

Software Engineering Department  
ORT Braude College

Capstone Project Phase A – 61998

**Voice2Textify**

**Danial Jarrous 211844881**

**Amne Salame 322732561**

**Amne Salame 322732561**

**Contents**

[***Abstract***](#_xtqj6ofggjqj) ***4***

[***1. Introduction***](#_bkcr6b9r1uc2) ***5***

[***2. Literature Review***](#_j1vnfyilt7ds) ***6***

[*2.1 Similar Real-Time Transcription Apps*](#_nui44i9n30ot) *6*

[*2.2 Possible Approaches and Tools for Real-Time Transcription of Spoken Language into Text*](#_dazn2l6k10fg)*.*

[***3. Expected Achievements***](#_2i5f8zcx2d91) ***10***

[*3.1 User-Friendly Interface*](#_r9lobs8lg1jc) *11*

[*3.2 Real-Time Speech Recognition and Processing*](#_5211jard9vh) *11*

[*3.3 Success Criteria of the Project*](#_wbzaf3hlu671) *12*

[***4. Engineering Process***](#_ubne8skckgby) ***12***

[*4.1 The Process of Crafting Software and Conducting Research*](#_80dc9twgghjf) *12*

[*4.1.1 Requirements*](#_h8rdvr2rh4af) *14*

[*4.1.2 Deciding Which Technologies We Will Use*](#_4bg2aw17991d) *15*

[*4.1.3 Environments*](#_5do71sowt0ol) *16*

[*4.1.4 Development Methodology*](#_djw6lj1af6ef) *17*

[*4.1.5 Challenges*](#_iqy0ffn6qana) *17*

[*4.2 Product*](#_qwgd3ec5v3g0) *18*

[*4.2.1 Software Architecture Diagram*](#_80tfu5eewkn6) *18*

[*4.2.2 UseCase Diagram*](#_2rq2i9ncspsx) *20*

[*4.2.3 Activity Diagram*](#_bh5zl7yvdjom) *21*

[*4.2.4 User-Friendly Interface*](#_ufuxii9yndxa) *21*

[***5. Verification Plan:***](#_wrlqm3yanulb) ***24***

[*5.1 Tests*](#_wvap8v13lkp7) *24*

[*5.2 Constraints*](#_esadfqy4gwtf) *26*

[*5.3 Assumptions*](#_mi7hfatzpjt0) *27*

[*5.4 Other Important Considerations: User’s Evaluation and Feedback*](#_856ia1dl1n5u) *27*

[*5.4.1 User Interface Evaluation*](#_7menk24hgxan) *27*

[*5.4.2 Asking for General Feedback*](#_rccgwua9ugaq) *27*

[***References***](#_ib2w3ffr455r) ***28***

# Abstract

The demand for real-time transcription of spoken language into text has grown significantly across various sectors, driven by the need to enhance accessibility and communication efficiency. This project introduces a mobile application that utilizes cutting-edge machine learning algorithms to instantly transcribe spoken words into text. The technology aims to facilitate seamless access to live captions on mobile devices, empowering users with immediate comprehension of spoken content in diverse contexts.

The core objective of the application is to bridge communication barriers by providing instantaneous and accurate transcription of live conversations and events. By leveraging advanced speech recognition capabilities, the application enhances accessibility for users in educational, professional, and social settings. It supports real-time interaction by displaying transcriptions directly on users' devices, thereby facilitating better understanding and participation in spoken exchanges.

In addition to basic transcription, the application offers advanced features such as Speaker Identification & Customization, which allows users to differentiate and label speakers within a conversation, Smart Summarization, which condenses lengthy transcriptions into key points for easier consumption, and Real-Time Translation, enabling the text to be translated into multiple languages instantly. These features ensure the application not only captures spoken language nuances with reliability and accuracy but also caters to diverse user needs across different environments.

In conclusion, this project represents a significant advancement in real-time transcription technology, offering broader applications beyond specific user groups. By enabling instant scripting of spoken language and incorporating advanced features like speaker identification, summarization, and translation, the application supports enhanced communication and accessibility across educational, professional, and social domains, fostering inclusivity and communication efficiency in diverse settings.

# 1. Introduction

In contemporary society, the demand for advanced technologies to address accessibility challenges and enhance communication efficiency is ever-growing. Real-time transcription technology represents a pivotal innovation in this landscape, offering immediate conversion of spoken language into text. This technology not only aims to alleviate the communication barriers faced by deaf individuals by providing instant access to spoken content but also holds promise for enhancing interaction dynamics across diverse professional, educational, and social settings.

Deaf individuals often encounter significant challenges in accessing live conversations and events where subtitles or sign language interpreters are unavailable. Real-time transcription emerges as a transformative solution, leveraging sophisticated machine learning algorithms to accurately capture and display spoken words as text on mobile devices. This capability enables deaf individuals to participate actively in conversations, follow discussions, and engage in real-time interactions with greater ease and independence.

Beyond its application in addressing the accessibility needs of deaf individuals, real-time transcription technology has broader implications for enhancing communication efficiency in multilingual settings, overcoming language barriers, and facilitating seamless interaction among diverse linguistic communities. The inclusion of advanced features such as **Speaker Identification & Customization**, **Smart Summarization**, and **Real-Time Translation** further enhances the utility of this technology. Speaker Identification allows users to differentiate and label speakers within a conversation, Smart Summarization condenses lengthy transcriptions into key points, and Real-Time Translation enables the text to be translated into multiple languages instantly.

By providing immediate and enhanced textual representation of spoken language, this technology promotes inclusivity and equal participation in public events, educational lectures, business meetings, and social gatherings. Its ability to facilitate immediate access to spoken content, combined with advanced features, enhances communication efficiency in multilingual environments, bridges language gaps, and fosters seamless interaction among diverse linguistic communities. This comprehensive approach to real-time transcription supports inclusivity and participation across a wide range of societal contexts.

# 2. Literature Review

The rapid advancement in speech recognition technology has led to the development of various applications aimed at transcribing spoken language into text in real-time. These applications are instrumental in enhancing accessibility, improving communication efficiency, and bridging language barriers. In this section, we will explore existing solutions, their methodologies, and the technologies they employ to address the problem of real-time speech-to-text conversion.

## **2.1 Similar Real-Time Transcription Apps**

#### **[Google Live Transcribe](https://play.google.com/store/apps/details?id=com.google.audio.hearing.visualization.accessibility.scribe&hl=en)**



Google Live Transcribe is an Android application that provides real-time transcription of spoken words into text. The app leverages Google’s advanced speech recognition technology to offer highly accurate transcriptions across multiple languages. Users can customize the text size and switch to dark mode for better readability.

