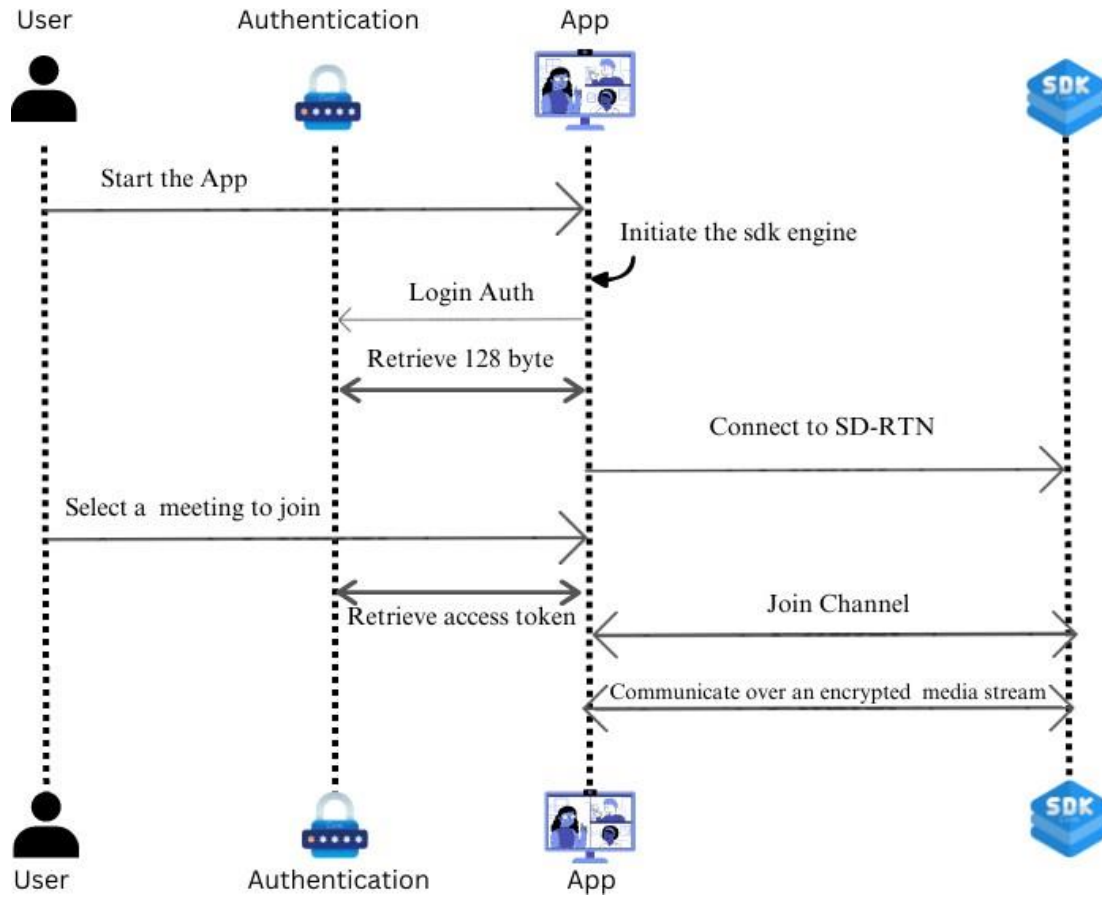


Figure 12: Sequence diagram of the E2E audio encryption System prototype

4.3.4 Flowcharts

This following represents flowchart diagrams that illustrate the sequence of steps or processes involved in the system. The purpose of these diagrams is to provide a visual representation of how the process works and help to identify potential problem areas, bottlenecks, or areas where improvements can be made. By following the various shapes and symbols in the flowchart, we can see the flow of information, decision points and actions taken at each stage of the process and help visualize complex processes to make them easier to understand.

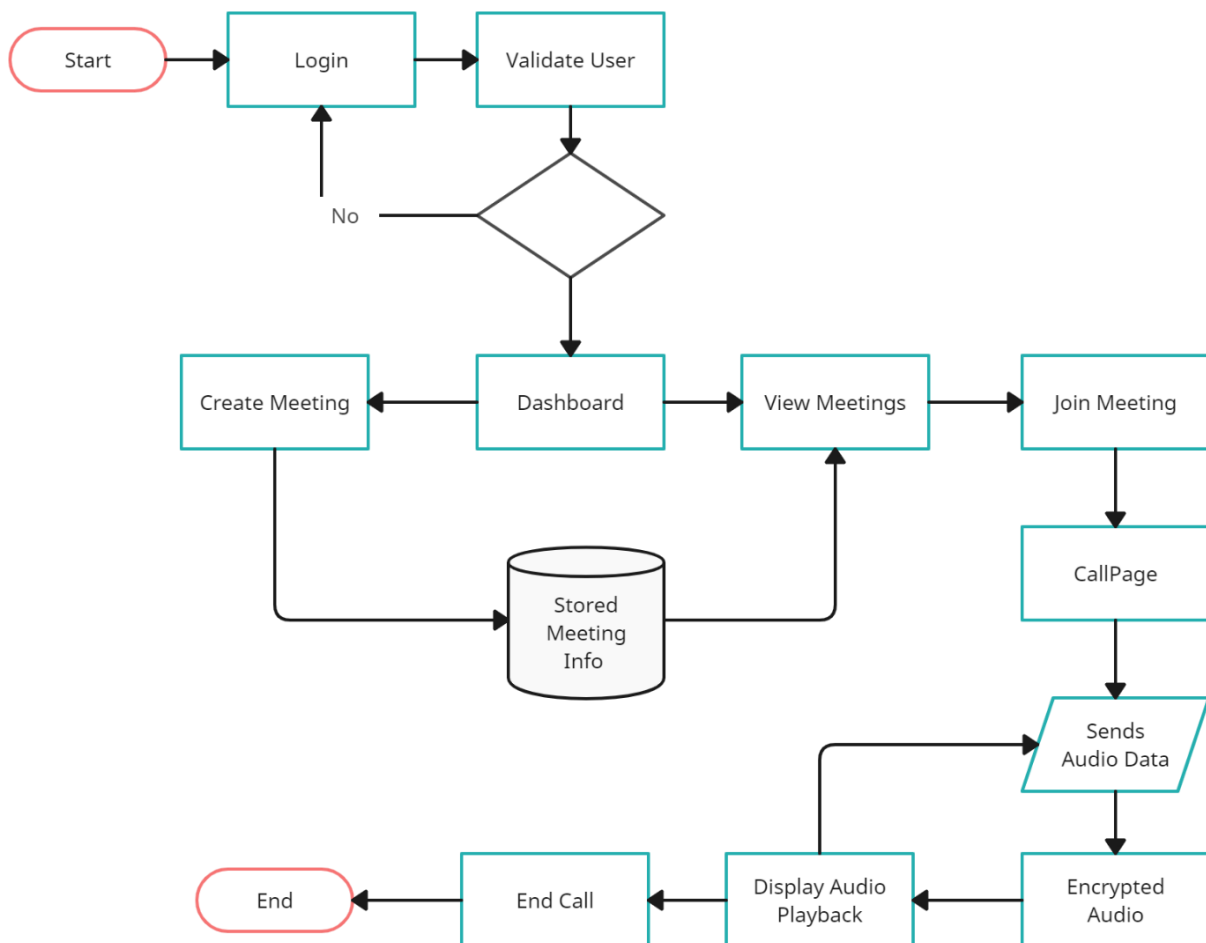


Figure 13: Showing the Transmission process of the audio message.

