

Software Engineering Department  
ORT Braude College

Capstone Project Phase B

**Voice2Textify**

**Danial Jarrous 211844881**

**Amne Salame 322732561**

**Amne Salame 322732561**

**Contents**

[***Abstract 3***](#_xtqj6ofggjqj)

[***1. Introduction 3***](#_bkcr6b9r1uc2)

[***2. Literature Review 4***](#_j1vnfyilt7ds)

[*2.1 Similar Real-Time Transcription Apps 5*](#_nui44i9n30ot)

[Google Live Transcribe 5](#_5o7ym8jsgm2g)

[Otter.ai 5](#_nm1k0meh3tef)

[*2.2 Possible Approaches and Tools for Real-Time Transcription of Spoken Language into Text 6*](#_dazn2l6k10fg)

[ASR Models 6](#_g32961ly9f0t)

[Recurrent Neural Networks (RNNs) 6](#_pc0z4fpp9df1)

[Speech Recognition Using the speech\_to\_text Flutter Package 7](#_hc9hxx4kc9bp)

[*3.1 User-Friendly Interface 9*](#_r9lobs8lg1jc)

[*3.2 Real-Time Speech Recognition and Processing 9*](#_5211jard9vh)

[*3.3 Success Criteria of the Project 10*](#_wbzaf3hlu671)

[***4. Engineering Process 10***](#_ubne8skckgby)

[*4.1 The Process of Crafting Software and Conducting Research 10*](#_80dc9twgghjf)

[Stage A – The Design: 10](#_tzgmmhp9s1x9)

[Stage B – The Implementation: 11](#_3zwvxyfvei0u)

[*4.1.1 Requirements 12*](#_h8rdvr2rh4af)

[4.1.1.1 Functional Requirements 12](#_t5qa9rvelhtm)

[4.1.1.2 Non-Functional Requirements 13](#_9tss0tqudzv)

[*4.1.2 Deciding Which Technologies We Will Use 13*](#_4bg2aw17991d)

[*4.1.3 Environments 14*](#_5do71sowt0ol)

[*4.1.4 Development Methodology 14*](#_djw6lj1af6ef)

[*4.1.5 Challenges 15*](#_iqy0ffn6qana)

[*4.2 Product 16*](#_qwgd3ec5v3g0)

[*4.2.1 Software Architecture Diagram 16*](#_80tfu5eewkn6)

[Cloud Services 16](#_roz7esx35voh)

[*4.2.2 UseCase Diagram 17*](#_2rq2i9ncspsx)

[*4.2.3 Activity Diagram 18*](#_bh5zl7yvdjom)

[User Interaction: 19](#_4h49pq2mbgxf)

[System Processing: 19](#_fmmv88uyr2sf)

[Key Features and Flexibility: 19](#_lzoj97o8yy9r)

[*4.2.4 User-Friendly Interface 20*](#_ufuxii9yndxa)

[***5. Verification Plan: 25***](#_wrlqm3yanulb)

[*5.1 Tests 25*](#_wvap8v13lkp7)

[*5.2 Constraints 28*](#_esadfqy4gwtf)

[*5.3 Assumptions 28*](#_mi7hfatzpjt0)

[*5.4 Other Important Considerations: User’s Evaluation and Feedback 29*](#_856ia1dl1n5u)

[*5.4.1 User interface Evaluation 29*](#_7menk24hgxan)

[*5.4.2 Asking for General Feedback 29*](#_rccgwua9ugaq)

[**6. The Process of Crafting Software 29**](#_9q9d43mpq5pi)

[**7. Description of Challenges Faced 30**](#_hnawgdh62a1t)

[**8. Conclusions 31**](#_qrowrfhsogs5)

[9. Overall Effectiveness of the Work 32](#_igqqq91n6njk)

[**10. Lessons Learned 32**](#_tm6xjujbxfa1)

[**11. Meeting Project Objectives 33**](#_y414pg79w14l)

[**12. User Guide 33**](#_66a1ceyg7c4s)

[**13. Maintenance Guide for Voice2Textify 39**](#_dp5mswi2vxx1)

[Contact and Support 41](#_j2k5l8spr944)

[***References 42***](#_ib2w3ffr455r)

# 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 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 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.

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.

### **Speech Recognition Using the speech\_to\_text Flutter Package**

The speech\_to\_text Flutter package is an efficient and lightweight solution for implementing speech recognition in Flutter applications. It enables converting spoken language into text directly on the device without the need for external APIs or cloud services. This approach significantly reduces latency and eliminates potential data privacy concerns since the audio data is processed locally on the user's device.

The package offers real-time audio input transcription, making it highly suitable for applications requiring immediate text output, such as transcription apps, note-taking tools, and accessibility solutions. Additionally, the speech\_to\_text package provides features like:

* **Support for Multiple Languages:** Users can select different languages for transcription, enhancing its versatility for global applications.
* **Real-Time Transcription:** The package processes audio input instantly, providing an interactive transcription experience.
* **Platform Support:** The package is compatible with Android and iOS devices, ensuring cross-platform functionality.

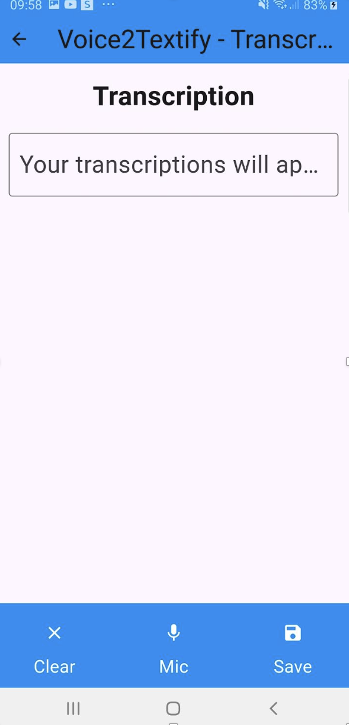
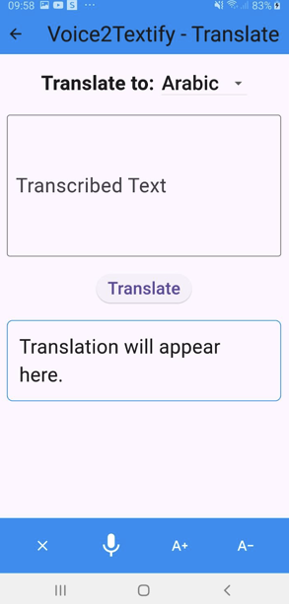
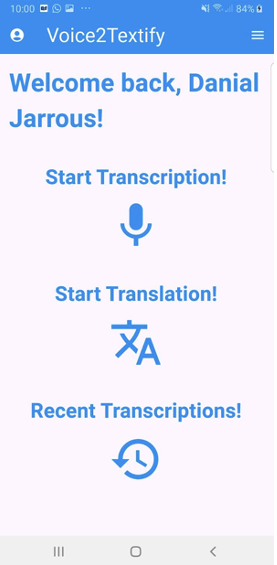
**Expected Achievements**Our project aims to deliver a highly reliable, real-time voice-to-text transcription mobile application designed to enhance communication by instantly converting spoken language into text. The application is user-friendly, accurate, and efficient, offering accessibility and convenience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

By incorporating advanced features, the application addresses 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 creates a comprehensive solution that supports inclusive communication, facilitates better understanding in multilingual environments, and promotes active participation in discussions and events.

Key Objectives:

* Developing an intuitive user interface for seamless interaction.
* Utilizing the speech\_to\_text Flutter package for real-time speech recognition with device-based processing.
* Ensuring real-time processing with minimal latency by leveraging on-device capabilities.
* Enhancing accessibility through multilingual support and text translation features.

