

# **SIGNON TRANSLATIONS**

PROJECT REPORT

Submitted by

**ALEENA ROSE (SNM19CS003)**  
**ANANDHAKRISHNAN T.U. (SNM19CS006)**  
**ROHITH A.R. (SNM19CS024)**  
**SNEHA M.A. (SNM19CS028)**

**To**

The APJ Abdul Kalam Technological University

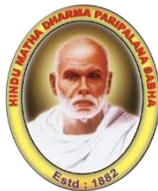
In partial fulfillment of the requirements for the award of the Degree

Of

Bachelor of Technology

In

Computer Science and Engineering



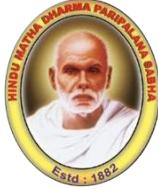
**Department of Computer Science and Engineering**

**SREE NARAYANA MANGALAM INSTITUTE OF MANAGEMENT**  
**& TECHNOLOGY**  
**MALIANKARA**

April, 2023

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**SREE NARAYANA MANGALAM INSTITUTE OF  
MANAGEMENT & TECHNOLOGY, MALIANKARA**



**CERTIFICATE**

This is to certify that the report entitled ‘**SIGNON TRANSLATIONS**’ submitted by **ALEENA ROSE, ANANDHAKRISHNAN T.U., ROHITH A.R, SNEHA M.A** to the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science Engineering is a bonafide record of the project work carried out by him/her under my/our guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.

**Ms. SONEY R.NADH**

Internal Supervisor

**Ms. SAREENA K.K.**

Coordinators

**Ms. SNEHA T. SUBRAHMANIAN**

External Expert

External Expert

**Dr. SANJUNA K.R.**

HEAD OF THE DEPARTMENT

## DECLARATION

We undersigned hereby declare that the project report “**SIGNON TRANSLATIONS**”, submitted for partial fulfillment of the requirements for the award of degree of Bachelor of Technology of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by we under supervision of Ms. Soney R.Nadh, Assistant Professor , Department of Computer Science and Engineering , S.N.M.I.M.T. , Maliankara. This submission represents our ideas in our own words and where ideas or words of others have been included, We have adequately and accurately cited and referenced the original sources. We also declare that we have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in our submission. We understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

Place: Maliankara

Signature

Date

ALEENA ROSE

ANANDHAKRISHNAN T.U.

ROHITH A.R.

SNEHA M.A.

# CONTENTS

Contents	Page No
ACKNOWLEDGEMENT	I
ABSTRACT	II
Chapter 1. INTRODUCTION	1
1.1 Purpose	1
1.2 History	2
1.3 Machine learning	4
1.4 Types of machine learning	4
1.5 Overview	7
1.6 Modules	7
1.7 Scope	8
Chapter 2. PROBLEM STATEMEN AND OBJECTIVES	9
2.1 Problem statement	9
2.2 Objectives	10
Chapter 3 LITERATURE SURVEY	11
Chapter 4. SYSTEM REQUIREMENTS	20
4.1 Hardware	20
4.2 Software	20
Chapter 5 SYSTEM ARCHITECTURE	23
Chapter 6 APPLICATION ARCHITECTURE	27
Chapter 7 MODULES	31
7.1 Login/sign up	31
7.2 Text Translations	32
7.3 Voice Translations	33
7.4 Sign Language Translations	33
7.2 Detection	31
Chapter 8 USER INTERFACE	35
8.1 Welcome page	35
8.2 Login Page	35
8.3 SignUp Page	36
8.4 Main page	36
8.5 Text-to-Text Translation page	37
8.6 Text-to-Voice Translation page	37
8.7 Voice-to-Text Translation page	38
8.8 Voice-to-Voice Translation page	38
8.9 Sign Language Translation page	39
Chapter 9 CONCLUSION	39
Chapter 10 SOURCE CODE	40
REFERENCES	37

## LIST OF TABLES

TITLE	PAGE NO:
TABLE 1: Arabic Language Data-set	12
TABLE 2: List of Body Joints Used as Input to the SKN	17
TABLE 3: Register	25

## LIST OF FIGURES

TITLE	PAGE NO:
FIG 1: General Overview of the system	14
FIG 2: Example of OPL output on the ARSL Dataset	15
FIG 3: Example of the OPL output on the ARSL Datasets, Static sign—joints correctly identified	15
FIG 4: Example of the OPL output on the ARSL Datasets, Static sign—joints correctly identified	16
FIG 5: Hand Skeleton Generation Errors, Right Hand fully/Partially blurred (Dynamic sign)- Finger joints not identified	17

FIG 6: System Architecture Diagram	25
FIG 7: Flowchart Diagram	26
FIG 8: Application Architecture Diagram	29
FIG 9: Welcome page	35
FIG 10: Login page	35
FIG 11: Signup page	36
FIG 12: Main page	36
FIG 13: Text-to-Text Translation page page	37
FIG 14 : Text-to-Voice Translation page	37
FIG 15 : Voice-to-Text Translation page	38
FIG 16 : Voice-to-Voice Translation page	38
FIG 17 : Sign Language Translation page	39

# ACKNOWLEDGEMENT

The satisfaction that accompanies that the successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success. We would like to express our deep-felt gratitude to GOD Almighty for showing his grace upon us, without which our SIGN ON TRANSLATIONS would not been materialized.

We are extremely grateful to **DR. SANJUNA K.R.** ,our principal , and Head of the department of Computer Science and Engineering, for her valuable guidance through this humble endeavour. Her highly enterprising attitudes, patience for listening to our doubts, timely suggestions and guidance made our project a so-called reality in stipulated time.

We would also like to thank our project coordinators **Ms. SAREENA K.K** , **Ms. SNEHA T SUBRAHMANIAN** , Assistant Professors, Department of Computer Science and Engineering and our project guide **Ms. SONEY R.NADH**, Assistant Professor, Department of Computer Science and Engineering for their gracious of encouragement and the valuable assistance through out our project.

We express our sincere gratitude to all the faculty members of the department of Computer Science and Engineering for their co-operation and support to the Project.

We also express our sincere gratitude to all our friends and our parents for their cooperation and constant.

## ABSTRACT

Language is a structured system of communication .It consists of grammar and vocabulary. It is the primary means by which humans convey meaning, both in spoken and written forms, and may also be conveyed through sign languages. But it creates a big problem when the opposite person doesn't know the spoken language and it's meaning. This situation makes the conversation impossible. So , in this paper we are proposing a system that helps as a mediator to help the people who is not having enough knowledge to communicate without any problem. That is , if the other person is communicating in a different language and the first person wants to communicate with the other person, he/she can use our software and have a uninterrupted and hassle free communication. We are using the Machine Learning technology to implement this system. In our system it will be able to translate text, voice and sign language to the desired target language. We are using the mediapipe which has different pipelines which can help to implement translation process using machine learning . This system provides real-time translation , efficient & accurate meaning , better communication , and it enriches relationships because of better communication.



# **CHAPTER 1**

## **INTRODUCTION**

A translation system is a set of tools and technologies that are used to facilitate the translation of text or speech from one language to another. A translation system can be made up of both machine translation (MT) and human translation (HT) components, and may include software, hardware, and human resources. The primary purpose of a translation system is to enable communication between speakers of different languages. It works by taking input in one language and generating output in another language. The input can be in the form of written text, spoken language, or even images with text. The system analyzes the input using natural language processing (NLP) techniques, which involve breaking down the text into smaller units like words and phrases, identifying the grammatical structure, and analyzing the meaning of the text. The translation process can involve both automated and human components. Machine translation engines can be used to automatically translate the input text into another language. However, machine translation can have limitations in terms of accuracy and idiomatic expression. As a result, human translation is often used in combination with machine translation to ensure quality and cultural sensitivity.

### **1.1 PURPOSE**

The purpose of a translation system is to facilitate communication between individuals who speak different languages. The primary purpose of a translation system is to enable communication between people who speak different languages. This can be done through various modes of communication, such as written text, spoken words, or sign language. Language barriers can prevent people from accessing important information or participating fully in society. Translation systems help to overcome these barriers by providing access to information and enabling communication across linguistic and cultural boundaries.

Translation systems can help to promote cross-cultural understanding by facilitating communication and conveying cultural nuances and context. This can help to reduce misunderstandings and promote tolerance and respect for different cultures. In today's

*S.N.M.I.M.T. , Maliankara*

globalized world, businesses need to communicate effectively with partners, suppliers, and customers from different parts of the world. Translation systems can help businesses to overcome language barriers and reach new markets. Diplomacy and international relations rely on effective communication and understanding between nations. Translation systems can help to bridge linguistic and cultural differences and promote cooperation and diplomacy. It can also help to preserve cultural heritage by enabling the translation of historical texts, literature, and other cultural artifacts into different languages. This can help to promote cultural diversity and preserve important aspects of human history and culture.

## **1.2 HISTORY**

The history of sign language translation is closely intertwined with the history of sign language itself. Sign language is a natural language that has been used by deaf communities for centuries. While ad hoc versions of sign language have existed throughout the course of human civilization, it was not until the 17th century that the Western world saw a systematic study of the teaching and learning of sign language. In 1620, a Spanish priest by the name of Juan Pablo Bonet published the first modern text to propose a method of educating deaf children through the use of manual signs. His *Reducción de las letras y arte para enseñar a hablar a los mudos* (Summary of the Letters and Art of Teaching Speech to the Mute) paved the way for another philanthropic educator, Abbé Charles-Michel de l'Épée, who in 1755 founded the first public school for the deaf. Known as the “Father of the Deaf” for his extensive work with sign language, Abbé de l'Épée gleaned much of his teachings from his observations of deaf people on the streets of Paris. Synthesizing this information with typical gestures and hand signs, he was able to publish a manual alphabet that was used, mostly

unchanged, until the present. Modern American Sign Language is derived from the continental system created in the 18th century, but it also has its own unique history. American minister Thomas Hopkins Gallaudet was a prominent figure in the education of the deaf, traveling to France in 1815 to study methods of communication. In 1817, upon his return to the United States, Gallaudet founded the country's first school for the deaf, in Hartford, Connecticut. By 1863, there were a total of 22 functioning schools throughout the nation, and by 1864, a college dedicated to the education of deaf people. Prior to that, in the 17th century, the United States was the birthplace of Martha's Vineyard Sign Language. That area of the country was home to a large deaf population; by 1854, the percentage of deaf people on the island was almost 35 times higher than the national average. For this reason, a specialized sign language had been worked out in the area before it had spread to other parts of the country. The first recorded use of sign language dates back to ancient Greece, where it was used in schools for the deaf. The first known deaf education program was established in the early 19th century by Thomas Hopkins Gallaudet and Laurent Clerc, who developed a sign language based on French Sign Language (LSF) that is now known as American Sign Language (ASL). This marked the beginning of the formal use of sign language in education and as a means of communication between deaf and hearing individuals.

In the early 20th century, the first sign language interpreters began to appear, as deaf and hard of hearing individuals increasingly needed access to education, employment, and other services. Sign language interpreters, also known as sign language translators, are trained professionals who are fluent in both sign language and spoken language, and are able to translate back and forth between the two. They use their knowledge of both languages, as well as their understanding of deaf culture and the needs of deaf and hard of hearing individuals, to accurately convey the meaning of spoken language into sign language, and vice versa. Over time, the use of sign language interpreters has become more widespread, and they are now commonly used in a variety of settings, including schools, workplaces, hospitals, and other public settings, to facilitate communication between deaf and hard of hearing individuals and hearing individuals.

## **1.3 MACHINE LEARNING**

Machine learning (ML) is a field of inquiry devoted to understanding and building methods that 'learn', that is, methods that leverage data to improve performance on some set of tasks. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as in medicine, email filtering, speech recognition, agriculture, and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks. A subset of machine learning is closely related to computational statistics, which focuses on making predictions using computers, but not all machine learning is statistical learning. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a related field of study, focusing on exploratory data analysis through unsupervised learning. Some implementations of machine learning use data and neural networks in a way that mimics the working of a biological brain. In its application across business problems, machine learning is also referred to as predictive analytic

## **1.4 TYPES OF Machine Learning**

1. Supervised Learning
2. Unsupervised Learning
3. Semi-supervised Learning
4. Reinforcement Learning.

### **Supervised Learning**

Supervised learning is a type of machine learning method in which we provide sample labelled data to the machine learning system in order to train it, and on that basis, it predicts the output. The system creates a model using labelled data to understand the

data-sets and learn about each data, once the training and processing are done then we test the model by providing a sample data to check whether it is predicting the exact output or not. The goal of supervised learning is to map input data with the output data. The supervised learning is based on supervision, and it is the same as when a student learns things in the supervision of the teacher. The example of supervised learning is spam filtering.

## **Unsupervised Learning**

The training is provided to the machine with the set of data that has not been labelled, classified, or categorized, and the algorithm needs to act on that data without any supervision. The goal of unsupervised learning is to restructure the input data into new features or a group of objects with similar patterns. In unsupervised learning, we do not have a predetermined result. Unsupervised learning cannot be directly applied to a regression or classification problem because unlike supervised learning, we have the input data but no corresponding output data. The goal of unsupervised learning is to find the underlying structure of dataset, group that data according to similarities, and represent that dataset in a compressed format.

Unsupervised learning is helpful for finding useful insights from the data. Unsupervised learning is much similar as a human learns to think by their own experiences, which makes it closer to the real AI. Unsupervised learning works on unlabelled and uncategorized data which make unsupervised learning more important. In real-world, we do not always have input data with the corresponding output so to solve such cases, we need unsupervised learning.

## **Semi-Supervised Learning**

Semi-Supervised learning is a type of Machine Learning algorithm that represents the intermediate ground between Supervised and Unsupervised learning algorithms. It uses the combination of labelled and unlabelled datasets during the training period. To overcome these drawbacks of supervised learning and unsupervised

learning algorithms, the concept of Semi-supervised learning is introduced. In this algorithm, training data is a combination of both labelled and unlabelled data. However, labelled data exists with a very small amount while it consists of a huge amount of unlabelled data. Initially, similar data is clustered along with an unsupervised learning algorithm, and further, it helps to label the unlabelled data into labelled data. It is why label data is a comparatively, more expensive acquisition than unlabelled data.