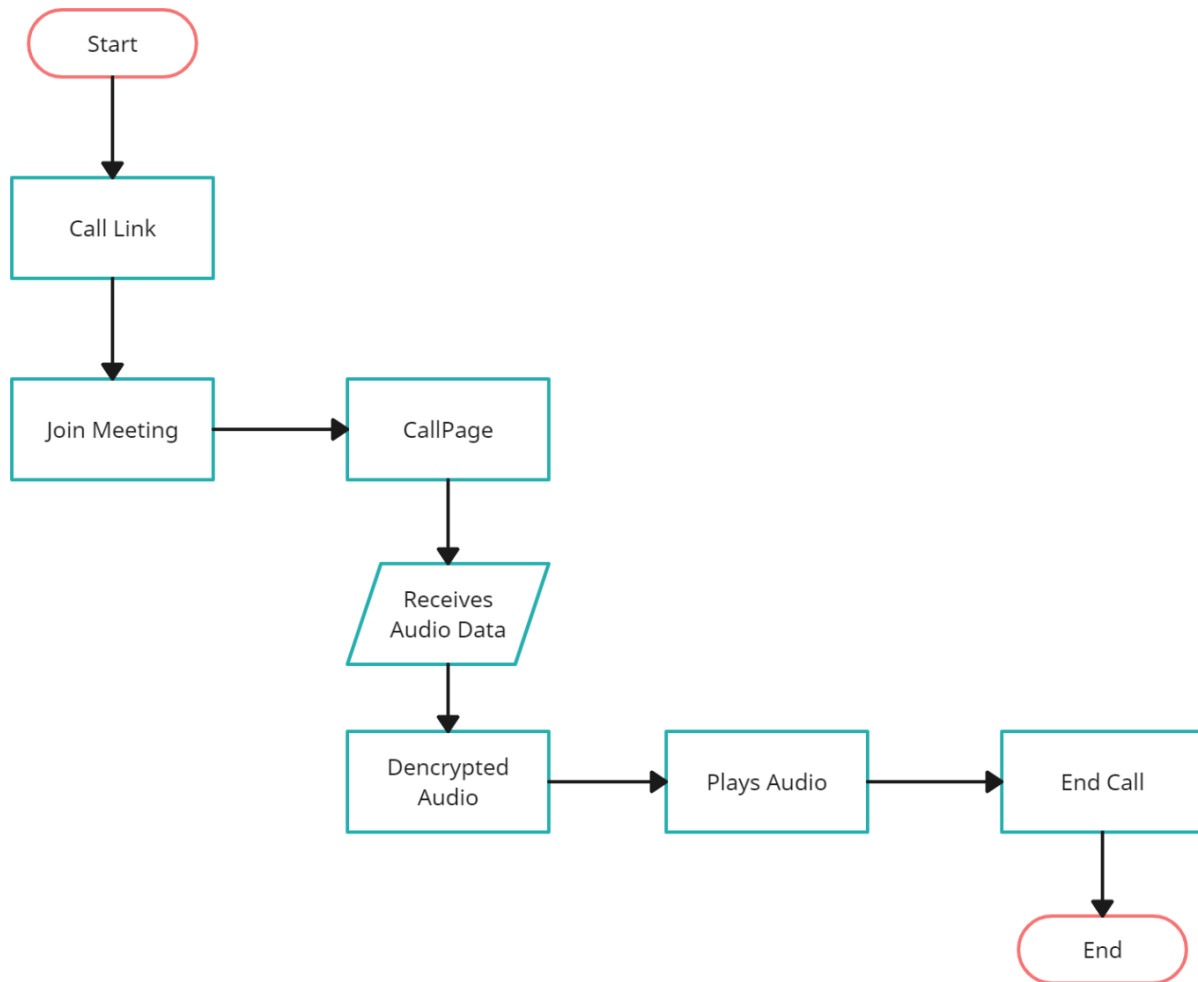


Figure 14: Showing the Receiver Side flowchart.

4.3.5 User-Interface Design

The following pictures entail high-fidelity wireframes for the graphical user interface (Frontend) that the end users would be able to send audio messages securely.

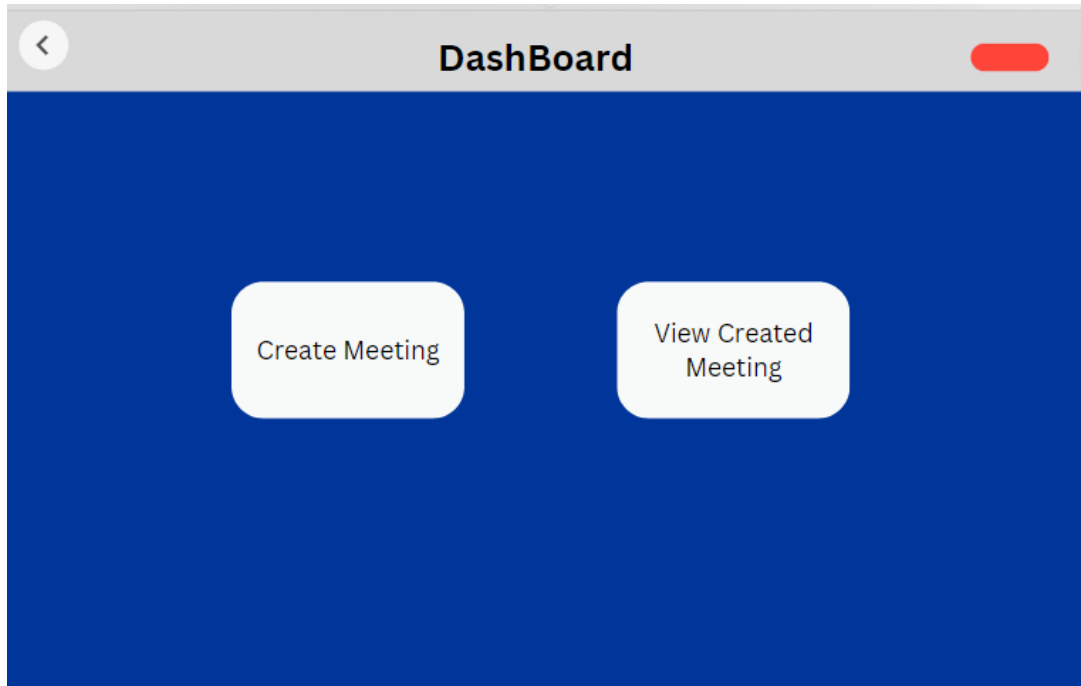


Figure 15: UI Design for the Dashboard Page.

