

Software Engineering Department  
Braude College

Capstone Project Phase A – 61998

**Global Communication System**

**24-2-R-21**

**Advisors:**

**Dr. Dan Lemberg,** [**lembergdan@braude.ac.il**](mailto:lembergdan@braude.ac.il)

**Mrs. Elena Kramer,** [**elenak@braude.ac.il**](mailto:elenak@braude.ac.il)

**Students:**

**Razi Mograbi,** [**razi.mograbi@e.braude.ac.il**](mailto:lembergdan@braude.ac.il)

**Dolev Seri,** [**Dolev.Seri@e.braude.ac.il**](mailto:lembergdan@braude.ac.il)

**Contents**

1. Abstract………………………………………………………………………………………………………………………………………4

[1. Introduction 4](#_Toc177843731)

[2. Related Work 5](#_Toc177843732)

[3. Background 5](#_Toc177843733)

[3.1 AI-Powered Contextual Translation 6](#_Toc177843734)

[3.2 Real-Time Communication (RTC) 6](#_Toc177843735)

[3.2.1 RTC for Live Audio Streaming 6](#_Toc177843736)

[3.3 Speech-to-Text Conversion with Whisper API 6](#_Toc177843737)

[3.3.1 Features of Whisper API 6](#_Toc177843738)

[3.3.2 Handling Multilingual Speech Recognition 7](#_Toc177843739)

[3.4 ChatGPT-4 API 7](#_Toc177843740)

[3.5 Google Cloud (GCP) 7](#_Toc177843741)

[3.6 Voice Conversion 7](#_Toc177843742)

[3.7 Voice Separation 7](#_Toc177843743)

[3.8 Text-to-Speech (TTS) Using OpenAI 8](#_Toc177843744)

[3.8.1 OpenAI TTS for Voice Output 8](#_Toc177843745)

[3.9 Flutter for App Development 8](#_Toc177843746)

[3.9.1 Cross-Platform Development with Flutter 8](#_Toc177843747)

[3.9.2 Benefits for Real-Time Applications 8](#_Toc177843748)

[4. Expected Achievements 9](#_Toc177843749)

[4.2 Unique Features 9](#_Toc177843750)

[4.2.1 Speech to Speech Translation 9](#_Toc177843751)

[4.2.2 Voice Personalization 10](#_Toc177843752)

[4.2.3 Stream-Based Communication System 10](#_Toc177843753)

[4.2.4 User-Friendly Interface 10](#_Toc177843754)

[4.3 Criteria for Success 10](#_Toc177843755)

[5. The Process 11](#_Toc177843756)

[5.1 Research - Speech-to-Speech Translation 11](#_Toc177843757)

[5.1.1 Constraints and Challenges 11](#_Toc177843758)

[5.1.2 Conclusion 12](#_Toc177843759)

[5.2 Research - Voice Conversion and Cloning 12](#_Toc177843760)

[5.2.1 Constraints and Challenges 13](#_Toc177843761)

[5.2.2 Conclusion 13](#_Toc177843762)

[5.3 Research Noise Suppression and Voice Separation 13](#_Toc177843763)

[5.3.1 Constraints and Challenges 14](#_Toc177843764)

[5.3.2 Conclusion 14](#_Toc177843765)

[5.4 Methodology and Development Process 14](#_Toc177843766)

[6. Product 15](#_Toc177843767)

[6.1 Requirements 15](#_Toc177843768)

[6.1.1 Functional 15](#_Toc177843769)

[6.1.2 Non-functional 16](#_Toc177843770)

[6.2 Architecture overview 16](#_Toc177843771)

[6.2.1 GlobalCommunicationapplication 17](#_Toc177843772)

[6.2.2 Screens 18](#_Toc177843773)

[6.3 Diagrams 21](#_Toc177843774)

[6.3.1 Use case: 21](#_Toc177843775)

[6.3.2 Activity Diagram: 22](#_Toc177843776)

[6.3.3 Sequence Diagram 22](#_Toc177843777)

[7.Verification and Evaluation 23](#_Toc177843778)

[7.1 Evaluation 23](#_Toc177843779)

[7.2 Verification 23](#_Toc177843780)

[7.2.1 Testing 23](#_Toc177843781)

[8. References 24](#_Toc177843782)

#### Abstract

Today, the human population is growing rapidly, making the world more connected than ever before while increasing the chances of needing to communicate in foreign languages. Modern methods of translation can be slow and error prone, especially in real-time scenarios. This highlights the urgent need for innovative solutions that enable real-time multilingual communication. As a result of advanced AI models like ChatGPT, we now have the capability to translate speech in near real time across languages. Our project will use the latest AI technologies to build a communication app for both Android and iOS, designed to eliminate language barriers. The project will allow only two people to communicate in different languages in near real- time. While communicating, we will save the context to ensure better translations. We will combine the latest speech recognition, translation, and voice Cloning to create a system where people can communicate naturally using their own languages. The result is a tool that not only translates in near real-time but also maintains each speaker's unique voice. The applications for this technology are wide-ranging, from changing day to day interactions between global citizens to military communication where language barriers have long required skilled interpreters to keep up with the real-time translation. This project goes beyond just making communication easier for everyone. It sets a new bar for how we connect across the globe, bringing us closer to a world where language doesn't get in the way of people understanding each other and working together.

# 1. Introduction

Today, there are approximately 8 billion people living in the world, which has significantly increased the demand for trade and communication between them. Also, with the raise of the Internet the global society became more interconnected, making the ability to communicate across different languages more crucial than ever. From sealing big business deals overseas to smoothing out talks between countries, or just catching up with loved ones abroad, being able to chat easily across languages is key. However, current translation methods often struggle to meet the demands of our fast-paced world. They tend to be slow, inaccurate, and handle emotional conversations poorly, as context is often lost, frequently missing the finer points and subtle meanings in a speaker's message. As a result, businesses are often forced to hire multiple translators to communicate with foreign partners, leading to increased costs and inefficiencies. Many are looking for new solutions[[1]](#footnote-1) that can overcome these issues and provide a more natural feel to conversations between people who speak different languages. The goal is to make cross-language communication as smooth and effective as possible and to preserve the essence of what's being said. To combat these challenges, our project will use advanced AI technologies to develop a communication platform that allows people to converse in their native languages, with each participant hearing the other’s voice translated into their own language. This system not only translates speech in real time but also preserves the speaker's unique vocal characteristics, for a more natural experience. This new approach has the potential to revolutionize how people around the world talk to each other. It breaks down language barriers, helping people connect and work together more easily.

