CogniCare

CO3302 Computer Engineering Project

Group No: 3

Hami B.D.J.K.

Palliyaguruge O.G.W.

Hasarinda H.R.A.

Abstract

This project utilizes Dart and Flutter for the mobile application interface, complemented by a Python backend employing natural language processing (NLP) techniques. The app empowers non-verbal individuals, specifically in the Deaf and Dumb community, by converting text into animated sign language. Leveraging a dataset of signs, the technology stack ensures accurate and contextually relevant translations, fostering seamless communication and inclusivity. The integration of NLP techniques enhances the app's ability to capture linguistic nuances, providing a transformative tool for real-time communication and continuous learning within the Deaf community.

Acknowledgement

We would like to express our sincere gratitude to all those who contributed to the successful completion of this project. Firstly, we would like to thank our project supervisor for providing us with valuable guidance and support throughout the project. Their expertise and insights were instrumental in helping us develop a high-quality web application.

We would also like to thank our team members for their hard work and dedication throughout the project. Each team member brought unique skills and perspectives that contributed to the success of the project. We worked collaboratively and effectively as a team, overcoming challenges and achieving our objectives.

We would like to acknowledge the support of our families and friends, who provided us with encouragement and motivation throughout the project. Their unwavering support helped us to stay focused and committed to the project.

Finally, we would like to express our gratitude for everyone's assistance in making this initiative a success. This project would not have been feasible without their help, direction, and comments.

Table of Contents

Table of Figures

1. Introduction

This innovative mobile application, developed using Flutter and Dart, serves as a transformative tool for non-verbal individuals, specifically the Deaf and Dumb community, empowering them to communicate effortlessly in public settings. The app's primary functionality involves converting text into sign language, providing animated outputs that enable seamless understanding and interaction for individuals with hearing and speech impairments. This breakthrough technology eliminates the need for a constant companion or translator, offering independence to users in their daily lives.

Another main feature of this app is the learning platform. It provides supplementary resources; such as sign language learning materials (videos). This educational feature aims to enhance users' sign language proficiency, enabling them to expand their vocabulary and communicate more effectively within the Deaf community.

Moreover, the app incorporates natural language processing techniques, leveraging Python to ensure accurate and contextually relevant translations from text to sign language. The fusion of cutting-edge technology and linguistic expertise creates a user-friendly and efficient platform that significantly improves the quality of life for non-verbal individuals.

By seamlessly integrating communication and education, this app stands as a beacon of inclusivity, breaking down barriers for the Deaf and Dumb community. It not only facilitates real-time communication but also fosters continuous learning, contributing to the empowerment and integration of individuals with hearing and speech impairments into mainstream society.

2. Objectives

Creating the User Interface (UI)

The user interface is an important part of the application since it directly affects the user experience. The user interface (UI) of a text-to-sign language translation program should be simple and easy to use. Developing with Flutter and Dart enables for the building of a visually beautiful and responsive interface that caters to the demands of nonverbal users. Designing a user-friendly text input mechanism, presenting animated sign language outputs, and implementing interactive elements for the learning platform are all part of this.

Tokenizing Text

Text tokenization is a crucial process in natural language processing, involving the breakdown of a text string into smaller units called tokens. In the context of a text-to-sign language conversion app, this step is important for understanding sentence structure and extracting key linguistic information. By accurately tokenizing text, the app ensures that movements and expressions in sign language align with the intended meaning of the input text, contributing to precise and contextually relevant translations.

Creating a Suitable Dataset

For this application after doing certain findings about several datasets, finally we decided to create suitable dataset using available signs for English alphabet.

Finalizing Architecture

The architecture of the application involves defining the structure and flow of information between different components. In this case, the architecture needs to seamlessly integrate the text processing (natural language processing) and sign language animations. The architecture should be scalable and adaptable to accommodate potential future updates or improvements.

Generating Sign Language Outputs

This involves mapping the letters to corresponding sign language gestures and animations. The generated sign language outputs are then displayed in an animated format on the user interface, providing a visual representation of the translated text.

Hosting

Hosting involves making the application accessible over the internet. This step is essential for users to interact with the app from their devices. It involves deploying the application and its associated databases or resources on a server that can handle user requests, providing a seamless experience for users, whether they are learning sign language or using the communication feature in real-time.

