

Software Engineering Department  
ORT Braude College

Capstone Project Phase B – 61998

**24-1-R-2**

****

Supervisor:

Alex Keselman

[Alex@vir-tec.net](mailto:Alex@vir-tec.net)

GitHub: [ShayZak1/SonicCipher (github.com)](https://github.com/ShayZak1/SonicCipher)

Authors:

Shay Zak 312330749

[Shay.zak@e.braude.ac.il](mailto:Shay.zak@e.braude.ac.il)

Michael Ioffe 315388900

[Michael.Ioffe@e.braude.ac.il](mailto:Michael.Ioffe@e.braude.ac.il)

Table of Contents

[1. Abstract 3](#_Toc177819341)

[2. Introduction 3](#_Toc177819342)

[3. Background & Related Work 4](#_Toc177819343)

[3.1 API 5](#_Toc177819344)

[3.1.1 Google Voice To Text API 5](#_Toc177819345)

[3.1.2 OpenAI API 5](#_Toc177819346)

[3.1.3 Google Translate API 6](#_Toc177819347)

[3.2 Security 6](#_Toc177819348)

[3.2.1 Peer-To-Peer 6](#_Toc177819349)

[3.2.2 Asymmetric encryption 7](#_Toc177819350)

[3.3 Development Tools 7](#_Toc177819351)

[3.3.1 Preact 7](#_Toc177819352)

[3.3.2 HTML 8](#_Toc177819353)

[3.3.3 CSS 8](#_Toc177819354)

[3.3.5 Node.JS 8](#_Toc177819355)

[3.3.5 Preact Vs React 9](#_Toc177819356)

[3.3.6 Tailwind CSS 9](#_Toc177819357)

[3.3.7 Web Speech API 10](#_Toc177819358)

[3.3.7 Peer JS 10](#_Toc177819359)

[4. Engineering Process 11](#_Toc177819360)

[4.1 The process 11](#_Toc177819361)

[4.2 The Challenges 12](#_Toc177819362)

[5. Product 13](#_Toc177819363)

[5.1 Proposed Solution 13](#_Toc177819364)

[5.2 Implementation Process 14](#_Toc177819365)

[5.3 Used Technology 14](#_Toc177819366)

[6. Activity Diagram 15](#_Toc177819367)

[6.1 Web Screens 16](#_Toc177819368)

[7. Evaluation/Verification Plan 18](#_Toc177819369)

[**Test Cases Implemented:** 18](#_Toc177819370)

[8. Results and Conclusions 19](#_Toc177819371)

[9. User Documentation 20](#_Toc177819372)

[9.1 General Description 20](#_Toc177819373)

[9.2 Usage Scenarios 21](#_Toc177819374)

[10. References 22](#_Toc177819375)

# Abstract

In today's connected world, overcoming language barriers is crucial for enhancing communication and fostering understanding among diverse populations. Our capstone project at ORT Braude College introduces an innovative web application aimed at bridging these barriers through cutting-edge voice technologies. Utilizing Google's Speech-to-Text and Translate APIs alongside OpenAI's ChatGPT, our application offers real-time voice-to-text conversion and translation across multiple languages in two distinct modes: basic translation for quick understanding and advanced translation for context-sensitive, nuanced communication. This dual approach allows users to choose the level of translation that best suits their needs, enhancing flexibility and user experience. The application stands out for its ability to deliver seamless and accurate communication solutions that are accessible on both desktop and mobile platforms. It is designed to improve user satisfaction by providing intuitive and responsive interactions, ensuring that users can easily engage in cross language related conversations without the need for technical expertise. Additionally, the application incorporates essential security measures to ensure that user interactions remain private and protected, supporting a safe communication environment. This project aims to enhance global communication, making it simpler and more inclusive, thereby transforming the world into a more connected and accessible place.

# Introduction

The start of globalization has made it necessary to develop technologies that helps

Others around the world to surpass the language related barriers, helping others around the world to talk with each other in easier way. Our project introduces an innovative web application that works both for desktop and smartphone users, which aims to close the gap between languages through the power of voice. This application will stand at the junction of advanced natural language processing (NLP) with encryption technologies,

Utilizing the Google translate API and ChatGPT API to record the spoken language and convert it into written text and afterwards translate it into the user’s preferred language with outstanding accuracy.in this era where our world is connected more than ever,

the need for secure and efficient communication between languages is now more important than ever. Our project addresses this demand by enabling users to easily communicate in their origin language, while guaranteeing that their messages are understood by the application accurately and translated accurately to the language of the recipient. More than only translation, our application integrates Peer-to-peer and asymmetric encryption, which promises that the exchange of the communication during the translations and meetings not only flawless but also secure, protecting the privacy of the communication from potential eavesdropping. Created with a strong emphasis on user satisfaction, our application is addressing this need by enabling users to effortlessly communicate in their native language, whether you are on your phone or sitting at a computer. with this app, we strive to improve and promote a more inclusive digital world.