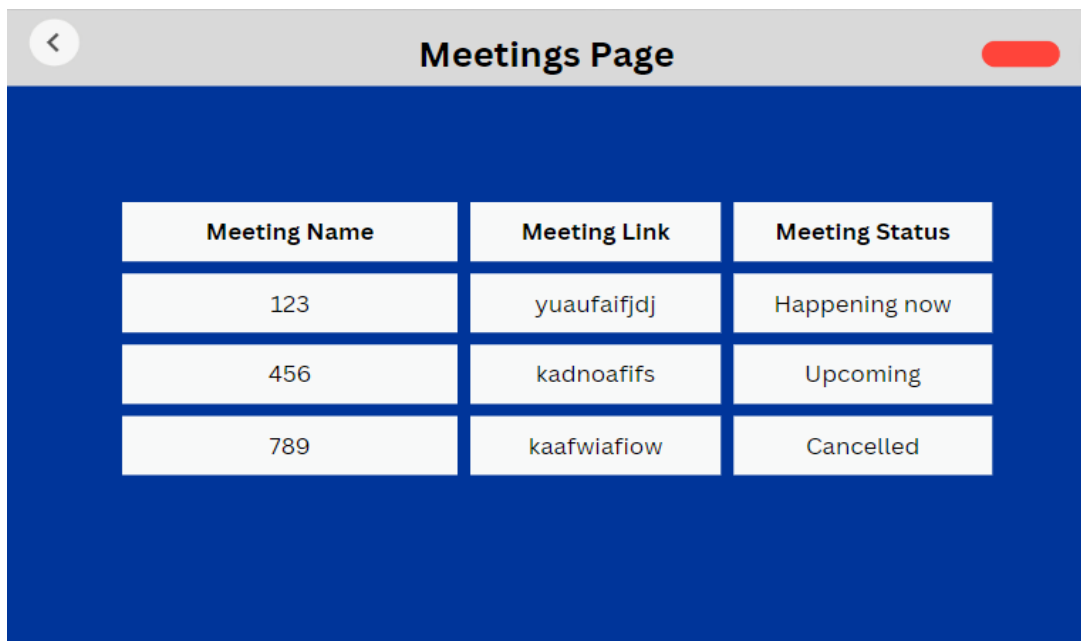


Figure 16: UI Design for the Meetings Page.

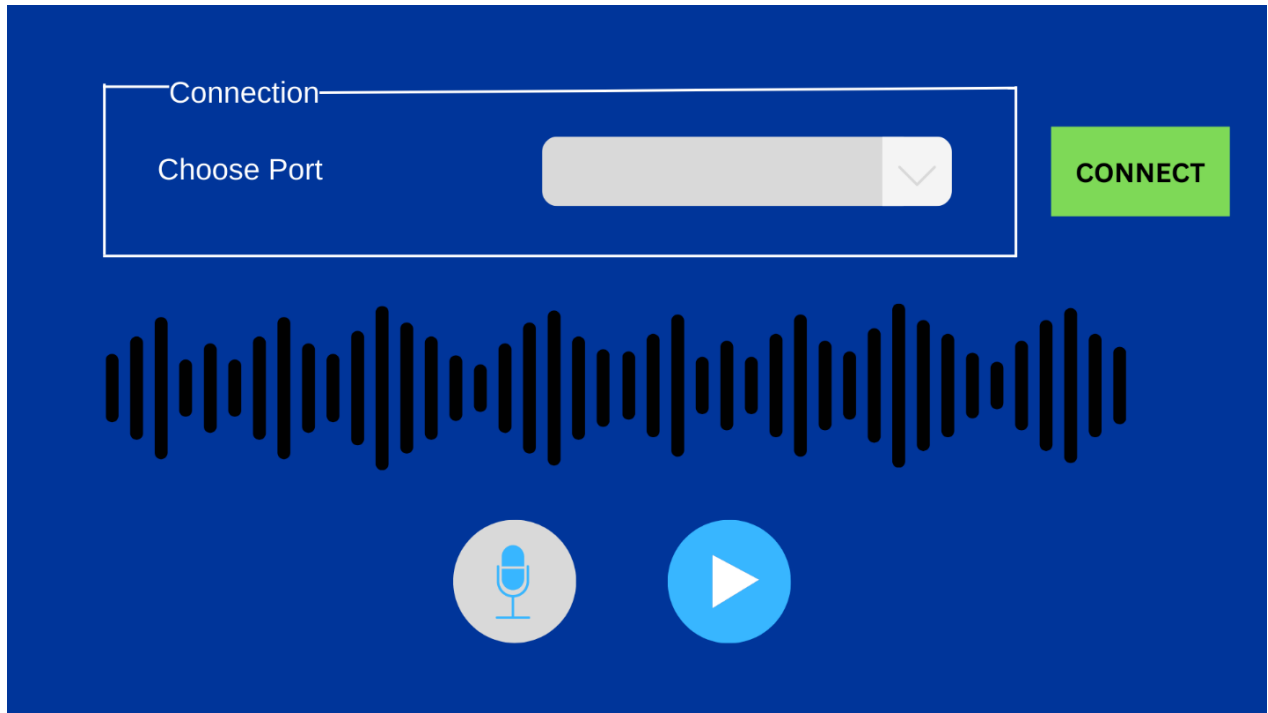


Figure 17: UI design of the GUI Call Page.

5 CHAPTER FIVE: SYSTEM IMPLEMENTATION

5.1 Chapter Overview

This chapter discusses the various activities and considerations involved in the system implementation process. It explores the steps taken to translate the system design into executable code.

5.2 Programming Tools, Techniques and Technologies

The following technologies were used in the development of the system.

1. **NodeJS**- An asynchronous event-driven JavaScript runtime designed to build scalable network applications.
2. **ReactJS**- JavaScript library for building user interfaces (UIs) for web applications.
3. **Elastic UI Framework** – This is a design and development framework created by Elastic to help developers create consistent and user-friendly user interfaces for web applications and websites.
4. **CryptoJS**-Crypto.js is a JavaScript library that provides cryptographic functionalities for web applications. It offers a wide range of cryptographic algorithms, including hashing, encryption, decryption, digital signatures, and more.
5. **WebRTC**- An open-source protocols and APIs that enables real-time communication and peer-to-peer data transfer directly between web browsers or other compatible applications.
6. **Postman** - An API platform used to design, build, test, and document APIs.
7. **Firebase Authentication** - Firebase Authentication provides easy-to-use SDKs for user authentication using including email/password, social media logins.
8. **Cloud Firestore** - Firestore is Firebase's next-generation NoSQL database, designed for scalability, real-time synchronization, and offline support.
9. **Version Control**- The use of Git version control was a necessity to track any changes made to the system source code on every update. All changes were committed and pushed to Github for hosting my project repository. This provides an online backup for the system source code if an accident occurs to the local repository. Github was also used as a project management tool to manage my tasks.
10. **Continuous Integration/Continuous Deployment (CI/CD)** - CI/CD tools were set up to automate the build and deployment processes.

5.3 Implementation Results

The implementation process involved two major parts:

1. Audio chat development.
2. Encryption implementation.

Both modules of the development process were of significance to the one unified system that was seeking to fulfill the objectives set.

5.3.1 Audio Chat Development

Establishing peer-to-peer connections and enabling audio communication was done through WebRTC using the agora sdk to relay the audio data from the sender to the receiver.

Once the Peer-to-Peer connection is established, use the `getUserMedia` API to access the user's microphone and capture the audio stream.

```
const initWebRTC = () => {  
  navigator.mediaDevices.getUserMedia({  
    video: true,  
    audio: true  
  })  
  .then((stream) => {  
    setStreamObj(stream);  
  })  
}
```

Figure 18: WebRTC functionality

5.3.2 Encryption Implementation

This was the main aspect of the development process which involved securing the data that is being relayed. Since it is a mandatory requirement of all WebRTC products to be encrypted. The SDK already provided us with its own encryption key to encrypt and decrypt the media stream that it provided. An AppID was provided which is the project's unique identification for secure tokenization, and the AppSign was the security key used to encrypt and decode data exchanged between the SDK and the Cloud servers.

```
client > .env  
1 REACT_APP_HOST="localhost:3000"  
2 REACT_APP_APP_ID="984465085"  
3 REACT_APP_SERVER_SECRET="8eb38e5d33742411845e568f674e25a6"  
4
```

Figure 19: Picture of AppID and AppSign.