3. Architecture of the System

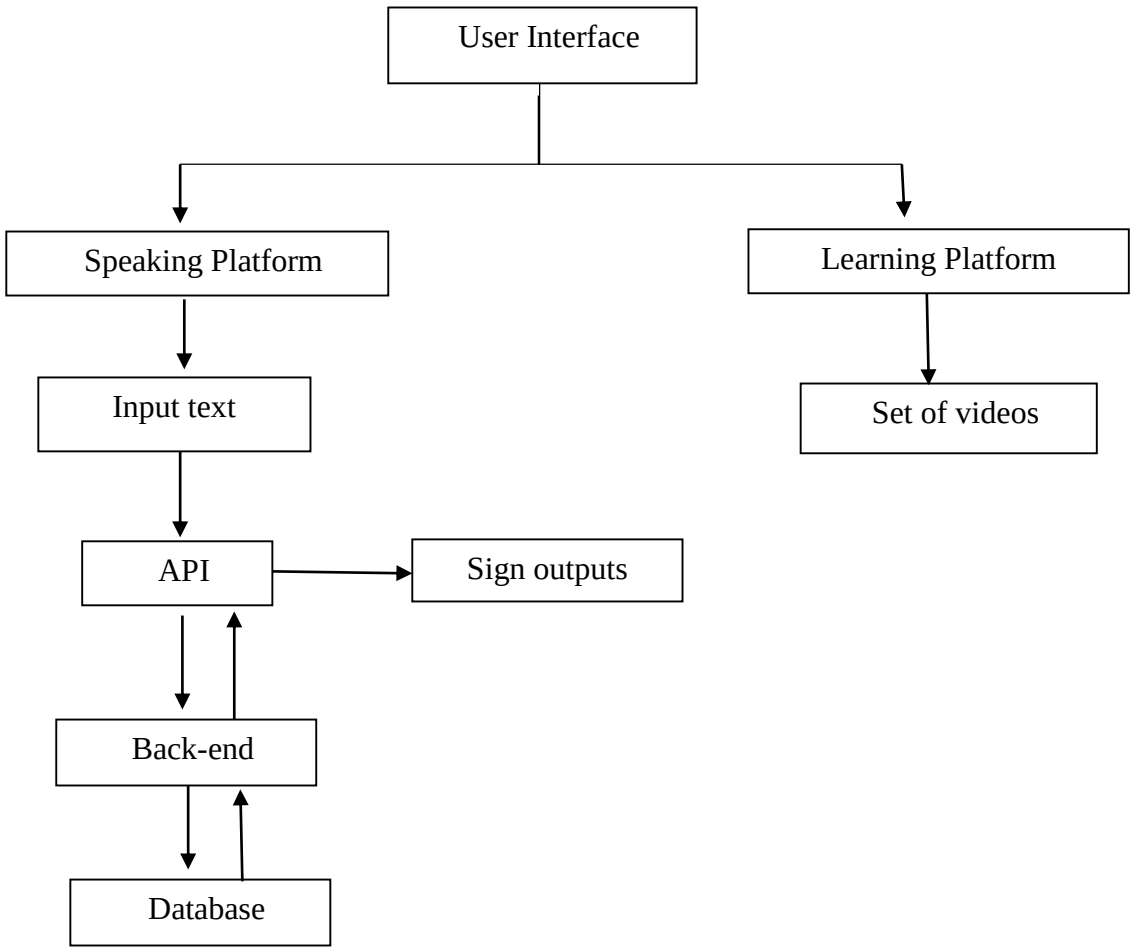

Figure 2:Architecture Diagram

1) User Interface

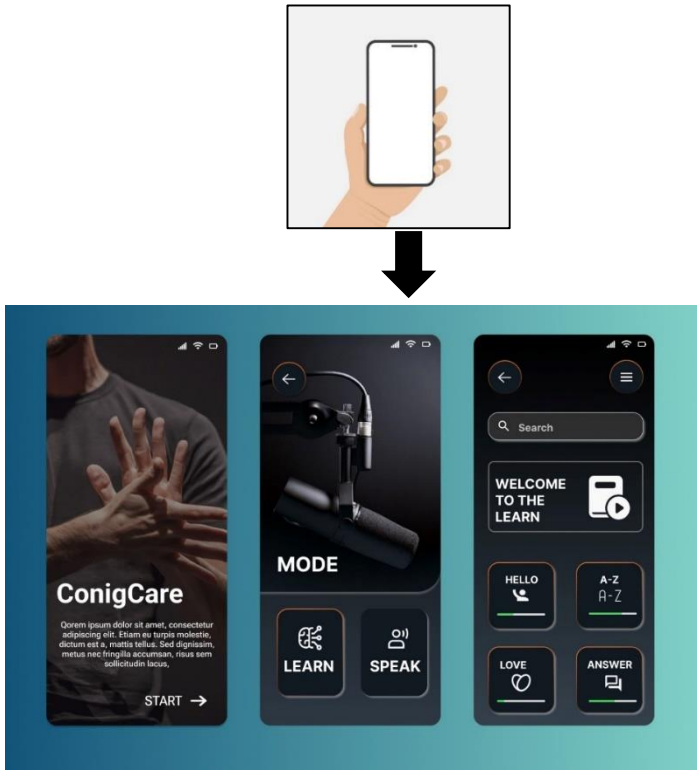

Figure 2: User Interfaces

Speaking Platform Interface

- Input Interface - Allows users to input text through text input field by typing.
- Output Interface - Displays the generated sign language animations to the user.

Learning Platform Interface

- Provides access to supplementary resources, such as sign language learning videos.

2) Back-end

Text Analysis

Receives the input text and performs natural language processing (NLP) tasks such as word and character tokenization. This helps in understanding the structure and meaning of the text.

Get Signs from Database

Associated numeric tag for each tokenized letter and send those tags to database to get relevant signs.

Generate Output

Generate the output animation from receive signs

3) Database

This contains dataset of sign language images.

Sign Language Mapping

Maps the received tags from backend to the corresponding sign language signs or gestures.