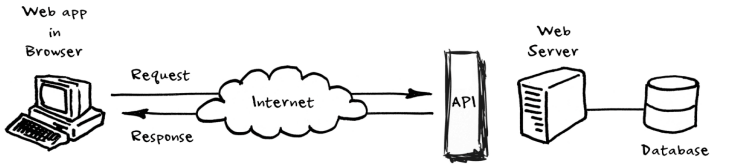
**2.1** motivation

This project is motivated from recognizing the limitations of the current translation tools, we use the ChatGPT API which understands in advanced way the language that the user seeks to translate his input or translate by Google translate API that stores a huge language library. This combination of the APIs aims to close the gap between the language barriers more effectively and ensure that the communication will stay private. By using these technologies together, we are assisting the users to have a clearer conversation with different languages. Furthermore, many popular translation tools like Google Translate, iTranslate, Microsoft Translate are well known and commonly used across the world for quick text and speech translation across many languages. However those tools makes mistakes when it comes to fully understanding the variations of the given language. And by understanding this problem This is where our project comes in, which will use ChatGPT with the advanced understanding of languages or with Google Translate API in order to translate our recorded text in basic way.

# Background & Related Work

In our increasingly globalized world, the need to overcome language barriers has never been more critical. Voice-based communication tools offer a promising solution, allowing people to interact naturally in their native languages while relying on technology to bridge the gap. However, ensuring these communications are secure, especially when sensitive information is shared, is equally important. This literature survey explores the current landscape of voice recognition, real-time translation technologies, and secure communication methods.

## 3.1 **API**

API (Application Programming Interface) is a set Procedures for building and interacting with software applications. It enables different software programs to interact with each other, allowing them to share data and features without the need to know how they're implemented with each other. In web-based systems and development, APIs are crucial for enabling fusion and communication between different software applications and services over the web. They allow web applications to receive data from servers without needing to reload the webpage and more. API’s enables many functions like payment processing and data analyzation in web applications which makes them more interactive with the user

### 3.1.1 Google Voice To Text API

Googles Voice To Text API, is part of Google Clouds Speed To Text service which represents a complex and sophisticated solutions for making deep learning techniques

Much better in transforming spoken language into text. Voice to Text APIs works by capturing audio input which afterwards is handled by deep learning algorithms which transform the speech into text. It is designed to recognize over 120 languages. This API works in many different environments, such as noisy backgrounds by using advanced noise cancellation techniques. It has many other features Such as supporting real time transcription, distinguish between multiple speakers and more. The API also is customizable, enabling users to train the model on domain-specific vocabulary for improved accuracy. This precision make it an invaluable tool for developers looking to incorporate voice recognition into their applications.



### 3.1.2 OpenAI API

OpenAI API is a new solution from OpenAI. Its designed to bring sophisticated natural language processing capabilities into many applications. the API enables the integration of conversational AI, making machines understand and generate human-like text responses. its built on the latest versions of the GPT models, which are known for the ability to understand context, making relevant text, and even being a sensible activity.

this API supporting many various of languages and can handle with some conversational scenarios. From simple interactions to complex dialogue management, the GPT allows real-time conversations based in real-time texts and asynchronous across various domains. It is especially adapting at understanding nuances in language, thanks to its extensive training in many various internet texts, what makes it appropriate to the relevant applications which is required for the interactions between human and computer.

Developers can fit the API to their specific needs with matched prompts so they can guide responses to particular tones or content styles. This matching, combined with its ability to improve responses for a long time, and making the ChatGPT to a valuable tool for businesses and developers which are strive to improve their user experience with conversational AI. Whether if its for customer support or for entertainment , the ChatGPT API provides us strong framework for AI communication into apps, websites and etc.