# 2. Related Work

In recent years, there has been significant progress in the development of technologies that enhance real-time communication across languages. These advancements include speech recognition, machine translation, noise suppression, and voice conversion—each playing a crucial role in making conversations more natural and seamless, even when users speak different languages. Speech recognition has become a core component in enabling real-time conversations. By converting spoken words into text, speech recognition systems allow communication to flow smoothly between speakers of different languages. Research has shown that modern speech recognition systems can handle a wide range of languages and accents, making them highly effective in real-time applications where accuracy and speed are critical[[2]](#footnote-2). However, capturing speech is just one part of the process; translating that text into another language is equally important. Machine translation (MT) systems enable automatic language translation, allowing users to communicate seamlessly across language barriers. With advancements in language models, translation systems can handle more complex language structures and provide accurate translations in real-time. These systems are essential for maintaining natural, fluid conversations between users who speak different languages[[3]](#footnote-3). In addition to recognizing and translating speech, maintaining the clarity of communication is crucial, especially in noisy environments. Noise suppression technologies are designed to reduce background noise, ensuring that conversations remain clear, even in challenging environments. Recent advancements in noise suppression have made it easier for users to communicate without interference, which is especially important in real-time translation systems where clear audio is vital for accurate recognition and translation[[4]](#footnote-4). Voice conversion technologies have been developed to ensure that the speaker’s voice remains recognizable, even when their speech is translated into another language. This helps maintain the naturalness of the conversation, allowing users to hear the translated message in a voice that still sounds familiar. Research has demonstrated that voice conversion systems can adapt to different languages while preserving the unique qualities of the speaker’s voice, making conversations feel more personal and authentic[[5]](#footnote-5). Existing tools like Google Interpreter Mode and Skype Translator provide real-time translation but often fail to maintain the speaker’s original voice, which can feel impersonal. Our app aims to address this by integrating voice conversion technology, with the goal of ensuring that translated speech retains the speaker’s voice. By incorporating technologies such as speech recognition, machine translation, noise suppression, and voice conversion, our app is designed to create a real-time translation system that allows users to communicate naturally, regardless of the language they speak. By leveraging these advancements, we hope to overcome many of the challenges that have traditionally hindered cross-language communication, offering a more fluid and immersive experience for users.

# 3. Background

This section provides a description of the technologies, methodologies, and challenges involved in developing a real-time multilingual communication app with voice cloning capabilities.

## 3.1 AI-Powered Contextual Translation

AI-powered contextual translation refers to the use of artificial intelligence to translate text while maintaining the overall context of the conversation or document. This method employs AI models that focus on understanding and preserving meaning, intent, and nuance, rather than just translating words or phrases in isolation.

AI-powered contextual translation goes beyond simple word or phrase translation by considering the broader conversation or document context. Models like OpenAI’s ChatGPT-4 are designed to grasp the intent behind sentences and account for linguistic subtleties, idiomatic expressions, and conversational flow. By using advanced natural language processing (NLP) techniques, this method produces translations that are more human-like, maintaining accuracy and coherence, even in complex or idiomatic language scenarios.

## 3.2 Real-Time Communication (RTC)

Real-Time Communication (RTC) refers to technologies that enable the instantaneous transmission of data, audio, and video between users with minimal delay. RTC is crucial for applications where immediate interaction is required, such as video conferencing, voice calls, and online gaming. It relies on protocols designed to maintain low latency and high reliability, ensuring that data is exchanged in real-time between users.

### 3.2.1 RTC for Live Audio Streaming

RTC enables live audio streaming by providing a direct, low-latency communication channel between two or more parties. This allows for seamless conversations without interruptions, making it essential for real-time voice communication applications.

## 3.3 Speech-to-Text Conversion with Whisper API

speech-to-text conversion is a critical component in modern communication systems, enabling the transformation of spoken language into text. OpenAI's Whisper API offers advanced capabilities in this area by providing accurate and efficient transcription of audio into text, even in noisy environments or with diverse accents. Whisper uses state-of-the-art models that support a wide range of languages, making it a powerful tool for multilingual speech recognition.

### 3.3.1 Features of Whisper API

The Whisper API provides high accuracy in speech recognition, supports multiple languages, and performs well in challenging environments with background noise or varying speaker accents. It is essentially an encoder-decoder Transformer that has been trained on 680,000 hours of multilingual and multitask supervised data[[6]](#footnote-6).

### 3.3.2 Handling Multilingual Speech Recognition

Whisper API is designed to handle various languages and dialects, making it effective for global applications where users may speak different languages. Its ability to recognize and transcribe multilingual speech ensures accessibility across diverse language groups.

## 3.4 ChatGPT-4 API

The ChatGPT-4 API by OpenAI is an advanced language model designed for natural language understanding and generation. It excels in translating text between languages, maintaining context, and capturing nuances in communication. Unlike traditional translation tools, ChatGPT-4 uses deep learning to provide more accurate and context-aware translations. It can handle complex sentence structures and idiomatic expressions, ensuring fluid and natural translations in various languages.

## 3.5 Google Cloud (GCP)

Google Cloud (GCP) is a cloud computing platform by Google that provides a range of services, including computing, storage, machine learning, and big data analysis. It offers scalable tools for building, deploying, and managing applications, with support for frameworks like TensorFlow and access to powerful resources like GPUs and TPUs for efficient training and deployment of neural networks.

## 3.6 Voice Conversion

Voice conversion is a technique in speech processing where the voice characteristics (such as timbre, pitch, or accent) of a speaker are altered to sound like another speaker, while retaining the original linguistic content. In this process, the system modifies the audio signal of one speaker to match the vocal traits of a target speaker without changing the words or meaning of the spoken content.

The process typically involves using machine learning models to analyze and map the acoustic features of one speaker’s voice to another. This mapping adjusts factors like pitch, tone, and cadence to ensure that the output sounds like the target speaker, even though the original speaker’s words are retained.

## 3.7 Voice Separation

Voice Separation is a machine learning technique that isolates a specific speaker's voice from an audio recording containing multiple overlapping voices or background noise. Voice separation models, such as those described in VoiceFilter-Lite by Google[[7]](#footnote-7), utilize techniques like speaker-conditioned neural networks to isolate a target speaker’s voice from overlapping audio. These models typically rely on features like d-vectors, which represent the target speaker, to filter out other voices and noise while maintaining performance in various acoustic conditions​.

## 3.8 Text-to-Speech (TTS) Using OpenAI

### 3.8.1 OpenAI TTS for Voice Output

The OpenAI TTS API enables the generation of realistic, human-like speech from text inputs. It supports a wide range of languages and can handle different accents, providing a natural voice experience. The API is designed for efficiency and accuracy, making it suitable for real-time applications.

## 3.9 Flutter for App Development

Flutter is an open-source UI software development kit (SDK) created by Google for building cross-platform applications. It allows developers to create natively compiled applications for mobile (iOS, Android), web, and desktop from a single codebase. The key advantage of Flutter is its ability to deliver high-performance apps with a consistent user interface across different platforms, reducing development time and costs.

### 3.9.1 Cross-Platform Development with Flutter

Flutter’s cross-platform capabilities make it a popular choice for developers who need to deploy applications across multiple devices. By using a single codebase, developers can create apps that work on Android, iOS, web, and desktop without having to write separate versions for each platform. This drastically reduces the time and resources needed for development and maintenance. Flutter also offers a rich set of pre-designed widgets that make it easy to create custom UIs that work consistently across different screen sizes and devices.