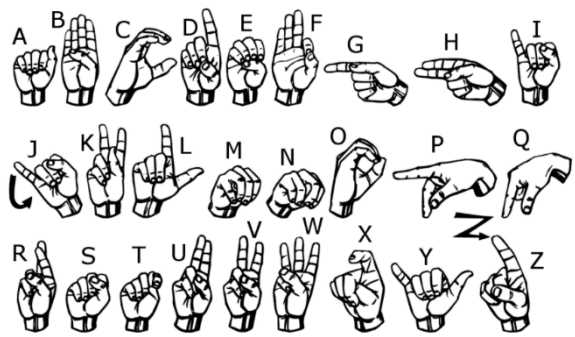

Figure 3: Signs for Alphabet

4) Application Programming Interface (API)

Enables the communication between user interface (text input) and back-end.

4. Technologies used in developing

Software Development Tools

Android Studio - Version 2022.2

PyCharm - Community Edition 2022.1.3

Operating Systems

Android OS

Libraries and APIs

Flutter – Version 3.10.5

Fast API

NLTK (Natural Language Toolkit)

Programming Languages

Dart – Version 3.0.5

Python – Version 3.10

Minimum Requirements

- For android Studio

  CPU Architecture : 2nd Generation Inter Core or newer or AMD CPU with support for a windows Hypervisor

  RAM: 8GB or more

  Storage: 8GB of available disk space minimum (IDE + Android SDK + Android Emulator)

  Resolution: 1280 × 800 minimum screen resolution
- For PyCharm Community Edition:

  CPU: Any modern CPU

  RAM: 4GB or more

  Storage: 3.5GB or more

  Resolution: 1024 × 768 minimum monitor resolution

5. GUI with guidance

Home Page interface and Mode Page interface

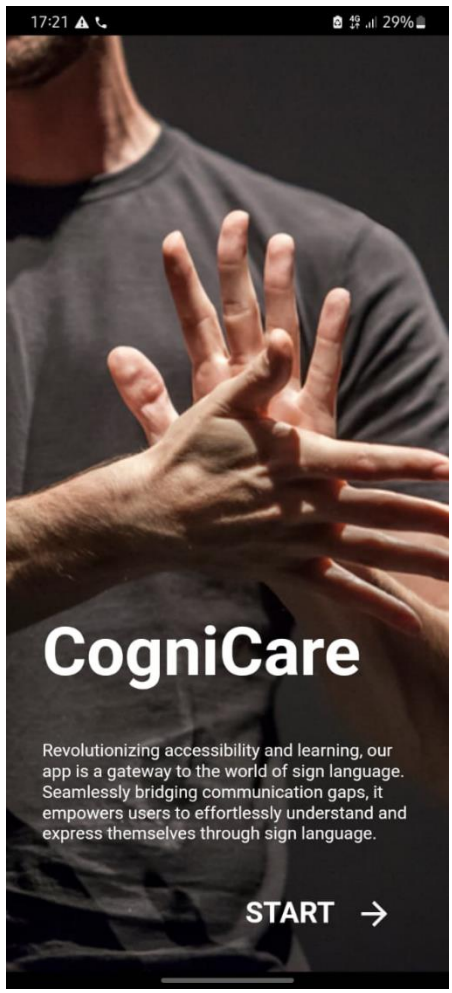

Figure 4) a: Home Page

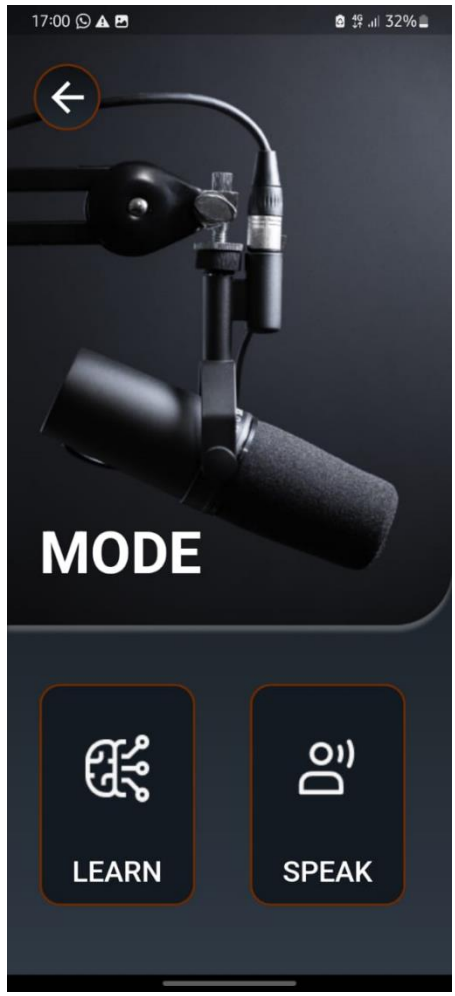

Figure 4) b: Mode Page

- When entering this application figure 4) a will be the home page. By clicking Start or forward arrow it moves to Mode page.
- In Mode page (figure 4) b) there are two platforms Learn platform and Speak platform. Learn platform is to learn about the sign language gestures. Speak platform is to convert text into sign language animations.
- Backward arrow facilitates to go back to previous page. (Home page)