# 



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

## 3.1 User-Friendly Interface

We aim to create an intuitive interface that seamlessly displays live transcriptions while offering users the flexibility to customize their experience. The interface will include features such as adjustable text size for improved readability, the ability to view previously saved transcriptions for easy reference, and a convenient option to toggle between the original text and its translated version, enhancing accessibility and user engagement.

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

The app will utilize the mobile device's microphone to capture audio and process it in real-time, providing accurate and immediate transcriptions. Leveraging the robust capabilities of the Speech-To-Text Flutter package, the app is designed to handle a variety of accents, speech patterns, and varying speeds. Additionally, it is optimized to perform reliably even in noisy environments, ensuring a seamless and user-friendly transcription experience across diverse use cases.

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

* **Accuracy of Speech-to-Text Transcription:** Aims to deliver reliable transcriptions with accuracy rates aligning with the capabilities of the Speech-To-Text Flutter package. The app is optimized to handle a variety of accents and speech speeds, though perfect accuracy in noisy environments or for less common accents may vary based on the package's limitations.
* **Efficiency of Processing:** Ensures that transcriptions are displayed in near real-time, providing updates within seconds of speech being spoken, offering a smooth and responsive user experience.
* **User Interface Usability:** Features a streamlined and intuitive interface that allows users to easily view live transcriptions, adjust text size, toggle between original and translated text, and access past transcriptions. Usability is prioritized for seamless interaction across different user demographics.
* **Accessibility and Multilingual Support:** Successfully supports multiple languages through text translation. While real-time transcription is primarily optimized for single-language inputs, the translation feature bridges language barriers effectively.
* **Positive User Feedback:** Focuses on delivering strong, positive feedback by providing an accessible, customizable, and practical solution for transcription needs. Real-world usability, including the ability to save and revisit past sessions, enhances the app's overall utility and user satisfaction.

# **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 extensive research on the use of speech recognition technologies and their applications in various settings. The focus was on identifying user needs, including real-time transcription, basic translation capabilities, and customizable user interfaces. This research guided us in tailoring the app to enhance accessibility and usability for diverse audiences.
* **Exploring Existing Apps and Speech Recognition Technologies:** We analyzed popular transcription apps and the available Flutter packages for speech-to-text. This included identifying strengths and limitations in terms of accuracy, real-time performance, and ease of integration. Based on this research, we selected the speech\_to\_text Flutter package, ensuring reliable and efficient transcription capabilities.
* **Defining Project Requirements and Features:**

We outlined the app's core functionalities, including:

* Real-time transcription with live updates.
* Multilingual translation using Google Translate API.
* Easy-to-navigate interface with features like text resizing and past transcription access.

We prioritized features based on user needs and technical feasibility, simplifying the scope while ensuring essential functionalities were included.

* **Selecting Tools and Technologies:** After evaluating various tools and frameworks, we chose Flutter for its cross-platform development capabilities and integration with the speech\_to\_text package. For translations, the Google Translate API was integrated, and its use was clearly documented. Firebase was selected for authentication, storage, and database management due to its seamless compatibility with Flutter.
* **Designing the System Architecture:**
  + **Frontend:** Flutter framework for a responsive and intuitive user interface.
  + **Backend:** Firebase for managing user authentication and storing transcription data.
  + **APIs:** Integration of the Speech-To-Text Flutter package for real-time transcription and Google Translate API for multilingual support.
* **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 Speech-to-Text Package:**
  + Integrate the speech\_to\_text Flutter package for real-time transcription. The package will be configured to process audio input efficiently and display live transcription on the app interface.
  + Integrate the Google Translate API for multilingual translation, allowing developers to insert their own API keys for ease of use.
* **Building the Backend Infrastructure:** Utilize Firebase Firestore as the backend to manage user sessions and store transcriptions. Firebase Authentication will be used to distinguish between guest users and logged-in users.
* **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 and translations.
* **Implementing Real-Time Processing Features:**
  + Plan to implement real-time transcription using the speech\_to\_text package, ensuring minimal latency and high accuracy.
  + Integrate Google Translate API to allow real-time text translation into multiple languages.
  + Future scope includes the addition of advanced features, such as speaker identification and smart summarization, to enrich user experience.
* **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.
  + Guest users are permitted to access limited features, such as saving transcriptions.
  + Secure authentication is ensured through Firebase 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.
  + The system saves each transcription session uniquely for logged-in users in the database, allowing easy retrieval.
* **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 ensures high transcription accuracy, but this may vary depending on the language and background noise.
  + Users can customize their experience by adjusting text size directly from the interface.
* **Multilingual Translation:**
  + The system offers real-time translation of transcribed text into multiple languages using the Google Translate API.
  + Users must configure their own API key for translation functionality.
* **Recent Transcriptions:**
  + Logged-in users can view a list of their recent transcription sessions, enabling easy access to previous work.

#### **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 and language preferences, to enhance their experience.
* **System Performance:**
  + The system processes audio data efficiently, ensuring that transcriptions 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, transcription sessions, and personalized settings, using Firebase Authentication and Firestore.
  + Transcriptions and user data are accessible only to the authenticated user and are not shared with third parties without explicit user consent.

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

* **Transcription Technology:**

**Speech-to-Text Flutter Package:** This package will provide real-time transcription capabilities directly within the app. It offers a lightweight and efficient solution for capturing and transcribing spoken language into text in real-time. This package will be evaluated for its accuracy, handling of accents, and performance in various environments.

* **Translation Technology:**

**Google Cloud Translation API:** This API will allow the app to translate transcribed text into multiple languages. By integrating this API, the app aims to make transcription results accessible to a global audience, supporting diverse linguistic needs.

* **API Integration:**

**Integration Approach:** The app will utilize RESTful API calls to integrate translation functionality. The Speech-to-Text Flutter package will handle transcription directly within the app, reducing dependency on external API calls and ensuring efficient performance.

* **Data Management:**

**Firebase Firestore:** Firebase Firestore will be used for real-time data storage and synchronization. It will manage transcription sessions, user profiles, and other app data while ensuring data integrity and scalability. This will allow users to access their data seamlessly across devices.

* **Summary of Benefits:** By leveraging the Speech-to-Text Flutter package, Google Cloud Translation API, and Firebase Firestore, the app will deliver a reliable, scalable, and efficient solution.

### **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:**
  + **Firebase:** Utilized for authentication, data management, and handling user sessions directly, eliminating the need for a custom backend infrastructure.
* **Database:**
  + **Firebase Firestore:** Chosen for its real-time cloud-based data storage, enabling synchronization of transcription sessions and user-specific data.
* **APIs:**
  + **Google Cloud Translation API:** ntegrated to provide text translation, making transcriptions accessible in multiple languages.
  + **Firebase Authentication:** Handles user sign-up, login, guest access, and secure authentication without the need for custom server implementation.
* **Deployment Tools:**
  + **Xcode:** For iOS app deployment.
  + **Android Studio:** For Android app deployment.

### **4.1.4 Development Methodology**

The dynamic and evolving requirements of our transcription app have led us to adopt the **Agile methodology** as the foundation for development. Agile emphasizes iterative improvement, flexibility, and a user-centered approach, allowing the team to adapt to feedback and deliver a product that meets real-world needs effectively. The development process is divided into iterative cycles, known as sprints, each focused on refining existing features and implementing new functionality.