* **Limitation:** Google Live Transcribe does not have built-in speaker identification, meaning it cannot distinguish between different speakers in a conversation.

#### **[Otter.ai](http://otter.ai)**



Otter.ai is a versatile web and mobile application designed for transcribing conversations, meetings, and lectures in real-time. It incorporates AI algorithms to provide live transcription, collaborative editing features, and speaker identification. The app integrates seamlessly with various productivity tools, enhancing its utility in professional settings.

* **Limitation:** Otter.ai provides detailed transcriptions but does not offer automatic summarization, which can be cumbersome when dealing with lengthy meetings or conversations.

## **2.2 Possible Approaches and Tools for Real-Time Transcription of Spoken Language into Text**

### **ASR Models**

Automatic Speech Recognition (ASR) models are systems that convert spoken language into written text in real-time. These models are essential in various applications, including virtual assistants, transcription services, and accessibility tools for the hearing impaired. The primary function of ASR models is to process audio inputs, recognize speech patterns, and accurately transcribe spoken words, enabling smooth interaction between humans and machines.

### **Core Components of ASR Models**

ASR models rely on several key components for effective speech recognition. **Feature extraction** involves using Convolutional Neural Networks (CNNs) to analyze audio spectrograms, capturing important phonetic details such as pitch, tone, and rhythm. **Sequence modeling** addresses the temporal dependencies in speech data, using Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) networks, and Gated Recurrent Units (GRUs) to maintain context over time and generate coherent transcriptions.

### **Advanced Architectures**

Modern ASR models are increasingly incorporating **Transformer** architectures, which have revolutionized natural language processing. Unlike RNNs, transformers use self-attention mechanisms to process entire input sequences simultaneously, capturing long-range dependencies and contextual information more effectively. This leads to higher accuracy and efficiency in transcribing speech, making transformers valuable in advanced ASR systems. Models like BERT and GPT exemplify the advanced capabilities of transformers in generating human-like text.

### **Recurrent Neural Networks (RNNs)**

Recurrent Neural Networks (RNNs) are a cornerstone technology in speech-to-text systems, specifically designed to handle sequential data like audio signals. RNNs are particularly effective because they have loops in their architecture, allowing information to persist and be used in subsequent steps of the sequence. This makes them well-suited for tasks that require the retention of context over time, such as transcribing spoken language into text.

### **Core Components of RNNs**

RNNs consist of several layers that work together to process and transcribe sequential data. The **Input Layer** receives the audio signal, which is typically represented as a sequence of features extracted from the raw audio data. **Recurrent Layers** then process this input one step at a time, maintaining a hidden state that captures the contextual information from previous steps.

To enhance the network’s ability to learn and remember long-term dependencies, RNNs often incorporate **Long Short-Term Memory (LSTM)** layers or **Gated Recurrent Units (GRUs)**. These specialized layers are crucial for handling longer sequences and preventing issues like vanishing gradients, which can hinder learning in traditional RNNs. **Fully Connected Layers** take the output from the recurrent layers and transform it into a sequence of probabilities over possible output tokens, such as characters or phonemes.

### **Output and Application**

The final **Output Layer** produces the transcription as a sequence of characters or words, completing the speech-to-text process. RNNs, with their ability to retain and utilize context, are integral to the accuracy and coherence of transcriptions in speech recognition systems.

### **Google Cloud Speech-to-Text API**

The **Google Cloud Speech-to-Text API** is a highly efficient and accurate service that converts spoken language into text using Google’s advanced machine learning models. The API processes audio input in real-time, making it highly suitable for applications requiring immediate transcription, such as in meetings, lectures, and multilingual environments.

The key idea behind this API is its ability to process audio streams with advanced models that handle real-time transcription, automatically recognizing various speakers through speaker diarization. Additionally, the API supports over 120 languages and variants, making it extremely versatile for global applications. Automatic punctuation is also added to improve the clarity and readability of the transcriptions.

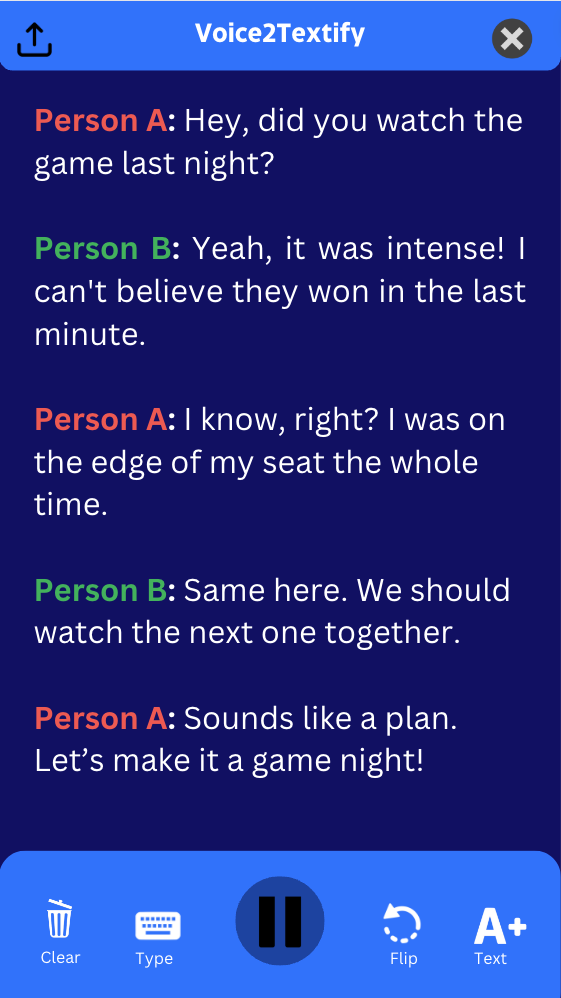
The **Google Cloud Speech-to-Text API** enables real-time performance with just a single pass through its machine learning model for each segment of audio input. This allows applications like ours to generate instant, accurate transcriptions with minimal delay.

# **3. Expected Achievements**

Our project aims to deliver a highly reliable, real-time voice-to-text transcription mobile application designed to enhance communication by instantly converting spoken language into text. The application is intended to be user-friendly, accurate, and efficient, offering accessibility and convenience across various settings, including educational, professional, and social environments.

By incorporating advanced features, the application will address the diverse needs of users, ensuring that it not only captures spoken words accurately but also enhances usability and accessibility in various contexts. The project aims to create a comprehensive solution that supports inclusive communication, facilitates better understanding in multilingual environments, and promotes active participation in discussions and events.