Speak Platform interface

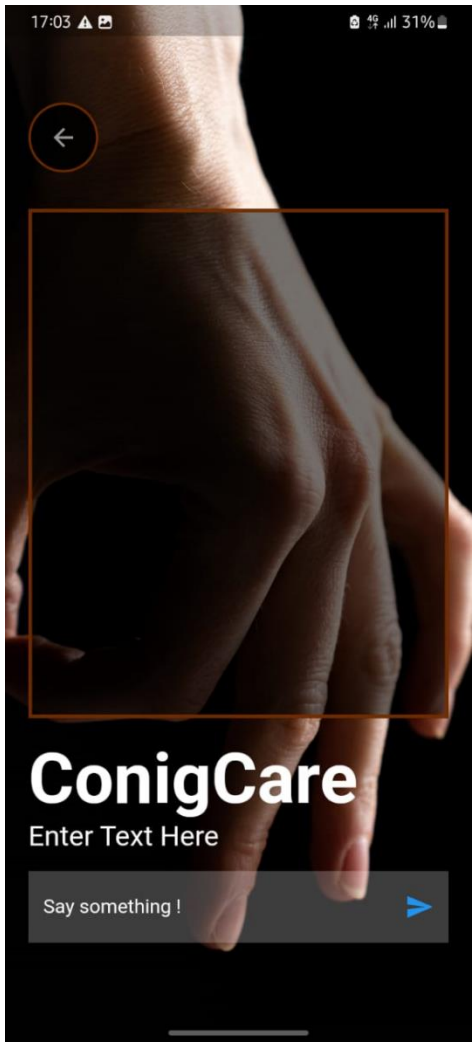

Figure 5) a: Before entering text

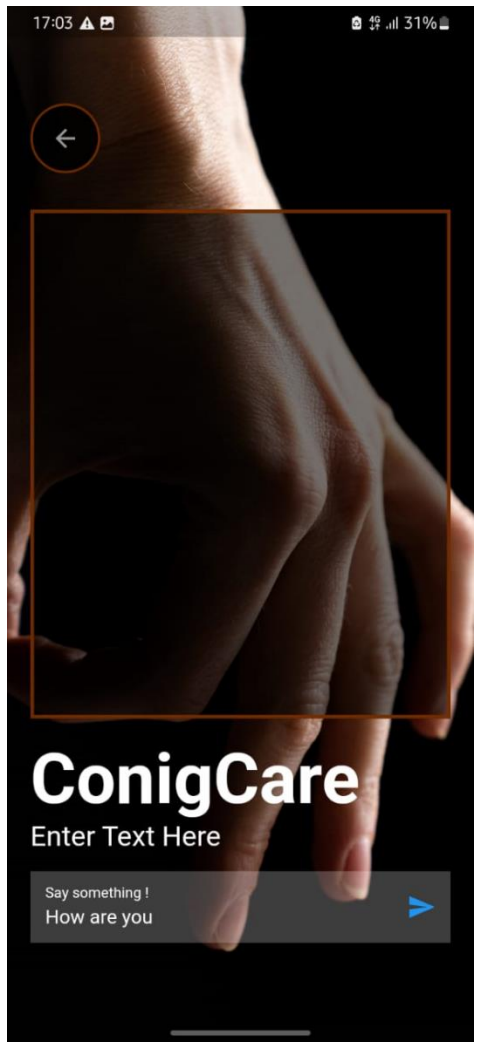

Figure 5) b: After entering text

Figure 5: Text Entering Interface

Figure 5) a and 5) b shows the entering interface when move to the Speak platform. It is text entering interface which gives user to enter the text into the bottom text field as shown in figure 5) a. Figure 5) b shows how it appears after entering the input text. Arrow in right corner of the text input field should be clicked after entering the input text. After clicking the arrow, it appears in green color. Then it takes the user input and generate sign language animation.