### 3.9.2 Benefits for Real-Time Applications

Flutter’s real-time capabilities are enhanced by its fast rendering engine, which delivers high performance even in complex, interactive applications. The framework provides smooth animations and rapid response times, making it ideal for applications that require real-time user interactions. Additionally, Flutter supports integration with various APIs and real-time communication protocols, ensuring that developers can build feature-rich, responsive applications. Its hot-reload feature allows for quick iterations during development, enabling faster testing and debugging for real-time features.

A screenshot of a computer program

Description automatically generated

Figure 1: Fluter Architectural layers

# 4. Expected Achievements*ראש הטופס*

Outcomes:

We expect our project to deliver a mobile app that provides near real-time voice translation, enabling users to communicate across language barriers with minimal delays. While achieving true real-time translation presents challenges due to the complexity of various technical components, our app will aim to optimize these processes to create a natural and smooth conversation flow that feels intuitive and effortless. The app will feature a user-friendly interface designed with accessibility and ease of use in mind, ensuring that users of varying technical skill levels can comfortably navigate and utilize its features. Whether used in casual settings or business environments, the app will provide a seamless experience for a wide range of users. To make the interactions feel more personalized, we will incorporate voice cloning technology, which ideally will help the translated speech retain the original speaker’s unique voice characteristics, for supported languages. This will foster a more natural communication experience, allowing users to feel more connected during conversations. Our ultimate goal is to deliver a solution that empowers users to communicate confidently, minimizing the discomfort caused by language differences.

## 4.2 Unique Features

### 4.2.1 Speech to Speech Translation

A cornerstone of our app is the Speech to Speech Translation feature, which enables users to communicate across languages by translating speech in near real-time. To create this feature, we will need to integrate and fully understand multiple technologies, including noise suppression and separation, speech-to-text (STT), translation, and text-to-speech (TTS) APIs. Our aim is to combine these components and optimize their performance to achieve near real-time voice translation.

### 4.2.2 Voice Personalization

Another key feature of our app is Voice Personalization through voice cloning and conversion, which ensures that the speaker's unique voice characteristics are retained, even after translation. To achieve this, we will use voice cloning and conversion technologies to replicate the user's voice and adapt it to the translated output. This ensures that the translated speech closely resembles the original speaker's voice, making the conversation feel personal and natural. One challenge is obtaining clean and high-quality voice samples from the user, as this is crucial for producing a reliable and consistent cloned voice.

### 4.2.3 Stream-Based Communication System

A key aspect of our app will be the stream-based communication system, which is intended to allow for continuous, fluid conversations between users, despite slight delays experienced in the translation process. Our goal is to design a system that enables both sides to take turns naturally, with only brief waiting times for the translated content to be delivered, rather than interrupting the conversation with lengthy pauses. Although we aim to simulate real-time conversation as closely as possible, we anticipate small delays as the translation and voice cloning processes occur. Each user will briefly wait for the translated content to arrive before responding, but we expect the conversation flow to remain smooth, making it feel as natural as possible.

### 4.2.4 User-Friendly Interface

One important consideration is ensuring that our app is viable for a wide range of users. To achieve this, we will develop a User-Friendly Interface designed with simplicity and functionality in mind. Our goal is to create an interface that is intuitive and easy to navigate, making it accessible to users regardless of their technical experience. The interface will prioritize clarity and ease of use, similar to popular phone call apps. By following a familiar design, users will intuitively know how to navigate the app and engage with its features, allowing them to focus on communication rather than the technology itself. By maintaining a simple layout and providing clear visual cues, we aim to minimize the learning curve and ensure that users can quickly and effectively use the app.

## 4.3 Criteria for Success

1. A final product that delivers near real-time voice translation.
2. A cloned voice that ideally retains the speaker's unique characteristics
3. A user-friendly interface that is easy to navigate for a wide range of users.
4. Seamless integration of noise suppression, translation, voice cloning, and user interaction components.
5. Train and develop Optimized stream- based communication for smooth and continuous conversation flow.

# 5. The Process

## 5.1 Research - Speech-to-Speech Translation

In our first step toward creating an effective translation app, we asked ourselves two essential questions:

* what components would our app need to successfully handle speech-to-speech translation?
* what platform would ensure the best user experience?

Based on our advisor's suggestion, we decided to leverage OpenAI’s ChatGPT for translation, as it is widely regarded as one of the most advanced tools available, offering both accuracy and versatility. One of its key strengths is its ability to support a wide range of languages, making it an excellent choice for global communication. Additionally, it offers a well-structured API that allows for smooth integration. After thorough consideration, we turned our focus to two key technologies to complement ChatGPT: OpenAI’s Whisper and Text-to-Speech (TTS). Whisper is a powerful Speech-to-Text (STT) system capable of converting spoken language into text with high precision across various languages. Integrating Whisper and TTS with ChatGPT created a seamless pipeline, ensuring that each step—from transcription to translation to speech synthesis—works efficiently within our app’s architecture. When considering platform, we realized that the most accessible solution for users would be personal smartphones paired with earphones. Smartphones are widely used, making the app accessible to many people, while earphones are recommended to improve both voice input and output. Using earphones helps reduce background noise and ensures clearer audio capture for accurate transcription and translation, which greatly enhances the user experience.

### 5.1.1 Constraints and Challenges

In our research for the speech-to-speech translation, several technical constraints and challenges have emerged. One of the major challenges is achieving low-latency performance across the entire translation pipeline. Since our app is designed for conversational interactions, minimizing delays is crucial to maintaining a smooth experience. The time it takes for each component—Speech-to-Text (STT) using Whisper, translation via ChatGPT, and Text-to-Speech (TTS)—to process and deliver results can add up, leading to a noticeable delay. Reducing this latency is essential for keeping conversations natural. Another challenge lies in learning how to integrate and use multiple APIs effectively within our app. While Whisper, ChatGPT, and TTS are highly efficient, integrating them into a seamless workflow requires an understanding of how to use their functionalities correctly. This process can be difficult, but it's crucial for delivering a low-latency, cohesive user experience.

### 5.1.2 Summary

From our research, we have identified several strategies to overcome the challenges faced in building our speech-to-speech translation app. To address latency, we will use chunk-based processing, where audio is divided into smaller segments and processed progressively. This approach will help reduce overall delay by allowing translations to happen in parallel with ongoing audio input, making the conversation feel more responsive and natural. Additionally, we plan to implement a sliding window mechanism to ensure that context is preserved across chunks, enhancing the accuracy of translations. Additionally, Integrating and using multiple APIs, such as Whisper, ChatGPT, and TTS, is simplified by the fact that these APIs are well-documented and widely supported by extensive online guides, which will aid our development process.