In the initial sprint, the team concentrated on establishing a functional prototype that included basic transcription capabilities using the Speech-to-Text Flutter package. This phase also involved developing a foundational user interface that supports real-time transcription sessions and ensures a seamless user experience. Subsequent sprints are planned to enhance these core functionalities, incorporating features such as viewing past transcriptions, enabling text translation, and managing transcription sessions for logged-in users. Each iteration will also include rigorous testing to validate the system’s usability and performance in real-world scenarios.

User feedback plays a pivotal role in this process, guiding refinements and adjustments throughout the project. Early prototypes and implemented features will be evaluated by target users, focusing on improving transcription accuracy, multilingual support, and overall app responsiveness. This feedback loop ensures that the app evolves in alignment with user expectations and preferences.

The Agile methodology enables the team to remain responsive to changes and emerging requirements. Features planned for future development, such as offline functionality or advanced analytics, will be integrated based on priority and feasibility in subsequent sprints. By prioritizing flexibility and continuous improvement, this methodology ensures that the app is not only functional but also reliable, intuitive, and inclusive, fostering user confidence and delivering a valuable transcription solution.

### 

### 

### 

### **4.1.5 Challenges**

* **Real-Time Processing and Latency:**
  + Ensuring that transcriptions are delivered with minimal delay is critical for maintaining a smooth user experience. While real-time transcription is achievable, balancing performance and responsiveness remains a challenge, particularly for advanced features like translation or summarization.
* **Accuracy of Speech Recognition:**
  + Maintaining high accuracy in transcription, especially in noisy environments, is essential.
* **Data Privacy and Security:**
  + Securing transcribed data, including user credentials, 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**

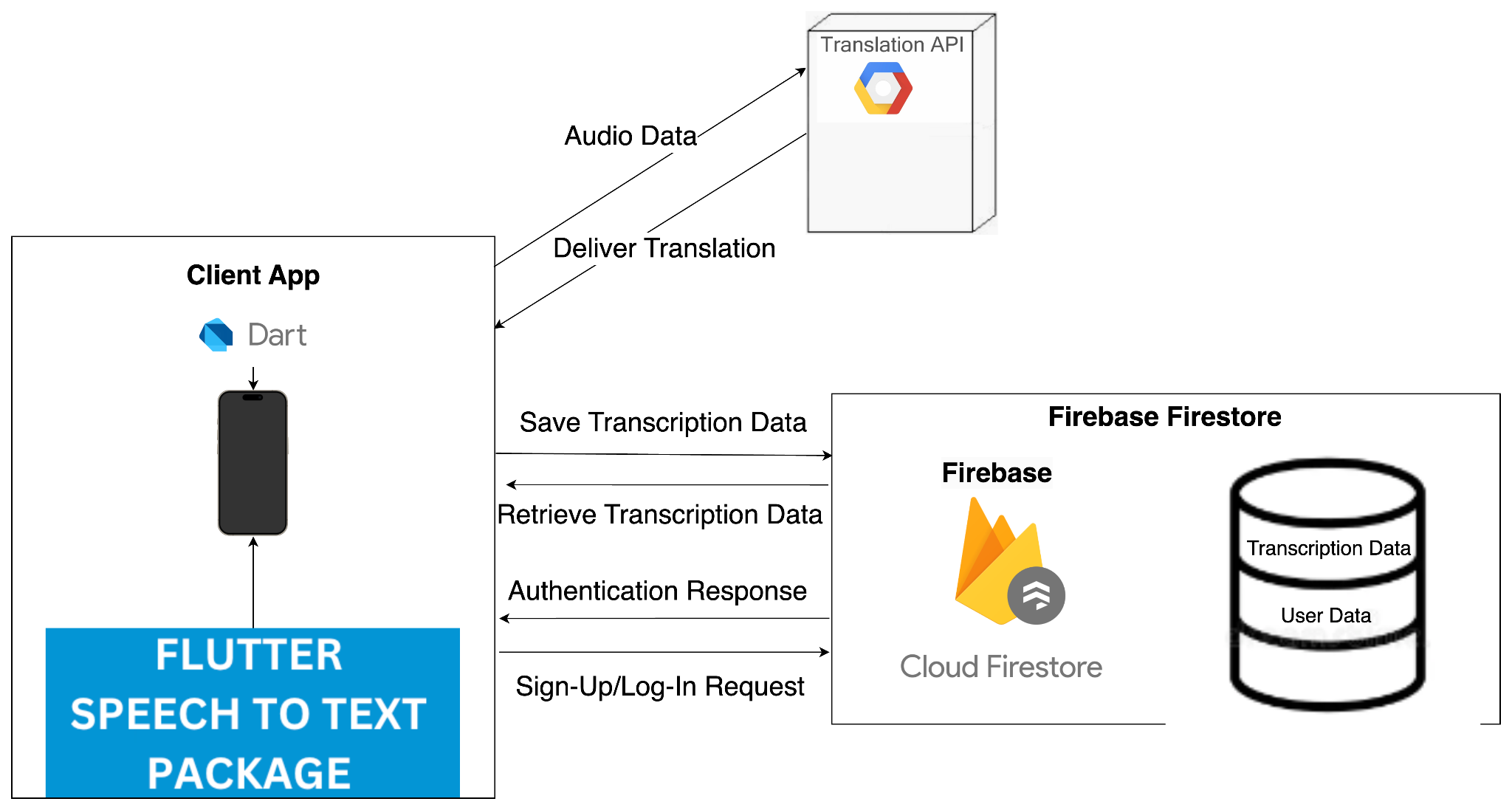
This section outlines the architecture of the Voice2Textify app, showcasing its components and how they work together to deliver transcription, translation, and other features. The app is built as a client-centric solution using Flutter, integrated directly with cloud-based services like Firebase and Google APIs.

**Client App:**

* **Flutter Framework**: The entire application is developed using Flutter, a cross-platform framework, which ensures consistent performance on both Android and iOS.
* **Dart Programming Language**: Handles the app’s user interface, logic, and integration with external services.
* **Native Features**: Flutter’s plugins, such as speech\_to\_text and Firebase plugins, are used to access native functionalities like the microphone and real-time database connectivity.

#### **Cloud Services**

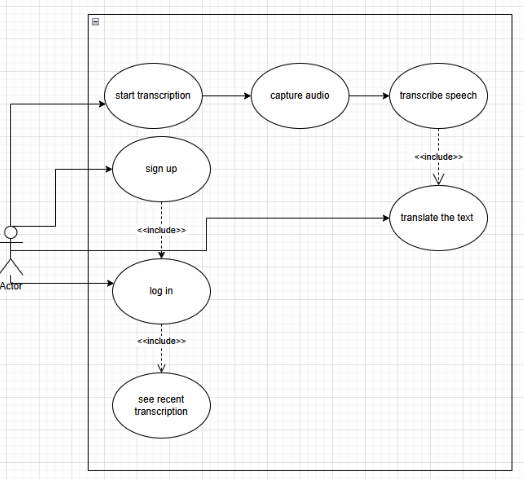
1. **Firebase**:
   * **Authentication**: Handles user authentication (email/password and guest access).
   * **Firestore Database**: A NoSQL cloud database is used to store transcriptions, user settings, and session data in real time.
2. **Google APIs**:
   * **Google Cloud Translation API**: Allows for text translation into multiple languages to enhance accessibility.