### 3.1.3 Google Translate API

Google Translate API is a tool from Google. It helps computers change text from one language to another. This API lets apps understand and translate text in many languages. It uses smart tech to do translations that make sense. This API can be used in different apps and websites. It's good for when you need to show text in many languages. You can use it for simple words or for big paragraphs. It makes talking or sharing information with people who speak different languages easier. People who make apps or websites can use this API to help their users read or talk in the language they know best. It's helpful for businesses or anyone who wants to reach more people in their own language. Google Translate API is easy to fit into your app or website. You can start using it to translate text right away. It's a powerful tool for anyone who needs to communicate across different languages.

## 3.2 Security

### 3.2.1 Peer-To-Peer

Our project Applies Peer-to-Peer (P2P) technology to break down language barriers by employing a distributed approach, real time communication. Different from other traditional models, P2P allows direct data exchange among users, enhancing and improving the privacy and reducing latency of the process. This technology enhances and makes efficient voice to text conversion and translation without relying on central servers, making our system scalable and durable. We will use WebRTC for peer connections, ensuring seamless interaction. Security is addressed through encryption and peer authentication. P2P's integration underscores our commitment to providing a secure, efficient platform for overcoming language obstacles, embodying our vision for a more inclusive digital world. P2P technology will be implemented in our project when we develop our live video website conversations which will allow others to translate their speech during live conversation.

### 3.2.2 Asymmetric encryption

Asymmetric encryption is a way of scrambling messages so that only the person meant to read them can understand. Each user has a public key and a private key. Unlike a simple lock and key system where one key opens one lock, this method uses two keys for each person. This method ensures that only the intended recipient, who has the correct private key, can decrypt the message. Anyone can use this key to encrypt a message for you, but once it's locked, only you have the private key to decrypt and read it. This ensures that even if someone intercepts the message, they can't read it, only the intended person can. The process of key generation involves complex mathematics, basically depends on problems considered computationally infeasible, such as the factorization of large numbers or the computation of discrete logarithms. This is what makes the keys secure and keeps the messages safe. In our project, asymmetric encryption plays a crucial role in securing communication between users speaking different languages. When you send a message, it gets locked with the receiver's public key. This is not only secures the message from unauthorized access but also verifies the sender's identity through digital signatures, improving trust and privacy in our communication app.. This way, we're not just keeping the messages safe from snooping, we're also making sure that the message really comes from you, building trust and keeping things private in our app for chatting across languages. Asymmetric encryption will be implemented in our project when we develop our live video website conversations which will allow others to translate their speech during live conversation.

## 3.3 Development Tools

### 3.3.1 Preact

Preact is a lightweight JavaScript library that mirrors the capabilities of React but at a fraction of the size, making it ideal for projects where performance and payload are critical. It employs the same component-based architecture as React, allowing developers to manage complex UIs efficiently. With its compatibility with React, projects can seamlessly transition to Preact using the preact-compat add-on, facilitating easy migration without significant codebase changes. Preact's small footprint (around 3KB) ensures faster load times, enhancing performance especially in progressive web applications and other high-demand environments. It maintains high compatibility with React’s ecosystem, providing developers a robust alternative that doesn't compromise on the ability to deliver dynamic and responsive interfaces. By adopting Preact, developers leverage its fast rendering capabilities and efficient update handling through a virtual DOM, ensuring a smooth and performant user experience in web applications.

### 3.3.2 HTML

HTML (HyperText Markup Language) is the foundational building block of the web, providing the structural framework for web pages. It enables the creation of structured documents by defining elements such as headers, paragraphs, links, and other content units. HTML is essential for web development as it dictates the organization and presentation of content on the Internet. Its simplicity and versatility allow developers to easily integrate multimedia resources, form elements, and other interactive features, making it indispensable in creating user-friendly web applications. By leveraging HTML, developers ensure that content is accessible and functional across various devices and web browsers, enhancing the reach and effectiveness of web applications.

### 3.3.3 CSS

CSS (Cascading Style Sheets) is a stylesheet language used to describe the presentation of documents written in HTML or XML. CSS empowers web developers to control layout, colors, fonts, and the overall visual aspect of web pages, enabling consistent styling across multiple pages of a website. The separation of content (HTML) from design (CSS) simplifies site maintenance and ensures greater flexibility and control in the presentation of web pages. By using CSS, developers can create responsive designs that adapt to different screen sizes and resolutions, providing an optimal viewing experience. CSS's capabilities contribute significantly to the aesthetic appeal and user experience of web applications, playing a crucial role in user engagement and retention.