## 5.2 Research - Voice Conversion and Cloning

In our research on voice conversion, we found that this process, commonly performed using neural networks, offers the ability to transform one person's voice to sound like another’s. Voice conversion has potential applications in creating more personalized and immersive user experiences by maintaining the characteristics of the original speaker while changing the spoken language. As we searched for suitable methods, we identified several options, such as FreeVC and SpeechT5-VC, that could provide voice conversion functionality. However, both options showed a common issue of high latency , which may affect the fluidity of real-time communication. This concern requires further testing during our app development to decide whether to integrate voice conversion into the final product. The decision will hinge on balancing the potential advantages of voice conversion against the increased latency it introduces. After identifying potential latency issues with voice conversion methods, we turned to voice cloning as an alternative approach. Voice cloning offers a more efficient solution for retaining a user's unique vocal characteristics, even across different languages. During our research, we encountered various voice cloning models, but many were either too costly or lacked efficiency. Our search eventually led us to Coqui AI models, particularly xTTS. XTTS was trained in 17 languages and achieved state-of-the-art (SOTA) results in most of them, making it an excellent choice for multilingual applications. One of the strengths of xTTS lies in its use of Zero-Shot Text-to-Speech (ZS-TTS) technology, which enables the system to generate speech for previously unseen speakers with minimal data, allowing for efficient voice cloning with only a few seconds of audio samples[[8]](#footnote-8). Furthermore, for the languages that xTTS supports, we can use it as a TTS solution, eliminating the need for an additional API call to another service, which can help save costs. Although our voice cloning model is a zero-shot model, we plan to train it further to improve the overall quality of the voice output for each user. This process involves collecting a good number of high-quality voice samples from each user, ensuring the cloned voice retains its authenticity and clarity.

### 5.2.1 Constraints and Challenges

During our evaluation of voice conversion and voice cloning technologies, several challenges have surfaced. As mentioned earlier, one of the main issues with voice conversion is its high latency, which could significantly impact the app's real-time performance. Although voice cloning is said to be faster, it can still introduce latency, particularly when adapting it to meet real-time streaming requirements. Both processes may therefore pose a challenge to maintaining fluid conversations, and further optimization will be required to minimize delays. Another limitation specific to voice cloning is that it does not support all languages, which adds complexity to the app's implementation. Different solutions may be needed depending on the language being processed, increasing the development workload. In both cases, another challenge is the collection of high-quality voice samples for training. Determining the optimal sample size and ensuring the quality of these samples is critical to achieving effective results. This process could be time-consuming, and even with training, improvements may not always be consistent.

### 5.2.2 Summary

After discussing the potential challenges, we decided that we will evaluate voice conversion during the development phase to determine whether it can function effectively within the app. This testing will help us assess whether the latency introduced by voice conversion is manageable and if the feature can be integrated without disrupting the user experience. To address the latency introduced by the voice cloning process, we will adapt the same strategy discussed in the speech-to-speech translation section—using chunk-based processing to minimize delays and improve responsiveness. Additionally, since xTTS does not support all languages, we will rely on OpenAI’s TTS for those languages, ensuring smooth transitions between supported and unsupported languages. From our research, we have also learned that many people have attempted to train similar models, and there are several resources, including videos, that provide guidance on how to set up and train models effectively. These resources show promising results, but we are also aware that achieving the desired quality in some cases may be challenging, despite training efforts.

## 5.3 Research Noise Suppression and Voice Separation

As we advanced in our project, we explored the current methods available for improving audio quality, particularly focusing on noise suppression and voice separation. In our research, we found that many modern approaches rely on AI and neural network models. For noise suppression, we considered models like RNNoise, which use a combination of traditional Digital Signal Processing (DSP) and deep learning. This model shows potential for real-time applications while aiming to keep computational complexity manageable. For voice separation, we looked into models such as SVoice, which uses neural networks to separate voices in environments with multiple speakers. These models aim to identify and separate individual voices, even when the exact number of speakers is unknown​ svoice[[9]](#footnote-9). We also explored SpeechBrain, an open-source toolkit that provides a variety of pre-built models for speech processing tasks, including both noise suppression and voice separation. SpeechBrain offers flexibility and modularity, which could be useful for testing different configurations and improving audio quality in our app.

### 5.3.1 Constraints and Challenges

While we explored several models for noise suppression and voice separation, a key challenge is the limited real-world testing and implementation. Models like RNNoise, though reliable, are older and may not perform optimally for modern applications. Newer models like SVoice also lack extensive deployment, making it harder to predict their real-world effectiveness, particularly in more complex systems like our app. Additionally, the processed audio from these models might not be ideal for Whisper’s speech-to-text accuracy or for producing high-quality voice samples. This could lead to transcription errors or affect voice cloning performance. Moreover, these models may require additional training to perform effectively, which could pose time and resource challenges. Ensuring real-time performance is another uncertainty, as processing complex audio inputs without introducing delays remains difficult.

### 5.3.2 Summary

Some of the challenges posed by the models we explored might be addressed with SpeechBrain, a newer tool with strong documentation that could assist in both noise suppression and voice separation. However, further testing will be necessary to determine which model or combination of models—whether it's SpeechBrain, SVoice, RNNoise, or others—best fits our needs. Whisper's robustness against background noise may allow us to use lighter noise suppression, but voice separation could still pose a challenge that requires additional testing during development. Ultimately, we will need to carefully assess all these tools after development and testing to decide how to integrate them effectively into our app.

## 5.4 Methodology and Development Process

For the development of our speech-to-speech translation app, we have chosen the Agile methodology due to its flexibility and iterative nature. This approach allows us to develop complex components—such as Whisper-based speech recognition, OpenAI-powered translation, and voice cloning—in manageable sprints. Agile ensures that we can test and improve each feature progressively, minimizing issues during integration. By continuously refining the app based on feedback, we can achieve smoother functionality and better user experience while accommodating changes in technology and requirements.Our app’s development will progress through the following key phases:

1. UI and Database setup: Set up the user interface and database to store user preferences, forming the foundation for connecting future components.
2. Basic server setup: Set up a basic server for handling Rest API.
3. Speech-to-Text (STT) Integration: Integrate Whisper to convert audio into text, creating the backbone of the speech-to-speech translation process.
4. Translation Integration: Integrate OpenAI’s ChatGPT API to handle text translation for accurate multilingual communication.
5. Text-to-Speech (TTS) Integration: Integrate OpenAI TTS or xTTS to convert translated text back into speech.
6. Noise Suppression and Voice Separation: Test models for noise suppression and voice separation to ensure clear input and improve transcription accuracy.
7. Integrate Noise Suppression and Voice Separation if performs well: we will deploy the Voice Separation model to google cloud (GCP) and integrate it into our server.
8. Experimental Voice Conversion and Cloning: Conduct training and evaluation of voice conversion and cloning models, focusing on their latency and performance after training.
9. Iterative Testing and Integration: After developing each component, test it in isolation, integrate it with the rest of the system, and refine the app through feedback loops.
10. Deploy Server on Google cloud (GCP).