****

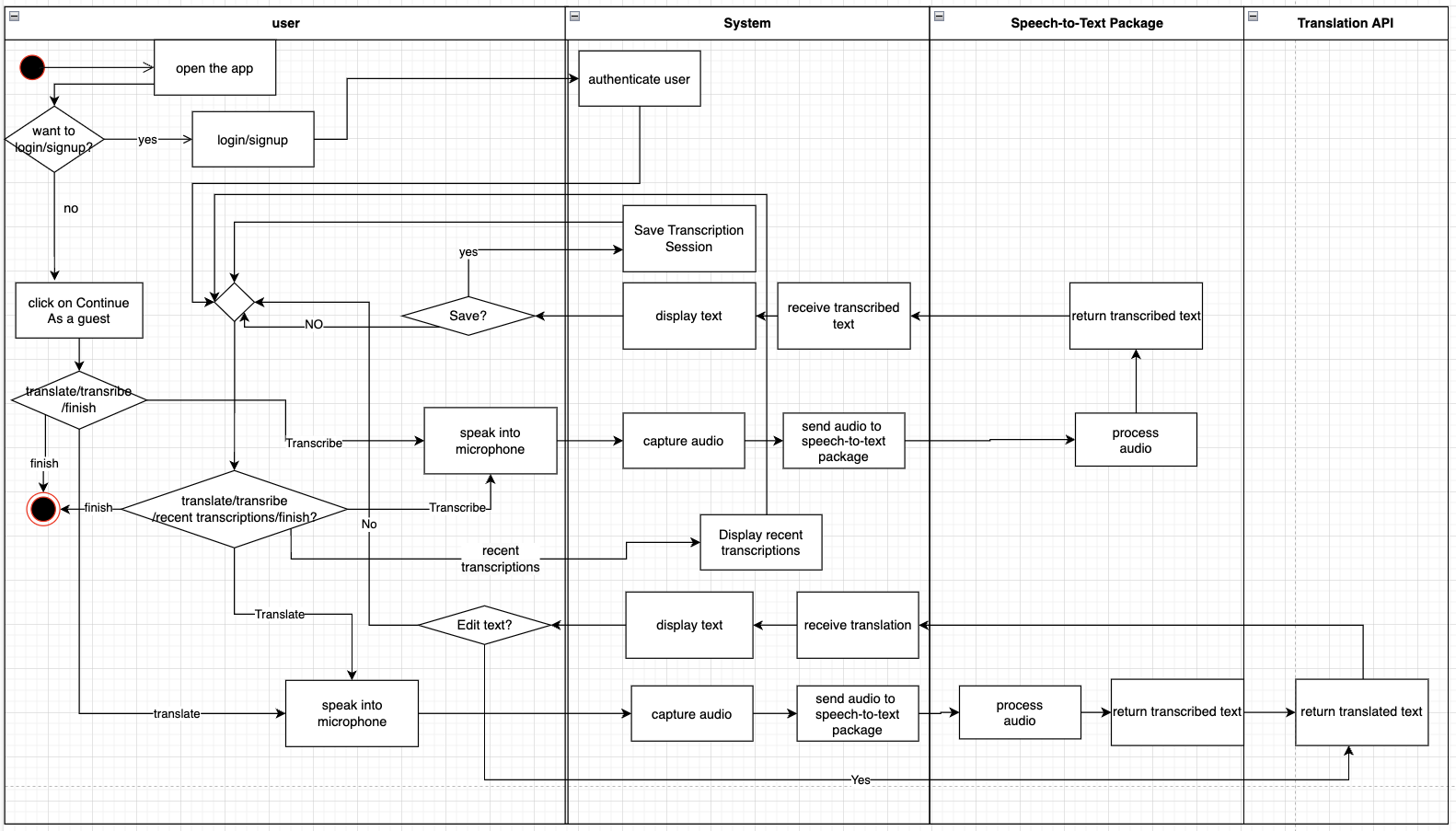
### **4.2.2 UseCase Diagram**

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

* **Start Transcription**: Allows the user to begin a transcription session, which involves capturing audio and transcribing speech into text. The feature includes an option to translate the transcribed text into another language.
* **Sign Up**: Enables new users to create an account and register in the system, granting access to personalized features such as saving and viewing previous transcriptions.
* **Log In**: Authenticates users, allowing them to access their saved transcriptions and use the app's features.
* **See Recent Transcriptions**: Provides logged-in users with access to their previously saved transcriptions.
* **Translate the Text**: An optional feature integrated into the transcription process, allowing users to convert the transcribed text into a different language.

****

### **4.2.3 Activity Diagram**



The activity diagram illustrates the workflow of the voice-to-text application, detailing the interactions between the user, the system, and external APIs for transcription and translation.

### **User Interaction:**

* **Opening the App**: The user starts by opening the app, where they can either log in, sign up, or continue as a guest. Guest users bypass the login process but have limited functionality.
* **Choosing an Action**: After logging in or continuing as a guest, the user selects an action: start transcription, translate text, view recent transcriptions, or finish their session.
* **Starting Transcription**: The user initiates a transcription session, during which they speak into the microphone, and the system captures audio in real time.
* **Saving Transcription**: Logged-in users have the option to save their transcription session. Guest users can view transcriptions but cannot save them.
* **Accessing Recent Transcriptions**: Logged-in users can view their previously saved transcriptions through the "recent transcriptions" feature.

### **System Processing:**

* **Capturing Audio**: The system captures the audio input from the user and processes it for transcription.
* **Transcription Processing**: Audio data is sent to the Speech-to-Text package, which processes the audio and returns the transcribed text.
* **Displaying Text**: The transcribed text is displayed to the user in real-time for review.
* **Translation**: If the user chooses to translate the text, the transcribed text is sent to the Translation API. The translated text is then returned and displayed to the user.

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

* **Editing Text**: Users can edit the translated text directly within the app for customization.
* **Finish Session**: Users can choose to finish their session, which redirects them to select another action or exit the app.
* **Real-Time Feedback**: The system provides immediate feedback through text display and allows the user to control the workflow by pausing, saving, or resuming transcription sessions.

### 

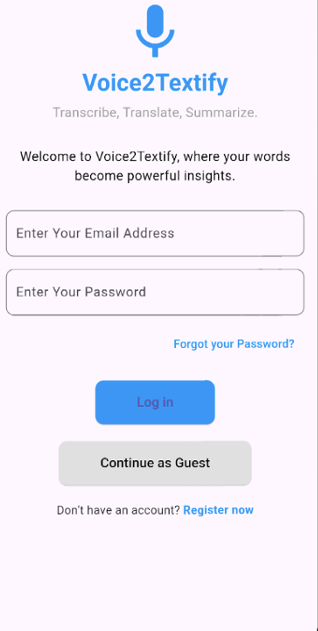
### 

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

**Login Screen**

This page allows users to log in, continue as a guest, or create a new account to access the app.

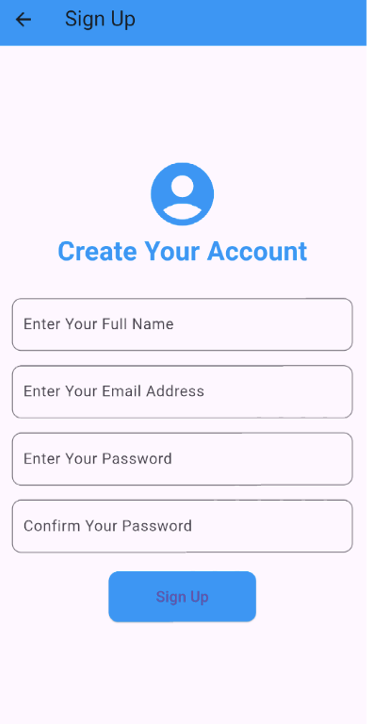
* **Login Form**: Users can enter their **Email Address** and **Password** to log in.
* **Guest Access**: Tap **Continue as Guest** to explore the app without logging in.
* **Password Recovery**: Tap **Forgot your Password?** to reset your password.
* **Sign-Up Prompt**: Tap on **Register now** if you don’t have an account.