The most basic disadvantage of any Supervised Learning algorithm is that the dataset has to be hand-labelled either by a Machine Learning Engineer or a Data Scientist. This is a very costly process, especially when dealing with large volumes of data. The most basic disadvantage of any Unsupervised Learning is that its application spectrum is limited.

To counter these disadvantages, the concept of Semi-Supervised Learning was introduced. In this type of learning, the algorithm is trained upon a combination of labelled and unlabelled data. Typically, this combination will contain a very small amount of labelled data and a very large amount of unlabelled data. The basic procedure involved is that first, the programmer will cluster similar data using an unsupervised learning algorithm and then use the existing labelled data to label the rest of the unlabelled data. The typical use cases of such type of algorithm have a common property among them – The acquisition of unlabelled data is relatively cheap while labelling the said data is very expensive.

## **Reinforcement Learning**

Reinforcement learning is a feedback-based learning method, in which a learning agent gets a reward for each right action and gets a penalty for each wrong action. The agent learns automatically with these feedbacks and improves its performance. In reinforcement learning, the agent interacts with the environment and explores it. The goal of an agent is to get the most reward points, and hence, it improves its performance.

- input: The input should be an initial state from which the model will start
- Output: There are many possible outputs as there are a variety of solutions to a

particular problem

- Training: The training is based upon the input; The model will return a state and the user will decide to reward or punish the model based on its output.
- The model keeps continues to learn.
- The best solution is decided based on the maximum reward.

## **1.5 OVERVIEW**

The Project “ SignOn ” is a advanced translator tool built using machine learning for better communication between people. This application aims to narrow the gap of communication barrier for people communicating different languages. It is capable of translating languages from text-to-text, text-to-voice, voice-to-text, voice-to-voice and also Sign language-to-text. It is a versatile , useful application which can be used in any computer system and can be installed in any system easily without any issues. It has a user-friendly environment for making the end user experience better.

When it comes to the usability the users can login/sign in to this application and then user is shown an interface which shows text translation, voice translation, and sign language translation options. The user can choose which translation he/she wants to do and then it will be directed to its respective page where the input from the user is taken and the translation is done.

## **1.6 MODULES**

### **1. Text-to-Text**

- The Input language is selected
- The text to be translated is Given in the input box
- The target Language is Selected
- Translation is then done
- The translated text is shown in the Output Box

### **2. Text-to-Voice**

- The Input language is selected
- The text to be translated is Given in the input box
- The target Language is Selected

- Translation is then done
- The translated voice is then played

### **3. Voice-to-Text**

- The Input language is selected
- The voice to be translated is spoken through the mic
- The target Language is Selected
- Translation is then done
- The translated text is then shown in the output box

### **4. Voice-to-Voice**

- The Input language is selected
- The voice to be translated is spoken through the mic
- The target Language is Selected
- Translation is then done
- The translated voice is then played

### **5. Sign Language to text**

- The Sign language to be translated is shown through the Webcam
- The system recognizes the Sign shown with its trained data-set
- The system Identifies the Sign Shown
- The translation is Shown in text in-real time

## **1.7 SCOPE**

The scope of language translators is limited to facilitating communication between people who use different languages and the other person doesn't know the same. They are not intended to fully translate or interpret all aspects of a conversation, but rather to provide a basic level of communication that allows people who use different language to communicate. Thus helping to narrow the gap communication gap between people and helping them to have a healthy conversation.



# **CHAPTER 2**

## **PROBLEM STATEMENTS AND OBJECTIVES**

### **2.1 Problem statement**

#### **Knowledge about the Language**

Every language is a complex and nuanced language, not everyone will have the knowledge about it, thus creating language barrier. Learning the language is also not that easy, it has its own learning curve and consumes a lot of time. Also people who has the knowledge is very few in number, thus there is a need of a translator

#### **Communication barrier**

There is very less knowledge about every language know. Even how many different types of different language exist is still not sure. So not everyone will be knowing about it which will lead to difficulty in communication, thus creating a communication barrier among people in turn which can lead to serious problems.

#### **Need of translator**

Language is one way people communicate with each other, if each person speaks different language and each one only has the knowledge of their language then there arises a problem which can lead to difficulties in many ways. This shows us the need of a language translator.

### **2.2 Objectives**

#### **Real-time Detection**

Real-time language detection allows for seamless communication between different individuals, without the need for an intermediary or written communication. This can improve the overall quality of communication and facilitate more effective and efficient communication. It can provide each individuals with greater access to information, as they can understand and participate in discussions, lectures, and other

events in real-time and narrowing their communication barrier.

### **Efficiency and Accuracy**

It allows for faster communication, as it eliminates the need for an intermediary or written communication. This can save time and improve the efficiency more. A reliable and accurate translator can increase the confidence of every individuals, as it allows them to communicate effectively and be understood by the other person. Thus people can rely on it more.

### **Better Communication**

A language translator can facilitate better communication between various individuals, as it allows for real-time communication without the need for an intermediary or written communication. It can enable each person to communicate with other person in real-time, either through text or voice, or through the use of sign language interpretation software that converts sign language into text or speech in real-time. Thereby reducing the communication barrier between people.

### **Faster Translation**

Fast recognition and translation are important for a language translators, as they can enhance the efficiency and accuracy of the translation process. The speed of recognition and translation is especially important, as it allows for more seamless and natural communication between individuals. Fast recognition and translation can also improve the overall quality of communication, as it reduces the potential for misunderstanding or miscommunication.

# **CHAPTER 3**

## **LITERATURE REVIEW**

### **Arabic Sign Language Recognition System Using 2D Hands and Body Skeleton Data**

Sign Language can be considered as the most preliminary means of human communication. If someone travels to any foreign country, ignoring their language completely, he can easily, and by instinct, find a way to make signs to at least drink water, eat food, and get a place to sleep. Throughout this context, deaf and hard of hearing (HoH) people and their surrounding families have developed, over decades, a set of hand gestures, facial expressions, and lip movements to communicate with each other. These signs vary between different countries and languages, though they have some similarities. Nowadays, researchers have conducted many advanced studies in sign languages to help the deaf in their daily life. In addition, through the emergence of the video calls, the deaf society and HoH are no more clustered and pouched in their bulbs, but contribute actively in many domains. Although, the task of recognizing signs in videos could be similar to recognizing actions and gestures it is actually more complex. The complexity comes from detecting singular gestural boundaries in long sequences. This full recognition process is dependent on what is recognized instantly, as a part of an independent sign, or as a boundary between two successive signs. The first point is the absence of a clear stopping sign. In theory, the conventional “hands down” can be used as a stop sign like a “silent pause” in speech. However, this pause might not come at every second, and the length of successive signs may undergo an nondeterministic duration. Moreover, some signers have their own way of stopping or pausing that is more related to the signer fatigue. Many sign experts

Ref	# Signers	# Signs	Collection	Devices	Modality	Gloss	Remark	Acc.	Year	Publicly Available	Country
[1]	3	23 x 50	Words	Camera	RGB	No	-	71%	2007	No	UAE
[2]	40	32	Alphabet	Camera Gray Images	Hands	No	54049 images One Hand	-	2018	Yes	Saudi Arabia
[3]	10	500 (Mixed signs)	Words Sentences	Canon Power Shot A490 Camera	RGB, + Face Emotion	No	Signs World Atlas	97% 95.28%	2014	No	Egypt
[4]	4	1216	Words Sentences	Leap Motion KinectV2, Digital Cam		No	4 Rec. angles		2015	-	Egypt
[5]	1	300		Camera		No	Gloves	93%	2007	-	-
[6]	2	20 x 10 repetitions	Words	Leap Motion +Camera	Face, Body	No		95%		-	-

Table 1 : Arabic language data set

may stop signing for a short instant, “time to recall the next sign”, to recapitulate the idea in their heads and continue, but the pause is not unified between signers, and does not appear to be a pause if one is not aware of it in advance. It is a similar approach to pauses in speech, but the hands are still used, unlike the speech, where speech sound stops for a short instant (time to take a breath or even drink some water in long discourses). From our discussions with Arabic Sign language (ArSL) experts, signers do this approach to make pauses, and care more about the next idea in the sign discussion. The second point is the inclusion of the facial expressions, including the lips and the eyebrows movement, which lead to additional vectors or modalities in the sign recognition process. Mouth closure is also used as a pause, but more research still needs to be done in this case. Nevertheless, many SL research papers simply ignore the face entirely, either it was not recorded, or recordings did not focus on the face. In this paper, we propose a new approach for recognizing dynamic and static signs. Our approach is a two steps fold, the first step, we estimate the body joints including the finger positions from input sequence of frames, in the second step a point CNN model is used to differentiate between the sign classes. In order to select the best number of successive frame images useful for a fast decision, we introduce the index inference score (IES), that relates the processing speed of the system, to the accuracy of the model. This parameter will determine the optimal number of frames to be fed to the network to decide and give a result. The rest of the paper is organized as follows, in section 2, we develop a literature review of ArSL datasets, methods and results. In section 3, we present our methodology in recording the dataset and the proposed methods to recognize automatically the signs. In section 4, we present the experimental results, while in section 5, we mention the

diverse problems encountered during the ArSL dataset recording stage. Finally, in the last section, we conclude our research and propose some extension ideas.

The main bottleneck problem in SL is the detection of the hands, the arms, the face, and the exact location of the fingers. Once the output from a camera or a video file is fed to the OPL network, the hands and body key points are generated, as shown in the global view architecture of Figure 1. Each RGB frame is thus converted to a set of 2D key points. The system waits to collect a small sequence of frames and sends this sequence to the SKN network to generate the recognized Arabic sign.

## **OPENPOSE NETWORK ARCHITECTURE**

One of the state of the art framework in pose estimation is the OpenPose library (OPL). The OPL is a very powerful framework, as it estimates, from a single image, a set of key points of one or more persons, and provides with a very high accuracy, the body, hands and finger joints. This human pose estimation library was developed in C++ and uses Caffe as a backbone deep neural network. The second network that will be used is a point convolution network that will be adapted and tuned to accept a two-dimensional series of data and will output the probability of an ArSL sign. OPL has surpassed the state-of-the-art research in recognizing or estimating the pose of individuals

The latest version of OPL can generate 21 key points per hand, 70 key points for the face, and 25 key points for the body/foot. The body key points might differ depending on the trained model (COCO, MPI), and the key points can be generated in real time. The OPL network, when fed by one or more successive frames, generates for each frame a set of X, Y coordinates, a probability or confidence score for each joint of the body, and relevant network, where input images are fed to the first ten layers of the VGG19. The output feature maps are then transferred to a second network consisting of two parallel six stages sub networks. OPL generates all the 2D key points from the input image points within the face. The initial structure of the OPL is identical to the VGG19 FIGURE 4 and FIGURE 5 show some sample output images from our KSU-ArSL dataset, where joints are cor recently identified, while FIGURE 6 focuses on some of the intermediate frames where the OPL could not generate a valid hand skeleton (No or bad finger candidate key point generated).

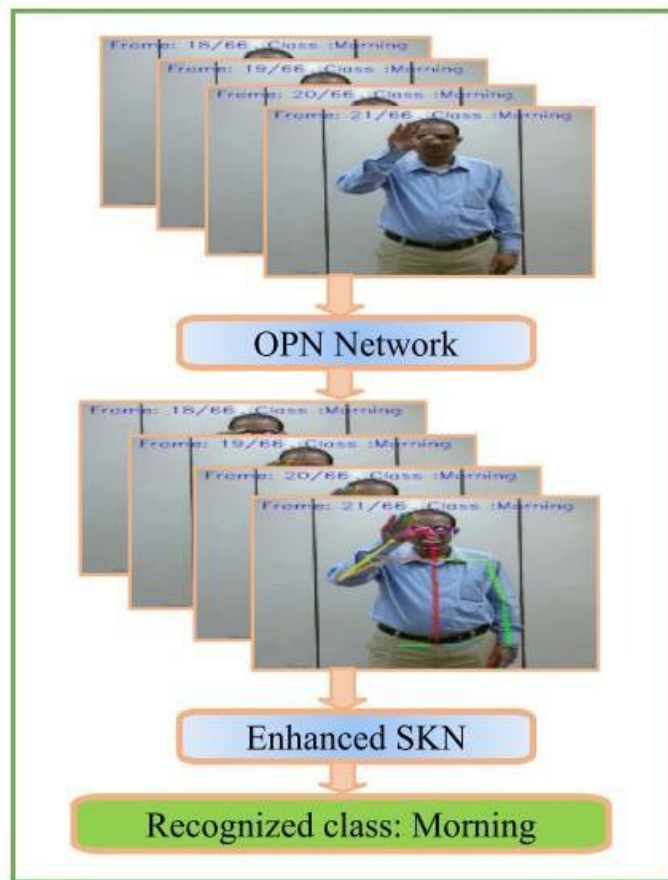


Fig 1: General overview of the system



Fig 2: Example of OPL output on the ArSL dataset

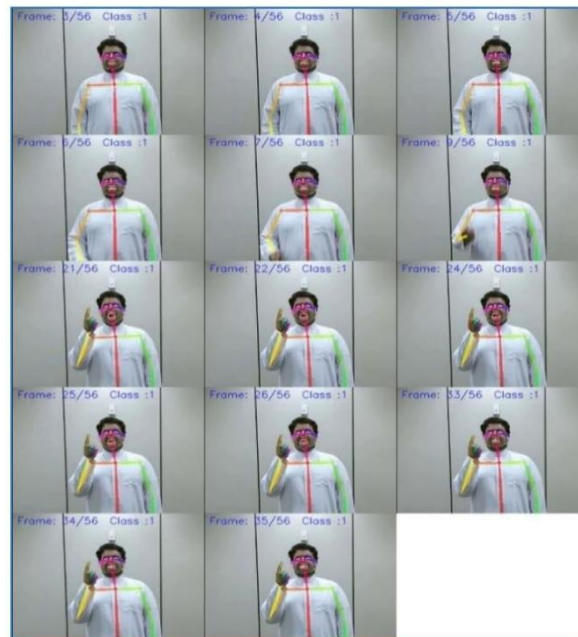


Fig 3: Example of the OPL output on the ArSL dataset, static sign—joints correctly identified.