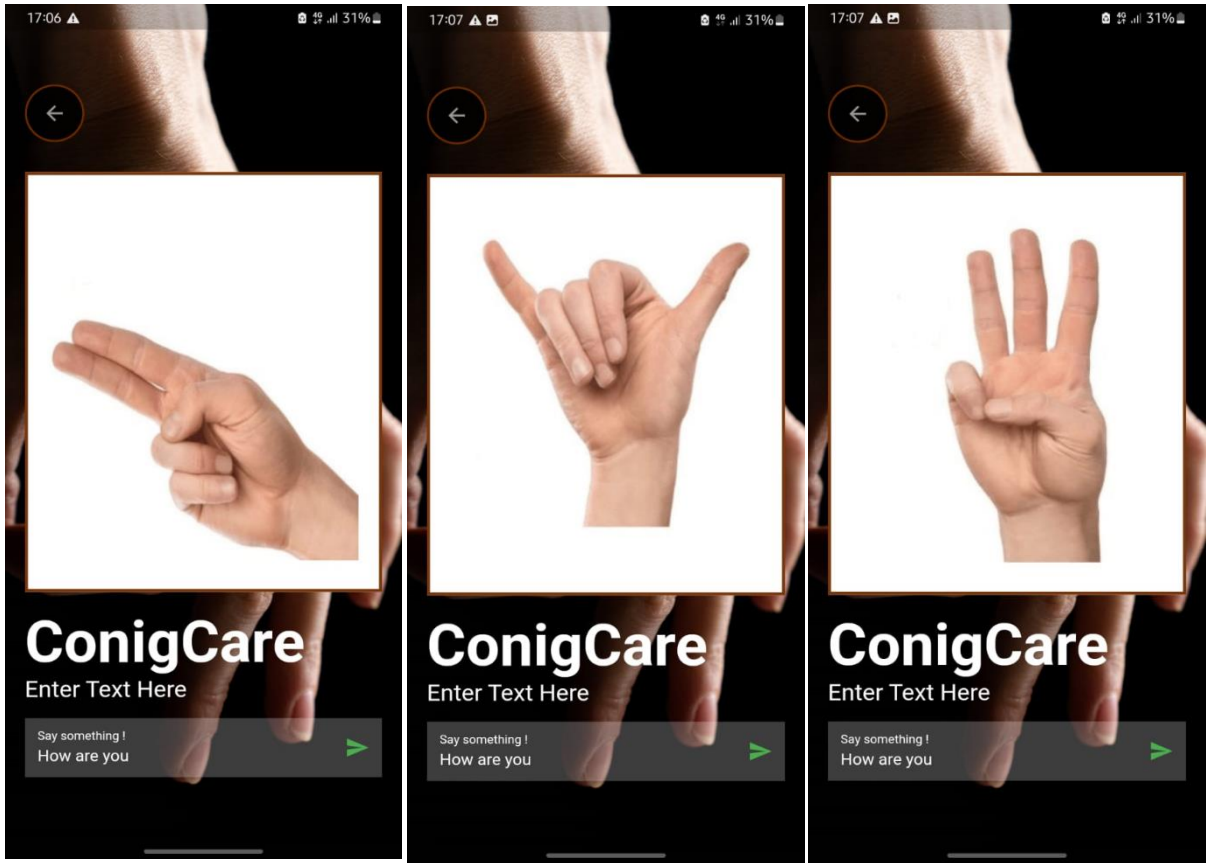

Figure 6: Output Animations

Learn Platform interface

- When moving to the learn platform it has many features in its interface as shown in figure 7. There are 4 types of categories in this interface as Hello, A-Z, Love and Answer. Each will give different type of learning materials.

- Backward arrow facilitates to go back to Mode page.

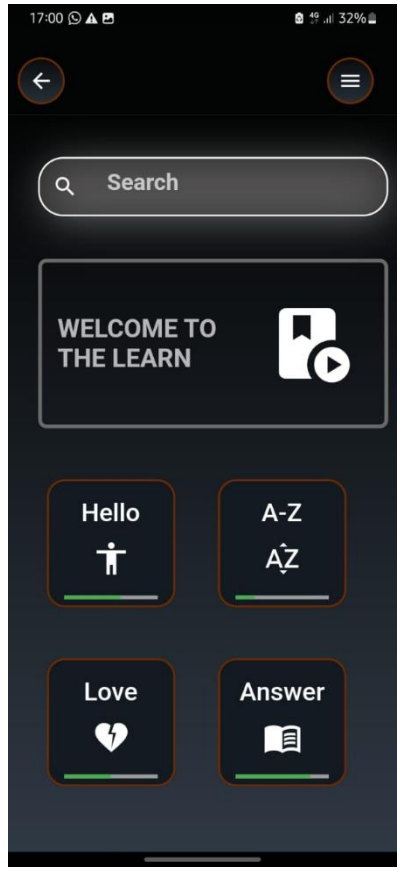

Figure 7: Learn Page Interface

Hello

This is to learn about how to show greeting patterns in American sign language.