# 6. Product

## 6.1 Requirements

### 6.1.1 Functional

|  |  |
| --- | --- |
| ID | Description |
| 1 | The system shall allow users to create and manage accounts, including sign-up, login, and logout functionalities. |
| 2 | The system shall enable users to make and receive calls to communicate with other users. |
| 3 | The system shall capture and convert spoken language to text efficiently. |
| 4 | The system shall translate text from the user's input language to the recipient's output language in a fast and efficient manner. |
| 5 | The system shall convert translated text back into speech for the recipient. |
| 6 | The system shall retain the original speaker's voice characteristics in the translated speech after sufficient voice samples have been collected and processed. |
| 7 | The system shall apply noise suppression to ensure clear and accurate audio input. |
| 8 | The system shall allow users to select and store their language preferences for future sessions. |
| 9 | The system shall support continuous and low-latency audio streaming between users, providing a smooth communication experience. |
| 10 | The system shall allow users to add and save friends for easy communication access. |
| 11 | The system shall store user data, including language preferences, voice samples, and friend lists, to optimize future sessions based on user preferences. |

### 6.1.2 Non-functional

|  |  |
| --- | --- |
| ID | Description |
| 1 | The system shall provide an intuitive and user-friendly interface, minimizing the learning curve for new users and ensuring ease of use. |
| 2 | The system shall perform efficiently, providing fast translations and communication to maintain a smooth user experience. |
| 3 | The system shall operate reliably, producing consistent and accurate translations for the provided inputs. |
| 4 | The system shall allow for future enhancements, with a well-organized and modular codebase to facilitate updates and new features. |
| 5 | The system shall support both iOS and Android platforms, ensuring cross-platform compatibility for users. |
| 6 | The system shall handle and display errors gracefully, ensuring that users are informed of issues without disruption to their experience. |

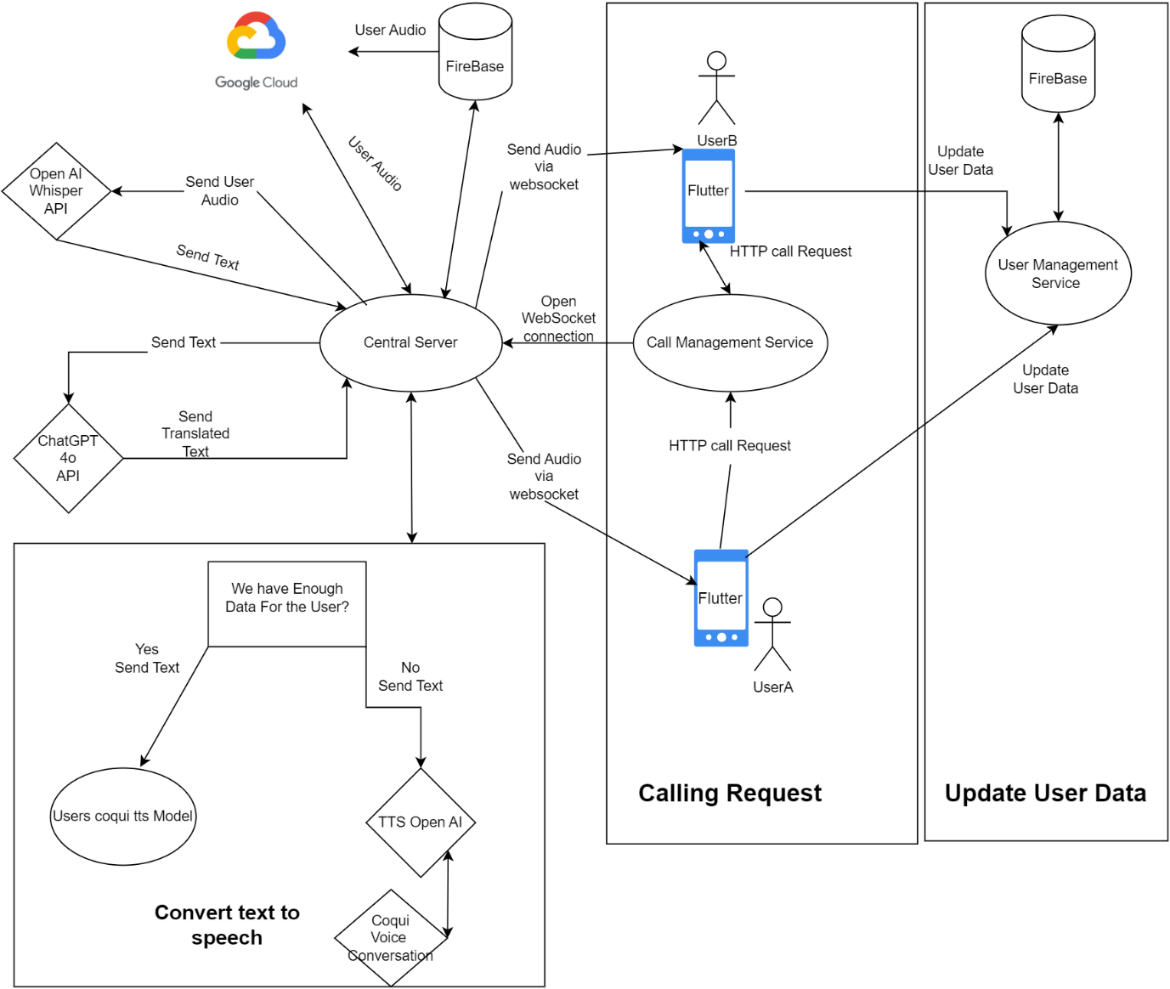
## 6.2 Architecture overview

Our architecture consists of several key components:

* **Call Management Service:** Its part of a server that is deployed to Google Cloud(GCP) that handles call requests between users It communicates through HTTP requests to initiate the call. Finally, it opens WebSocket connections for near real-time voice communication.
* **Central Server:** Its a server deployed to Google Cloud(GCP) that manages WebSocket connections for sending and receiving audio between the users and manages all the API calls.
* **Firebase:** for storing user audio samples for voice cloning training model. + users data.
* **Convert Text to Speech Module:** Coqui TTS: If enough voice data is available for the user, Coqui TTS generates the translated speech in the user's cloned voice.

OpenAI TTS + Coqui Voice Conversion: If there is insufficient voice data, OpenAI TTS generates speech, and Coqui's voice conversion adjusts it to match the user’s voice.

* **Google Cloud(GCP):** Saves the Voice Separation Model.

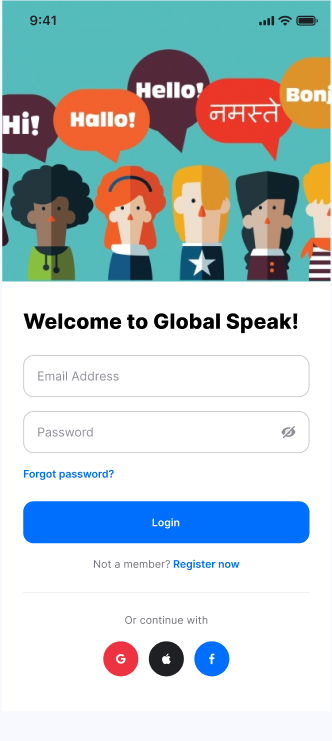


### 6.2.1 GlobalCommunicationapplication

It’s a multilingual communication app made for both Android & iOS, designed to facilitate near real-time communication between two users who speak different languages. It works by capturing the voice of each user, transcribing the speech into text, translating the text into the listener's preferred language, and converting it back into speech, which is then delivered in a voice that closely resembles the original speakers. The Service-Oriented Architecture (SOA) is Used in the backend side of the application because it involves multiple distinct services working together, including voice transcription, translation, and text-to-speech, each handled by different services. **Components of SOA in Backend:**