Fig 4: example of the OPL output on the ArSL dataset, static sign—joints correctly identified.

## ENHANCED PARALLEL SKELETAL DEEP CNN

Originally inspired from [1], the authors of proposed a skeletal CNN network (SKN) fed by three dimensional coordinates X, Y and Z, generated by the 3D Intel real sense depth camera software development kit. This type of camera generates 3D key points of the hands within a 3D space. Results from the paper have shown an accuracy of 91.28% for 14 gesture classes from the DHG datasets, but had limitations as it dropped by 7%, when doubling the number of classes to 28. Our contribution to the modified SKN network dealt with two major points: Firstly, we simplified the network architecture to use 2D points instead of 3D points, which led to 1/3 less computation per network branch. Secondly, we extended the input layers to use the full body 48 key points, see Table 7, instead of only the two hands 3D points generated by the 3D Intel real sense. Given the selected set of OPL key point coordinates, we fed these points to a second parallel network that inputs each key point to three parallel branches, a low-resolution, a



high resolution, and a pooling branch. The difference between the network proposed by and the OPN network, is that SKN convolutions are point convolutions and no connection or concatenation occurs at mid-stages, a detailed view of the network is presented in . Each of the channels of the network is a component of a multivariate time sequences.



Fig 5: Hand skeleton generation errors, right hand fully/partially blurred (Dynamic Sign)— finger joints not identified

	Number of original OP Key-points: (X, Y, Confidence probability)	List of selected key points used in the SKN
<b>Body</b>	(0 to 24): 25 x3	(0,1,2,3,4,5,6,7, 15,16,17,18): 12
<b>Hands</b>	(0 to 20): 21 x 3 x 2 hands	(0 to 20) 21 x 2 x 2 hands:84
<b>Face</b>	(0 to 69): 70 x 3	-
<b>Total</b>	$25 \times 3 + 21 \times 3 \times 2 + 70 \times 3 = 411$	$21 \times 2 \times 2 + 12 = 96$ <b>48 (X, Y) coordinate</b>

Table 2: List of body Joints used as input to the SKN.

Variations over each channel are independent and do not participate in the update of the weights of the other components. Each input channel size was modified to accept a two dimensional X, Y coordinates generated by the OPL network. We do not provide nor use the depth information, as in the original design, but we enrich the input with some additional selected body joints, instead of the hands only, as per the original design. Details of our model joints are presented in Table 2.

This research paper proposed an Arabic sign language automatic recognition framework, which consists of using a new ArSL Dataset recorded at our university premises. The dataset was recorded with three cameras, a Kinect V1, a Kinect V2, and a Sony handy cam. The 40 signers recorded each 80 signs, five times, resulting in a multimodality dataset that comprises RGB images, Depth images and body skeletons from the Kinect V2. In this paper, we investigated exclusively the RGB images of the Kinect V2 by proposing the concatenation in serial of two parallel networks, a 2D CNN network for key-points estimation and a second 1D CNN skeleton network. The automatic ArSL results are very promising, as our best network configuration recognized 98.39% for dynamic signs and 88.89% for static signs in the signer dependent mode, and an accuracy of 96.69% for dynamic signs and 86.34% for static signs in the signer independent mode. When the same network is trained by both dynamic and static signs, a test accuracy of 89.62% for the signer dependent and 88.09% for the signer independent mode were recorded. The use of the inverse efficiency score showed that with an optimal number of sufficient frames, as input to our system, the tradeoff between accuracy and speed could be enhanced, if such models are deployed on production. A complementary solution would be to add a sign boundary detector or a network that can decide whether the actual shape is transient gesture or a sign. More investigation will be axed on the frames between effective signs and their properties in the continuous ArSL. To enhance the accuracy of each signer in both dependent and independent modes, more research would be made on detecting the minimal set of distinguishing frames and/or key points that represent a sign by clustering the generated frame key points into representative reduced clusters, so that the deep model can be compact and lighter on mobile devices. Another additional point is to improve delay removal in the

whole pipeline via convolution suppression and optimal data propagation, aiming to reduce the network size and optimize the classification speed. Some other improvements would be to zoom on fingers and add an automatic technique to detect the palm orientations because we noticed that the drop in accuracy was mainly due to the difficulty to detect the similar signs that shapely resemble each other, but differ by one finger or by the palm orientation.

#### **Drawbacks of existing system :**

- It only does the recognition for recorded data only
- It has more latency
- Most of its frames were blurry and data was not usable
- It efficiency and accuracy was low

#### **Advantages of our system**

- The Recognition is done in real time
- Latency is very low ,thus giving use more efficiency and accuracy
- More accurate and improves its algorithm by self-learning with the use of machine learning

# CHAPTER 4

## SYSTEM REQUIREMENTS

### 4.1 HARDWARE

- Modern Operating System
  - Windows 7 or 10
  - Mac OS X 10.11 or higher, 64 bit
  - Linux: RHEL 6/7, 64 bit (almost all libraries also work in Ubuntu)
- x86 64-bit CPU (Intel / AMD architecture)
- 4 GB RAM
- 5 GB free disk space

### 4.2 SOFTWARE

- Visual Studio Code
- MySql Workbench
- GoogleTrans
- MediaPipe

#### Visual Studio Code

Visual Studio Code, also commonly referred to as VS Code, it is a source-code editor made by Microsoft with the for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality. Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including C#, Java, JavaScript, Go, Node.js, Python, C++, C, Rust and Fortran. It is based on the Electron framework,[21] which is used to develop Node.js web applications that run on the Blink layout engine. Visual Studio Code employs the same editor component (code named "Monaco") used in Azure DevOps. Visual Studio Code includes

basic support for most common programming languages. This basic support includes syntax highlighting, bracket matching, code folding, and configurable snippets. Visual Studio Code also ships with IntelliSense for JavaScript, TypeScript, JSON, CSS, and HTML, as well as debugging support for Node.js. Support for additional languages can be provided by freely available extensions on the VS Code Marketplace

## **MySql WorkBench**

MySQL Workbench is a unified visual tool for database architects, developers, and DBAs. MySQL Workbench provides data modeling, SQL development, and comprehensive administration tools for server configuration, user administration, backup, and much more. MySQL Workbench is available on Windows, Linux and Mac OS X.

MySQL Workbench is offered in the following Editions:

- MySQL Workbench Community Edition — Open Source (GPL License)
- MySQL Workbench Standard Edition — Commercial
- MySQL Workbench Enterprise Edition — Commercial

## **GoogleTrans**

Googletrans is a free and unlimited python library that implemented Google Translate API. This uses the Google Translate Ajax API to make calls to such methods as detect and translate. It supports 133 languages at various levels. GoogleTrans neural machine translation system uses a large end-to-end artificial neural network that attempts to perform deep learning in particular, long short-term memory networks. GNMT improves the quality of translation over SMT in some instances because it uses an example-based machine translation (EBMT) method in which the system "learns from millions of examples. GNMT's "proposed architecture" of "system learning" has been implemented on over a hundred languages supported by Google Translate. With the end-to-end framework, Google states but does not demonstrate for most languages that "the system learns over time to create better, more natural translations. The GNMT network attempts interlingual machine translation, which encodes the "semantics of the sentence rather than simply memorizing phrase-to-phrase translations" and the system did not invent its own universal language, but uses "the commonality found in between many languages".

## **Mediapipe**

Mediapipe is an open-source cross-platform framework for building real-time multimodal perceptual systems that can process audio, video, and sensor data. It was developed by Google and released in 2019 under the Apache 2.0 license. Mediapipe enables developers to build complex AI-based applications that can take advantage of machine learning and computer vision algorithms. The framework provides a set of pre-built pipelines that developers can use to quickly build applications that can perform tasks like object detection, face tracking, hand tracking, pose estimation, and more. These pipelines consist of a series of modular components that can be customized to fit specific use cases. The components are designed to be composable and reusable, making it easy to create complex pipelines that can handle multiple modalities of input. It is designed to be efficient and lightweight, with a focus on real-time processing. It can be used on a variety of platforms including Android, iOS, desktop, and cloud. The framework is written in C++ and Python and provides APIs for both languages. It also supports popular machine learning frameworks like TensorFlow, making it easy to integrate custom models into pipelines. One of the key features of Mediapipe is its support for hardware acceleration, including GPUs and specialized neural network accelerators like Google's Edge TPU. This makes it possible to perform complex tasks like object detection and pose estimation in real-time, even on resource-constrained devices like mobile phones. Mediapipe has gained popularity in the computer vision community due to its ease of use and the wide range of tasks that it can perform. It has been used to build a variety of applications including augmented reality filters, gesture recognition systems, and real-time video analytics tools. Overall, Mediapipe is a powerful framework that enables developers to build complex perceptual systems with ease. Its modular design and support for hardware acceleration make it a versatile tool for a wide range of applications.

# CHAPTER 5

## SYSTEM ARCHITECTURE

A system architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.

A system architecture can consist of system components and the sub-systems developed that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture, collectively these are called architecture description languages (ADLs).

Several types of systems architectures (underlain by the same fundamental principles) have been identified as follows:

- Hardware architecture
- Software architecture
- Enterprise architecture
- Collaborative systems architectures (such as the Internet, intelligent transportation systems, and joint air defense systems)
- Manufacturing systems architectures
- Strategic systems architecture.

A system architecture diagram is the distribution of the functional correspondences. These are formal elements, the embodiment of concepts and information. Architecture defines the relations between elements, among-st features, and the surrounding elements. Creating an Architecture diagram is not easy. An architecture diagram is a diagram that depicts a system that people use to abstract the software system's overall outline and build constraints, relations, and boundaries between components. It provides a complete view of the physical deployment of the evolution road-map of the software system.

A diagram is similar to a picture. The architecture diagram examples serve various functions. It always helps the relevant users to learn about system architecture and apply it in the decision-making procedures. It is crucial to communicate information regarding architecture. However, people must follow specific steps before making a diagram for architecture. These are:

- Breaking down communication barriers
- Reaching a consensus
- Decreasing ambiguity

System architecture diagram focuses on the structure along with the technological requirements, external services, servers, and databases.

Here, in the case of our software , First the user enters the software, the user will be given a interface with a start and quit option, the start to use the application and quit to quit out of the application. Once the use presses start, he/she will be given option sign-up to sign-up for the first time if it's a new user, Else the user can select the login option and login with his/her credentials. The user-data is stored in the local database. Once the account has be created or logged in , the user will be directed to the main page , where there will be three options, Text Translation, Voice Translation, and Sign translations. In the text translation module, the module can translate text-to-text translation, text-to-voice translation. In voice translation, the module can translate voice-to-text and voice-to-voice translations. In the Sign Translation module, Sign language-to-text is done.

Our Application runs on five phases, first phase is collection of frames using openCv. These frames are collected to extract data from each frame for the next step, i.e, giving hand landmarks for the collected frames. Hand landmarks can help a lot to extract more information out of the frame.

After the hand landmarks are set, then it is normalised to get a consistent value for the hand landmarks, so that wherever the hand is shown a consistent value is always achieved. Now using the normalised hand landmarks training is done with the data set. Once the training is done. System will be able to detect the signs when it is shown to the system through the webcam. The last and final phase is the tuning phase where the system improves itself by fine tuning its algorithm ,precision , recall by training for longer period of time.