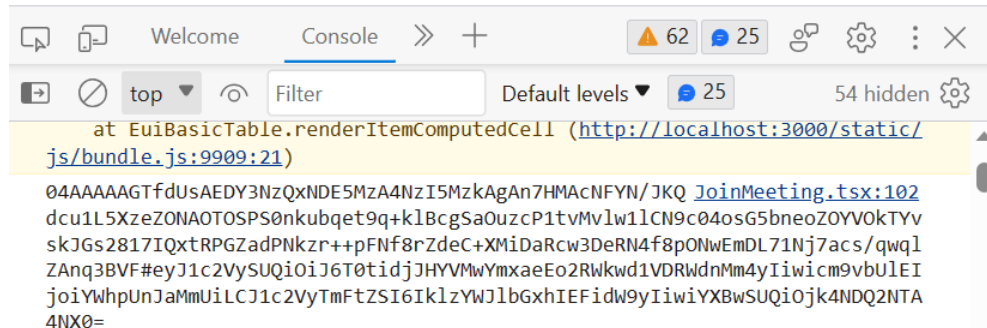
```

    parseInt(process.env.REACT_APP_APP_ID!),
    process.env.REACT_APP_SERVER_SECRET as string,
    params.id as string,
    user?.uid ? user.uid : generateMeetingID(),
    user?.displayName ? user.displayName : generateMeetingID()
  );
  console.log(kitToken)

```

Figure 20: Picture of the implementation of the AppId and AppSign.

For testing purposes, the following is the token being sent over the web



The screenshot shows a web browser's developer console with the 'Console' tab selected. It displays a long alphanumeric string, which is a token, sent from a web application. The token is: 04AAAAAGTfdUsAEDY3NzQxNDE5MZA4NzI5MzkAgAn7HMACNFYN/JKQ JoinMeeting.tsx:102 dcu1L5XzeZONAOTOSPS0nkubqet9q+klBcgSa0uzcP1tvMv1w1lCN9c04osG5bneoZ0YV0kTYv skJGs2817IQxtRPGZadPNkzr++pFNf8rZdeC+XMiDaRcw3DeRN4f8pONwEmDL71Nj7acs/qwql ZAnq3BVF#eyJ1c2VysUQiOiJ6T0tidjJHYVMwYmxaeEo2RWkwd1VDRWdnMm4yIiwicm9vbU1EI joiYWhpUnJaMmUiLCJ1c2VyTmFtZSI6IklzYWJlbGxhIEFidW9yIiwiaXBwSUQiOjk4NDQ2NTA4NX0=

Figure 21: Picture of the token on the web

On top of the AppId and AppSign, another encryption method was added for additional security and implementation purposes. This development process involved some keys parts which including:

1. Key Generation

The following is the encryptionKey variable would typically hold the secret key that is used for encryption and decryption.

```

var encryptionKey = "2a3c5e7g9i1k3m5o7q9s1u3w5y7a9c1e";
var encryptionSaltBase64 = "R0JZVmJqbGxhVzVrY2lWbFoybHVhVzUwYjNKaGJXVWdKR1Z6ZEdGc2JHbG1hVzRn";
var encryptionMode = "AES-GCM";

```

Figure 23: Picture of Key Generation.

2. Encryption and Decryption

The next section enabling the encryption and decryption functionality. The following is the encryption and decryption functions that were responsible for encrypting and decrypting the media being relayed.

Firstly, we converted the encryptionSaltBase64 to base64ToUint8Array, the encryptionKey to hex2ascii, the set the encryptionMode variable to a encryption mode and finally called setEncryptionConfig and pass encryptionMode, encryptionKey, and encryptionSaltBase64 as parameters.

5.4 Software Testing

Software testing is a crucial process in software development that involves evaluating the functionality, quality, and performance of a software application. It is performed to identify defects, errors, or any gaps between expected and actual results, ensuring that the software meets the specified requirements and delivers reliable user experience.

5.4.1 Unit Testing

It refers to testing of an individual program or module. It identifies and eliminates execution errors that could cause the program to terminate prematurely. The modules tested includes Create meeting module, Join meeting module, WebRTC module, Encryption/Decryption module.

5.4.2 Integration Testing

It involves testing two or more programs that depend on each other. This ensures that the client and the server, API, are working efficiently with no errors on integration.

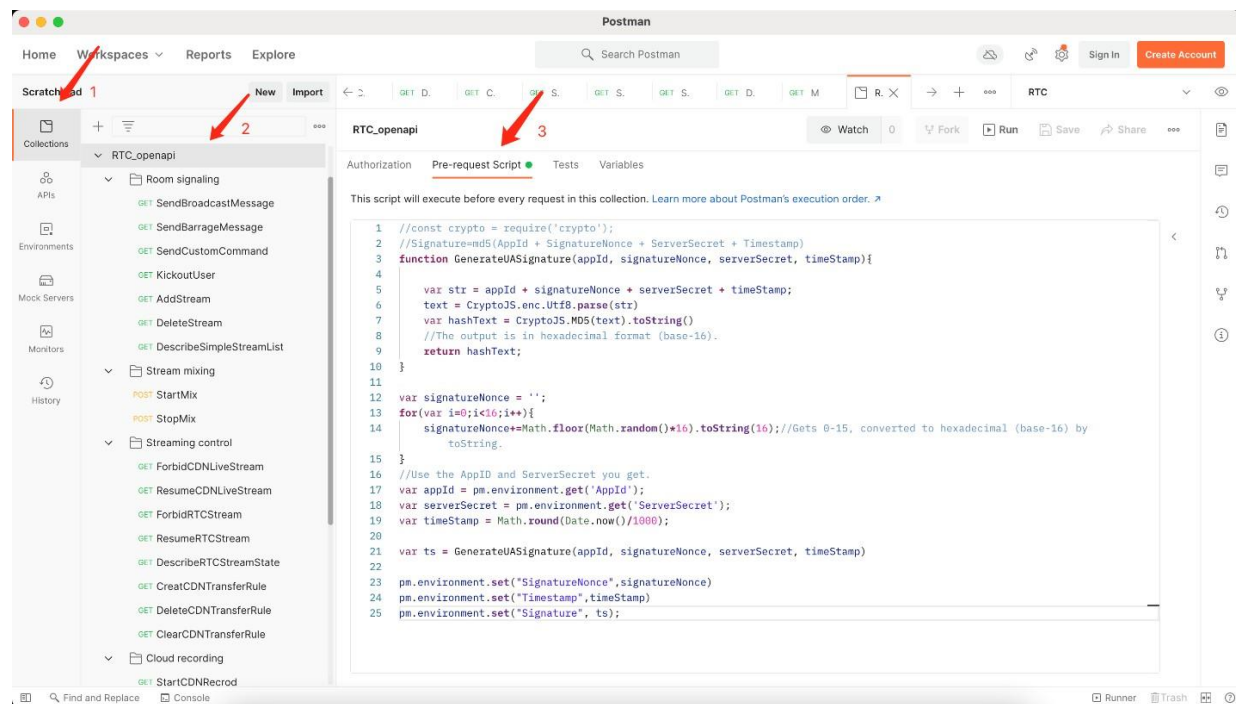


Figure 24: Picture of Unit testing the RTC api.