1. **Authentication & User Management Service (Firebase):** Handles user authentication (login, signup) and stores essential user data like email, password, and language preferences.
2. **Voice Data Service:** Collects voice data for voice cloning, stores it securely, and communicates with Coqui to generate or process voice models.
3. **Real-Time Communication Service:** Manages WebSocket connections to allow two users to communicate in real-time. It handles call initiation, termination, and the real-time transfer of audio streams.
4. **Transcription Service (OpenAI Whisper API):** Converts the audio streams from both users into text, sending the results to other services like translation.
5. **Translation Service (ChatGPT-4 API):** Translates the transcribed text into the target language, ensuring context and accuracy are maintained.
6. **Text-to-Speech Service:** Converts translated text into voice using either Coqui (if the voice clone is available) or OpenAI TTS (if insufficient voice data is available), applying voice conversion if necessary.
7. **Database Service (Firebase):** Stores user data, call history, and other relevant details like voice model information. Firebase will handle the persistence layer and communication with the backend.
8. **Google Cloud (GCP):** The Place where we will train the voice separation model for each user, Taking the audio for training from the Firebase and receiving and sending voice audio files with the central server after the model has been trained using enough data.
9. **Central Server:** The central hub where different services interact. It routes user requests, manages WebSocket communications, forwards data to the APIs, and handles the overall workflow (e.g., transcription, translation, voice conversion).

### 6.2.2 Screens



**Login Screen:**

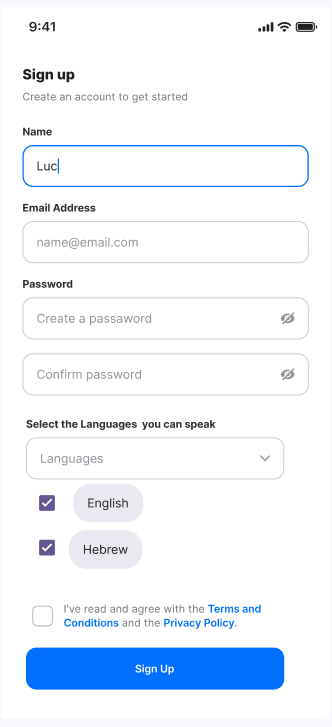
This is the Login Screen.

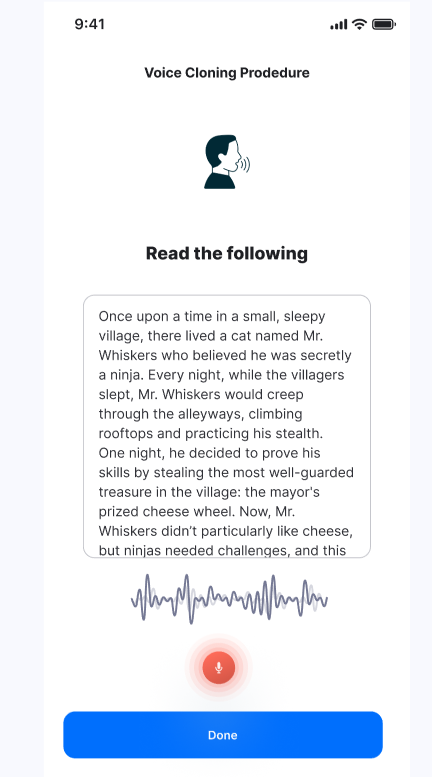
If the user has not signed it yet,

He can click on registration.

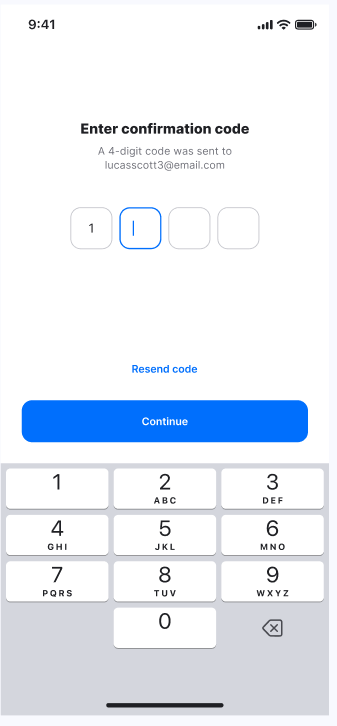
**SignUp:**

This is a basic sign-up screen but at the end the user must choose the languages that he can speak.



**Voice Cloning:**

After selecting your preferred languages in the sign-up page, you'll be prompted to choose one language and read a short paragraph aloud to help capture your voice.



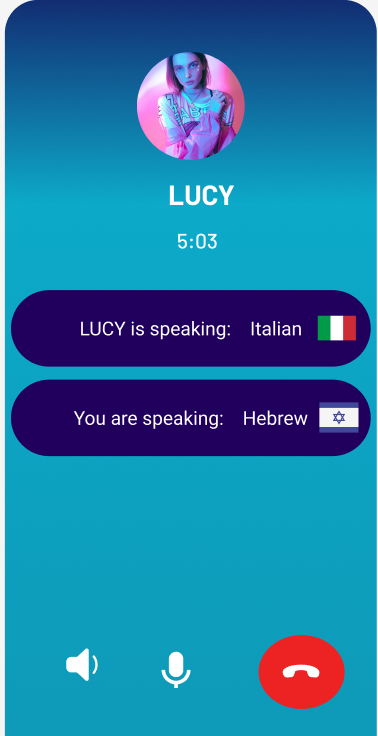
**Confirmation Screen:**  
Prompted after signing up. This screen is for security purposes, for making

Sure that the user email is valid.

A screenshot of a phone

Description automatically generated**Calls Screen:**

Here you can see all the saved contacts who use the app.