****

**Sign-Up Screen**

This screen allows new users to create an account and register for full access to the app features.

* **Input Fields**:  
  Users must fill out the following fields to create an account:
  + **Full Name**: Enter your full name for account identification.
  + **Email Address**: Provide a valid email address to be used for login and communication.
  + **Password**: Create a secure password for your account.
  + **Confirm Password**: Re-enter the password to confirm it matches.
* **Sign-Up Button**:  
  Tap the *Sign Up* button to submit your details and complete the registration process.
* **Navigation**:
  + Use the back arrow at the top left to return to the previous screen.
  + Ensure all fields are filled correctly before tapping *Sign Up*. Errors or mismatches will prompt a notification to correct them.

****

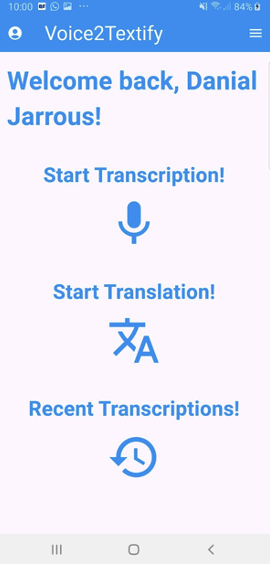
**Home Screen**

This is the main page where users can access key app features.

* **Start Transcription**:  
  Tap the microphone icon to begin voice-to-text transcription.
* **Start Translation**:  
  Tap the translation icon to start translating spoken language into another language.
* **Recent Transcriptions**:  
  Tap the clock icon to view previous transcription sessions.
* **Menu (Burger Icon)**:  
  Tap the menu icon to access additional options, including the **Log Out** button to securely log out.

**Notes:**

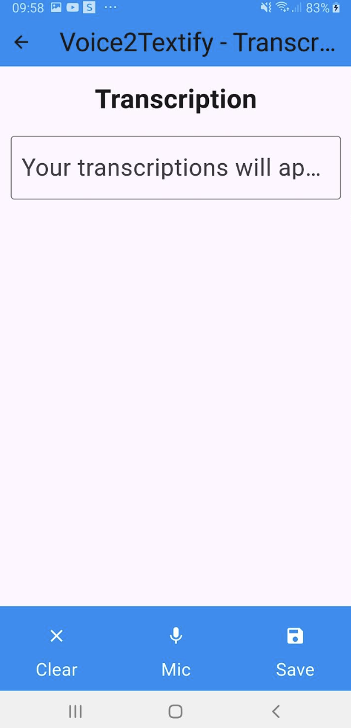
* **Guest Users**:  
  Some features, such as saving transcriptions, are restricted for guest users. These users can explore the app but will have limited functionality.
* **User-Specific Features**:  
  Data displayed, such as recent transcription sessions, is personalized and unique to the logged-in user. Guest users will not have access to saved sessions or personalized data.



**Transcription Screen**

This page allows users to transcribe speech into text in real time. The features available on this page include:

* **Mic Button**:  
  Tap to start or pause the transcription process. Speech will be transcribed and displayed on the screen in real time.
* **Clear Button**:  
  Tap to clear the current transcription from the screen.
* **Save Button**:  
  Tap to save the current transcription to your account. This feature is unavailable for guest users.
* **Real-Time Transcription Display**:  
  The transcription appears on the screen as you speak, updating in real time.



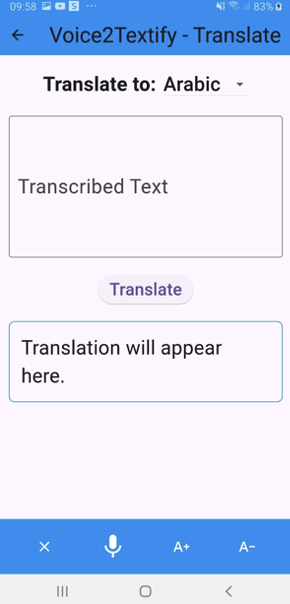
**Translate Screen**

This page allows users to translate transcribed English text into a selected language.

* **Language Selection Dropdown**:  
  Choose the target language for translation from the dropdown menu.
* **Transcribed Text Input Field**:  
  Displays the text you wish to translate.
* **Translate Button**:  
  Tap to translate the displayed text into the selected language. The translated text will appear below.
* **Translation Output Field**:  
  Displays the translated text after clicking the Translate button.
* **Font Size Adjustments**:
  + **A+ Button**: Increases the font size of both the input and output fields.
  + **A- Button**: Decreases the font size of both the input and output fields.
* **Clear Button**:  
  Tap to clear the transcribed text and translation fields.

**Notes for Users**:

* You can input text manually or through voice transcription for translation.

****

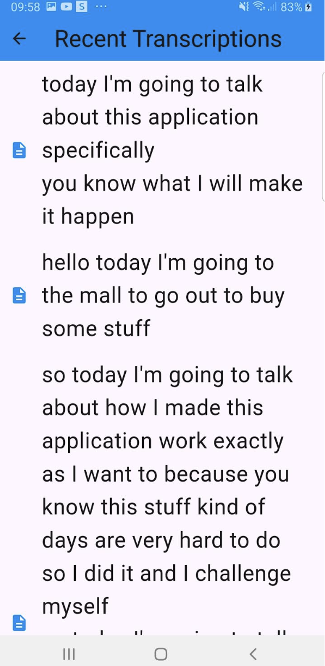
**Recent Transcriptions Screen**

This page allows users to view their previously saved transcription sessions. Features include:

* **List of Transcriptions**:  
  Each item displays the text from a saved transcription session, organized in chronological order.
* **Copy Button**:  
  Tap the copy icon next to a transcription to copy its content to the clipboard for use in other applications.

**Notes**:

* Transcriptions are saved only for logged-in users. Guest users do not have access to saved sessions.
* Each transcription represents a session recorded during usage, ensuring easy access to past conversations or notes.



# **5. Verification Plan:**

## **5.1 Tests**

**Test Real-Time Transcription Accuracy**

* **Expected Result**: The system captures spoken language and provides text transcriptions with at least 85% accuracy.
* **Measurement**: Compare the transcription output with a manually transcribed version of the same audio.
* **Explanation**: Ensures the app reliably transcribes spoken language, even in varying accents and moderate background noise.

**Test Translation Accuracy**

* **Expected Result**: The system translates transcribed text with at least 80% accuracy into supported languages.
* **Measurement**: Compare the system-generated translation with professionally translated text.
* **Explanation**: Verifies the app's ability to translate transcriptions effectively, ensuring cross-lingual accessibility.

**Test Guest Mode Functionality**

* **Expected Result**: Guest users can use transcription features but are restricted from saving or accessing advanced options like translation.
* **Measurement**: Verify that guest users cannot perform actions like saving transcriptions or accessing past sessions.
* **Explanation**: Confirms that the app behaves as expected for guest users while limiting access to premium features.

**Test Sign-Up Process**

* **Expected Result**: The app successfully registers new users and saves their data in Firebase.
* **Measurement**: Compare the input data during sign-up with the corresponding entries in Firebase Firestore.
* **Explanation**: Ensures new users can create accounts and access full features without issues.

**Test Login Process**

* **Expected Result**: The app correctly authenticates registered users and grants access to their profiles.
* **Measurement**: Track login success rates and verify that logged-in users can access their saved data.
* **Explanation**: Ensures secure and seamless login functionality for registered users.

**Test Transcription Session Management**

* **Expected Result**: The app allows users to start, pause, and stop transcription sessions smoothly.
* **Measurement**: Verify that all actions (start, pause, stop) function without errors and produce expected outputs.
* **Explanation**: Ensures the app’s transcription session management is user-friendly and reliable.