5.4.3 System Testing

This involves testing the system as a whole unit to demonstrate that the system performs both functionally and operationally as specified. This testing involves the following tests.

5.4.4 Test Cases

Several test cases were performed to view how the system would respond when different inputs were given by the user. Some of the tests conducted are listed below:

Table 2: Test Cases

Test Case ID	Module	Description	User Scenario	Expected Result	Actual Output
1	User Login	To test User Authentication	Enter Valid login. credentials	Redirect to Dashboard Page	Redirected to Dashboard Page
			Enter invalid login credentials	Displays Error in the incorrect input fields	An Error Message below the input fields
2	Create Meeting	To test create meeting functionality	Enter valid meeting details	Redirects to the Dashboard and display and notification that the meeting has be successfully created	Redirected to the Dashboard and display of success message.
			Enter invalid meeting details	Displays Error in the incorrect input fields	An Error Message below the input fields
			User cancels before submitting the create meeting form	Redirects to the Dashboard	Redirected to Dashboard Page

3	Edit Meeting	Testing the Edit functionality	User edits meeting with valid details	Meeting status changes according to the changes made. (Upcoming, Cancelled, Join Now)	Meeting status changed
			User edits meeting with invalid details	Displays Error in the incorrect input fields	An Error Message below the input fields
4	Copy Meeting	Testing the Copying functionality	User clicks the copy button	The meeting link is copied and it is displayed without the #init	The meeting link is copied and it is displayed without the #init
5	Join now	Testing the Audio communication functionality	User attempts to join a meeting that they have created	Displays the meeting call page with #init url	Displayed the meeting call page with #init url
			User attempts to join a meeting that they have been invited to	Displays the meeting call page with without #init url	Displayed the meeting call page with without #init url
			User attempts to join a meeting that they are not invited to	Displays an error message.	Displayed an error message.
6	Playback button	Encrypted audio playback	User clicks the play button	Encrypted audio is played	Static audio was heard on the testing phase only and didn't not play on the webapp.

6 CHAPTER SIX: CONCLUSIONS AND RECOMMENDATIONS

6.1 Chapter Overview

This chapter details the achievements and challenges that were faced while undertaking this project. It also recommends further work for advancement of this project.

6.2 Achievements

The successful completion of this project has resulted in the development of an end- to end audio encrypted teleconsultation system for the case of healthcare organizations. The achievements of the project were based on whether the system objectives were achieved:

“To prepare system requirements analysis specification for E2EE audio communication systems.”

Requirements analysis specification was done through forums, informal interviews, and document analysis hence the objective was achieved.

“To design a communication system that securely sends and receives audio data through RF.”

Designs were drawn to ensure the functionality both functional and non-functional were achieved.

“To ensure the audio data is sent and received through radio frequency.”

Audio was successfully rendered through WebRTC using agora sdk and was able to be received by the receiver hence the objective was achieved.

“To incorporate a highly secure algorithm for audio transmission and reception at low costs.”

The AES algorithm was implemented but with challenges leading to logical errors during testing.

“To test and evaluate the proposed system.”

Unit, System and Integration testing was done to ensure the functionality of the system.

“To write documentation on the proposed system.”

Project documentation was written and covers all aspects of the system, including its purpose, features, design, implementation approach, and potential benefits.

6.3 Challenges and limitations

There were several challenges encountered while implementing the system such as delayed responses from healthcare and cybersecurity professionals which hindered the project progress. Similarly, some users completely failed to respond which prompted the use of assumptions.

The system was first intended to be developed using low costs microcontrollers such as Arduino but the audio quality was not satisfactory and the recommended Arduino due was not available in the local hardware stores.

Complexity of the system: The system involves multiple components, including encryption modules, key management systems, communication protocols, and integration with existing infrastructure, the overall system architecture can become complex. Ensuring that all components work together seamlessly, without introducing vulnerabilities or performance bottlenecks, requires careful design, testing, and monitoring. This complexity of the system required more research than I intended it to, hence affecting and hindering the overall performance and progress of the project.

Furthermore, development of the audio playback was difficult in that it could not store continuous encrypted audio data hence for demonstration of the encrypted audio and testing of the encryption algorithm was harder than I anticipate it to be.

6.4 Conclusions

In conclusion, the development of an end-to-end audio encrypted teleconsultation system represents a powerful stride towards securing sensitive audio data and safeguarding privacy in the healthcare industry and I personally encourage healthcare organizations to implement the system for it would transform the way we live and ensure that the teleconsultation capabilities are secure to protect the patient's personal information.

Moreover, implementing this system was a great learning experience for it helped me help a lot about network security, encrypting algorithms, technical documentation, software analysis and design and many more. The project was most definitely a personal success.

6.5 Recommendation for further work

Despite the successes that have been made in implementing this project, there is more that can be done arising from this project to improve it. Some of my recommendations are as follows:

1. Implementing more health sector coded features for example file sharing and videoconferencing for seamless communication within the organization.
2. Integrating the system with a doctor's appointment system/ Electronic health record.
3. Implementing steganography in the system to conceal information and Zero trust technology to enhance the security of the system.

REFERENCES

1. Abuor, I. (2021) 'RF Secure Coded Communication System'. Available at: https://drive.google.com/file/d/1mOpXorJGvIILxX2vg_HvGF7JLWAWGm/view?usp=share_link.
2. Fahmy, S. (2018) 'ENCRYPTION AND DECRYPTION OF AUDIO SIGNAL BASED ON RSA ALGORITHM', 5, pp. 57–64. Available at: <https://doi.org/10.5281/zenodo.1341956>.
3. Javatpoint (2016) *Java Code for DES - Javatpoint*. Available at: <https://www.javatpoint.com/java-code-for-des> (Accessed: 9 March 2023).
4. Karleigh, M., W, E. and Dean, E. (2016) *Enigma Machine / Brilliant Math & Science Wiki*. Available at: <https://brilliant.org/wiki/enigma-machine/> (Accessed: 9 March 2023).
5. Katz, J. and Lindell, Y. (2015) *Introduction to Modern Cryptography*. Available at: https://eclass.uniwa.gr/modules/document/file.php/CSCYB105/Reading%20Material/%5BJonathan_Katz%2C_Yehuda_Lindell%5D_Introduction_to_Modern_Cryptography.pdf (Accessed: 16 February 2023).
6. Iutkevich, B. (2021) *What is End-to-End Encryption (E2EE) and How Does it Work?*, Security. Available at: <https://www.techtarget.com/searchsecurity/definition/end-to-end-encryption-E2EE> (Accessed: 9 March 2023).
7. Neha, T. (2020) 'What is Advanced Encryption Standard (AES)? Definition, Encryption, Decryption, Advantages and Disadvantages', *Binary Terms*, 5 May. Available at: <https://binaryterms.com/advanced-encryption-standard-aes.html> (Accessed: 9 March 2023).
8. Puneet (2020) 'What is RSA? How does an RSA work?', *Encryption Consulting*, 23 September. Available at: <https://www.encryptionconsulting.com/education-center/what-is-rsa/> (Accessed: 9 March 2023).
9. Ravi, D. (2017) *An Examination of the Caesar Methodology, Ciphers, Vectors, and Block Chaining*, Infosec Resources. Available at: <https://resources.infosecinstitute.com/topic/an-examination-of-the-caesar-methodology-ciphers-vectors-and-block-chaining/> (Accessed: 9 March 2023).
10. Richards, K. (2021) *What is Cryptography? Definition from SearchSecurity*, Security. Available at: <https://www.techtarget.com/searchsecurity/definition/cryptography> (Accessed: 15 February 2023).
11. Simplilearn (2020) *What Is Data Encryption: Algorithms, Methods and Techniques [2022*

Edition]/ *Simplilearn*, *Simplilearn.com*. Available at: <https://www.simplilearn.com/data-encryption-methods-article> (Accessed: 9 March 2023).