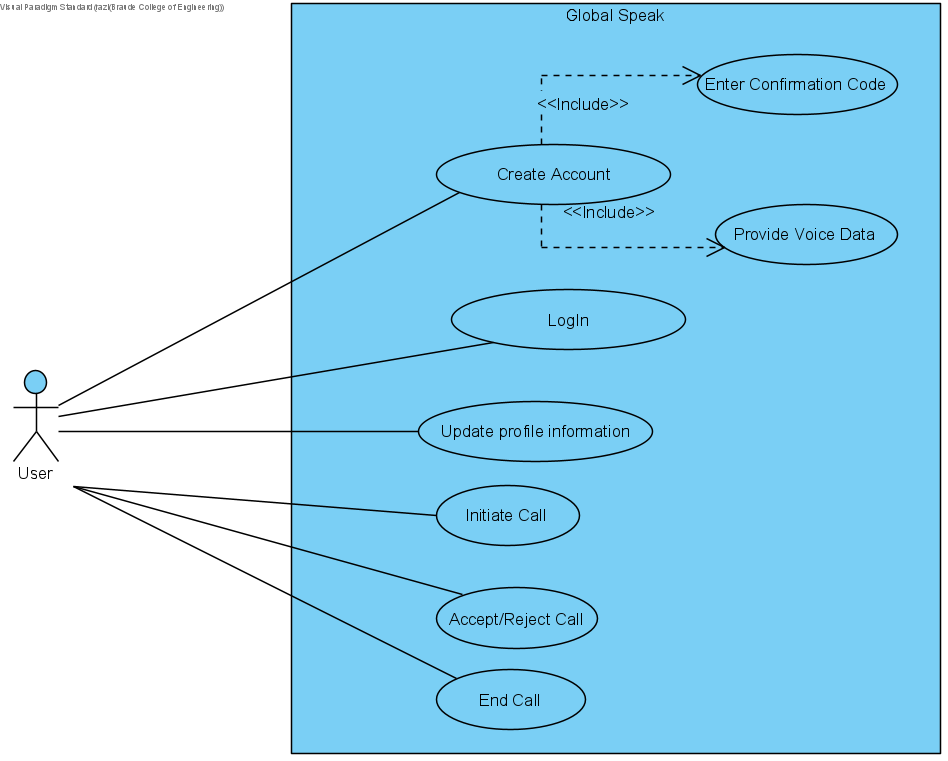
**Calling a friend:**

As you can see you will be able to view the langauge that the reciever is speaking and the language that you are currently speaking.

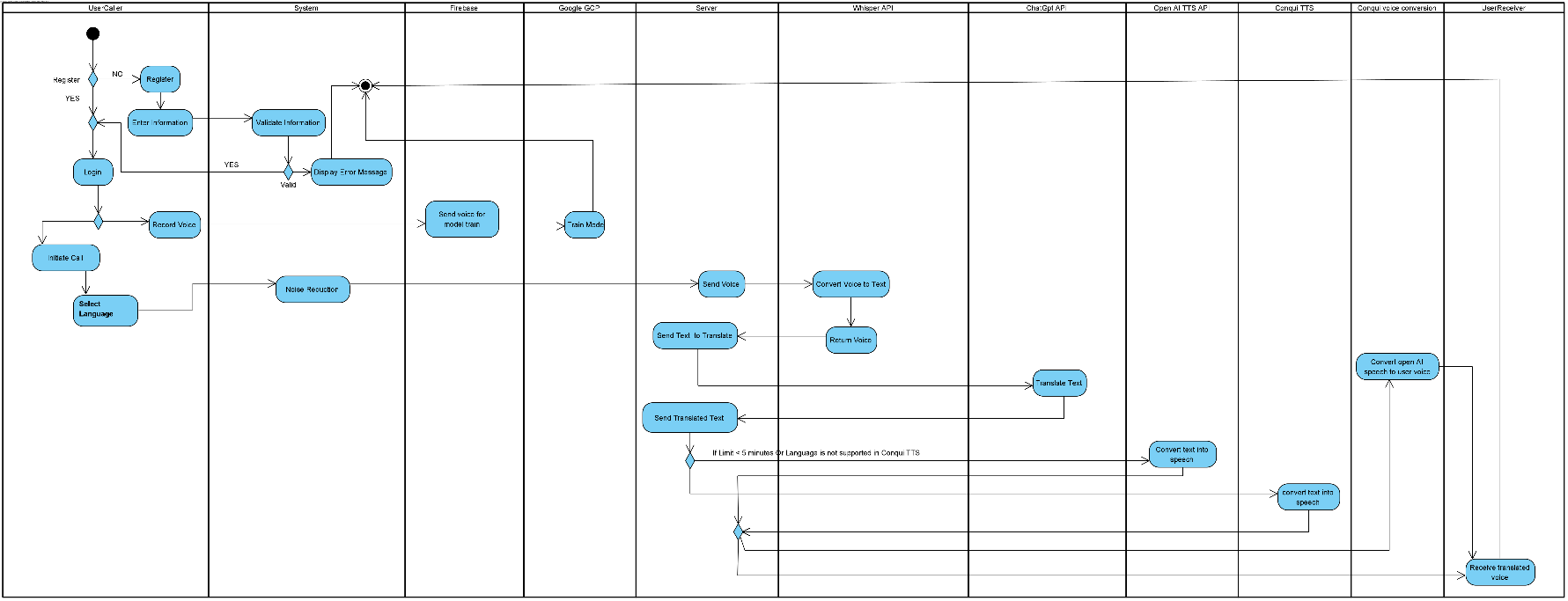
## 6.3 Diagrams

### 6.3.1 Use case:

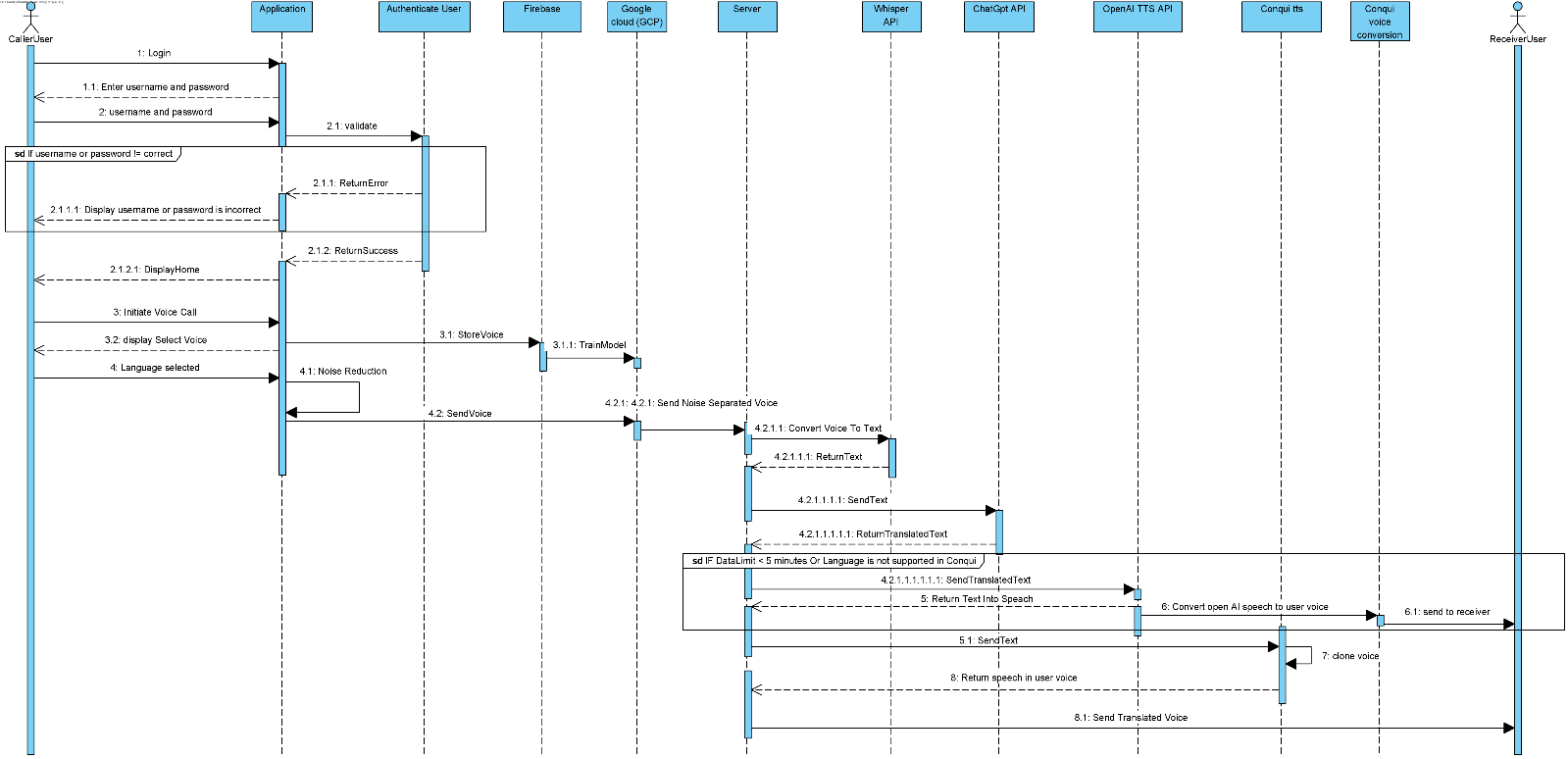
The following Use Case diagram shows the user interaction with the system.