### 3.3.5 Node.JS

Is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser. Node.js lets developers use JavaScript to write command line tools and for server-side scripting and running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web-application development around a single programming language, rather than different languages for server-side and client-side scripts. Though .js is the standard filename extension for JavaScript code, the name "Node.js" doesn't refer to a particular file in this context and is merely the name of the product. Node.js has an event-driven architecture capable of asynchronous I/O. These design choices aim to optimize throughput and scalability in web applications with many input/output operations, as well as for real-time Web applications (e.g., real-time communication programs and browser games). The Node.js distributed development project was previously governed by the Node.js Foundation and has now merged with the JS Foundation to form the OpenJS Foundation, which is facilitated by the Linux Foundation's Collaborative Projects program.

### 3.3.5 Preact Vs React

Preact and React are both popular JavaScript libraries for creating user interfaces, but they serve slightly different purposes due to their inherent characteristics: Preact is a streamlined version of React, designed to be small and efficient. It retains a similar API to React, which allows developers who are experienced with React to adapt easily to Preact. The main goal of Preact is to offer a lighter and faster alternative for projects where size and performance are critical, such as mobile applications, single-page applications, and progressive web apps. On the other hand, React is more robust and supports a broader range of features, making it suitable for developing complex, large-scale applications, including those for enterprises. One of the criticisms of React is its relatively large size and slower initial load times, which stem from its comprehensive feature set and the use of a virtual DOM. This overhead can affect performance, particularly on devices with limited resources. These performance concerns with React have led to the development of alternatives like Preact, which strips away some of the less commonly used features and optimizations inherent in React to provide a nimbler, more resource-efficient library. For projects requiring fast performance and efficient resource utilization, Preact is often the preferred choice over React.

### 3.3.6 Tailwind CSS

Tailwind CSS is a utility-first CSS framework that allows for rapid development by applying pre-defined classes directly in HTML. This approach reduces the need for custom CSS, promoting efficiency and maintainability. Tailwind's customization options enable us to create a unique and responsive design tailored to our project's needs. Its utility classes simplify responsive design, ensuring our application looks great on all devices. Additionally, Tailwind's focus on reusability and component-based styling aligns with our goal of building a consistent and visually appealing user interface for our language translation web application.

### 3.3.7 Web Speech API

Web Speech API is a powerful browser-based technology that enables real-time speech recognition and voice capture directly within web applications. This API allows us to efficiently capture voice input during live interactions, converting spoken words into text, which is then used for translation. By leveraging the Web Speech API, we can offer a seamless and interactive experience, eliminating the need for external plugins or software installations. Its ability to handle continuous speech recognition makes it ideal for our video translation feature, providing accurate and quick transcriptions that enhance the overall user experience. The API's straightforward integration and broad browser support align with our project's objective to deliver accessible and responsive language translation solutions, ensuring that voice input is captured clearly and processed instantly for live communication.

### 3.3.7 Peer JS

For securing peer-to-peer connections in our video translation feature, we utilize **PeerJS** with WebRTC, a technology that inherently supports secure data channels and media streams through encryption protocols like DTLS (Datagram Transport Layer Security) and SRTP (Secure Real-time Transport Protocol). WebRTC ensures that all video, audio, and data transferred between peers are encrypted end-to-end, preventing unauthorized access and ensuring the privacy of communications. By leveraging PeerJS, which abstracts the complexities of WebRTC, we are able to easily implement secure peer-to-peer interactions, aligning with our goal of maintaining a protected and private communication environment within our application.

# Engineering Process

In Part A of the project, we explored the field of real-time voice translation and the role of various APIs in enhancing cross-language communication. We delved into the functionality and limitations of existing voice-to-text and translation technologies, examining how they could be integrated with secure communication methods like peer-to-peer and asymmetric encryption to meet the privacy and security needs of our application. We assessed current solutions in natural language processing and reviewed the relevant APIs, including Google Speech-to-Text, Google Text-to-Speech, Google Translate, and OpenAI’s ChatGPT.

In Part B of the project, we will design and develop a web application that demonstrates our integrated solution. This application will showcase our ability to capture voice input, convert it to text, translate it into the desired language, and read it back aloud—and to enable users to have an videos conversation which will translate their speech live all while ensuring that communications remain private and secure. The focus will be on providing an intuitive and user-friendly experience that enables seamless, real-time multilingual conversations, whether on desktop or mobile platforms.