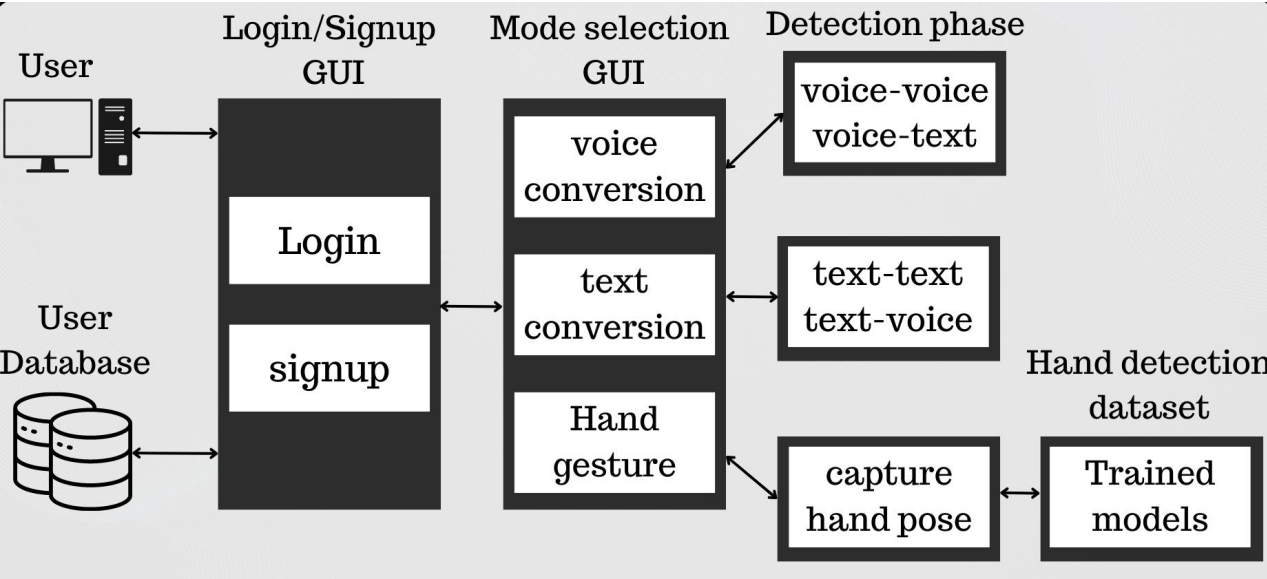


Fig 6: System architecture Diagram

Username	Mail id	Password
----------	---------	----------

Table 3: Database Schema

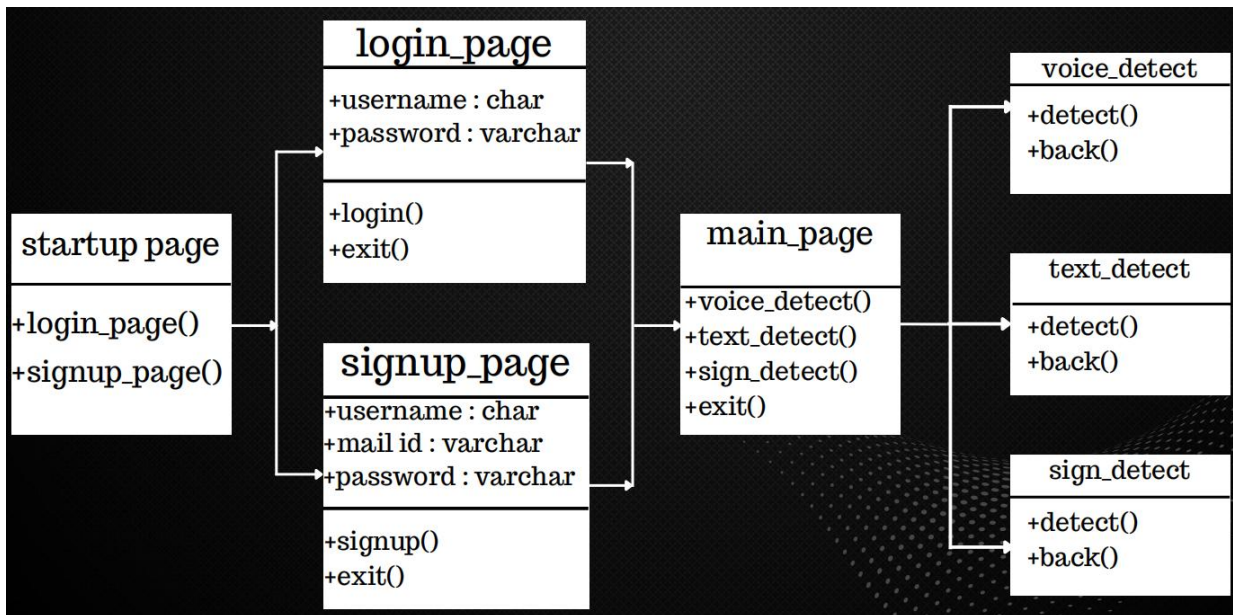


Fig 7: UML Diagram

# **CHAPTER 6**

## **APPLICATION ARCHITECTURE DIAGRAM**

An application architecture diagram provides a high-level graphical view of the application architecture, and helps you identify applications, sub-applications, components, databases, services, etc, and their interactions. The application architecture diagram is a well-known and highly useful figure in both the software engineering and cloud-native applications world. The application architecture is a pictorial representation of the application showing the whole application and its components like front-end, back-end, databases, cloud, micro-services, sub-applications, etc.

For making an application architecture diagram, you have to gather all the stakeholders' information that will be merged to make an application. Some of the useful components of the application architecture diagrams are.

### **6.1 STARTUP PAGE**

Here the user is given the first glance of our application, where he/she is treated with the startup page, where the user can see the application name in the centre and two options below it . To start the application or to quit from it. To continue to use the application the user can hit start to proceed forward, else if want to exit from the application the user can just click quit and quit out from the application.

### **6.2 HOMEPAGE**

The homepage comes after the startup page after the user clicking “ start “ option in the startup page. Here the user can see the same application logo in the centre and a signup option below it. If the user is a new user then he/she can just go with the signup option. Else if the user is a existing user then he/she can select the login option just below the signup option. There is also a back button on the right top corner, the user can use it if he/she wants to go back to the startup page.

### 6.3 SIGNUP PAGE

A signup page (also known as a registration page) enables users to independently register and gain access to our system. This software allows user to register by providing username, email id and password. SIGNON validates the each fields, username should be unique ,email should be valid,password must follow the password policy

- At least 12 characters
- A mixture of both uppercase and lowercase letters.
- A mixture of letters and numbers.
- Inclusion of at least one special

character, e.g @ #If any error occurs, it displays an error message.

User is then navigated to the main page after signing up for the first time. There is also a Home option on the top right corner to go back to the home page.

### 6.4 LOGIN PAGE

The login page allows a user to gain access to an application by entering their username and password . a user can login to our SIGNON application .Username and password should exist in the database and display an error if user does not exist or mismatch in username and password occurs . There is also a Home option on the top right corner to go back to the home page.

### 6.5 MAIN PAGE

The main page is the most important page , where the user can start using out our application for language translation. There will be three option available for using the various translation system.Text Translation, Voice Translation, And Sign Language Translations. Clicking on any one of them will lead to their respective window where the translation is done. There is also a Home option on the top right corner to go back to the home page.

## **6.6 TEXT-TO-TEXT TRANSLATION PAGE**

Here Text is translated to text. After clicking the text translation button on the main page it takes the user to text to text translation page first. Here we can see a drop-down menu to select the input language, just below it is a text-box where the user can type the text to be translated. After entering the text, the user can choose the language which the text is to be translated. A separate drop-down list is provided for the same. After choosing it, to implement the translation, the user can click translate button. The system processes everything and the text is translated in the user required language below.

Now if the wants to do text-to-voice translation, click the option below to convert it into voice, it will take the user to its respective page.

## **6.7 TEXT-TO-VOICE TRANSLATION PAGE**

Now if the wants to do text-to-voice translation, click the option below to convert it into voice in the text-to-text translation page, it will take the user to text-to-voice translation page page where the user can choose the input language from the first drop-down list, enter the required text to be translated in the input text box, then choose the target language from the next drop-down list. After doing this click translate and the system will translate the text to voice in the required target language and the system will dictate it loud to the user through the speaker.

## **6.8 VOICE-TO-TEXT TRANSLATION PAGE**

Here Voice is translated to text. After clicking the voice translation button on the main page it takes the user to voice to text translation page . Now the user can select the input language and the target language, then click start recording button. The user can Dictate the sentence or words through the mic, the software will detect it and will convert it into the target language and the output ,i.e, the sentence in target language will be displayed in the text box below.

Now if the user wants to go back to text to text translation page, then, he/she can click the button below which will take the user back to the text-to-text translation page. Also a Home button is present at the top corner , on clicking will take the user make to the main page.

## 6.9 VOICE-TO-VOICE TRANSLATION PAGE

Now if the wants to do voice-to-voice translation, click the option below to convert it into voice in the voice-to-text translation page, it will take the user to voice-to-voice translation page page where the user can choose the input language from the first drop-down list, then choose the target language from the next drop-down list. After doing this click start recording and the system will translate the voice to voice in the required target language and the system will dictate it loud to the user through the speaker. Now if the user wants to go back to the voice to text translation page, he/she can click the prompt below to go back to voice to text page. Also a Home button is present at the top corner , on clicking will take the user make to the main page.

## 6.10 SIGN LANGUAGE TRANSLATION PAGE

On clicking the Sign translations button on the main page, the user will be guided to the Sign Language Translation page. In the Sign Language translation page the user can show signs to the system through the system's webcam, and the system will recognize the signs in real-time, a box will appear everytime the system detects the sign and the meaning of the sign will be shown in the bottom of the box. If the user wants to go home then the user can click the home button which is on the top right corner of the console.

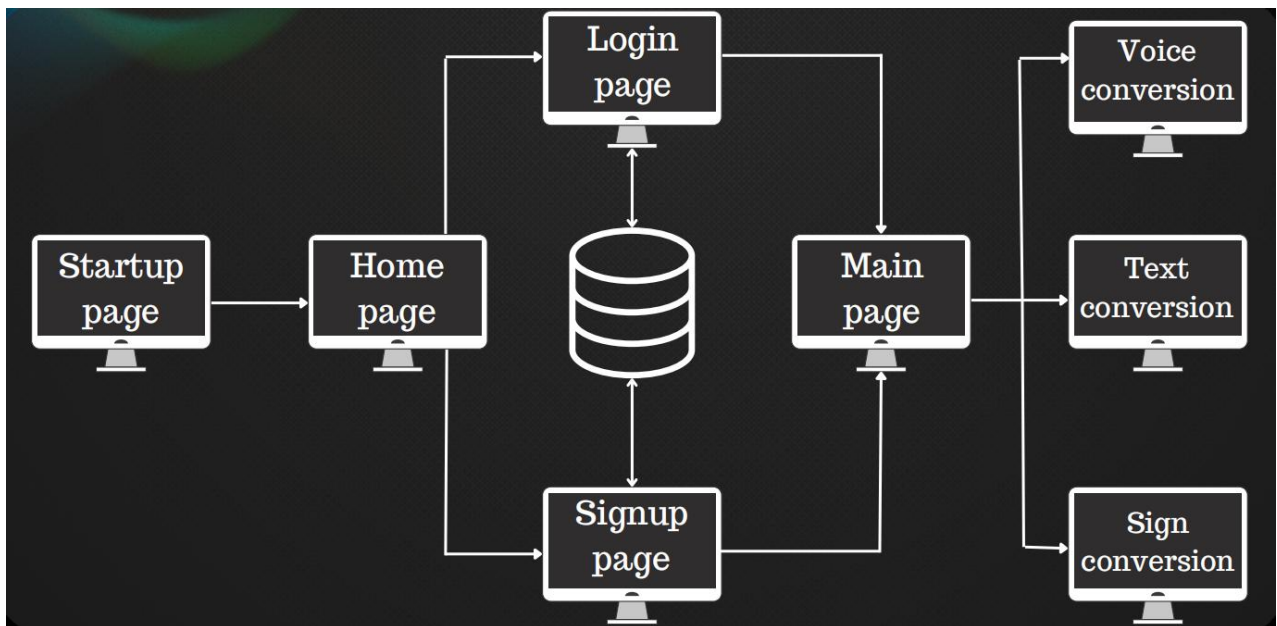


Fig 8: APPLICATION ARCHITECTURE DIAGRAM

# CHAPTER 7

## MODULES

### 7.1 LOGIN/SIGNUP

When the login button is clicked we are redirected to the login page. Here we have two text-fields and three buttons. The first text field is for entering the username and the second text field is for entering the password. We are connected to our page with a database. In this database there is a table for the registered persons details. When the username and password is entered the user clicks the button. The software checks here the username entered is compared with the usernames that exist in the table and if it gets a match then it looks for the password to check if it is right or wrong. If the password is wrong it shows an error message that invalid password or if the password is entered correctly then the user gets login and redirected to the main page which is the detecting page. Syntax for database connectivity:

```
mydb =
mysql.connector.con
nect(host="localhost",
user="yourusername",
password="yourpass
word"
)
mycursor =
mydb.cursor()
mycursor.exe
cute
("CREATE DATABASE mydatabase")
```

Login signup pages have username ,email,password validation .The datas entered in the signup frame are stored in the SIGNON database.

If the person is a new user he can create a new account by clicking on the signup button

and after this he is redirected to the Sign up page. In the Sign up page there are three entry fields and one button for registering. Entry field is for entering the username and second entry field is for entering email id and the third entry field is for entering password. After entering these details the user clicks the register button, the software checks if the username entered is compared with the usernames that exist in the database table and if there is any matching username then it shows in error message by saying the user already exist and if it is not matching then it checks for the email address if it is in a valid form and also the password for checking if it is in a valid form. If all the conditions above get right then the new user account is created and a message shows by saying this. There are also two other buttons in this page login and home. After creating a new account, the user can go back to the login page by using this login button and he can now login by using his details or if he doesn't want to create a new account then he can go back to the first page by clicking on the home button.

## **7.2 TEXT TRANSLATIONS**

Here Text is translated to text or voice. After clicking the text translation button on the main page it takes the user to text to text translation page first. Here we can see a drop-down menu to select the input language, just below it is a text-box where the user can type the text to be translated. After entering the text, the user can choose the language which the text is to be translated. A separate drop-down list is provided for the same. After choosing it, to implement the translation, the user can click translate button. The system processes everything and the text is translated in the user required language below. The Software does the translation using the googletans module, which converts the text to the required target language

Now if the wants to do text-to-voice translation, click the option below to convert it into voice, it will take the user to its respective page. After that, In the text-to-voice translation page, click the option below to convert it into voice in the text-to-text translation page, it will take the user to text-to-voice translation page page where the user can choose the input language from the first drop-down list, enter the required text to be translated in the input text box, then choose the target language from the next drop-down list. After doing this click translate and the system will translate the text to voice in the required target language and the system will dictate it loud to the user through the speaker.



## 7.3 VOICE TRANSLATIONS

Here Voice is translated to text. After clicking the voice translation button on the main page it takes the user to voice to text translation page. Now the user can select the input language and the target language, then click start recording button. The user can Dictate the sentence or words through the mic, the software will detect it and will convert it into the target language and the output, i.e., the sentence in target language will be displayed in the text box below.

Now if the user wants to go back to text to text translation page, then, he/she can click the button below which will take the user back to the text-to-text translation page. Here the user can choose the input language from the first drop-down list, then choose the target language from the next drop-down list. After doing this click start recording and the system will translate the voice to voice in the required target language and the system will dictate it loud to the user through the speaker. Now if the user wants to go back to the voice to text translation page, he/she can click the prompt below to go back to voice to text page. Here the entire translation is done using the googlettrans module, it converts the input given by the user to the target language in text or voice based on user's preference

## 7.4 SIGN LANGUAGE TRANSLATION

After the user clicking sign translations option in the main page, it is redirected to the detecting console. In that interface the webcam of the system will be prompted to open and the video interface will be show. The user can start showing the sign to the software through the webcam. When the system detects a sign shown by the user, it freezes that frame takes that frame and analyze the data in that frame, using the machine learning algorithm provided the software will detect that it's a specific sign with a meaning that the system identified, this sign shown by the user will be highlighted by the system using a box and the meaning of the sign shown by the user will be also show below the box highlighted by the software.