### 6.3.2 Activity Diagram:

The following Activity diagram shows the whole process Starting from signing in/up to the call procedure.

### 6.3.3 Sequence Diagram

This sequence diagram shows the whole process Starting from signing in/up to the call procedure.

# 7.Verification and Evaluation

## 7.1 Evaluation

We will evaluate GlobalCommunication based on several key performance factors, including translation latency, the quality of voice cloning, and the app’s overall functionality. Our primary focus will be on how efficiently the app can process and translate speech with minimal delay, ensuring that conversations flow as naturally as possible despite the inherent processing times involved. Additionally, we will assess the accuracy and naturalness of the voice cloning, particularly how well the converted speech retains characteristics of the original speaker. Any latency in processing or noticeable deviations in voice quality will be measured, with the goal of minimizing both to create a smooth and realistic conversational experience. Success will be determined by the app’s ability to deliver translations quickly, with low latency, while maintaining acceptable quality in voice replication across the supported languages.

## 7.2 Verification

### 7.2.1 Testing

We will conduct a series of tests to evaluate the core functionalities of GlobalCommunication, focusing on translation speed, voice cloning quality, and system responsiveness. Each module, such as speech-to-text, translation, and text-to-speech, will be tested individually before performing end-to-end testing.

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Module | Tested Function | Expected Result |
| 1 | User Interface | UI responsiveness and navigation | UI is responsive, easy to navigate, and works seamlessly across devices. |
| 2 | Database | Data storage and retrieval | User preferences and data are stored and retrieved without errors or delays. |
| 3 | Speech-to-Text Module | Accuracy of speech recognition | transcribes speech into text accurately with minimal errors |
| 4 | Speech-to-Text Module | Latency of speech recognition | STT processes speech quickly, with low latency. |
| 5 | Translation Engine | Accuracy of translation | Translates text accurately between selected languages. |
| 6 | Translation Engine | Latency of translation | Translates text with minimal delay. |
| 7 | Text-to-Speech Module | Accuracy of synthesized speech | Synthesized speech is natural-sounding. |
| 8 | Text-to-Speech Module | Latency of synthesized speech | Synthesized speech is generated quickly with minimal delay. |
| 9 | Noise Suppression | Effectiveness of background noise handling | Effectively reduces noise with minimal distortion of the speaker’s voice. |
| 10 | Voice Separation | Multi-speaker voice separation | Accurately separates voices in noisy or multi-speaker environments. |
| 11 | Voice Cloning | Retention of original voice characteristics | Cloned voice retains characteristics of the original speaker. |
| 12 | Voice Conversion | Retention of original voice characteristics | Converted voice retains characteristics of the original speaker. |
| 13 | App Performance | Overall system accuracy | System accurately handles speech recognition, translation, and speech output. |
| 14 | App Performance | End-to-end latency of integrated components | All components work together with minimal delays, maintaining ≤ 3-5 seconds of latency. |

# 8. References

Ferraz, T. P. (2024). Efficient Compression of Multitask Multilingual Speech Models. arXiv preprint arXiv:2405.00966.

Chang, K. H. (2023). Natural Language Processing: Recent Development and Applications. Applied Sciences, 13(20), 11395.

Cambouropoulos, E. (2008). Voice and stream: Perceptual and computational modeling of voice separation. Music Perception, 26(1), 75-94.

Sadov, K., Hutter, M., & Near, A. (2023). Low-latency Real-time Voice Conversion on CPU. arXiv preprint arXiv:2311.00873.‏

Valin, J. M. (2018, August). A hybrid DSP/deep learning approach to real-time full-band speech enhancement. In *2018 IEEE 20th international workshop on multimedia signal processing (MMSP)* (pp. 1-5). IEEE.‏

Brown, T. B. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.‏

Graham, C., & Roll, N. (2024). Evaluating OpenAI's Whisper ASR: Performance analysis across diverse accents and speaker traits. *JASA Express Letters*, *4*(2).‏

Casanova, E., Davis, K., Gölge, E., Göknar, G., Gulea, I., Hart, L., ... & Weber, J. (2024). XTTS: a Massively Multilingual Zero-Shot Text-to-Speech Model. *arXiv preprint arXiv:2406.04904*.‏

Nachmani, E., Adi, Y., & Wolf, L. Voice separation with an unknown number of multiple speakers. In *International Conference on Machine Learning* (pp. 7164-7175). PMLR.

1. Chang, K. H. , Natural Language Processing: Recent Development and Applications. [↑](#footnote-ref-1)
2. Graham et al., Evaluating OpenAI's Whisper ASR: Performance analysis across diverse accents and speaker traits [↑](#footnote-ref-2)
3. Brown et al., Language models are few-shot learners.  [↑](#footnote-ref-3)
4. Valin, J. M. A hybrid DSP/deep learning approach to real-time full-band speech enhancement. [↑](#footnote-ref-4)
5. Sadov et al., Low-latency Real-time Voice Conversion on CPU. [↑](#footnote-ref-5)
6. Ferraz, T. P. , Efficient Compression of Multitask Multilingual Speech Models. [↑](#footnote-ref-6)
7. Cambouropoulos, E. (2008). Voice and stream: Perceptual and computational modeling of voice separation. Music Perception, 26(1), 75-94. [↑](#footnote-ref-7)
8. Casanova, E., Davis, K., Gölge, E., Göknar, G., Gulea, I., Hart, L., ... & Weber, J. (2024). XTTS: a Massively Multilingual Zero-Shot Text-to-Speech Model. *arXiv preprint arXiv:2406.04904*. [↑](#footnote-ref-8)
9. Nachmani, E., Adi, Y., & Wolf, L. Voice separation with an unknown number of multiple speakers. In *International Conference on Machine Learning* (pp. 7164-7175). PMLR. [↑](#footnote-ref-9)