12. Vardhaman, R. and Ivanov, I. (2023) *What Is AES and Why You Already Love It?* Available at: <https://techjury.net/blog/what-is-aes/> (Accessed: 9 March 2023).
13. *WhatsApp* (2020) *WhatsApp.com*. Available at: <https://www.whatsapp.com/privacy> (Accessed: 9 March 2023).

APPENDICES

APPENDIX A: User Manual

The user first encounters the login page in which they can login in with email and password or via google for a faster login. If the user has an email two factor authentication will then be prompted for further security. If they user has no account, they can click signup page to create an account.

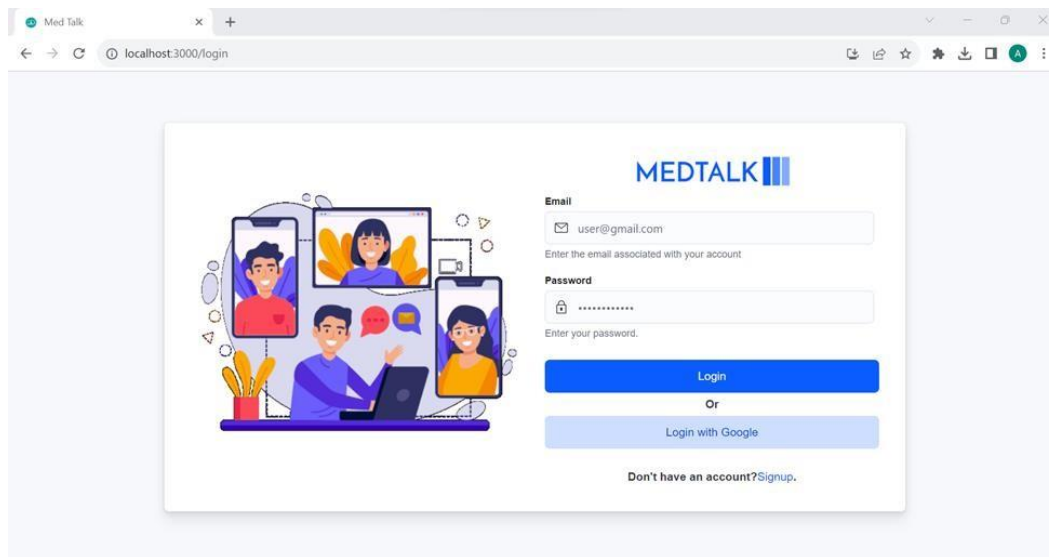


Figure 25: Final Login page

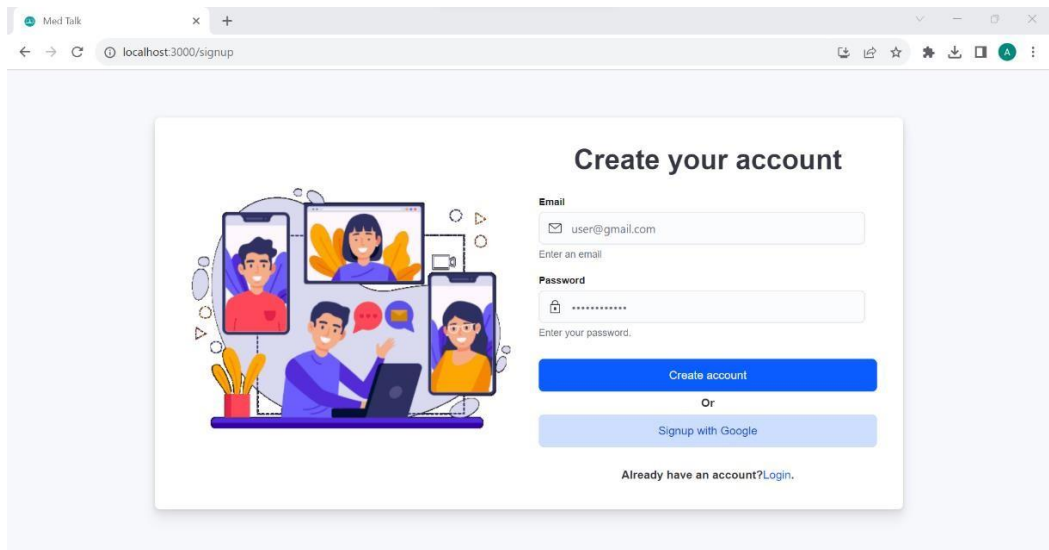


Figure 26: Final Signup page

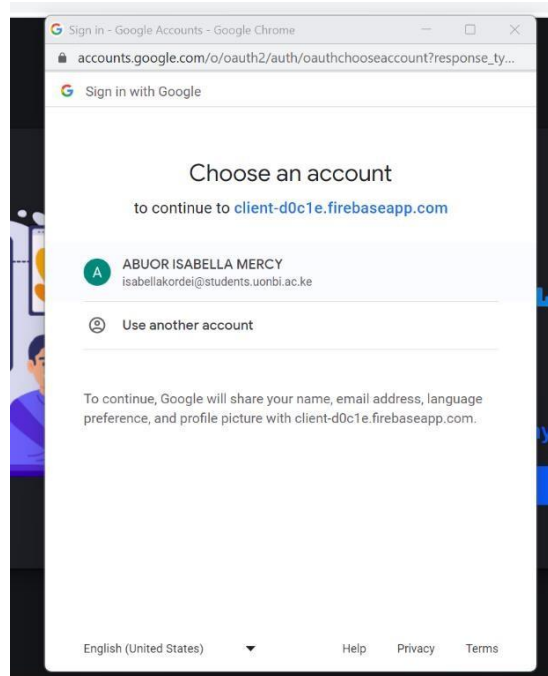


Figure 27: Google Login Popup

After the user has logged in, he/she is directed to the dashboard page where they are able to either create a new meeting, view the meeting they have created or the ones they have been invited two.