The detection module works on these steps:

- Collecting frames using WebCam using OpenCv
- Creating hand landmarks for the collected frames
- Normalizing the hand landmarks
- Training dataset using the normalized hand landmarks
- Performance Tuning

**Collecting frames using WebCam using OpenCv :** The first part of this software is collecting frames from the webcam for analyzing, This is done by using OpenCV. It will help us to take a particular frame and extract its data from it, which is a crucial part for our software.

**Creating hand landmarks for the collected frames:** Hand Landmarks are points which is denoted by each pixel, and this helps to mark each points in a hand thus giving us data about the hand.

**Normalizing the hand landmarks:** In this phase , the hand landmarks which are gives is being normalized,i.e, it is made into a base value which will remain constant, even if the hand is shown anywhere on the screen.

**Training dataset using the normalized hand landmarks:** In the training part, dataset is trained using the normalized hand landmarks,i.e, giving it more meaning and labelling each signs according to it.

**Performance Tuning:** The phase where the entire software can be made more accuracte and efficiency by giving more data to work with, training more number of times, improving and making chances to the algorithm, Overall making the precision high and recall low.

# CHAPTER 8

## USER INTERFACE

### 8.1 WELCOME PAGE

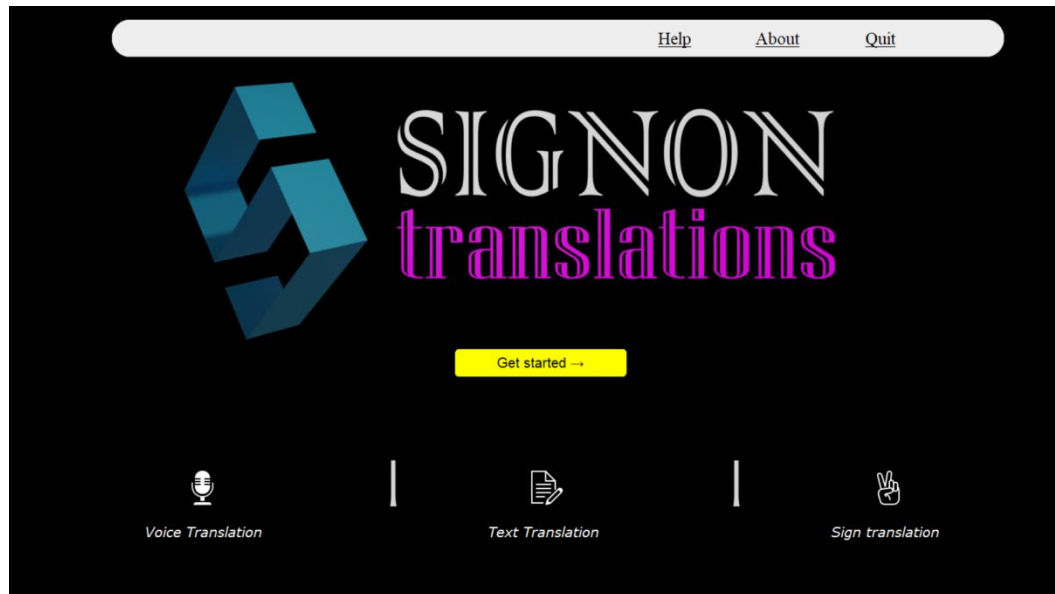


Fig 9: Welcome Page

### 8.2 LOGIN PAGE

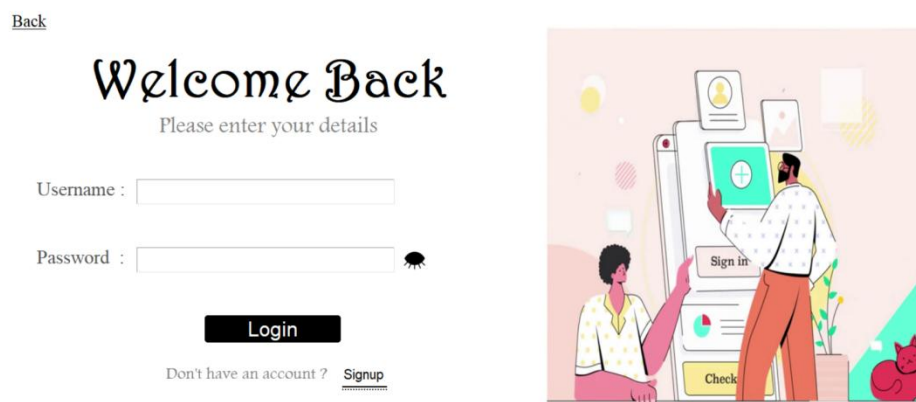
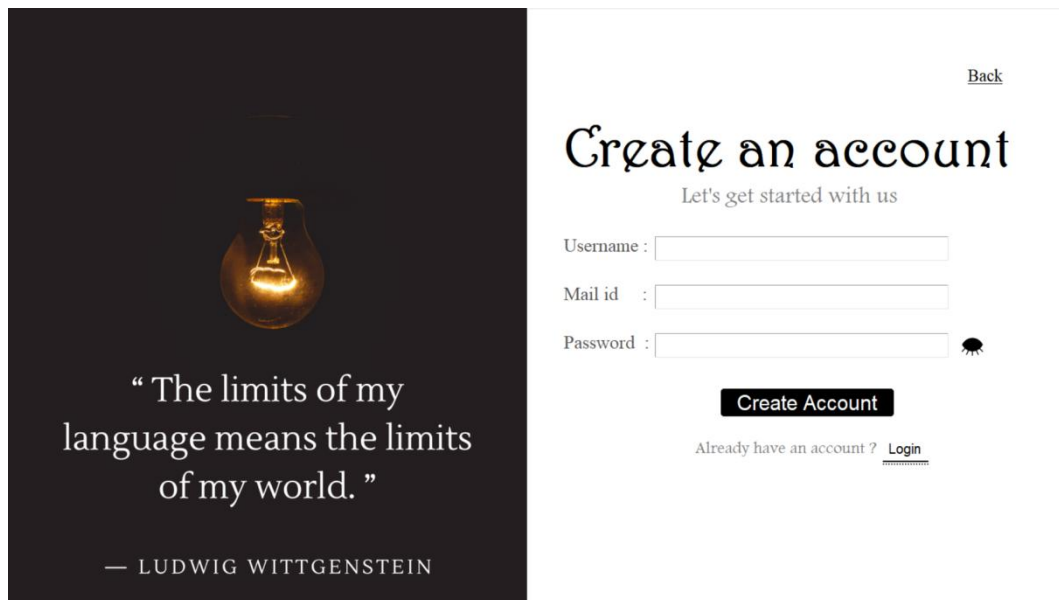


Fig 9: Login Page

### 8.3 SIGNUP PAGE



The Signup Page is divided into two main sections. The left section has a dark background and features a glowing lightbulb icon. Below the icon is a quote: "The limits of my language means the limits of my world." attributed to — LUDWIG WITTGENSTEIN. The right section is white and contains the heading "Create an account" with the subtext "Let's get started with us". At the top right of this section is a [Back](#) link. The form includes three input fields: "Username :", "Mail id :", and "Password :". The "Password :" field has a small eye icon to its right. Below the inputs is a black "Create Account" button. At the bottom, there is a link "Already have an account ? [Login](#)".