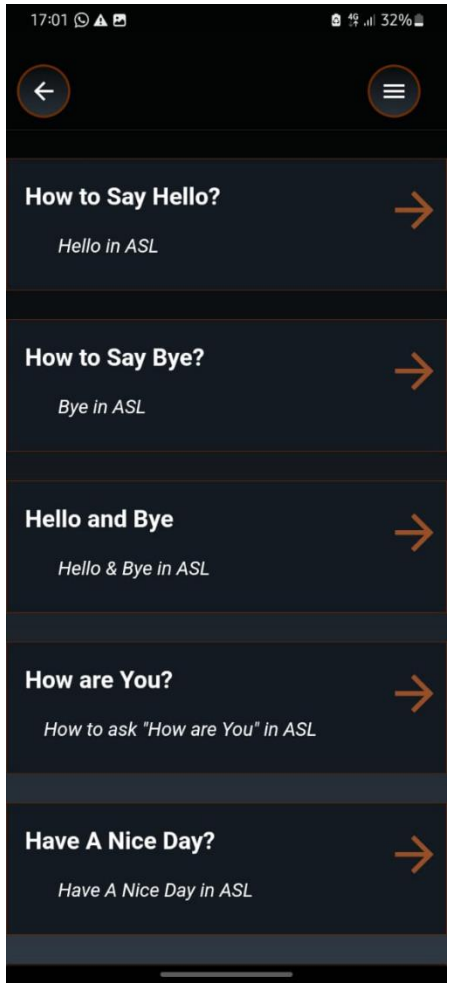

Figure 8) a: Different greetings

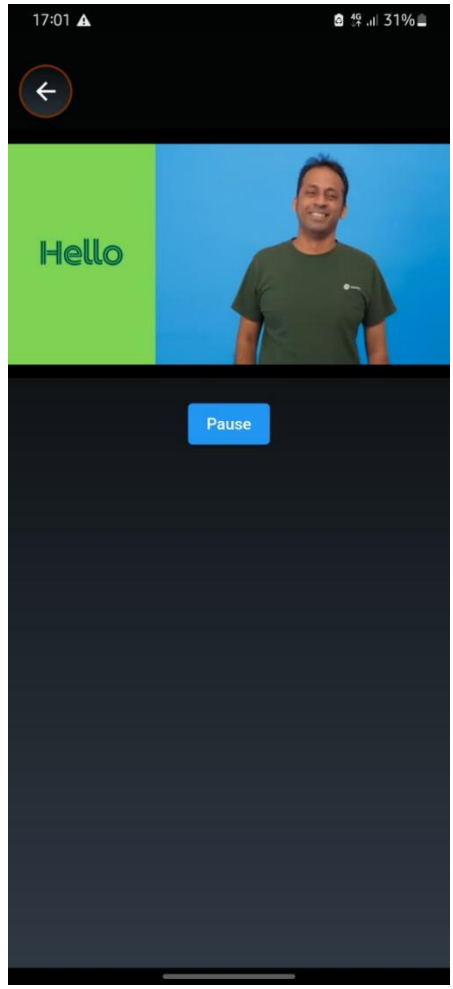

Figure 8) b: How to say Hello

Figure 8: Hello Interface

When user click any of this it will show videos how to say that selected terms and phrases. Figure 8) b shows how to say ‘Hello’ in American sign language using video. User can pause the video when he needs using blue color button which indicates ‘pause’. After pause the video button text change to ‘play’. So by click on the button again user can play the video.

A-Z

This is to learn about how to show English alphabet in American sign language using video.

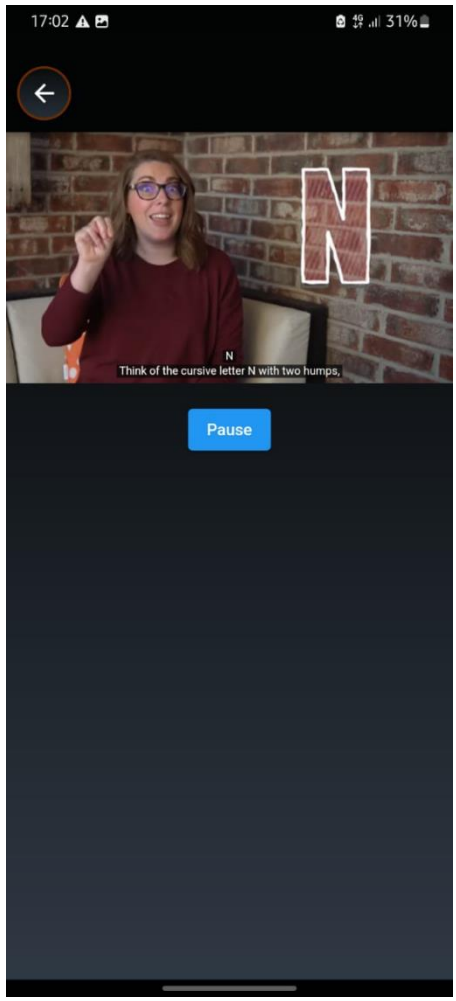

Figure 9: A-Z Interface

User can pause the video when he needs using blue color button which indicates ‘pause’. After pause the video button text change to ‘play’. So by click on the button again user can play the video.

Love

This is to learn about how to show emotions, love, angry in American sign language.