1. **Developing an Intuitive User Interface for Seamless Interaction**
2. **Leveraging Google API for High-Accuracy Real-Time Speech Recognition**
3. **Ensuring Real-Time Processing with Minimal Latency**
4. **Enhancing Accessibility Through Multilingual Support**



*Visual representation showcasing the anticipated appearance and functionality of our app*

## 3.1 User-Friendly Interface

We plan to build an intuitive interface that not only displays live transcriptions but also allows users to customize their experience. The interface will include features such as adjusting text size, switching between themes (e.g., dark mode), and selecting the language of transcription. Additionally, users will have the ability to customize speaker labels, view summarized transcriptions, and toggle between original and translated text. Real-time feedback, such as visual cues or notifications, will inform users of the transcription process's status.

## **3.2 Real-Time Speech Recognition and Processing**

The app will capture audio using the mobile device's microphone, processing it in real-time to deliver accurate transcriptions. By leveraging the advanced capabilities of the Google Speech-to-Text API, which incorporates state-of-the-art machine learning technologies, the app will effectively handle various accents, speech speeds, and noisy environments. The app will also employ Speaker Identification to differentiate and label multiple speakers during a transcription session. Optimization techniques will be employed to ensure that transcriptions are delivered with minimal latency.

## **3.3 Success Criteria of the Project**

* **Accuracy of Speech-to-Text Transcription:** Aims to achieve up to 95% accuracy in recognizing and transcribing spoken words across various languages and accents, including the correct identification of multiple speakers.
* **Efficiency of Processing:** Ensures that transcription appears within 1 second of the speech being spoken.
* **User Interface Usability:** Receives positive feedback on ease of use, customization options, and overall user experience from a diverse group of users.
* **Accessibility and Multilingual Support:** Successfully supports multiple languages, making the app accessible to a wide audience, and effectively bridges language barriers through real-time translation.
* **Positive User Feedback:** Gains strong, positive feedback from users regarding the app's performance, accuracy, and overall utility in real-world scenarios.

# **4. Engineering Process**

## **4.1 The Process of Crafting Software and Conducting Research**

## This process consists of two stages:

### **Stage A – The Design:**

## **Researching Speech Recognition and Accessibility Needs:** We conducted thorough research on existing technologies and challenges faced by individuals with hearing impairments. This research helped us identify essential features for our app, including real-time transcription accuracy, multilingual support, and user-friendly interfaces.

## **Exploring Existing Apps and Speech Recognition Technologies:** We analyzed current transcription apps and technologies, identifying their strengths and weaknesses. This exploration guided us in enhancing our app with features like Speaker Identification, Smart Summarization, and Real-Time Translation.

## **Defining Project Requirements and Features:** We outlined the specific needs our app will address, such as real-time transcription, multilingual support, and accessibility. We also considered advanced capabilities like speaker identification and summarization to meet diverse user needs, clearly defining the project’s scope and objectives.

## **Selecting Tools and Technologies:** We determined the criteria for selecting the most suitable tools, technologies, and frameworks. Key considerations included integrating APIs that support advanced features like speaker identification and real-time translation. The final selection will be detailed in a later section.

## **Designing the System Architecture:** We defined the overall structure of our application, including component interactions, data flow, and system integration. Diagrams were created to represent user interactions and real-time data processing, ensuring a seamless experience while supporting advanced features and maintaining scalability.

* **Writing the activity and the use case diagrams:** we want to visually represent the flow of activities and the actions within the system.The use cases describe interactions between users (actors) and the system to achieve specific goals.

## **Stage B – The Implementation:**

* **Integrating and Customizing Google APIs:** Integrate the Google Cloud Speech-to-Text, Translation, and Natural Language APIs, customizing them for optimal transcription accuracy, speaker identification, and real-time translation and summarization.
* **Building the Backend Infrastructure:** Develop the backend using Node.js with Express to handle API requests, data processing, and integration with Firebase Firestore for real-time data storage and synchronization.
* **Developing the Frontend Interface:** Create the user interface using Flutter, ensuring it is intuitive and responsive across different devices, with a focus on real-time display of transcriptions, translations, and summaries.
* **Implementing Real-Time Processing Features:** Develop and integrate real-time transcription, speaker identification, translation, and summarization features, ensuring they work seamlessly together with minimal latency.
* **Testing the Application:** Conduct comprehensive testing, including unit, integration, and system tests, to validate the app’s functionality, performance, and user experience across various environments and devices.
* **Managing Errors and Debugging:** Address and resolve any errors or bugs identified during testing, ensuring the app operates smoothly and reliably.
* **Gathering User Feedback:** Collect feedback from users after initial development to refine and improve the app’s usability, features, and overall experience, making necessary adjustments based on user input.

### **4.1.1 Requirements**

#### **4.1.1.1 Functional Requirements**

* **User Authentication:**
  + The system allows users to log in to their accounts using a username and password.
  + The system provides a sign-up feature for new account creation.
  + The system restricts access to registered accounts only, ensuring secure authentication.
* **Session Management:**
  + The system enables users to initiate a transcription session with a single action.
  + The system provides options to pause and resume transcription sessions as needed.
  + The system allows users to end the transcription session, with the final transcription output presented upon completion.
* **Real-Time Transcription:**
  + The system captures spoken language and transcribes it into text in real-time.
  + Transcriptions are delivered with minimal latency, aiming for less than 1 second delay.
  + The system maintains a high level of accuracy, targeting at least 95% across various languages and accents.
  + Users can customize their experience by adjusting text size and toggling between light and dark themes.
  + The system supports multiple languages, allowing users to select their preferred transcription language.
  + **Speaker Identification & Customization:** The system identifies different speakers during a transcription session and allows users to customize speaker labels.
  + **Smart Summarization:** The system provides a summary of the transcription, condensing lengthy transcriptions into key points.
  + **Real-Time Translation:** The system offers real-time translation of transcribed text into multiple languages, based on the user's selection.
* **User Profile Management:**
  + The system allows users to view and manage their profile information.
  + Users can edit details such as their name, preferred language, and theme settings directly from their profile.
* **Feedback and Statistics:**
  + The system provides real-time feedback during transcription, including visual indicators that confirm audio capture.
  + Users can access session statistics, such as transcription accuracy, speaker participation, summary content, and total word count, after each session.