Fig 9: SignUp Page

### 8.4 MAIN PAGE

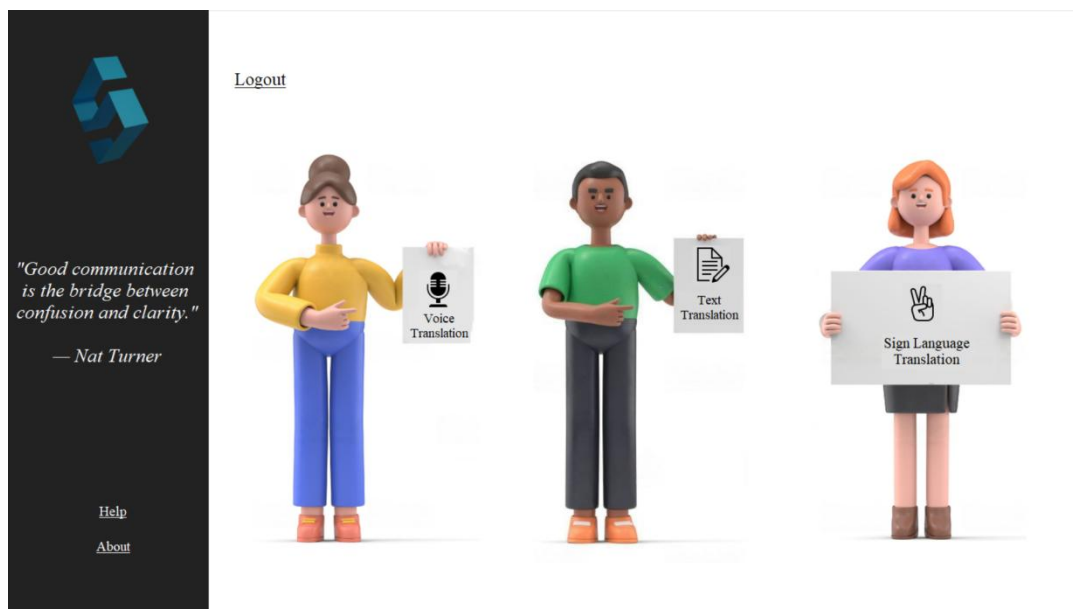


Fig 9: Main Page

## 8.5 TEXT-TO-TEXT TRANSLATION PAGE

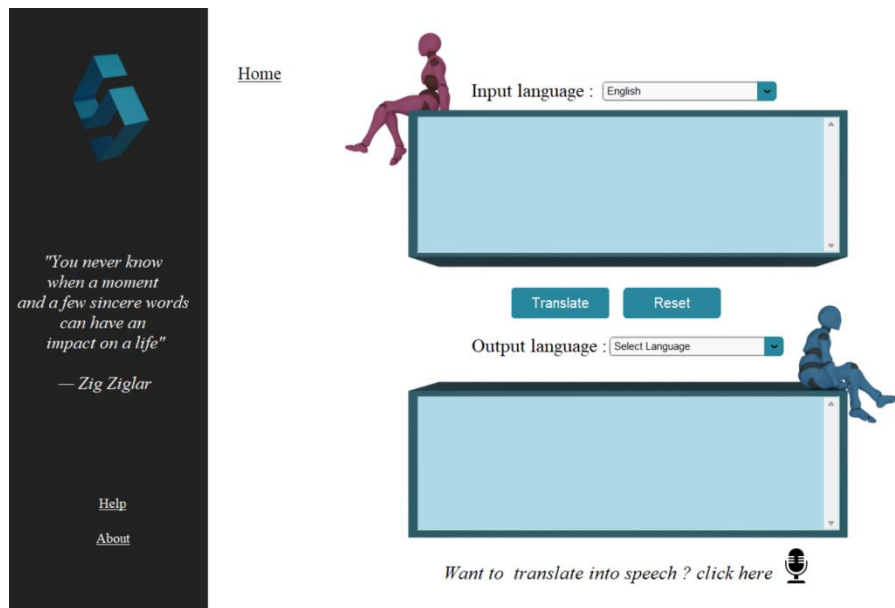


Fig 9: Text-to-text Translation Page

## 8.6 TEXT-TO-VOICE TRANSLATION PAGE

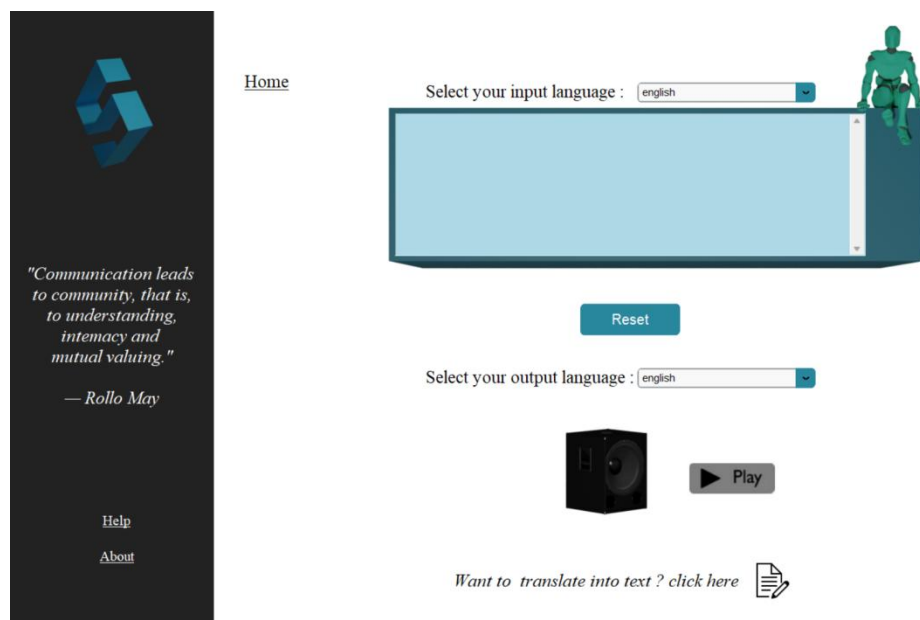


Fig 9: Text-to-Voice Translation Page

## 8.7 VOICE-TO-TEXT TRANSLATION PAGE

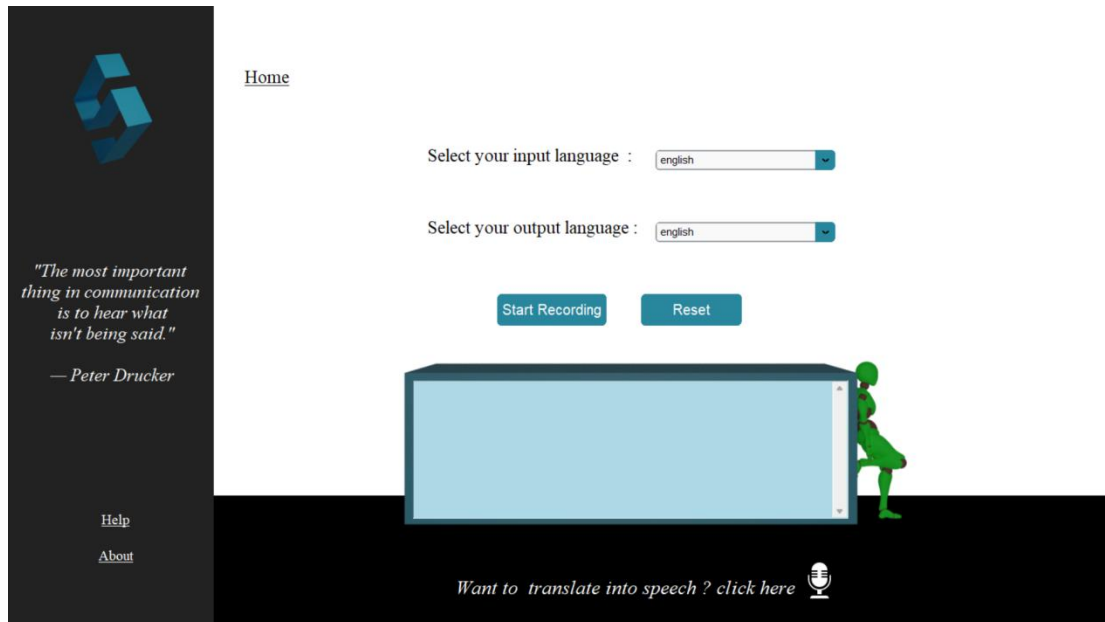


Fig 9: Voice-to-text Translation Page

## 8.8 VOICE-TO-VOICE TRANSLATION PAGE

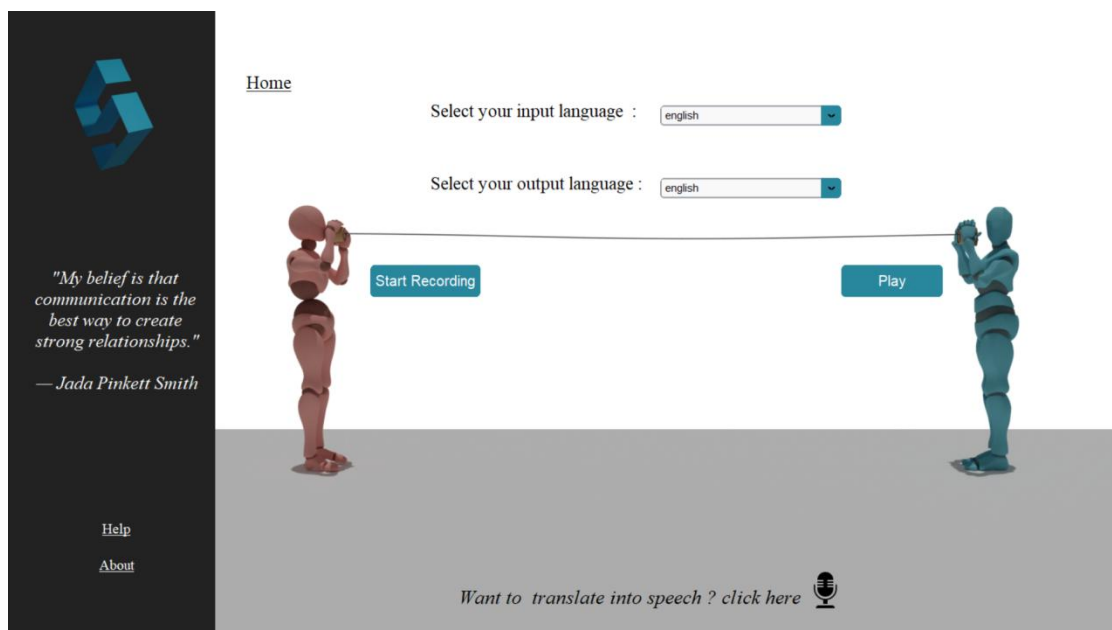


Fig 9: Voice-to-Voice Translation Page

## 8.9 SIGN LANGUAGE TRANSLATION PAGE

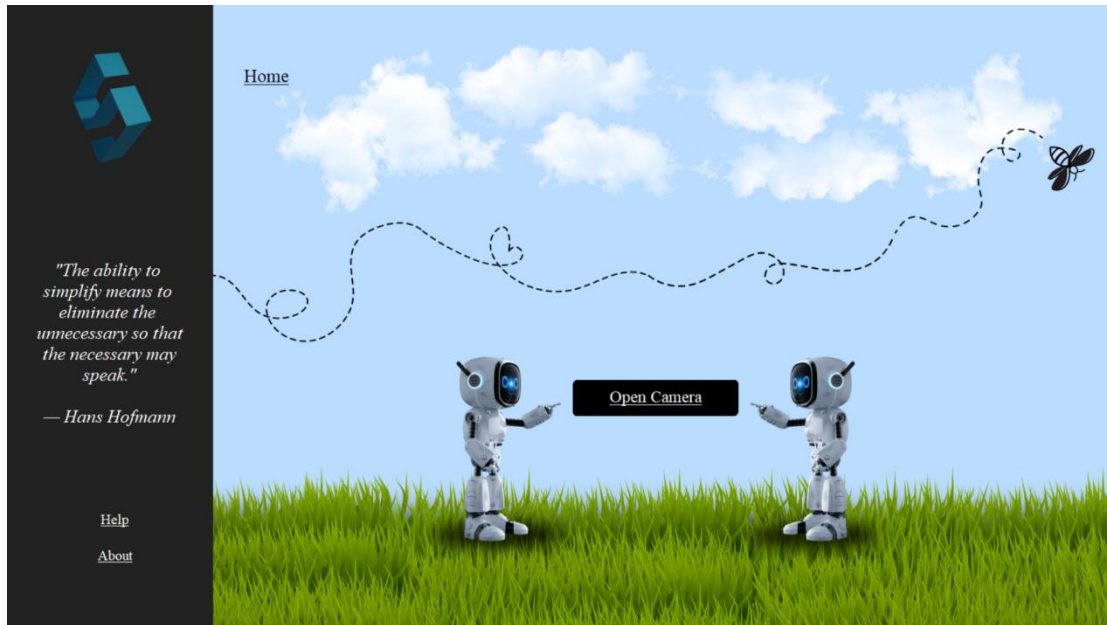


Fig 9: Sign Language Translation Page

## **CHAPTER 9**

### **CONCLUSION**

In modern world ,communication plays a vital role in everything that a person do in his/her daily life. A little mistake or misunderstanding in communication can lead to serious problems among people. With our software we aim to narrow the gap that has been there for the communication of deaf and mute people.

As for people using different languages from all over the world, we aim to provide a platform that will greatly help them in all ways to make their communication as clear as possible if it is in the form of Text, Voice or even Sign Language, our software will help them in having a healthy communications without any trouble.



## CHAPTER 10

### SOURCE CODE

```
##### importing modules #####
```

```
import tkinter as tk
from tkinter import *
import customtkinter
from tkinter import messagebox,ttk
import tkvideo
from tkvideo import tkvideo
import mysql.connector
import googletrans
from googletrans import Translator
import socket
from playsound import playsound
import speech_recognition as sr
from gtts import gTTS
import os
import time
```

```
##### creating tkinter #####
```

```
root=tk.Tk()
root.state("zoomed")
root.overrideredirect(True)
```

```
##### creating frames #####
```

```

fstartup=tk.Frame(root)
flogin=tk.Frame(root)
fsignup=tk.Frame(root)
fmain=tk.Frame(root)
ft_t=tk.Frame(root)
ft_v=tk.Frame(root)
fv_t=tk.Frame(root)
fv_v=tk.Frame(root)
fhand=tk.Frame(root)

##### function to check the button working or not #####

def clicked():
    messagebox.showinfo("button clicked","User have clicked the button")

##### Internet connection #####

# img24=PhotoImage(file="wifiok.png")

# img25=PhotoImage(file="nowifi.png")

# l48=Label(root,image=img24,background="white")

# def test_net():
#     try:
#         socket.create_connection(('google.com',80))
#         l48.configure(image=img24)
#         l48.place(x=1350,y=20)
#     except OSError:
#         l48.configure(image=img25)
#         l48.place(x=1350,y=20)

```

```
# root.after(1000,test_net)
```

```
##### startup page #####
```

```
cstartup=Canvas(fstartup,height=1080,width=1920,  
    background="black",bd=0,highlightthickness=0,relief="ridge")  
cstartup.place(x=0,y=0)
```

```
img1=PhotoImage(file="logo_title.png")
```

```
l1=Label(cstartup,image=img1,background="black",border=0,height=540,width=960)  
l1.place(x=250,y=20)
```

```
img2=PhotoImage(file="mic.png")
```

```
l2=Label(cstartup,image=img2,background="black",height=100,width=100)  
l2.place(x=230,y=650)
```

```
img3=PhotoImage(file="text.png")
```

```
l3=Label(cstartup,image=img3,background="black",height=100,width=100)  
l3.place(x=730,y=650)
```

```
img4=PhotoImage(file="hand.png")
```

```
l4=Label(cstartup,image=img4,background="black",height=100,width=100)  
l4.place(x=1230,y=650)
```

```
l10=customtkinter.CTkLabel(cstartup,text=" ",fg_color="#EEEEEE",  
    S.N.M.I.M.T. , Maliankara
```

Department of CSE

```

width=1300,height=54,bg_color="black",corner_radius=50)
l10.place(x=150,y=20)

b1=customtkinter.CTkButton(cstartup,text="Help",font=("times",25,"underline"),bg_color="#EEEE
EEE",
    fg_color="#EEEEEEE",hover_color="grey",text_color="black",command=clicked)
b1.place(x=900,y=30)

b2=customtkinter.CTkButton(cstartup,text="About",font=("times",25,"underline"),bg_color="#EE
EEEE",
    fg_color="#EEEEEEE",hover_color="grey",text_color="black",command=clicked)
b2.place(x=1050,y=30)

def quit():
    root.destroy()

b3=customtkinter.CTkButton(cstartup,text="Quit",font=("times",25,"underline"),bg_color="#EEE
EEE",
    fg_color="#EEEEEEE",hover_color="grey",text_color="black",command=quit)
b3.place(x=1200,y=30)

b4=customtkinter.CTkButton(cstartup,text=("Get started
→"),font=("ROBOT",20),fg_color="yellow",
    text_color="black",hover_color="grey",height=40,width=250,
    command=lambda:[fstartup.place_forget(),flogin.place(x=0,y=0,height=1080,width=1920)])
b4.place(x=650,y=500)

img5=PhotoImage(file="line.png")

l5=Label(cstartup,image=img5,background="black")
l5.place(x=550,y=660)

l6=Label(cstartup,image=img5,background="black")

```

```
l6.place(x=1050,y=660)
```

```
l7=Label(cstartup,text="Voice  
Translation",background="black",fg="white",font=("verdana",15,"italic"))  
l7.place(x=195,y=750)
```

```
l8=Label(cstartup,text="Text  
Translation",background="black",fg="white",font=("verdana",15,"italic"))  
l8.place(x=695,y=750)
```

```
l9=Label(cstartup,text="Sign  
translation",background="black",fg="white",font=("verdana",15,"italic"))  
l9.place(x=1195,y=750)
```

```
##### Login page #####
```

```
def userlogin():
```

```
mydb=mysql.connector.connect(host="localhost",user="root",passwd="root",database="signon",a  
utocommit=True)
```

```
mycur=mydb.cursor()
```

```
if e1.get()==" or e2.get()==":
```

```
    messagebox.showerror("Error","all field required")
```

```
url3=("select * from register;")
```

```
mycur.execute(url3)
```

```
datas=mycur.fetchall()
```

```
flag1=0
```

```
flag2=0
```

```
if e1.get():
```

```

for i in range(len(datas)):
    if datas[int(i)][0]==e1.get():
        flag1=1
        break

if flag1==0:
    messagebox.showerror("Error","user Doesnt exists !!!")
if e2.get() and flag1:
    url4=("select pass from register where uname=%s;")
    #print(logentry0.get())
    pass2=[e1.get()]
    #print(pass2)
    mycur.execute(url4,pass2)
    uname1=mycur.fetchone()
    #print(uname1)
    #print(datas[1][1])
    if uname1[0]==e2.get():
        flag2=1

if flag2==0:
    messagebox.showerror("Error","Invalid password !!!")
else:
    login_reset()
    flogin.place_forget(),fmain.place(x=0,y=0,height=1080,width=1920)

clogin=Canvas(flogin,height=1080,width=1920,relief="ridge",highlightthickness=0,background="
white")
clogin.place(x=0,y=0)

def login_reset():
    for widget in clogin.winfo_children():
        if isinstance(widget,tk.Entry):
            widget.delete(0,'end')

```

```

b5=customtkinter.CTkButton(clogin,text="Back",font=("Times",25,"underline"),
    width=50,hover_color="grey",fg_color="White",text_color="black",

command=lambda:[login_reset(),flogin.place_forget(),fstartup.place(x=0,y=0,height=1080,width=
1920)])
b5.place(x=80,y=100)

l11=Label(clogin,text="Welcome
Back",font=("harrington",60,"bold"),background="white",fg="black")
l11.place(x=200,y=150)

l12=Label(clogin,text="Please      enter      your      details",font=("footlight      mt
light",25),background="white",fg="grey")
l12.place(x=300,y=250)

l45=Label(clogin,text="Username : ",font=("times",22),bg="white",fg="#555555")
l45.place(x=120,y=345)

l46=Label(clogin,text="Password : ",font=("times",22),bg="white",fg="#555555")
l46.place(x=120,y=445)

e1=Entry(clogin,bg="white",fg="grey",font=("Bahnschrift",20),width=25)
e1.place(x=270,y=350)

e2=Entry(clogin,bg="white",fg="grey",font=("Bahnschrift",20),show="*",width=25)
e2.place(x=270,y=450)

img22=PhotoImage(file="eye1.png")

img23=PhotoImage(file="eye2.png")