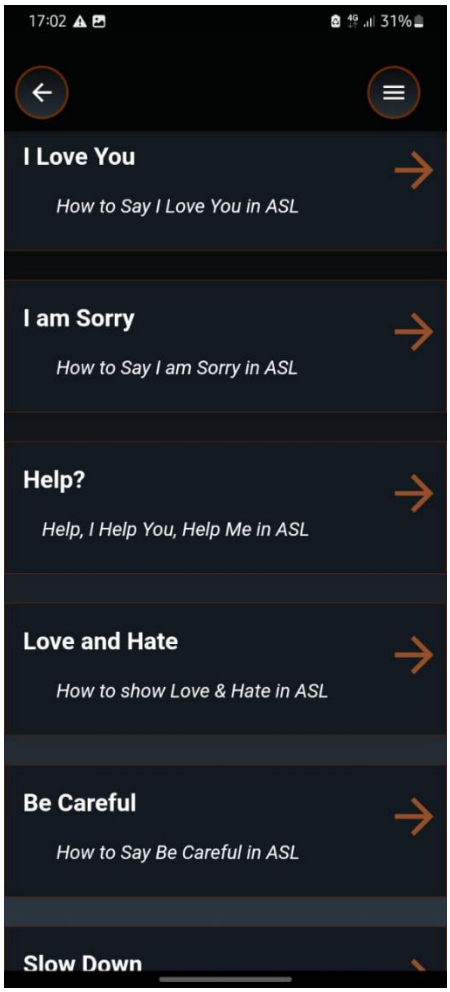

Figure 10) a: Different emotions

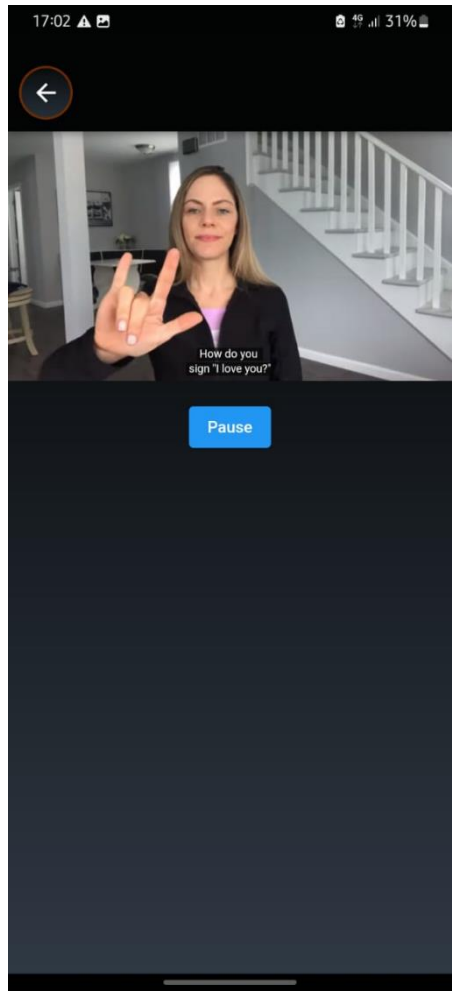

Figure 10) b: How to say I love you

Figure 10: Love Interface

When user click any of this it will show videos how to say that selected terms and phrases. Figure 10) b shows how to say ‘I love you’ in American sign language using video. User can pause the video when he needs using blue color button which indicates ‘pause’. After pause the video button text change to ‘play’. So by click on the button again user can play the video.

Answer

This is to learn about how to show different signs categories, family members, animals, patterns in American sign language. This shows sign sets other than one related sign using videos.

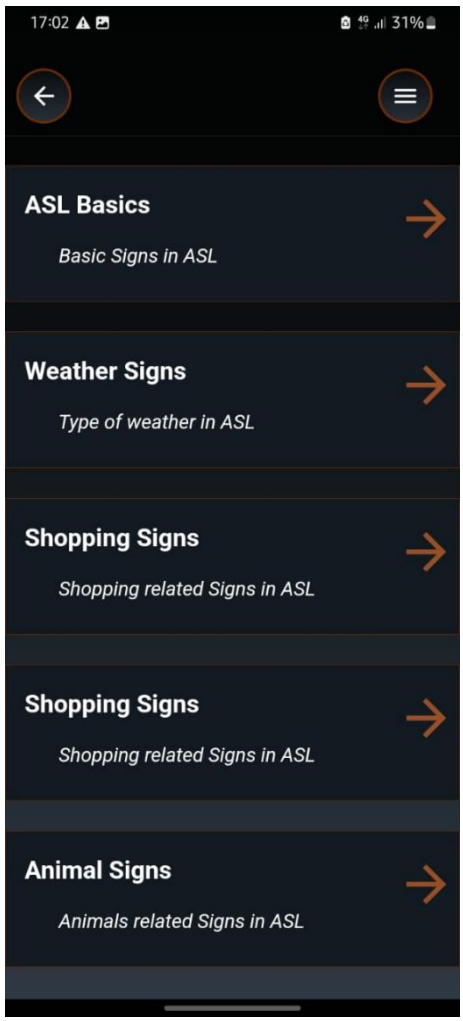

Figure 11) a: Different sign sets

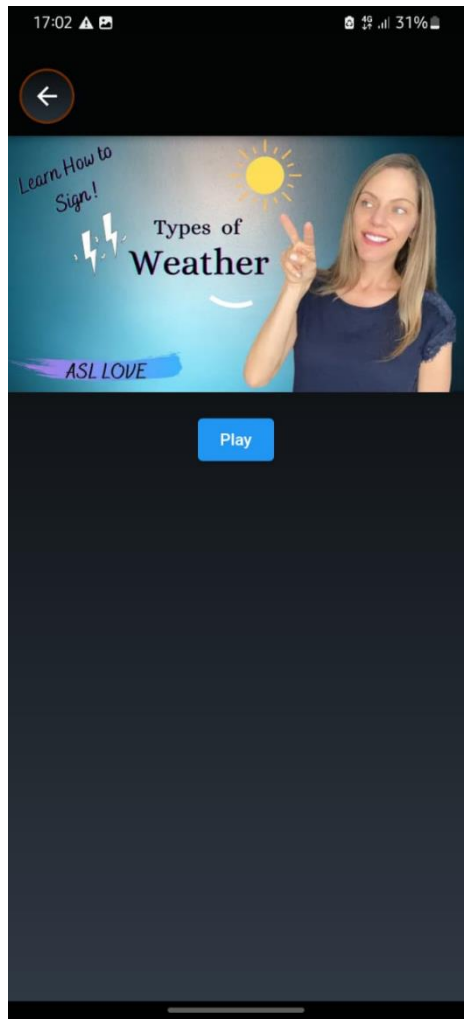

Figure 11) b: Weather signs

Figure 11: Answer Interface

When user click any of this it will show videos how to say that selected terms and phrases. Video in figure 11) b shows how to show weather signs. User can play the video when he needs using blue color button which indicates ‘play’.

Drawer button

Drawer button in Mode page and other pages gives list of page to move quickly. In here drawer allows only to move Home page and Mode page.