#### **4.1.1.2 Non-Functional Requirements**

* **User Interface Usability:**
  + The user interface is designed to be intuitive and easy to navigate, offering clear options for starting, pausing, and ending transcription sessions.
  + The system allows users to customize the interface, including text size adjustments, theme selection, and language preferences, to enhance their experience.
* **System Performance:**
  + The system processes audio data efficiently, ensuring that transcriptions, summaries, and translations are delivered with minimal latency.
  + It consistently handles multiple languages and speaker identification with high accuracy and speed, regardless of the language or accent.
* **Privacy and Security:**
  + The system securely stores user data, including login credentials, personalized speaker labels, and summary content.
  + The system ensures that all transcribed, summarized, and translated data is handled according to privacy standards and is not stored or shared without user consent.

### **4.1.2 Deciding Which Technologies We Will Use**

* **Transcription Technology:**

**Google Cloud Speech-to-Text API:** This API provides real-time transcription with high accuracy, including Speaker Diarization to identify and label different speakers. This ensures that transcriptions are contextually clear and useful for users.

* **Translation Technology:**

**Google Cloud Translation API:** This API enables real-time translation of transcribed text into multiple languages, making the app accessible globally. It ensures smooth interaction for users from diverse linguistic backgrounds.

* **Summarization Technology:**

**Google Cloud Natural Language API:** This API generates concise summaries of transcriptions, helping users quickly grasp key points. It’s particularly useful in professional and educational settings where time-efficient review is essential.

* **API Integration:**

**Integration Approach:** RESTful API calls will integrate the transcription, translation, and summarization APIs into the Flutter app. This approach ensures efficient processing and instant display of results within the user interface.

* **Data Management:**

**Firebase Firestore:** Firebase Firestore will handle real-time data storage and synchronization across all user devices, managing transcription data, user settings, speaker labels, summaries, and translations. Its flexibility and scalability support the app’s growth.

* **Database Integration:**

**Integration Approach:** Firebase Firestore will be integrated using the Firebase SDK for real-time data storage and updates. This ensures seamless data synchronization across devices and reliable performance as the app scales.

* **Summary of Benefits:** By using Google Cloud’s Speech-to-Text, Translation, and Natural Language APIs with Firebase Firestore, the app delivers a robust, scalable, and efficient solution. These technologies ensure high accuracy, minimal latency, and real-time synchronization, enhancing the user experience with features like speaker identification, summarization, and translation.

### **4.1.3 Environments**

* **Front-End (UI) Development:**
  + **Flutter:** For building the mobile app’s user interface on both iOS and Android.
* **Back-End Development:**
  + **Node.js with Express:** For server-side logic and managing API requests, including transcription, summarization, and translation.
* **Database:**
  + **Firebase Firestore:** For cloud-based, real-time data storage and synchronization.
  + **SQLite (Optional):** For local data storage on the device when offline access is needed.
* **APIs:**
  + **Google Cloud Speech-to-Text API:** For real-time transcription and speaker identification.
  + **Google Cloud Translation API:** For real-time translation of transcribed text.
  + **Google Cloud Natural Language API:** For smart summarization of transcriptions.
  + **Firebase Authentication:** For managing user sign-up, login, and secure access.
* **Platform-Specific Features:**
  + **Native Code Integration:** Kotlin/Java for Android and Swift/Objective-C for iOS as needed.
* **Deployment Tools:**
  + **Xcode:** For iOS app deployment.
  + **Android Studio:** For Android app deployment.

### **4.1.4 Development Methodology**

### Given the dynamic nature and evolving requirements of our real-time transcription app, we've chosen the Agile methodology. Throughout the project, we will operate in iterative cycles, known as sprints, allowing us to continuously refine and enhance the app based on user feedback and testing outcomes.

### Upon completing the initial sprint, we've established a foundational understanding of the system's requirements, developed a prototype of the user interface, and integrated preliminary API functionality. Moving forward, each subsequent sprint will focus on iterating upon these components, implementing new features—such as Speaker Identification & Customization, Smart Summarization, and Real-Time Translation—and validating the system's performance and usability.

### This iterative approach ensures that we can adapt to changes quickly, deliver functional increments regularly, and maintain a user-centered focus throughout the project. By prioritizing flexibility and continuous improvement, we foster a development process that is both responsive and resilient, building confidence as we progress toward our final goal.

### 

### 

### **4.1.5 Challenges**

* **Real-Time Processing and Latency:**
  + Ensuring instant transcriptions, translations, and summaries without delays is critical for user experience.
* **Accuracy of Speech Recognition:**
  + Maintaining high accuracy in transcription and speaker identification, especially in noisy environments, is essential.
* **Data Privacy and Security:**
  + Securing transcribed data, including speaker labels and translations, is crucial for user privacy and compliance.
* **Integration with Third-Party APIs:**
  + Dependence on external APIs, like Google’s, introduces reliability risks and potential service disruptions.
* **Cross-Platform Consistency:**
  + Ensuring consistent performance of all features across iOS and Android despite hardware variations is challenging.

## **4.2 Product**

### **4.2.1 Software Architecture Diagram**

This diagram represents the components of the software system of our app, showcasing the functions, their implementations, and their interrelationships. It primarily involves the client app (Frontend) component, the server (Backend) component, and the database component.

**Client App:**

* The client app is built using Flutter, a cross-platform framework for developing mobile applications.
* It utilizes Dart for programming the user interface and application logic.
* Platform-specific code may be written in Kotlin/Java for Android or Swift/Objective-C for iOS, especially for accessing native features like the microphone. Flutter’s integration capabilities allow this code to be seamlessly incorporated into the Dart codebase, enabling smooth access to native hardware functionalities.