**Test Data Storage and Retrieval**

* **Expected Result**: Transcriptions, translations, and session data are reliably stored in Firebase Firestore and retrieved accurately when accessed.
* **Measurement**: Compare saved data in Firestore with what is displayed in the app.
* **Explanation**: Confirms that user data is securely stored and accessible as needed.

**Test Save Feature for Logged-In Users**

* **Expected Result**: Logged-in users can save their transcription sessions successfully.
* **Measurement**: Verify that saved data appears in the "Recent Transcriptions" section and matches the transcription content.
* **Explanation**: Ensures that saving transcriptions is functional for logged-in users.

**Test User Interface Responsiveness**

* **Expected Result**: The app displays consistently and functions seamlessly across different devices and screen sizes.
* **Measurement**: Check UI layout and functionality on various devices, including smartphones and tablets.
* **Explanation**: Confirms that the app is visually and functionally optimized for a wide range of devices.

**Test Text Size Adjustment**

* **Expected Result**: Users can adjust the transcription text size in real time, and the changes reflect immediately.
* **Measurement**: Verify that text size increases or decreases as per user input without disrupting the layout.
* **Explanation**: Ensures usability by allowing users to customize their transcription experience.

**Test Recent Transcriptions Retrieval**

* **Expected Result**: Logged-in users can view their past transcription sessions from the "Recent Transcriptions" section.
* **Measurement**: Check the accuracy and completeness of retrieved sessions compared to saved data in Firestore.
* **Explanation**: Verifies that the app correctly displays past session data for logged-in users.

**Test Firebase Authentication Functionality**

* **Expected Result**: The app handles authentication errors gracefully (e.g., incorrect password, invalid email).
* **Measurement**: Simulate authentication errors and verify that appropriate error messages are displayed.
* **Explanation**: Ensures that the app provides clear feedback to users when authentication issues occur.

**Test API Key Configuration**

* **Expected Result**: The app functions as expected when a valid API key is provided for the translation feature.
* **Measurement**: Remove or replace the API key and observe if the app notifies users about the missing key.
* **Explanation**: Ensures that the app provides clear guidance for setting up required APIs during deployment.

**Test Transcription Playback (Optional)**

* **Expected Result**: Users can view their saved transcriptions in a readable format and navigate through them easily.
* **Measurement**: Open recent transcriptions and verify the readability and accessibility of saved data.
* **Explanation**: Enhances usability by allowing users to revisit past sessions.

## **5.2 Constraints**

1. **Hardware Limitations**The app's performance may be affected by the processing capabilities of users' mobile devices. Older devices with limited memory and processing power might struggle to handle real-time transcription and translation features, leading to potential delays or reduced accuracy.
2. **Network Connectivity**As the app relies on cloud-based services (e.g., Google Cloud Translation API) and Firebase Firestore, its performance is highly dependent on the quality of the user's internet connection. Poor or unstable connectivity may cause delays, interruptions, or inaccuracies in transcription and translation.
3. **Environmental Factors**Background noise, overlapping speech, and diverse accents can affect the accuracy of the transcription feature. These factors are often unpredictable and may lead to inconsistent results during real-world use.
4. **Data Diversity**The accuracy of the transcription and translation features depends on the diversity and quality of the data used to train the underlying machine learning models. Limited or biased data may reduce performance, particularly for underrepresented languages, dialects, and accents.
5. **Compliance with Privacy Regulations**The app must adhere to data privacy laws, such as GDPR. These legal requirements may restrict how user data (e.g., transcriptions) is collected, stored, and processed. Ensuring compliance may necessitate additional safeguards that could impact performance or limit certain features, such as data retention for analytics or improvement.

## **5.3 Assumptions**

1. **Consistent User Interaction**It is assumed that users will interact with the application in a predictable and consistent manner. This includes speaking clearly into the device, correctly initiating, pausing, and stopping transcription sessions, and selecting the appropriate language for translation when needed. These predictable usage patterns will ensure reliable testing results.
2. **Optimal Audio Conditions**The application is assumed to be used in environments with minimal background noise and clear audio input. Background noise, overlapping speech, or low-quality audio can negatively impact the performance of the speech recognition feature. This assumption ensures that testing is conducted under conditions where the app can perform optimally.
3. **Stable Network Connectivity**It is assumed that users will have stable and reliable network connectivity while using the application. Stable internet access is essential for features such as transcription, translation, user authentication, and data synchronization with Firebase Firestore. Any network instability could result in delays or interruptions in these functionalities.

## **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 and translation 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 and translation.
* Some of the questions that they will get:
  + Can you share your overall impressions of the app’s usefulness and effectiveness, particularly in its transcription and translation features?
  + How well does the system assist in managing and understanding spoken language?
  + What recommendations or suggestions do you have for improving the app’s functionality or user experience?
  + Are there any specific features or adjustments you believe could enhance the transcription, translation, or overall usability of the app?

# 6. The Process of Crafting Software

**Speech-to-Text Integration:**The **Speech-to-Text** Flutter package was implemented to enable real-time transcription of spoken words into text. This package processes audio input efficiently, ensuring minimal delay in displaying transcriptions.

**Translation API Integration:**The **Google Cloud Translation API** was utilized to translate transcriptions into multiple languages. Users can select their preferred language, and the system provides accurate translations, helping bridge communication gaps.

**User Interface Development:**Several user-friendly UI components were developed, including:

* **Login and Sign-Up Pages:** Allow users to create accounts, log in securely, or continue as guests.
* **Home Page:** Provides quick access to core features, such as starting a transcription, viewing translations, or accessing past transcriptions.
* **Recent Transcriptions Page:** Displays previously saved transcription sessions for logged-in users.

**Real-Time Transcription Workflow:**

* The app captures audio input in real-time through the device microphone and processes it with the Speech-to-Text package.
* Transcriptions are displayed instantly on the screen, with minimal latency, ensuring a smooth user experience.
* Users can save completed transcriptions, view them later, or translate them into their chosen language.

**Error Management and Feedback:**

* Common issues, such as misinterpreted speech or translation inaccuracies, were addressed by improving input handling and testing under diverse conditions (e.g., accents and background noise).
* User feedback was gathered to refine features, simplify navigation, and enhance usability, resulting in a more efficient app experience.

**Database and Storage:**

* **Firebase Firestore** was integrated for storing user data, transcription sessions, and translation history. The database ensures secure storage and quick retrieval for logged-in users.
* **Guest Mode** ensures that users can access basic transcription features without requiring data storage, maintaining privacy.

**Testing and Refinement:**

* Comprehensive testing was conducted to validate accuracy, speed, and user experience. Issues identified during testing were resolved through iterative updates and improvements.
* Real-world testing was performed to ensure the app's performance aligns with user expectations across various environments and devices.

#### 

# 7. Description of Challenges Faced

During the development and implementation of the Voice2Textify app, several technical and operational challenges arose. Below is a detailed account of these challenges and the solutions we devised to overcome them:

1. **Setting Up Google Cloud Translation API**
   * **Challenge:** Configuring the Google Cloud Translation API required managing API keys securely and understanding the integration process. Ensuring that the API handled multiple requests efficiently while staying within usage limits posed an additional challenge.
   * **Solution:** We implemented a secure mechanism for storing API keys using environment variables and ensured the API integration was robust. The app was optimized to send requests efficiently, batching translations where applicable to reduce API call frequency.
2. **Speech-to-Text Package and Real-Time Processing**
   * **Challenge:** The Flutter Speech-to-Text package required fine-tuning to deliver real-time transcription with minimal latency. Achieving this required handling background noise and varying speech patterns effectively.
   * **Solution:** We tested the Speech-to-Text package under diverse conditions and adjusted settings such as audio input sensitivity and pause durations to optimize real-time transcription accuracy. Background noise reduction techniques were also applied to enhance performance.
3. **Dependency on gRPC for Streaming APIs**
   * **Challenge:** While exploring Google Cloud Speech-to-Text API, the need for gRPC for real-time audio streaming introduced complexity in the integration process, particularly in mobile environments.
   * **Solution:** To simplify implementation, we opted for the Flutter Speech-to-Text package, which provided a more straightforward integration method while meeting real-time transcription requirements.