## 4.1 The process

In the initial phase of our project, we defined the primary goals and identified the key challenges of breaking language barriers through secure, real-time translation. We formulated specific objectives, such as providing accurate translations across multiple languages. We conducted a comprehensive literature review to explore current voice-to-text and translation technologies. This helped us understand the existing solutions and identify gaps, such as enhancing accuracy and implementing stronger security measures. During development, we integrated key APIs—Google Speech-to-Text for voice recognition, Google Translate for basic translations, and OpenAI API for contextual translations. We also included Google Text-to-Speech to provide auditory feedback of translated text, enhancing the user experience. For security, we implemented peer-to-peer (P2P) communication using WebRTC for direct connections in our video live translation, minimizing latency and improving privacy.. We developed a comprehensive testing plan to evaluate the application’s performance and security, including testing under heavy load, verifying translation accuracy.

**Advantages of the System**

Our system improves upon traditional translation tools by prioritizing security and real-time performance. P2P communication safeguard data during live translation, reducing the risk of unauthorized access and enhancing user privacy. The application is also scalable, allowing for the addition of languages and features to meet the needs of a diverse user base, making it suitable for large-scale deployments where secure and efficient communication is essential. Overall, our project delivers a secure, user-friendly platform.

## 4.2 The Challenges

During the development of our web application, we encountered several significant challenges, including unfamiliarity with new programming technologies, achieving secure real-time translation, ensuring seamless performance, and maintaining robust security measures. Addressing these challenges required extensive research, learning, and the implementation of advanced technologies and methodologies. One of the primary challenges was our lack of familiarity with the programming languages and tools necessary for building the application. To develop the web application, we needed to learn new technologies, including Preact, JavaScript, Node.js, and various APIs such as Google Speech-to-Text, Google Translate, OpenAI, and Google Text-to-Speech. This steep learning curve demanded significant time and effort as we mastered these technologies, integrated them into our application, and optimized their performance. Another major challenge was establishing a secure and efficient communication channel between users during live translations. This was critical to protect against potential attacks, such as man-in-the-middle attacks, where an attacker could intercept and manipulate the translated data. To address this, we implemented peer-to-peer (P2P) communication using WebRTC, which allows direct, encrypted connections between users without relying on central servers. We also faced challenges related to real-time performance and synchronization of the translation process. Ensuring that voice input was accurately captured, translated, and vocalized without significant delays required optimizing the integration of multiple APIs. To achieve this, we implemented a strategy that prioritizes low-latency connections and efficient data processing, allowing the application to handle voice recognition and translation tasks seamlessly. Implementing live translation across all platforms proved to be particularly difficult. Some platforms block real-time voice data transmission due to security restrictions, making it challenging to establish consistent functionality. We faced compatibility issues with specific browsers and operating systems, and had to work around platform limitations and permissions that restricted microphone and audio access. These obstacles required continuous adjustments and testing to find suitable solutions that work across different environments. Scalability and reliability were further challenges we needed to address, especially as our application aims to support a diverse user base with varying language needs. We developed a modular architecture that can handle increasing loads and support multiple languages, accents, and dialects. This approach ensures that our system remains responsive and adaptable as usage grows. Testing and validation posed another significant challenge, as we needed to ensure the accuracy of translations and the security of the communication channels. Developing a comprehensive testing plan was essential to identify potential weaknesses and optimize the performance of our APIs and security protocols. We conducted extensive testing across various scenarios to validate the application’s functionality and security measures. Overall, the challenges we faced pushed us to expand our knowledge and skills, allowing us to develop a secure, reliable, and efficient application that meets our project’s objectives of breaking language barriers while ensuring privacy and security.

# Product

## 5.1 Proposed Solution

In our project, we identified the primary challenge of breaking down language barriers through secure, real-time translation. Our proposed solution is a web application that leverages advanced APIs for voice recognition, translation, and audio playback, combined with robust security protocols to ensure user privacy.

Our application captures spoken language, converts it to text, translates it into the target language, and then reads it back to the listener in real time. The solution involves multiple integrated components and follows these key steps:

1. **Voice Capture and Conversion**: The application uses the Google Speech-to-Text API to capture the user's voice input and convert it into written text. This API is capable of recognizing various languages and dialects, allowing for accurate transcriptions even in noisy environments.
2. **Translation**: Once the speech is transcribed, the Google Translate API performs a basic translation of the text, providing a quick and straightforward conversion of the user's input. For more context-aware translations, we use the OpenAI API, which enhances accuracy by considering the tone and meaning of the conversation, making the translation more nuanced.
3. **Audio Playback**: After the translation, the application uses the Google Text-to-Speech API to vocalize the translated text, allowing the recipient to hear the message in their preferred language. This feature is particularly beneficial for users who are visually impaired or prefer auditory learning.
4. **Security Measures**: To secure the communication, we implemented peer-to-peer (P2P) communication using WebRTC, allowing direct, encrypted connections between users. This approach minimizes reliance on centralized servers, reducing latency and enhancing privacy. Additionally, secure transmission protocols were adopted to protect the data from interception during translation.
5. **Live Video Translation**: As part of the live translation feature, the application supports video meetings with real-time speech recognition and translation. However, implementing this across all platforms proved challenging due to platform-specific restrictions, such as browser limitations and blocked access to microphone and audio data. Overcoming these barriers required continuous testing and optimization to ensure consistent functionality across devices.

## 5.2 Implementation Process

The development of our system followed a structured approach, focusing on the independent creation of each component before integrating them into a cohesive application. This method allowed us to refine each feature and ensure its functionality met the required standards.

* **User Interface**: The front-end of the application was built using JavaScript and Preact. The interface was designed to be user-friendly, with clear prompts and controls for starting and stopping translations. The goal was to make the technology accessible to all users, regardless of their technical background.
* **Backend Gateway**: The backend, built with Node.js, serves as the gateway that manages the flow of data between the user interface and the APIs. This component handles the integration of voice capture, translation, and playback processes, ensuring that each step operates smoothly and efficiently.
* **Integration and Testing**: Each API was integrated with careful attention to performance and reliability. Testing involved multiple scenarios to validate the accuracy of speech recognition, the quality of translations, and the consistency of audio playback. We also tested the security protocols to ensure user data remained protected throughout the translation process.

## 5.3 Used Technology

**5.3.1 Preact, Node.js, and Tailwind CSS**

* **Preact**: A lightweight JavaScript library used to build the user interface, offering fast rendering and a responsive user experience.
* **Node.js**: Used for backend development, managing API requests, processing data, and ensuring the application’s logic is executed reliably.
* **Tailwind CSS**: Employed for styling the application, enabling a visually appealing and consistent design across all devices.

**5.3.2 WebRTC**

* **WebRTC**: Facilitates peer-to-peer connections for direct and secure communication between users during live video meetings.

Our proposed solution delivers a user-friendly and secure platform for real-time language translation, overcoming technical and security challenges to enhance global communication.