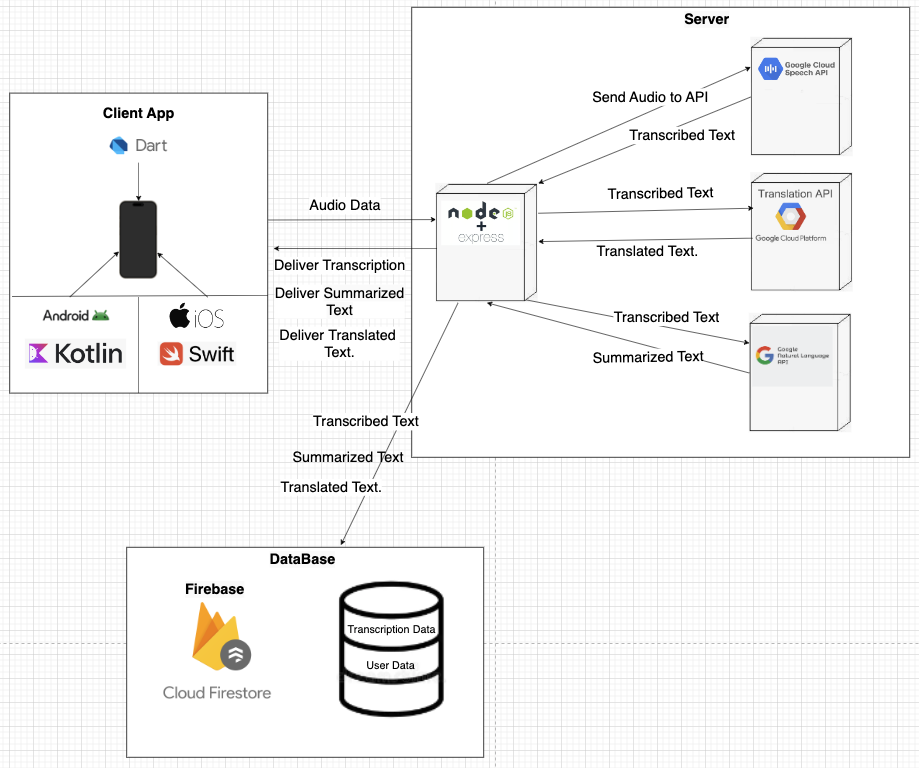
**Server:**

The server component consists of several sub-components:

* **Node.js:** A JavaScript runtime environment for executing server-side code.
* **Express:** A web application framework for Node.js, used to build the app’s backend APIs.
* **Google Cloud Speech-to-Text API:** Manages real-time transcription and speaker identification, processing audio data and converting speech to text.
* **Google Cloud Translation API:** Handles real-time translation of transcriptions into multiple languages.
* **Google Cloud Natural Language API:** Responsible for generating smart summaries of transcriptions.

**Database:**

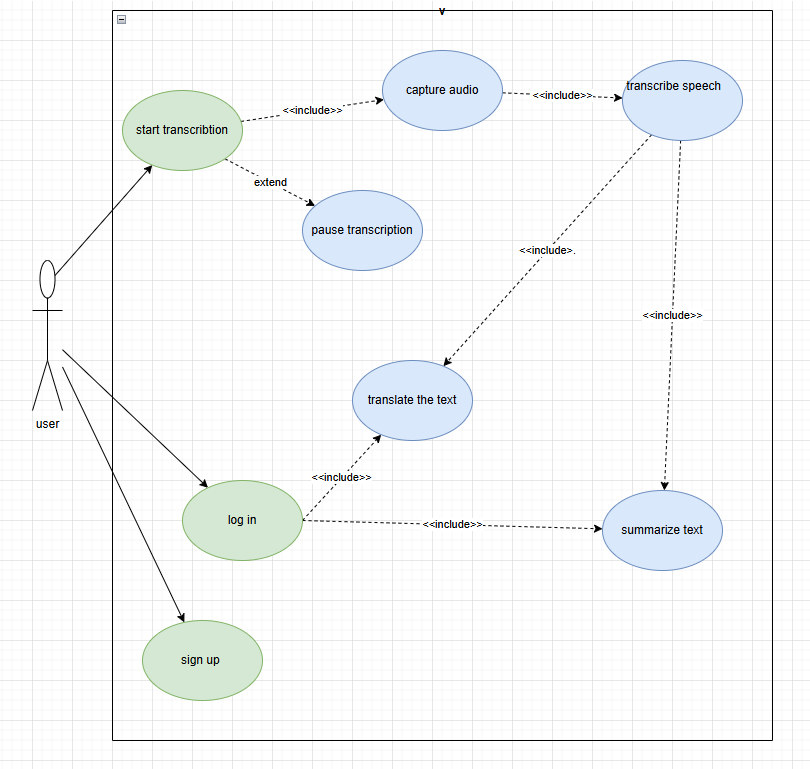
* The app uses Firebase Firestore, a NoSQL cloud database, for storing data, including transcriptions, speaker labels, summaries, translations, and user settings.