```

```
b20=Button(clogin,image=img22,borderwidth=0,background="white")
```

```
def view3(e):
```

```
    e2.configure(show="")
```

```
    b19.place_forget()
```

```
    b20.place(x=660,y=450)
```

```
def view4(e):
```

```
    e2.configure(show="*")
```

```
    b20.place_forget()
```

```
    b19.place(x=660,y=450)
```

```
b19=Button(clogin,image=img23,borderwidth=0,background="white")
```

```
b19.place(x=660,y=450)
```

```
b19.bind('<Button-1>',view3)
```

```
b20.bind('<Button-1>',view4)
```

```
b6=customtkinter.CTkButton(clogin,text="Login",font=("informal
```

roman

```
regular",30),fg_color="black",
```

```
    text_color="white",corner_radius=5,width=200,
```

```
    command=userlogin)
```

```
b6.place(x=370,y=550)
```

```
l13=Label(clogin,text="Don't have an account ? ",font=("footlight mt  
light",18),background="white",fg="grey")
```

```
l13.place(x=310,y=620)
```

```
img7=PhotoImage(file="under.png")
```

```
l14=Label(clogin,image=img7,background="white")
```

```
l14.place(x=563,y=615)
```

*S.N.M.I.M.T. , Maliankara*

*Department of CSE*



```

def clearsSignup():
    flogin.place_forget(),fsignup.place(x=0,y=0,height=1080,width=1920)
    e1.delete(0,END)

b7=customtkinter.CTkButton(clogin,text="Signup",font=("informal
regular",20),fg_color="white",
    text_color="black",border_width=0,width=45,hover_color="grey",
    command=lambda:[login_reset(),clearsSignup()])
b7.place(x=567,y=622)

vdo1=Label(clogin,borderwidth=0,border=0)
vdo1.place(x=750,y=100)

player=tkvideo("loginvdeo1.mp4",vdo1,loop=1,size=(800,600))

player.play()

##### signup page #####

def register():

mydb=mysql.connector.connect(host="localhost",user="root",passwd="root",database="signon",a
utocommit=True)

    mycur=mydb.cursor()
    if e3.get()==" or e4.get()==" or e5.get()=="":
        messagebox.showerror("Error","all field required")

    fmail=0
    funame=0
    fpass=0
    fcheck=0

```

```

if e3.get():
    url=("select * from register;")
    mycur.execute(url)
    datas=mycur.fetchall()

    for i in range(len(datas)):
        fcheck=fcheck+0
        if datas[int(i)][0]==e3.get():
            messagebox.showerror("Error","User already exist ...Please use login tab to login")
            fcheck=1
        else:
            funame=1

if e4.get():
    flag1=0
    for i in e4.get():
        if i=="@":
            flag1 =1
    if flag1==0:
        messagebox.showerror("Error","Invalid email!!!")
    else:
        fmail=1

if e5.get():
    paschk=e5.get()
    l=int(0)
    u=int(0)
    d=int(0)
    p=int(0)
    if (len(paschk) >= 8):
        for i in paschk:

```

```

        if (i.islower()):
            l+=1

        # counting uppercase alphabets
        if (i.isupper()):
            u+=1

        # counting digits
        if (i.isdigit()):
            d+=1

        # counting the mentioned special characters
        if(i=='!' or i=='@' or i=='#' or i=='$' or i=='%' or i=='^' or i=='&' or i=='*' or i=='(' or
i==')' or i=='_' or i=='-' or i=='+' or i=='=' or i=='[' or i==']' or i=='{' or i=='}' or i==';' or i==':' or
i=='"' or i=='/' or i=='\' or i=='?' or i=='>' or i=='<' or i=='.' or i==',' or i=='|'):
            p+=1
        else:
            messagebox.showerror("Error","Password should have minimum 8 characters !!!")

        if (l>=1 and u>=1 and p>=1 and d>=1 and l+p+u+d==len(paschk)):
            fpass=1
        else:
            messagebox.showerror("Error","\t"+"Invalid password format!!!"+"\n"+
            "Password should contain atleast one uppercase letter, lowercase letter,numeric
character and special symbol")

        if funame==1 and fmail==1 and fpass==1 and fcheck==0:
            val=(e3.get(),e4.get(),e5.get())
            url1=("insert into register value(%s,%s,%s)")
            mycur.execute(url1,val)
            messagebox.showinfo("success","Account has been Created Successfully!!!")

```

```

        signup_reset()

csignup=Canvas(fsignup,height=1090,width=1920,background="white",relief="ridge")
csignup.place(x=0,y=0)

def signup_reset():
    for widget in csignup.winfo_children():
        if isinstance(widget,tk.Entry):
            widget.delete(0,'end')

b8=customtkinter.CTkButton(csignup,text="Back",font=("Times",25,"underline"),
        width=50,hover_color="grey",fg_color="White",text_color="black",

command=lambda:[signup_reset(),fsignup.place_forget(),fstartup.place(x=0,y=0,height=1080,width=1920)])
b8.place(x=1380,y=80)

l15=Label(csignup,text="Create an account",font=("harrington",60,"bold"),background="white",fg="black")
l15.place(x=800,y=150)

l16=Label(csignup,text="Let's get started with us",font=("footlight mt light",25),background="white",fg="grey")
l16.place(x=970,y=250)

l47=Label(csignup,text="Username :",font=("Times",20),bg="white",fg="#555555")
l47.place(x=800,y=325)

l47=Label(csignup,text="Mail id :",font=("Times",20),bg="white",fg="#555555")
l47.place(x=800,y=395)

l47=Label(csignup,text="Password :",font=("Times",20),bg="white",fg="#555555")

```

```
l47.place(x=800,y=465)
```

```
e3=Entry(csignup,bg="white",fg="black",font=("Times",20),width=30)
e3.place(x=935,y=330)
```

```
e4=Entry(csignup,bg="white",fg="black",font=("Times",20),width=30)
e4.place(x=935,y=400)
```

```
e5=Entry(csignup,bg="white",fg="black",font=("Times",20),width=30,show="*")
e5.place(x=935,y=470)
```

```
img20=PhotoImage(file="eye1.png")
```

```
img21=PhotoImage(file="eye2.png")
```

```
b18=Button(csignup,image=img20,borderwidth=0,background="white")
```

```
def view1(e):
    e5.configure(show="")
    b17.place_forget()
    b18.place(x=1375,y=470)
```

```
def view2(e):
    e5.configure(show="*")
    b18.place_forget()
    b17.place(x=1375,y=470)
```

```
b17=Button(csignup,image=img21,borderwidth=0,background="white")
b17.place(x=1375,y=470)
```

```
b17.bind('<Button-1>',view1)
```

```
b18.bind('<Button-1>',view2)
S.N.M.I.M.T. , Maliankara
```

```

b9=customtkinter.CTkButton(csignup,text="Create Account",font=("informal roman
regular",30),fg_color="black",
    text_color="white",corner_radius=5,width=250,
    command=register)
b9.place(x=1030,y=550)

```

```

l17=Label(csignup,text="Already have an account ? ",font=("footlight mt
light",18),background="white",fg="grey")
l17.place(x=990,y=620)

```

```

img8=PhotoImage(file="under.png")

```

```

l18=Label(csignup,image=img8,background="white")
l18.place(x=1255,y=615)

```

```

b10=customtkinter.CTkButton(csignup,text="Login",font=("informal roman
regular",20),fg_color="white",
    text_color="black",border_width=0,width=45,hover_color="grey",

```

```

command=lambda:[signup_reset(),fsignup.place_forget(),flogin.place(x=0,y=0,height=1080,width
=1920)])

```

```

b10.place(x=1265,y=622)

```

```

img9=PhotoImage(file="signup4.png")

```

```

l20=Label(csignup,image=img9,background="white")
l20.place(x=-2,y=-2)

```

```

##### main page #####

```

```

cmain=Canvas(fmain,height=1080,width=1920,background="white",relief="ridge")

```

```
cmain.place(x=0,y=0)
```

```
def askokcancel():
```

```
    logoutanswer=messagebox.askyesno("WARNING","ARE YOU SURE YOU WANT TO LOG  
OUT ?")
```

```
    if(logoutanswer==True):
```

```
        fmain.place_forget()
```

```
        fstartup.place(x=0,y=0,height=1080,width=1920)
```

```
    elif(logoutanswer==False):
```

```
        return
```

```
b11=customtkinter.CTkButton(cmain,text="Logout",font=("Times",25,"underline"),border_width  
=2,
```

```
width=120,hover_color="grey",fg_color="white",text_color="black",border_color="white",corner  
_radius=20,
```

```
    command=askokcancel)
```

```
b11.place(x=300,y=80)
```

```
l19=Label(cmain,text=" ",height=1080,width=40,background="#222222")
```

```
l19.place(x=0,y=0)
```

```
b12=customtkinter.CTkButton(cmain,text="Help",font=("times",20,"underline"),bg_color="#2222  
22",
```

```
    fg_color="#222222",hover_color="black",text_color="white",command=clicked)
```

```
b12.place(x=80,y=700)
```

```
b13=customtkinter.CTkButton(cmain,text="About",font=("times",20,"underline"),bg_color="#222  
222",
```

```
    fg_color="#222222",hover_color="black",text_color="white",command=clicked)
```

```
b13.place(x=80,y=750)
```

```
l21=Label(fmain,text="\nGood communication \nis the bridge between \nconfusion and  
S.N.M.I.M.T. , Maliankara
```

Department of CSE

```

clarity.\n\n— Nat Turner",
    font=("Times",20,"italic"),fg="white",background="#222222")
l21.place(x=8,y=350)

img10=PhotoImage(file="logo.png")

l22=Label(fmain,image=img10,background="#222222")
l22.place(x=80,y=60)

img11=PhotoImage(file="signgirl.png")

l23=Label(fmain,image=img11,borderwidth=0)
l23.place(x=350,y=200)

img17=PhotoImage(file="voicetran.png")

l29=Label(fmain,image=img17,borderwidth=0)

def micenter(e):
    l29.place(x=498,y=75)
    b14.configure(bg="grey")

def micleave(e):
    l29.place_forget()
    b14.configure(bg="#e5e5e5")

img14=PhotoImage(file="mic1.png")

b14=Button(fmain,image=img14,relief="flat",bg="#e5e5e5",
    command=lambda:[fmain.place_forget(),fv_t.place(x=0,y=0,height=1080,width=1920)])
b14.place(x=585,y=370)

b14.bind("<Enter>",micenter)

```



```
b14.bind("<Leave>",micleave)
```

```
l26=Label(fmain,text="Voice\nTranslation",bg="#e5e5e5",font=("times",14))
```

```
l26.place(x=570,y=426)
```

```
img12=PhotoImage(file="signboy.png")
```

```
l24=Label(fmain,image=img12,borderwidth=0)
```

```
l24.place(x=750,y=210)
```

```
img18=PhotoImage(file="texttran.png")
```

```
l30=Label(fmain,image=img18,borderwidth=0)
```

```
def textenter(e):
```

```
    l30.place(x=770,y=15)
```

```
    b15.configure(bg="grey")
```

```
def textleave(e):
```

```
    l30.place_forget()
```

```
    b15.configure(bg="#d8d8d8")
```

```
img15=PhotoImage(file="text1.png")
```

```
b15=Button(fmain,image=img15,relief="flat",bg="#d8d8d8",
```

```
    command=lambda:[fmain.place_forget(),ft_t.place(x=0,y=0,height=1080,width=1920)])
```

```
b15.place(x=975,y=335)
```

```
b15.bind("<Enter>",textenter)
```

```
b15.bind("<Leave>",textleave)
```

```
l27=Label(fmain,text="Text\nTranslation",bg="#d8d8d8",font=("times",14))
```

```
l27.place(x=955,y=400)
```

```
img13=PhotoImage(file="signgirl4.png")
```

```
l25=Label(fmain,image=img13,borderwidth=0)
```

```
l25.place(x=1150,y=205)
```

```
img19=PhotoImage(file="handtran.png")
```

```
l31=Label(fmain,image=img19,borderwidth=0)
```

```
def handenter(e):
```

```
    l31.place(x=1100,y=15)
```

```
    b16.configure(bg="grey")
```

```
def handleave(e):
```

```
    l31.place_forget()
```

```
    b16.configure(bg="#dfdfff")
```

```
img16=PhotoImage(file="hand1.png")
```

```
b16=Button(fmain,image=img16,relief="flat",bg="#dfdfff",
```

```
    command=lambda:[fmain.place_forget(),fhand.place(x=0,y=0,height=1080,width=1920)])
```

```
b16.place(x=1275,y=390)
```

```
b16.bind("<Enter>",handenter)
```

```
b16.bind("<Leave>",handleave)
```

```
l28=Label(fmain,text="Sign Language\nTranslation",bg="#dfdfff",font=("times",16))
```

```
l28.place(x=1245,y=460)
```

```
##### voice to text #####
```

```
cv_t=Canvas(fv_t,height=1080,width=1920,background="white",relief="ridge")
```

```

cv_t.place(x=0,y=0)

def reset_vt():
    for widget in fv_t.winfo_children():
        if isinstance(widget,tk.Text):
            widget.delete('1.0','end')

b21=customtkinter.CTkButton(cv_t,text="Home",font=("Times",25,"underline"),border_width=2,

width=120,hover_color="grey",fg_color="white",text_color="black",border_color="white",corner
_radius=20,
    command=lambda:[fv_t.place_forget(),fmain.place(x=0,y=0,height=1080,width=1920)])
b21.place(x=300,y=80)

l32=Label(cv_t,text=" ",height=1080,width=40,background="#222222")
l32.place(x=0,y=0)

b22=customtkinter.CTkButton(cv_t,text="Help",font=("times",20,"underline"),bg_color="#22222
2",
    fg_color="#222222",hover_color="black",text_color="white",command=clicked)
b22.place(x=80,y=700)

b23=customtkinter.CTkButton(cv_t,text="About",font=("times",20,"underline"),bg_color="#2222
22",
    fg_color="#222222",hover_color="black",text_color="white",command=clicked)
b23.place(x=80,y=750)

l33=Label(cv_t,text="\nThe most important \nthing in communication \nis to hear what\nisn't
being said.\n\n— Peter Drucker",
    font=("Times",19,"italic"),fg="white",background="#222222")
l33.place(x=15,y=350)