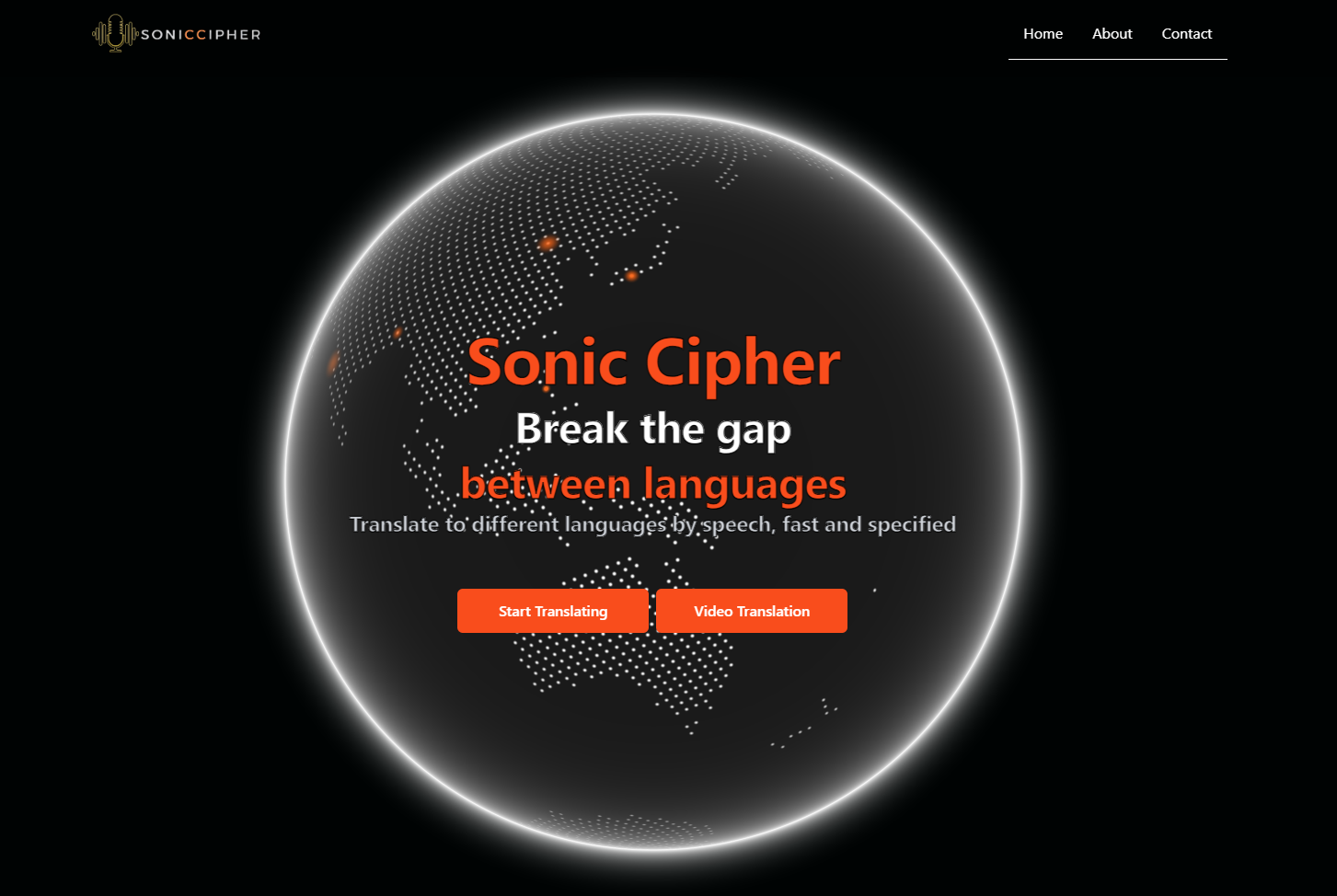
# Activity Diagram

Figure 1: System Architecture

**Sequence:**

1. **User Interaction**: The user starts on the home page, selects the translation method (basic or premium), and chooses input and output languages.
2. **Voice Input**: The user presses "Start Talking" to capture their voice, which is processed by the Google Speech-to-Text API.
3. **Text Display and Review**: The generated text is displayed to the user for review.
4. **Translation**: If approved, the text is sent to Google Translate API for basic translation or OpenAI API for contextual translation, based on the chosen method.
5. **Text-to-Speech**: The translated text is converted to speech using Google Text-to-Speech API and played back to the user.
6. **Repeat or Respond**: If the conversation continues, the user can engage in bidirectional translation using the "opposite language translation" button.

## 6.1 Web Screens



A screenshot of a computer

Description automatically generated

A screenshot of a phone

Description automatically generated

A screenshot of a video translation

Description automatically generated

# Evaluation/Verification Plan

Our web application serves as a testing platform to evaluate the performance and effectiveness of our real-time translation system. It allows us to simulate various user scenarios and test the application’s ability to handle different languages, accents, platform restrictions, and live translations. This approach helps us identify any issues, such as inaccuracies in translations, latency problems, or security vulnerabilities, and make necessary adjustments before deploying the application in a live environment. Additionally, it enables us to demonstrate the workflow of our translation process and highlight how it enhances communication while maintaining user privacy.

The evaluation focuses on assessing the accuracy, speed, and security of the translation system. This includes capturing voice inputs, processing them through the APIs, and verifying the correctness of the translations displayed and spoken back to the users. By using controlled testing environments, we simulate real-world conditions to ensure the application performs reliably across various devices and networks.

**Test Cases Implemented:**

|  |  |  |
| --- | --- | --- |
| Test Number | Test Case | Expected Result |
| 1 | Voice recognition accuracy | Text output matches the spoken input correctly |
| 2 | Translation accuracy using Google Translate and OpenAI | Translations are contextually accurate and understandable |
| 3 | Real-time text-to-speech playback | The translated text is played back without delay |
| 4 | Functionality on different platforms | Application works seamlessly across browsers and devices |
| 5 | Handling platform-specific restrictions | Secure P2P connections are established without errors |
| 6 | Security of data transmission | Voice and text data are encrypted and protected |

These test cases help ensure the application is functional, secure, and user-friendly. By evaluating each component under realistic conditions, we can refine the system to offer an optimal real-time translation experience.

# Results and Conclusions

The implementation of our real-time translation web application yielded positive results in improving multilingual communication. Through the integration of advanced voice recognition, translation, and text-to-speech APIs, the system successfully facilitated accurate and efficient translations in various real-world scenarios. The application effectively handled voice input, translated it into the desired language, and provided clear audio playback, demonstrating the accuracy and fluidity of the translation process.