****

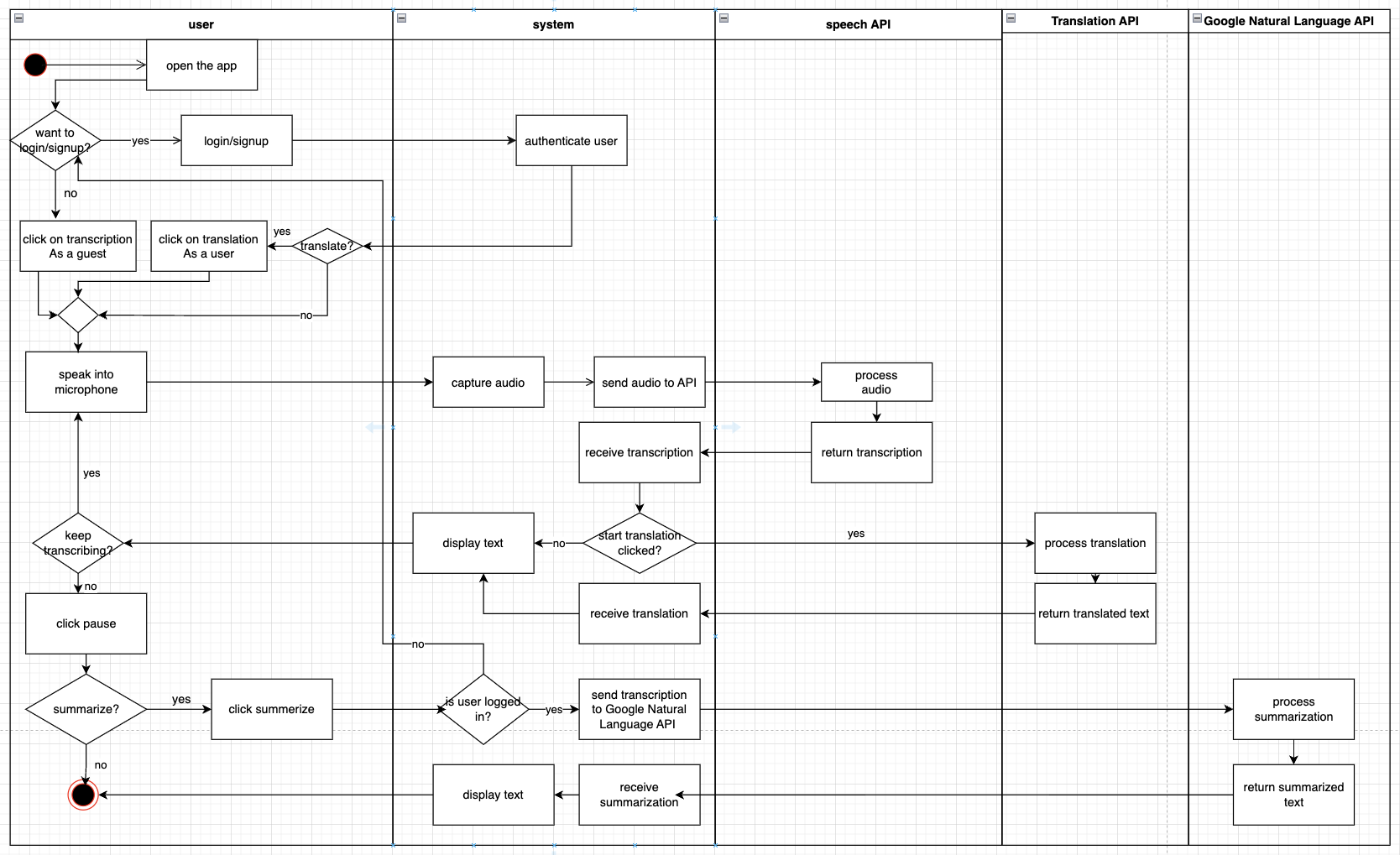
### **4.2.2 UseCase Diagram**

**The use case diagram outlines the key functionalities of the voice-to-text application:**

1. **Start Transcription: Initiates a transcription session, including capturing audio and real-time transcription. The user can also pause the transcription as needed.**
2. **Log In: Authenticates the user, granting access to the app’s features like transcription, translation, and summarization.**
3. **Sign Up: Allows new users to create an account and register with the system.**
4. **Translate the Text: Converts the transcribed text into another language, seamlessly integrated into the transcription process.**
5. **Summarize Text: Provides a concise summary of the transcribed content, useful for quickly understanding key points.**
6. **Pause Transcription: Extends the transcription functionality, enabling users to pause the process during a session.**

****

### **4.2.3 Activity Diagram**



### **The activity diagram presented illustrates the workflow of our voice-to-text application, detailing the interactions between the user and various system components, including the integration with external APIs.**

#### **User Interaction:**

### **Opening the App: The user begins by opening the app, which prompts them to either log in or sign up for a new account. If they choose to continue as a guest, they bypass the login process.**

### **Starting Transcription: Once logged in or as a guest, the user can start a transcription session. This initiates the process where the app begins capturing audio input from the user's microphone.**

### **Speaking into the Microphone: The user speaks into the microphone, and the system captures the audio in real-time.**

### **Pausing and Resuming: During transcription, the user has the option to pause the session and resume it later, adding flexibility to the transcription process.**

#### **System Processing:**

### **Capturing Audio: The system captures the user's audio and sends it to the Speech API for processing.**

### **Processing and Transcription: The Speech API processes the audio and returns the transcribed text to the system, which is then displayed to the user.**

### **Translation: If the user opts for translation, the transcribed text is sent to the Translation API, which processes the translation and returns the translated text.**

### **Summarization: If the user requests a summary, and they are logged in, the transcribed or translated text is sent to the Google Natural Language API. This API processes the text to provide a concise summary, which is then returned and displayed to the user.**

#### **Key Features and Flexibility:**

### **Text Display: Throughout the process, the text, whether transcribed, translated, or summarized, is displayed back to the user for review.**

### **User Choices: The diagram highlights the user's ability to control the workflow, with options to translate, summarize, or simply view the raw transcription.**

### **This activity diagram provides a comprehensive overview of how the app manages user input, processes it using advanced APIs, and delivers the results back to the user in various forms. It emphasizes the app's flexibility and the seamless integration of multiple functionalities to enhance the user experience.**

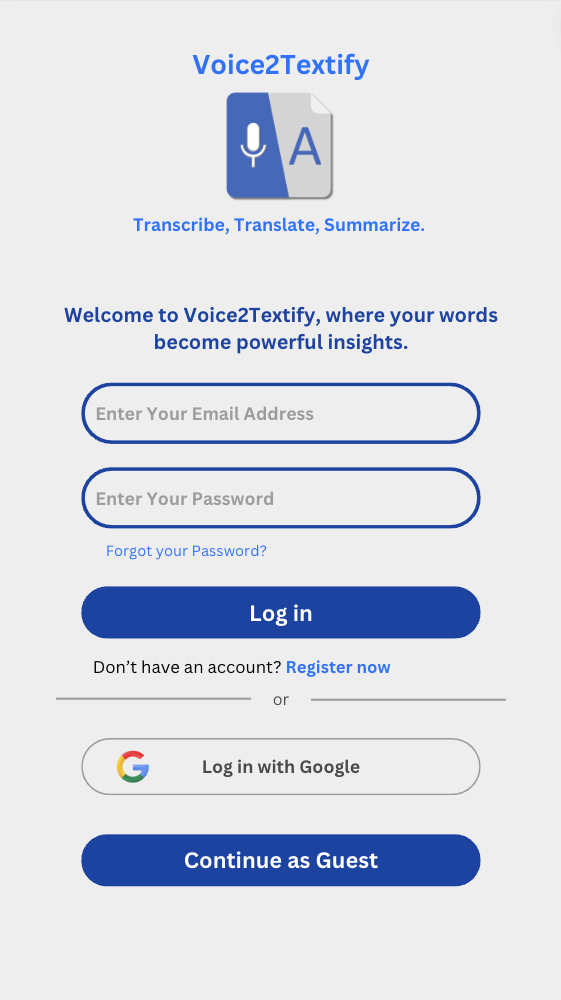
### 

### 

### **4.2.4 User-Friendly Interface**

The login page includes the following elements:

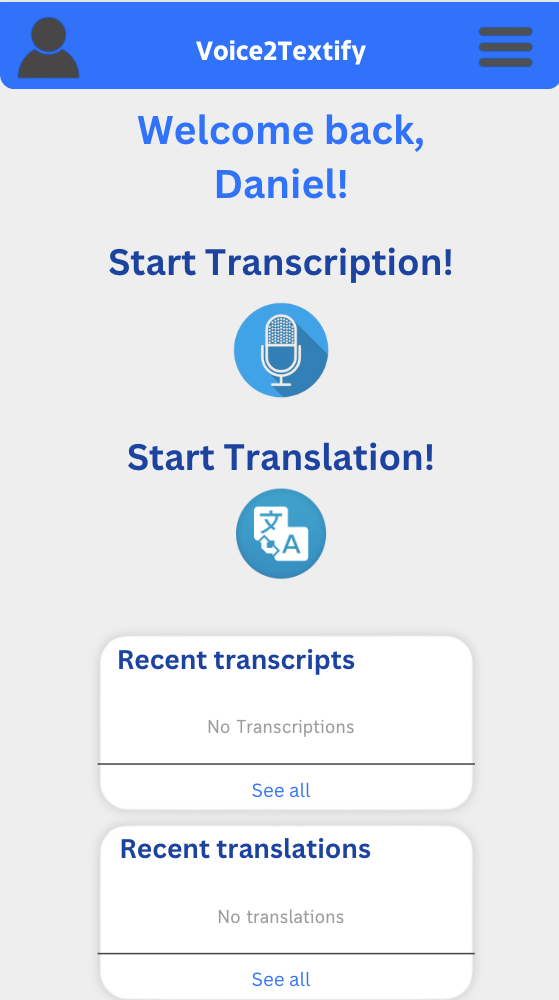
* **Email Field**: Allows the user to enter their email address to log in.
* **Password Field**: Enables the user to input their password for secure access.
* **Log In Button**: Press this button to log in to the app after entering your credentials.
* **Forgot Password Link**: Provides a way to recover the account if the user forgets their password.
* **Register Now Link**: Directs new users to a page where they can create a new account.
* **Google Login Button**: Offers an alternative login method using the user’s Google account for quick access.
* **Continue as Guest Button**: Allows the user to explore the app without logging in, providing a preview of its features.



### **Home Page**

The home page includes the following elements:

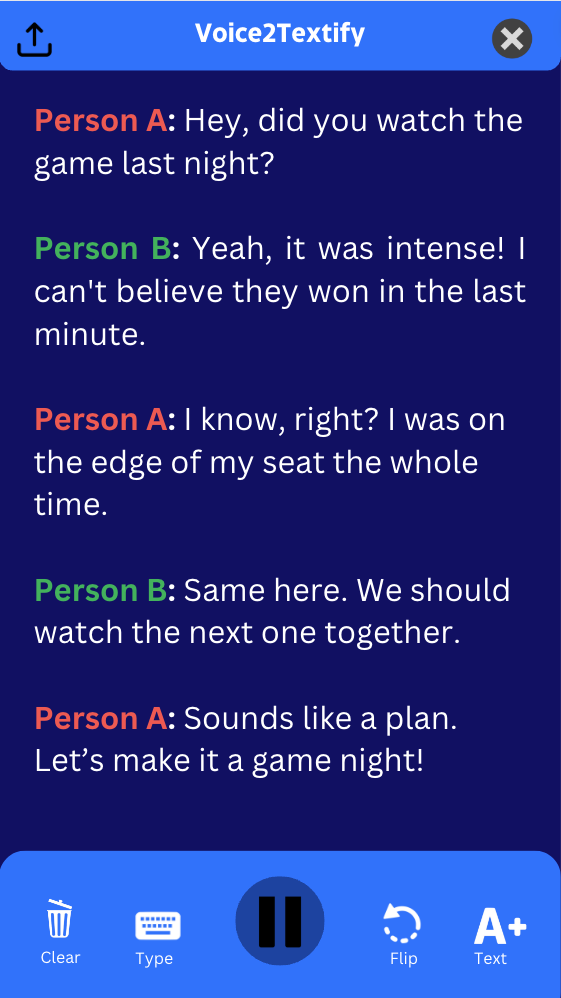
* **Welcome Message**: A personalized greeting that welcomes the user by name, enhancing the user experience by making it more personal.
* **Start Transcription Button**: Central action button to begin a new transcription session immediately.
* **Start Translation Button**: Central action button to begin a new Translation session immediately.
* **Recent Transcripts Section**: Displays the most recent transcriptions the user has completed. A "See all" link allows access to the full history of transcriptions.
* **Recent Translations Section**: Shows the latest translations performed by the user, with an option to view all past translations.
* **Recent Summarizations Section**: Lists the most recent text summarizations. The "See all" link gives access to a complete list of all previous summarizations.



### **Conversation Page**

The Conversation Page provides an interface to view transcriptions of ongoing or recorded conversations:

* **Speaker Identification**: The app differentiates between speakers using different colors and labels (e.g., "Person A" in red and "Person B" in green), making it easier to follow and understand the conversation flow.
* **Clear Button**: Allows the user to clear the current transcription from the screen.
* **Type Button**: Provides an option to manually input text if needed.
* **Pause/Play Button**: Controls the transcription process, allowing users to pause and resume as necessary.
* **Flip Button**: Reverses the order of text display or switches between transcription and another view.
* **Text Size Adjustment**: The "A+" button allows users to increase the text size for better readability.



# **5. Verification Plan:**

## **5.1 Tests**

**1. Test the Accuracy of Real-Time Transcription**

* **Expected Result:** The system achieves at least 95% accuracy in transcribing spoken language into text.
* **Measurement:** Evaluate the percentage of correctly transcribed words out of the total spoken words.
* **Explanation:** This test verifies the core functionality of the app by ensuring that the transcription is accurate across various accents and speech speeds.

**2. Test Speaker Identification and Labeling**

* **Expected Result:** The system correctly identifies and labels different speakers with 90% accuracy.
* **Measurement:** Compare the system’s speaker labels with a manually labeled transcript.
* **Explanation:** Ensuring accurate speaker identification is crucial for users to follow multi-speaker conversations effectively.

**3. Test Real-Time Translation Accuracy**

* **Expected Result:** The system provides accurate translations with at least 85% accuracy across multiple languages.
* **Measurement:** Evaluate the translation accuracy by comparing the system’s output with professional translations.
* **Explanation:** Accurate real-time translation is essential for facilitating communication across different languages.

**4. Test Smart Summarization Precision**

* **Expected Result:** The system generates concise and accurate summaries with at least 80% precision.
* **Measurement:** Compare the system-generated summaries with human-generated summaries for consistency and relevance.
* **Explanation:** This test ensures that the summarization feature provides users with meaningful, condensed versions of lengthy transcriptions.

**5. Test the Guest Mode Functionality**

* **Expected Result:** The system allows transcription without login but restricts access to advanced features.
* **Measurement:** Verify that guests can transcribe but cannot save or use features like translation and summarization.
* **Explanation:** This test ensures that the app behaves as expected in guest mode, providing limited functionality without compromising the user experience.

**6. Test the Sign-Up Process**

* **Expected Result:** The system successfully saves new user data in the database upon sign-up.
* **Measurement:** Compare the input data during sign-up with records stored in the database.
* **Explanation:** This test ensures the reliability of the sign-up process, allowing users to create accounts and access full app features.

**7. Test the Login Process**

* **Expected Result:** The system successfully logs in the user and grants access to their profile with a 95% success rate.
* **Measurement:** Track the success rate of login attempts and verify user profile access.
* **Explanation:** A reliable login process is crucial for user satisfaction, ensuring users can access their data securely.