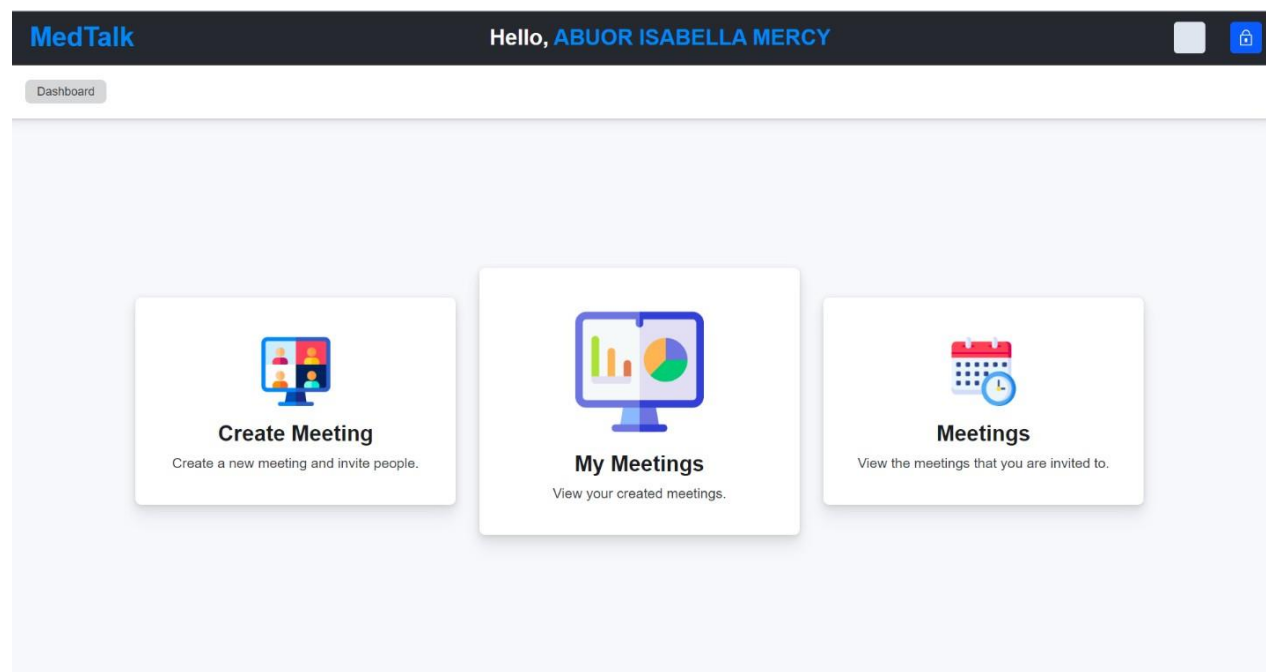


Figure 28: Picture of the Dashboard

In the create meeting page, the user is provided with a form to fill in the name, date, and users they want to invite. This system only allows one-to-one conversation hence both the sender and the receiver must be registered in the system.

Figure 29: Picture of the create meeting form.

Once the user clicks on create meeting they will be redirected to the dashboard and they can click on the My meetings page to be able to view the created meeting in they created.

Meeting Name	Meeting Date	Status	Edit	Copy...
ENT 2pm Consultation	06/02/2023	Join Now		
Demo1	06/23/2023	Upcoming		
Testing	06/02/2023	Cancelled		
Abc Therapy Session	06/02/2023	Join Now		
Consultation 1	05/31/2023	Ended		
test32	06/02/2023	Join Now		
Id24456	06/02/2023	Cancelled		
Id345657	06/02/2023	Join Now		
Test89	06/12/2023	Upcoming		
Consultation85	06/02/2023	Join Now		

Figure 30: Picture of My Meetings page

In my Meetings page, one can edit the meeting if and only if the meeting has not ended or been cancelled already.

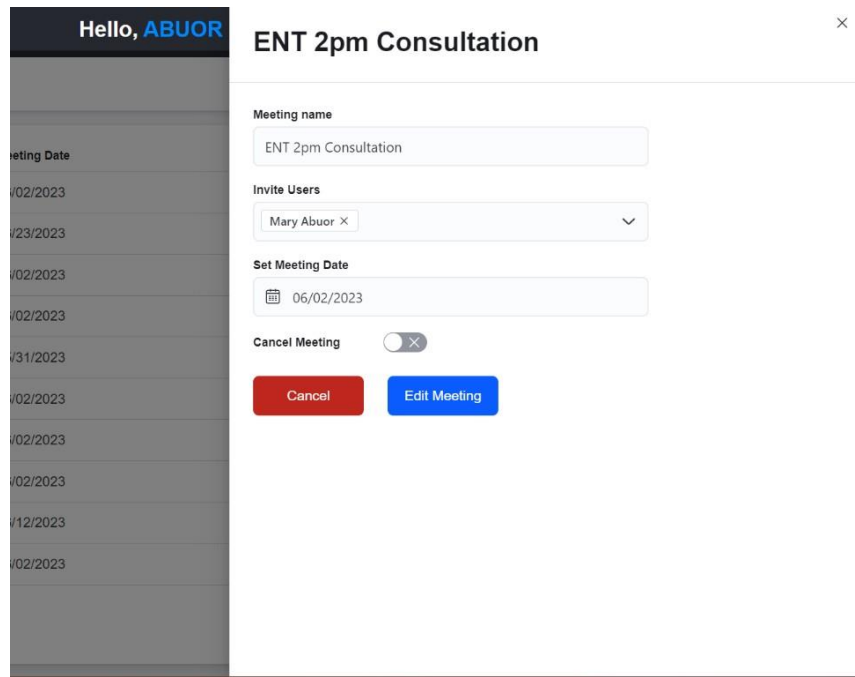
The image shows a mobile application interface. On the left is a dark sidebar with a 'Hello, ABUOR' header and a list of meeting dates. The main area displays a modal titled 'ENT 2pm Consultation' with a close button (X) in the top right. The modal contains the following fields: 'Meeting name' with the text 'ENT 2pm Consultation'; 'Invite Users' with a dropdown menu showing 'Mary Abuor X'; 'Set Meeting Date' with a calendar icon and the date '06/02/2023'; and a 'Cancel Meeting' toggle switch which is currently turned off. At the bottom of the modal are two buttons: a red 'Cancel' button and a blue 'Edit Meeting' button.

Figure 31: Picture of the Edit Functionality

The user can also copy the meeting link and share with one person through other communication browser if they would like.

If the user clicks on the Join Now button on the status section, they will be directed to the call homepage where they are able to change settings to their preferred liking.