# 8. Conclusions

**Achievement of Project Goals**:  
The Voice2Textify app successfully meets its primary objectives of providing an efficient and user-friendly real-time transcription and translation platform. The integration of the Flutter Speech-to-Text package ensures accurate and responsive transcriptions, while the Google Cloud Translation API extends accessibility to a multilingual audience. By incorporating Firebase Firestore, the app effectively manages and synchronizes user data.

The implementation also prioritizes user experience, offering features such as text size adjustments, saved sessions, and a responsive interface. Challenges such as API integration, data privacy, and cross-platform compatibility were addressed, resulting in a stable and scalable application. Overall, Voice2Textify demonstrates its potential as a valuable tool for enhancing communication and accessibility across various contexts.

## 

**Decision-Making Process**:  
The decision-making process for Voice2Textify focused on optimizing usability, accuracy, and performance. Key choices included integrating the Flutter Speech-to-Text package for reliable real-time transcription instead of implementing a custom speech model, ensuring faster deployment and reduced complexity. The decision to utilize Google Cloud Translation API was driven by its robust multilingual support and ease of integration. Additionally, Firebase Firestore was selected for its seamless real-time synchronization and scalability to manage user data efficiently.

When addressing trade-offs, such as balancing feature complexity with system responsiveness, the emphasis was placed on delivering a consistent and intuitive user experience. These informed decisions enabled the development of a functional application that achieves its objectives while remaining accessible and efficient for a diverse user base. In conclusion, methodical decision-making allowed the project to overcome challenges and deliver a practical and user-centric solution.

# 9. Overall Effectiveness of the Work

The project effectively achieved its primary objectives of providing accurate transcription, translation, and a user-friendly interface. By leveraging technologies like the Flutter Speech-to-Text package and Google Cloud Translation API, the app delivered a practical and functional solution for real-world use. However, there are areas where improvements could further enhance its overall effectiveness:

* **Testing in Diverse Environments:** Most testing was conducted in controlled settings. Expanding tests to include noisy or challenging environments earlier would have helped optimize transcription accuracy and performance.
* **Agile Workflow:** Incorporating more iterative development cycles and regular user feedback sessions would have identified usability challenges sooner and expedited their resolution.
* **Cross-Platform Optimization:** The app was successfully built using Flutter, which allowed deployment on both Android and iOS. However, additional optimization could ensure an even more seamless experience across devices with varying hardware capabilities.
* **Refining Translation Accuracy:** While the Google Cloud Translation API provided effective translations, further tuning and testing in diverse linguistic contexts could enhance its reliability for a broader audience.

# 10. Lessons Learned

During the implementation stage, we had to make some changes compared to the original plan. For instance, we initially planned to use the Google API for Speech-to-Text for real-time speech processing. However, during implementation, we discovered that using this API for real-time streaming was complex and required advanced gRPC configurations, which were not feasible given the time and resources available. Instead, we opted to use the Flutter Speech-to-Text package, which provided a simpler and more efficient solution that integrated seamlessly with our application.

Additionally, we had planned to include advanced features such as text summarization using the Google Natural Language API. However, due to time constraints and the need to focus on the core functionalities, we decided to forego some of these additional features. This allowed us to concentrate on delivering the primary goals of the app—transcription, translation, and data management—at a higher quality.

This process highlighted the importance of flexibility in planning and the need to make adjustments during development while staying focused on core objectives. It also reinforced the value of prioritizing simplicity and efficiency to achieve a functional and reliable product.

# 11. Meeting Project Objectives

Overall, we met most of the project objectives that were set during the planning stage. The core functionalities, including real-time transcription and text translation, were successfully implemented, allowing users to experience a functional and user-friendly application.

However, some features, such as real-time streaming with the Google Speech-to-Text API and text summarization, were not implemented as originally planned. This was due to technical complexities and time constraints, which required us to prioritize delivering a robust and functional app over additional features. Despite these adjustments, the project still achieved its primary goal of providing an accessible and efficient voice-to-text solution with translation capabilities.

These outcomes demonstrate the project's success in meeting its key objectives while emphasizing the importance of adaptability during development.

# 12. User Guide

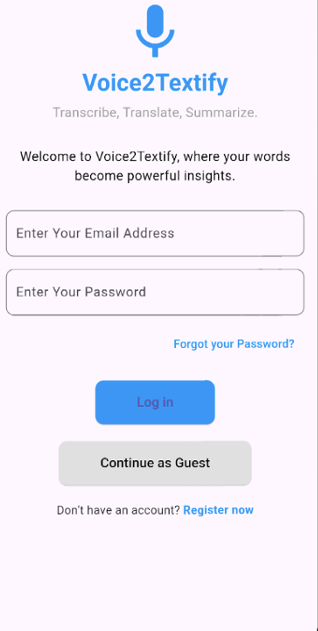
The file provides clear instructions to help users navigate the app and its features, ensuring a smooth experience. It serves as a quick reference for common tasks.

Creating a user-friendly interface for an app is crucial and essential for enhancing User Experience. Therefore, we are planning to build the following interface for our app.

**Login Screen**

This page allows users to log in, continue as a guest, or create a new account to access the app.

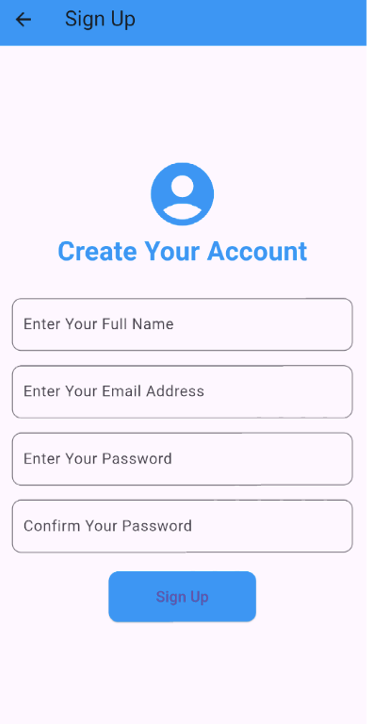
* **Login Form**: Users can enter their **Email Address** and **Password** to log in.
* **Guest Access**: Tap **Continue as Guest** to explore the app without logging in.
* **Password Recovery**: Tap **Forgot your Password?** to reset your password.
* **Sign-Up Prompt**: Tap on **Register now** if you don’t have an account.

****

**Sign-Up Screen**

This screen allows new users to create an account and register for full access to the app features.

* **Input Fields**:  
  Users must fill out the following fields to create an account:
  + **Full Name**: Enter your full name for account identification.
  + **Email Address**: Provide a valid email address to be used for login and communication.
  + **Password**: Create a secure password for your account.
  + **Confirm Password**: Re-enter the password to confirm it matches.
* **Sign-Up Button**:  
  Tap the *Sign Up* button to submit your details and complete the registration process.
* **Navigation**:
  + Use the back arrow at the top left to return to the previous screen.
  + Ensure all fields are filled correctly before tapping *Sign Up*. Errors or mismatches will prompt a notification to correct them.