**8. Test the Transcription Playback and Edit Features**

* **Expected Result:** The system allows users to play back and edit transcriptions smoothly.
* **Measurement:** Monitor the responsiveness and accuracy of the playback and editing features.
* **Explanation:** Ensuring that users can review and edit their transcriptions is vital for usability and overall app functionality.

**10. Test the Database's Data Integrity**

* **Expected Result:** The system reliably stores and retrieves user data, including transcriptions, translations, and summaries.
* **Measurement:** Compare database entries before and after saving or updating to ensure consistency.
* **Explanation:** Ensuring data integrity is essential for maintaining trust in the app’s ability to manage user information correctly.

**12. Test the User Interface on Different Devices and Screen Sizes**

* **Expected Result:** The app displays correctly and is fully functional across various devices and screen sizes.
* **Measurement:** Verify UI consistency on multiple devices (smartphones, tablets) and resolutions.
* **Explanation:** Ensuring that the app is responsive and visually consistent across devices is key to providing a good user experience.

**14. Test User Profile Management and Updates**

* **Expected Result:** The system accurately saves updates to user profiles and reflects changes immediately.
* **Measurement:** Check database entries after profile changes and confirm updates are reflected in the UI.
* **Explanation:** This test ensures that users can personalize and manage their profiles effectively, enhancing the overall user experience.

## **5.2 Constraints**

1. **Hardware Limitations:** The performance of the app may be constrained by the capabilities of the mobile devices used by participants. For example, older devices with limited processing power and memory may struggle to handle real-time transcription, speaker identification, translation, and summarization features, potentially leading to delays or reduced accuracy.
2. **Network Connectivity:** The app's reliance on cloud-based services for transcription, translation, and summarization means that its performance is heavily dependent on the quality of the user’s internet connection. Poor or unstable network connectivity can result in delays, interruptions, or reduced accuracy in the real-time processing of speech.
3. **Environmental Factors:** Real-world conditions, such as background noise, overlapping speech, or varying accents, can impact the accuracy of the transcription and speaker identification features. These environmental variables are difficult to control and may lead to inconsistent results in real-time usage.
4. **Data Diversity:** Since the app relies on machine learning algorithms for speech recognition, translation, and summarization, it may be constrained by the diversity and quality of the training data. Limited or biased training data may lead to suboptimal performance, especially when dealing with diverse languages, dialects, accents, and contextual nuances.
5. **Compliance with Privacy Regulations:** The app must comply with data privacy laws, such as GDPR, which restrict how user data can be collected, stored, and processed. These legal constraints may limit the types of data that can be used for improving the app’s features and may require additional safeguards that impact performance.

## **5.3 Assumptions**

1. **Consistent User Interaction:** The testing process assumes that users will interact with the application in a predictable and consistent manner. This means that users will speak clearly and follow the expected usage patterns as designed within the app, such as correctly initiating, pausing, and stopping transcription sessions, and selecting the appropriate language for translation.
2. **Optimal Audio Conditions:** The testing assumes that the application will be used in environments with minimal background noise and clear audio input. Excessive noise, overlapping speech, or poor audio quality can adversely affect the performance of speech recognition and speaker identification capabilities; thus, this presumption ensures that the app is tested under conditions where it can perform optimally.
3. **Stable Network Connectivity:** The testing presumes that there is stable network connectivity available during the use of the application. Reliable network access is necessary for functionalities such as real-time transcription, translation, summarization, user authentication, and data synchronization.

## **5.4 Other Important Considerations: User’s Evaluation and Feedback**

### **5.4.1 User Interface Evaluation**

* Gather feedback from 10 people (from different backgrounds and technical expertise) on the clarity and intuitiveness of the app's interface for users engaged in real-time transcription, translation, and summarization activities.
* Identify any areas for improvement or features that could enhance the overall user experience.
* Some of the questions that they will get:
  + How would you rate the app's interface in terms of clarity and intuitiveness?
  + Was the app easy to navigate and understand on your first use?
  + Were there any points where you felt lost or unsure of what to do next?
  + What specific features or areas do you think could be improved to enhance your experience?

### **5.4.2 Asking for General Feedback**

* Gather overall impressions from 10 people (from different backgrounds and technical expertise) of the usefulness and effectiveness of the real-time transcription app.
* Collect recommendations and suggestions for improvements from users, particularly regarding features that could enhance their experience with transcription, translation, and summarization.
* Some of the questions that they will get:
  + Can you share your overall impressions of the usefulness and effectiveness of the transcription and translation features?
  + How well do you think the system helps in managing and understanding spoken language in real-time?
  + What recommendations or suggestions do you have for improving the system?
  + Are there any specific features you believe could enhance your transcription, translation, or summarization experience?

# **References**

1. Google LLC. (n.d.). *Cloud Speech-to-Text* [API documentation]. Google Cloud. Retrieved from <https://cloud.google.com/speech-to-text?hl=en>
2. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems* (pp. 5998-6008).<https://arxiv.org/abs/1706.03762>
3. Hannun, A., Case, C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., ... & Zhu, Z. (2014). Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*.<https://arxiv.org/abs/1412.5567>
4. Chung, J. S., Weng, C. Y., Han, W., & Hsu, T. (2020). Real-time speech transcription and translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing* (pp. 2735-2744). Association for Computational Linguistics. <https://www.aclweb.org/anthology/2020.emnlp-main.222/>
5. Graves, A., Mohamed, A., & Hinton, G. (2013). Speech recognition with deep recurrent neural networks. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. <https://doi.org/10.1109/ICASSP.2013.6638947>
6. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735-1780. <https://doi.org/10.1162/neco.1997.9.8.1735>
7. Google LLC. (n.d.). *Cloud Translation* [API documentation]. Google Cloud. <https://cloud.google.com/translate>
8. Google LLC. (n.d.). *Cloud Natural Language* [API documentation]. Google Cloud. <https://cloud.google.com/natural-language>
9. Otter.ai. (n.d.). *Otter.ai: Conversation Transcription and Collaboration* [Web application software]. Retrieved from<https://otter.ai/>
10. Google LLC. (n.d.). *Live Transcribe* [Mobile application software]. Google Play. Retrieved from <https://play.google.com/store/apps/details?id=com.google.audio.hearing.visualization.accessibility.scribe>
11. IEEE. (2020). *Securing Data in Cloud and Mobile Computing*. IEEE Journals & Magazines. Retrieved from<https://ieeexplore.ieee.org/document/8570112>