Testing across multiple devices and platforms showed that the application maintained high performance and low latency, even when handling complex language inputs. The use of WebRTC for secure peer-to-peer communication ensured that data remained protected during live translations, addressing privacy concerns and enhancing the overall user experience.

Our evaluation highlighted the application’s ability to adapt to different accents, dialects, and user preferences, confirming the flexibility and robustness of the system. Despite challenges with platform restrictions and voice recognition accuracy, the results confirmed that the application met its objectives of providing secure, real-time translation with minimal user intervention.

In conclusion, our web application offers an effective and innovative solution for enhancing multilingual interactions, making communication more accessible and inclusive. The combination of advanced APIs, secure communication protocols, and a user-friendly interface provides a reliable tool for real-time translation, promoting seamless interaction in diverse linguistic settings.

# User Documentation

## 9.1 General Description

The system is designed as an innovative web application that provides secure, real-time voice translation across multiple languages. It integrates advanced APIs for voice recognition, translation, and speech synthesis, making communication smoother and more accessible. The application combines technologies such as Google Speech-to-Text, Google Translate, OpenAI, and Google Text-to-Speech APIs to deliver accurate translations with responsive audio playback.

The system architecture consists of a user interface built with JavaScript and Preact, and a main server that coordinates the translation workflow. The main server handles data processing, manages API interactions, and ensures secure communication between users via WebRTC, which establishes direct, encrypted connections during live translations.

The application caters to two main user groups: end-users seeking real-time translation services and developers integrating the system into websites or platforms that require multilingual support.

* **End-Users**: This group includes individuals who need to communicate in different languages for personal or professional reasons. The web application offers them an intuitive interface to speak, translate, and listen to translations instantly. Users can select their preferred languages, configure specific options like gender preferences for audio playback, and engage in seamless conversations.
* **Developers**: This group involves web developers and platform integrators who aim to embed translation capabilities into their own applications or services. Developers can configure the application to meet specific user needs, ensuring compatibility with existing systems and enhancing the multilingual features of their platforms. They focus on seamless integration, customizing API calls, and managing the security of data exchanges within the application.

the system serves both end-users who need reliable real-time translation and developers who seek to expand the language capabilities of their platforms. It provides a secure, easy-to-use solution that promotes effective communication in diverse linguistic environments.

## 9.2 Usage Scenarios

The system supports several usage scenarios that cater to different user needs and actions. The following scenarios highlight common interactions within the application:

**Real-Time Voice Translation:** In this scenario, the user initiates a translation session by selecting their input and output languages and pressing the "Start Talking" button. The application captures the user’s speech, converts it into text, and then translates it into the chosen target language. Finally, the translated text is converted into speech and played back to the listener, enabling seamless real-time communication.

**Live Video Meeting with Translation:** Users can engage in live video meetings where speech is translated in real time. In this scenario, the application utilizes WebRTC to establish a secure, direct connection between participants. As each person speaks, the application captures the audio, performs the translation, and reads the translated message aloud to the other participant, facilitating a fluid conversation in different languages.

**Bidirectional Conversation:** In a bidirectional conversation, users can take turns speaking, with the system capturing each user’s input and translating it back into the other’s language. This scenario is ideal for interactive discussions where both participants need real-time translation support, ensuring a smooth and responsive dialogue.

# References

*[1] Google Developers. (2022). "Speech-to-Text: Automatic Speech Recognition." Google Cloud. https://cloud.google.com/speech-to-text*

*[2] OpenAI. (2023). "OpenAI API." https://beta.openai.com/docs/*

*[3] Google Developers. (2022). "Translate API: Integrating real-time translation into your apps." Google Cloud. https://cloud.google.com/translate*

*[4] Mozilla Foundation. (2023). "HTML: Hypertext Markup Language." MDN Web Docs. https://developer.mozilla.org/en-US/docs/Web/HTML*

*[5] Mozilla Foundation. (2023). "CSS: Cascading Style Sheets." MDN Web Docs. https://developer.mozilla.org/en-US/docs/Web/CSS*

*[6] Preact. (2023). "Preact: Fast 3kB alternative to React with the same modern API." https://preactjs.com/*

*[7] WebRTC. (2021). "WebRTC: Real-time communication for the open web." https://webrtc.org/*

*[8] IEEE. (2018). "Explore the Principles of Asymmetric Encryption." IEEE Xplore, March 2018. https://ieeexplore.ieee.org/document/8374916*