****

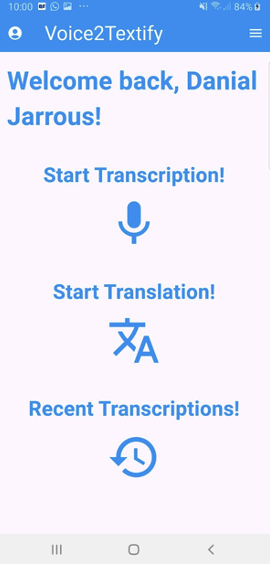
**Home Screen**

This is the main page where users can access key app features.

* **Start Transcription**:  
  Tap the microphone icon to begin voice-to-text transcription.
* **Start Translation**:  
  Tap the translation icon to start translating spoken language into another language.
* **Recent Transcriptions**:  
  Tap the clock icon to view previous transcription sessions.
* **Menu (Burger Icon)**:  
  Tap the menu icon to access additional options, including the **Log Out** button to securely log out.

**Notes:**

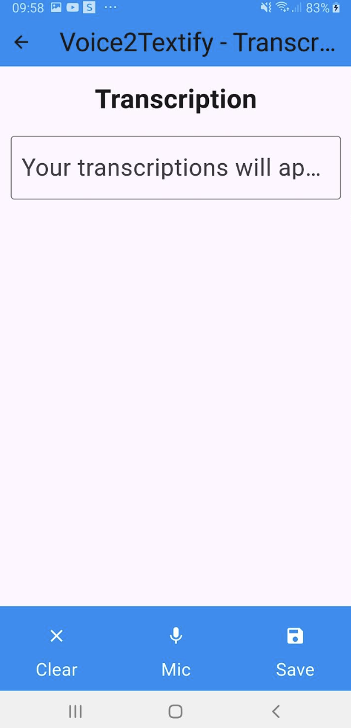
* **Guest Users**:  
  Some features, such as saving transcriptions, are restricted for guest users. These users can explore the app but will have limited functionality.
* **User-Specific Features**:  
  Data displayed, such as recent transcription sessions, is personalized and unique to the logged-in user. Guest users will not have access to saved sessions or personalized data.



**Transcription Screen**

This page allows users to transcribe speech into text in real time. The features available on this page include:

* **Mic Button**:  
  Tap to start or pause the transcription process. Speech will be transcribed and displayed on the screen in real time.
* **Clear Button**:  
  Tap to clear the current transcription from the screen.
* **Save Button**:  
  Tap to save the current transcription to your account. This feature is unavailable for guest users.
* **Real-Time Transcription Display**:  
  The transcription appears on the screen as you speak, updating in real time.



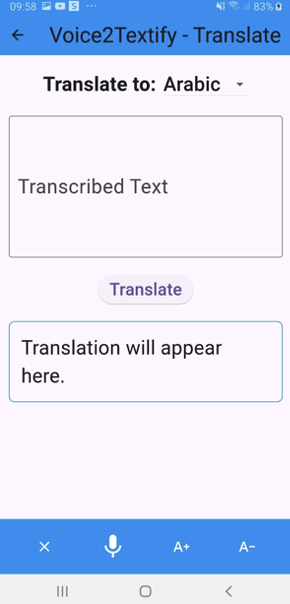
**Translate Screen**

This page allows users to translate transcribed English text into a selected language.

* **Language Selection Dropdown**:  
  Choose the target language for translation from the dropdown menu.
* **Transcribed Text Input Field**:  
  Displays the text you wish to translate.
* **Translate Button**:  
  Tap to translate the displayed text into the selected language. The translated text will appear below.
* **Translation Output Field**:  
  Displays the translated text after clicking the Translate button.
* **Font Size Adjustments**:
  + **A+ Button**: Increases the font size of both the input and output fields.
  + **A- Button**: Decreases the font size of both the input and output fields.
* **Clear Button**:  
  Tap to clear the transcribed text and translation fields.

**Notes for Users**:

* You can input text manually or through voice transcription for translation.

****

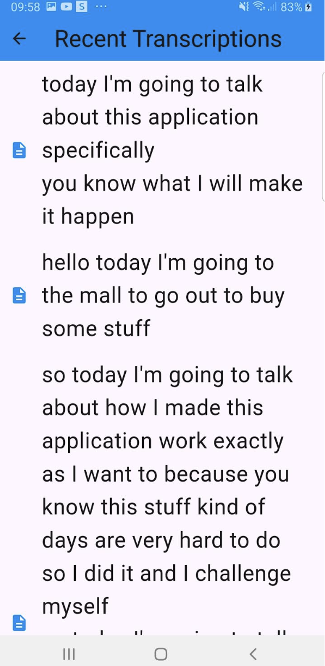
**Recent Transcriptions Screen**

This page allows users to view their previously saved transcription sessions. Features include:

* **List of Transcriptions**:  
  Each item displays the text from a saved transcription session, organized in chronological order.
* **Copy Button**:  
  Tap the copy icon next to a transcription to copy its content to the clipboard for use in other applications.

**Notes**:

* Transcriptions are saved only for logged-in users. Guest users do not have access to saved sessions.
* Each transcription represents a session recorded during usage, ensuring easy access to past conversations or notes.



# 13. Maintenance Guide for Voice2Textify

Overview

This guide outlines the steps required to maintain and troubleshoot the Voice2Textify application. It covers:

· Environment setup

· Updating dependencies

· Managing Firebase services

· Common troubleshooting steps

· Adding new features

· Ensuring app stability

Environment Setup

**Development Tools**:

o Flutter SDK (minimum version: 3.13.0 )

o Dart (minimum version: 3.5.4 )

o Android Studio or Visual Studio Code

**Clone the Repository**:

**Run in terminal**:

git clone <repository-url>

cd Voice2Textify

**Install Dependencies**:

**Run in terminal**:

flutter pub get

**Set Up Firebase**:

o Download the google-services.json file for Android and place it in android/app/.

o Download the GoogleService-Info.plist file for iOS and place it in ios/Runner/.

Add API Keys:

To enable the **translation feature**, you need to provide your Google Translate API key:

1. Open the translation\_api\_key.dart file located in the project.
2. Replace the placeholder text (INSERT\_YOUR\_API\_KEY) with your actual Google Translate API key.
3. Ensure that your API key has the required permissions enabled in the Google Cloud Console.

**Note:**

* The API key is not included in the project for security reasons. You must use your own key to enable translation functionality.
* To avoid exposing your API key, do not hard-code it into the app for production. Use environment variables or secure storage solutions like Firebase Remote Config if applicable.

**Environment Variables**:

o Add API keys and secrets in the appropriate files (e.g., .env or Flutter's dotenv package).

Firebase Management

**Firestore Database**:

o Ensure Firestore rules allow secure read/write operations:

rules\_version = '2';

service cloud.firestore {

match /databases/{database}/documents {

match /sessions/{sessionId} {

allow read, write: if request.auth != null;

}

}

}

**Authentication**:

o Enable email/password and guest sign-in methods in Firebase Console.

**Troubleshooting Firebase**:

If Firebase services fail, check:

o Internet connectivity

o API key validity

o Database quotas (Firebase plan)

Updating Dependencies

**Flutter/Dart Packages**:

To check outdated packages, run:

flutter pub outdated

Update packages with:

flutter pub upgrade

**Firebase SDK**:

o Update Firebase dependencies in pubspec.yaml.

**Testing Updates**:

o After updates, run:

flutter clean

flutter pub get

flutter run

#### ***Contact and Support***

* **Primary Developer**: DANIEL JARROUS
* **Contact Email**: danial.jarrous@gmail.com

# **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>