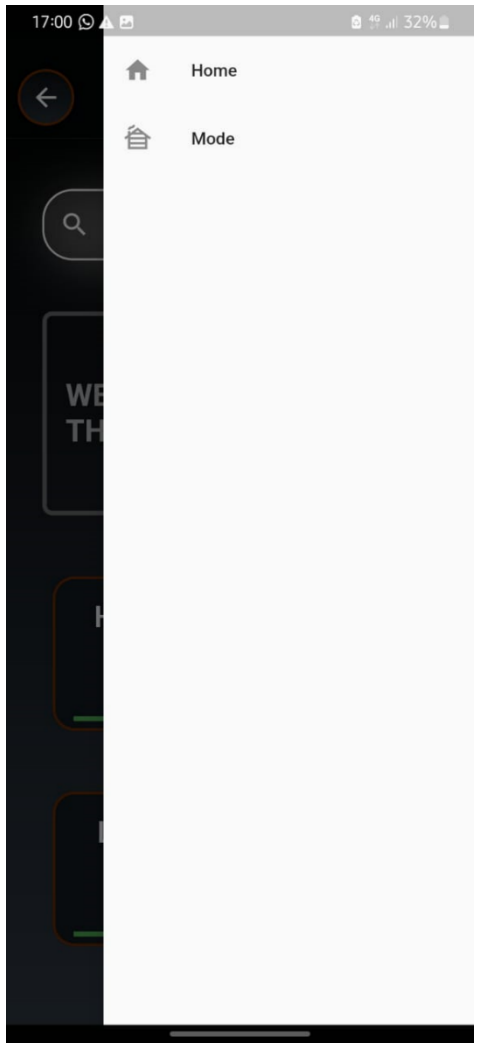

Figure 12: Drawer Option

6. Challenges

We ran across a number of technical difficulties when developing the sign language converter, which called for creative fixes. These included sophisticated coding, challenging algorithms, and hardware challenges that needed careful problem-solving. In order to assure the effectiveness of the converter, we had to carefully address user experience problems as designing an intuitive and user-friendly interface brought its own set of challenges. Additional challenges arose from integrating various parts and technologies, calling for calculated approaches to successfully resolve integration problems. There were obstacles in obtaining and analyzing user feedback, such as navigating divergent viewpoints and making wise decisions to improve the project.

Our restricted resources—budget and time—forced us to use resources as efficiently as possible in order to complete the project. We identified the obstacles to team collaboration and put tactics in place to promote productive teamwork, thereby overcoming these obstacles. Adaptability was essential during the iterative development process, as the project changed in response to important lessons discovered. The phases of testing and debugging posed additional difficulties, necessitating intense work to guarantee the stability and dependability of the sign language converter.

Careful planning and execution were necessary to maintain project quality within time constraints. In summary, the project's tenacity and eventual success highlight the fact that overcoming obstacles is a necessary component of project development, providing us with important takeaways and insightful knowledge that will help with current and next endeavors.

7. Future improvements

The Sign Language Converter project is a multi-pronged initiative that aims to transform communication for those who use sign language. Instantaneous communication between sign language users and non-signers is made possible by real-time translation capabilities. Users can add another level of personalization by teaching the system to recognize different signing styles through customizable gesture recognition. By introducing a gesture learning mode, users are able to enhance their skills through interactive training and feedback.

Language coverage is increased with multilingual support, which enables users to easily transition between regional variations or sign language dialects. Text-to-sign translation, which translates written text into sign language motions, was introduced to improve communication. Concerns about accessibility in places with spotty connectivity are addressed with an offline mode.

By using machine learning techniques, the accuracy of gesture detection is improved and false positives and negatives are decreased. Incorporating voice feedback and augmented reality (AR) into a system offers multimodal features that improve the user experience overall and accommodate people with visual impairments. Expressive facial recognition enhances the emotional expressiveness of communication by capturing emotions in sign language. A large gesture library with search capabilities makes it easier for users to locate and become familiar with particular sign language movements.

Interactive educational games add fun to learning by fusing pleasure and knowledge. Individual tastes can be catered to with customizable accessibility options, and cross-platform interoperability gives users freedom in a variety of scenarios. All of these attributes together represent a dedication to diversity, ease of use, and ongoing development in the field of sign language communication.

8. Project Setup

The steps below must be completed with the provided source code

1. Extract the source code folder.
2. Open the frontend folder in the Visual Studio Code application.
3. Open the backend folder using Pycharm community edition
4. Install the recommended libraries.
5. Run the project

We use render to host backend. For database we use firestore.

References

[1] L. Goyal and V. Goyal, “Text to sign language translation system,” International Journal of Synthetic Emotions, vol. 7, no. 2, pp. 62–77, 2016. doi:10.4018/ijse.2016070104

[2] N. K. Kahlon and W. Singh, “Machine translation from text to sign language: A systematic review,” Universal Access in the Information Society, vol. 22, no. 1, pp. 1–35, 2021. doi:10.1007/s10209-021-00823-1

[3] S. Stoll, N. C. Camgoz, S. Hadfield, and R. Bowden, “Text2Sign: Towards sign language production using neural machine translation and generative Adversarial Networks,” International Journal of Computer Vision, vol. 128, no. 4, pp. 891–908, 2020. doi:10.1007/s11263-019-01281- 2