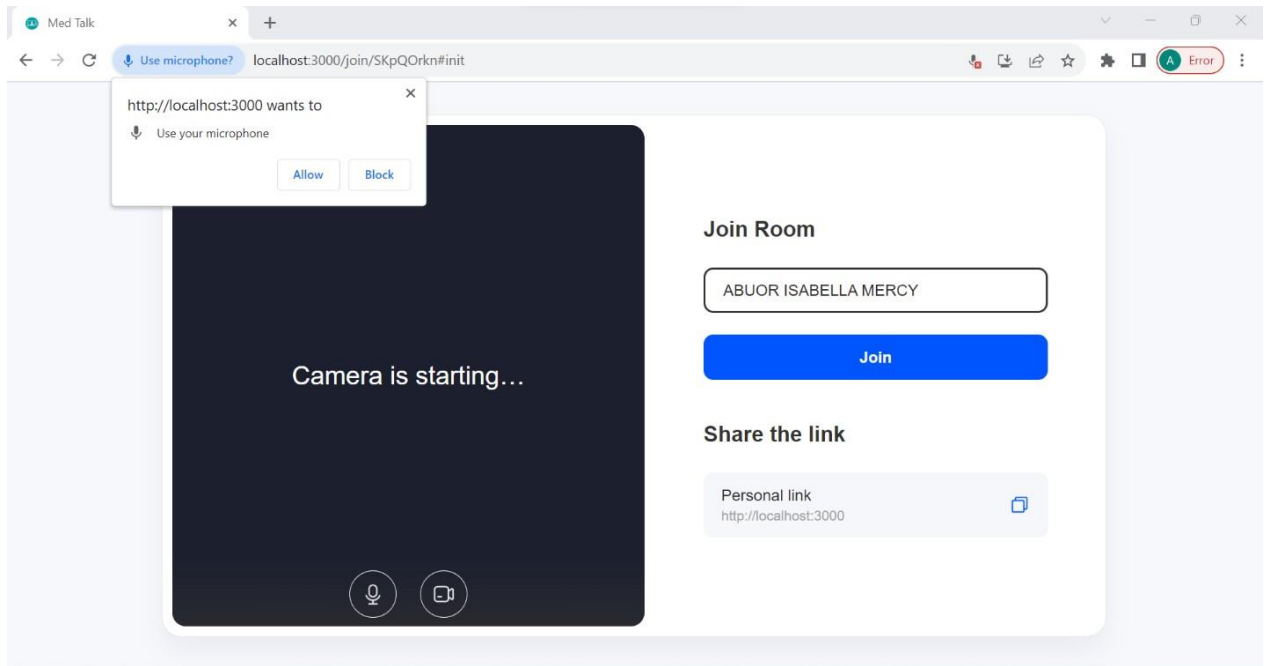


Figure 32: Picture of the call homepage

In the call page the user can converse with the intended receiver privately.

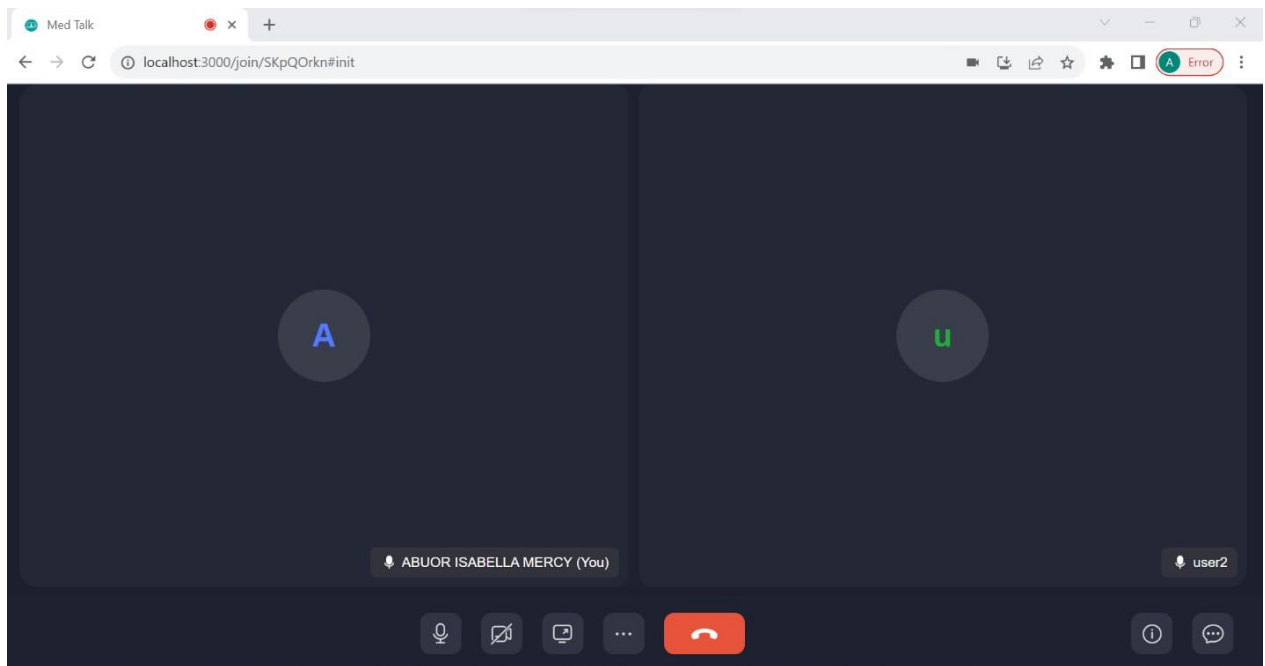


Figure 33: Picture of the Call page room.

APPENDIX B: Sample Code

Sample of the WebRTC Code Functionality:

```
MedTalk2 > client > src > pages > JoinMeeting > JS JoinMeeting.js > JoinMeeting > initWebRTC > then() callback
73 const initWebRTC = () => {
74   navigator.mediaDevices.getUserMedia({
75     video: true,
76     audio: true
77   })
78   .then((stream) => {
79     setStreamObj(stream);
80
81     myVideo.current.srcObject = stream;
82
83     peer = new Peer({
84       initiator: isAdmin,
85       trickle: false,
86       stream: stream,
87     });
88
89     if (!isAdmin) {
90       getReceiverCode();
91     }
92
93     peer.on("signal", async (data) => {
94       if (isAdmin) {
95         let payload = {
96           id,
97           signalData: data,
98         };
99         await postRequest(`${BASE_URL}${SAVE_CALL_ID}`, payload);
100       } else {
101         socket.emit("code", { code: data, url }, (cbData) => {
102           console.log("code sent");
103         });
104       }
105     });
106   });
107 }
```

Figure 34: Sample Code for the WebRTC

```
MedTalk2 > JS server.js > ...
1  require("dotenv").config();
2  const express = require("express");
3  const http = require("http");
4  const bodyParser = require("body-parser");
5  const cors = require("cors");
6  const port = process.env.PORT;
7  const app = express();
8  const server = http.createServer(app);
9  const Routes = require("./app/routes");
10
11 app.use([
12   cors({
13     origin: "http://localhost:3000",
14     methods: ["GET", "POST"],
15     // allowedHeaders: ["Content-Type", "Authorization"],
16     // credentials: true
17   }),
18   bodyParser.json(),
19   bodyParser.urlencoded({extended: false}),
20   Routes
21 ]);
22
23 const io = (module.exports.io = require('socket.io')(server, {
24   cors: {
25     origin: '*',
26   }
27 }));
28
29 // const io = require ("socket.io")(server);
30 const socketManager = require ("./app/socketManager");
31
32 io.on("connection", socketManager);
33
34 server.listen(port, () => {
```

Figure 35: Sample of the Server-side code

```

public class AesEncryption {
    private static final String ALGORITHM = "AES";
    private static final String TRANSFORMATION = "AES";

    public static void encrypt(String key, File inputFile, File outputFile)
        throws CryptoException {
        doCrypto(Cipher.ENCRYPT_MODE, key, inputFile, outputFile);
    }

    public static void decrypt(String key, File inputFile, File outputFile)
        throws CryptoException {
        doCrypto(Cipher.DECRYPT_MODE, key, inputFile, outputFile);
    }

    private static void doCrypto(int cipherMode, String key, File inputFile,
        File outputFile) throws CryptoException {
        try {
            Key secretKey = new SecretKeySpec(key.getBytes(), ALGORITHM);
            Cipher cipher = Cipher.getInstance(TRANSFORMATION);
            cipher.init(cipherMode, secretKey);

            FileInputStream inputStream = new FileInputStream(inputFile);
            byte[] inputBytes = new byte[(int) inputFile.length()];
            inputStream.read(inputBytes);

            byte[] outputBytes = cipher.doFinal(inputBytes);

            FileOutputStream outputStream = new FileOutputStream(outputFile);
            outputStream.write(outputBytes);

            inputStream.close();

```

Figure 36: Sample code of the AES encryption.