```

```
l34=Label(fv_t,image=img10,background="#222222")
l34.place(x=80,y=60)
```

```
lang_list = ('afrikaans','albanian','amharic','arabic',
             'armenian','azerbaijani','basque','belarusian',
             'bengali','bosnian','bulgarian','catalan','cebuano',
             'chichewa','chinese (simplified)','chinese (traditional)',
             'corsican','croatian','czech','danish','dutch',
             'english','esperanto','estonian','filipino','finnish',
             'french','frisian','galician','georgian','german',
             'greek','gujarati','haitian creole','hausa',
             'hawaiian','hebrew','hindi','hmong','hungarian',
             'icelandic','igbo','indonesian',
             'irish','italian','japanese','javanese',
             'kannada','kazakh','khmer','korean','kurdish (kurmanji)',
             'kyrgyz','lao','latvian','lithuanian','luxembourgish',
             'macedonian','malagasy','malay','malayalam','maltese',
             'maori','marathi','mongolian','myanmar (burmese)',
             'nepali','norwegian','odia','pashto','persian',
             'polish','portuguese','punjabi','romanian','russian',
             'samoan','scots gaelic','serbian','sesotho',
             'shona','sindhi','sinhala','slovak','slovenian',
             'somali','spanish','sundanese','swahili','swedish',
             'tajik','tamil','telugu','thai','turkish',
             'ukrainian','urdu','uyghur','uzbek','vietnamese','welsh',
             'xhosa','yiddish','yoruba','zulu')
```

```
lang_code = ('afrikaans', 'af', 'albanian', 'sq',
             'amharic', 'am', 'arabic', 'ar',
             'armenian', 'hy', 'azerbaijani', 'az',
             'basque', 'eu', 'belarusian', 'be',
             'bengali', 'bn', 'bosnian', 'bs', 'bulgarian',
```

'bg', 'catalan', 'ca', 'cebuano',  
'ceb', 'chichewa', 'ny', 'chinese (simplified)',  
'zh-cn', 'chinese (traditional)',  
'zh-tw', 'corsican', 'co', 'croatian', 'hr',  
'czech', 'cs', 'danish', 'da', 'dutch',  
'nl', 'english', 'en', 'esperanto', 'eo',  
'estonian', 'et', 'filipino', 'tl', 'finnish',  
'fi', 'french', 'fr', 'frisian', 'fy', 'galician',  
'gl', 'georgian', 'ka', 'german',  
'de', 'greek', 'el', 'gujarati', 'gu',  
'haitian creole', 'ht', 'hausa', 'ha',  
'hawaiian', 'haw', 'hebrew', 'he', 'hindi',  
'hi', 'hmong', 'hmn', 'hungarian',  
'hu', 'icelandic', 'is', 'igbo', 'ig', 'indonesian',  
'id', 'irish', 'ga', 'italian',  
'it', 'japanese', 'ja', 'javanese', 'jw',  
'kannada', 'kn', 'kazakh', 'kk', 'khmer',  
'km', 'korean', 'ko', 'kurdish (kurmanji)',  
'ku', 'kyrgyz', 'ky', 'lao', 'lo',  
'latin', 'la', 'latvian', 'lv', 'lithuanian',  
'lt', 'luxembourgish', 'lb',  
'macedonian', 'mk', 'malagasy', 'mg', 'malay',  
'ms', 'malayalam', 'ml', 'maltese',  
'mt', 'maori', 'mi', 'marathi', 'mr', 'mongolian',  
'mn', 'myanmar (burmese)', 'my',  
'nepali', 'ne', 'norwegian', 'no', 'odia', 'or',  
'pashto', 'ps', 'persian', 'fa',  
'polish', 'pl', 'portuguese', 'pt', 'punjabi',  
'pa', 'romanian', 'ro', 'russian',  
'ru', 'samoan', 'sm', 'scots gaelic', 'gd',  
'serbian', 'sr', 'sesotho', 'st',  
'shona', 'sn', 'sindhi', 'sd', 'sinhala', 'si',  
'slovak', 'sk', 'slovenian', 'sl',

'somali', 'so', 'spanish', 'es', 'sundanese',  
'su', 'swahili', 'sw', 'swedish',  
'sv', 'tajik', 'tg', 'tamil', 'ta', 'telugu',  
'te', 'thai', 'th', 'turkish',  
'tr', 'ukrainian', 'uk', 'urdu', 'ur', 'uyghur',  
'ug', 'uzbek', 'uz',  
'vietnamese', 'vi', 'welsh', 'cy', 'xhosa', 'xh',  
'yiddish', 'yi', 'yoruba',  
'yo', 'zulu', 'zu')

```
combo5=customtkinter.CTkComboBox(cv_t,values=lang_list,width=250)
combo5.place(x=300,y=200)
combo5.set("english")
```

```
combo6=customtkinter.CTkComboBox(cv_t,values=lang_list,width=250)
combo6.place(x=300,y=300)
combo6.set("english")
```

```
t4=Text(fv_t,font="Robot 16",bg="light blue",relief="groove",wrap="word",fg="black")
t4.place(x=585,y=555,height=192,width=605)
```

```
scrollbar4=Scrollbar(t4,width=20,cursor="hand2")
scrollbar4.pack(side="right",fill="y")
```

```
scrollbar4.configure(command=t4.yview)
t4.configure(yscrollcommand=scrollbar4.set)
```

```
b39=customtkinter.CTkButton(cv_t,text="start recording",command=clicked)
b39.place(x=300,y=400)
```

```
b40=customtkinter.CTkButton(cv_t,text="Reset",font=("Robot",20),
```

```
fg_color="#28869D",text_color="white",hover_color="#FFCCCB",bg_color="white",
S.N.M.I.M.T. , Maliankara
```

Department of CSE

```

border_width=0,height=45,command=reset_vt)
b40.place(x=880,y=400)

l62=Label(cv_t,text="Want to translate into speech ? click here",bg="white",fg="black",
font=("Times",20,"italic"))
l62.place(x=620,y=790)

b42=Button(cv_t,image=img14,bg="white",relief="flat",

command=lambda:[reset_vt(),fv_t.place_forget(),fv_v.place(x=0,y=0,width=1920,height=1920)])
b42.place(x=1100,y=770)

##### voice to voice

cv_v=Canvas(fv_v,height=1080,width=1920,background="white",relief="ridge")
cv_v.place(x=0,y=0)

b41=customtkinter.CTkButton(cv_v,text="Home",font=("Times",25,"underline"),border_width=2,

width=120,hover_color="grey",fg_color="white",text_color="black",border_color="white",corner
_radius=20,

command=lambda:[fv_v.place_forget(),fmain.place(x=0,y=0,height=1080,width=1920)])
b41.place(x=300,y=80)

##### text to text #####

ct_t=Canvas(ft_t,height=1080,width=1920,background="white",relief="ridge",bg="white")
ct_t.place(x=-1,y=-5)

# vdo2=Label(ct_t,borderwidth=0,border=0,bg="white")
# vdo2.place(x=-28,y=-28)

```

```

# player1=tkvideo("ttbgvideo.mp4",vdo2,loop=1,size=(1820,980))

# player1.play()

img29=PhotoImage(file="ttbgpic.png")

l64=Label(ct_t,image=img29,bg="white",border=0)
l64.place(x=-28,y=-28)

def reset_tt():
    for widget in ft_t.winfo_children():
        if isinstance(widget,tk.Text):
            widget.delete('1.0','end')

b24=customtkinter.CTkButton(ct_t,text="Home",font=("Times",25,"underline"),border_width=2,
width=120,hover_color="grey",fg_color="white",text_color="black",border_color="white",corner
_radius=20,

command=lambda:[reset_tt(),ft_t.place_forget(),fmain.place(x=0,y=0,height=1080,width=1920)])
b24.place(x=300,y=80)

l35=Label(ct_t,text=" ",height=1080,width=40,background="#222222")
l35.place(x=0,y=0)

b25=customtkinter.CTkButton(ct_t,text="Help",font=("times",20,"underline"),bg_color="#222222",
",
fg_color="#222222",hover_color="black",text_color="white",command=clicked)
b25.place(x=80,y=700)

b26=customtkinter.CTkButton(ct_t,text="About",font=("times",20,"underline"),bg_color="#2222
22",
fg_color="#222222",hover_color="black",text_color="white",command=clicked)

```



```
b26.place(x=80,y=750)
```

```
l36=Label(ct_t,
```

```
    text="\nYou never know \nwhen a moment \nand a few sincere words \ncan have an \nimpact  
on a life"\n\n— Zig Ziglar",
```

```
    font=("Times",19,"italic"),fg="white",background="#222222")
```

```
l36.place(x=10,y=350)
```

```
l37=Label(ft_t,image=img10,background="#222222")
```

```
l37.place(x=80,y=60)
```

```
language=googletrans.LANGUAGES
```

```
languageV=list(language.values())
```

```
lang1=language.keys()
```

```
l43=Label(ft_t,text="Select your input language : ",font="Times 20",fg="black",bg="white")
```

```
l43.place(x=650,y=100)
```

```
combo1=customtkinter.CTkComboBox(ft_t,values=languageV,font=("Robot",15),
```

```
    button_color="#28869D",button_hover_color="cyan",bg_color="white",text_color="black",  
    width=250)
```

```
combo1.place(x=980,y=105)
```

```
combo1.set("English")
```

```
t1=Text(ft_t,font="Robot 16",bg="light blue",relief="groove",wrap="word",fg="black")
```

```
t1.place(x=585,y=155,height=195,width=605)
```

```
scrollbar1=Scrollbar(t1,width=20,cursor="hand2")
```

```
scrollbar1.pack(side="right",fill="y")
```

```
scrollbar1.configure(command=t1.yview)
```

```
t1.configure(yscrollcommand=scrollbar1.set)
```

```
l44=Label(ft_t,text="Select your output language : ",font="Times 20",fg="black",bg="white")
l44.place(x=550,y=465)
```

```
combo2=customtkinter.CTkComboBox(ft_t,values=languageV,font=("Robot",15),
    button_color="#28869D",button_hover_color="cyan",bg_color="white",text_color="black",
    width=250)
combo2.place(x=880,y=470)
combo2.set("Select Language")
```

```
t2=Text(ft_t,font="Robot 16",bg="light blue",relief="groove",wrap="word",fg="black")
t2.place(x=585,y=555,height=192,width=605)
```

```
scrollbar2=Scrollbar(t2,width=20,cursor="hand2")
scrollbar2.pack(side="right",fill="y")
```

```
scrollbar2.configure(command=t2.yview)
t2.configure(yscrollcommand=scrollbar2.set)
```

```
b27=customtkinter.CTkButton(ft_t,text="Translate",font=("Robot",20),
    fg_color="#28869D",text_color="white",hover_color="light blue",bg_color="white",
    border_width=0,height=45,command=clicked,corner_radius=5)
b27.place(x=720,y=400)
```

```
b28=customtkinter.CTkButton(ft_t,text="Reset",font=("Robot",20),
    fg_color="#28869D",text_color="white",hover_color="#FFCCCB",bg_color="white",
    border_width=0,height=45,command=reset_tt)
b28.place(x=880,y=400)
```

```
l38=Label(ft_t,text="Want to translate into speech ? click here",bg="white",fg="black",
    font=("Times",20,"italic"))
```

```
l38.place(x=620,y=790)
```

```
b29=Button(ft_t,image=img14,bg="white",relief="flat",
           command=lambda:[reset_tt(),ft_t.place_forget(),ft_v.place(x=0,y=0,width=1920,height=1920)])
b29.place(x=1100,y=770)
```

```
##### text to voice #####
```

```
ct_v=Canvas(ft_v,height=1080,width=1920,background="white",relief="ridge")
ct_v.place(x=0,y=0)
```

```
# vdo3=Label(ct_v,borderwidth=0,border=0,bg="white")
# vdo3.place(x=-50,y=-70)
```

```
# player2=tkvideo("tvbgvideo.mp4",vdo3,loop=1,size=(1820,980))
```

```
# player2.play()
```

```
img28=PhotoImage(file="tvbgpic.png")
```

```
l63=Label(ct_v,image=img28,bg="white",border=0)
l63.place(x=-50,y=-70)
```

```
def reset_tv():
    for widget in ft_v.winfo_children():
        if isinstance(widget,tk.Text):
            widget.delete('1.0','end')
```

```
b30=customtkinter.CTkButton(ct_v,text="Home",font=("Times",25,"underline"),border_width=2,
```

```
width=120,hover_color="grey",fg_color="white",text_color="black",border_color="white",corner
S.N.M.I.M.T. , Maliankara
```

Department of CSE

```

_radius=20,

command=lambda:[reset_tv(),ft_v.place_forget(),fmain.place(x=0,y=0,height=1080,width=1920)])
b30.place(x=300,y=80)

l49=Label(ct_v,text=" ",height=1080,width=40,background="#222222")
l49.place(x=0,y=0)

b31=customtkinter.CTkButton(ct_v,text="Help",font=("times",20,"underline"),bg_color="#222222",
fg_color="#222222",hover_color="black",text_color="white",command=clicked)
b31.place(x=80,y=700)

b32=customtkinter.CTkButton(ct_v,text="About",font=("times",20,"underline"),bg_color="#222222",
fg_color="#222222",hover_color="black",text_color="white",command=clicked)
b32.place(x=80,y=750)

l50=Label(ct_v,text="\nCommunication leads \ninto community, that is,\ninto
understanding,\nintemacy and \nmatural valuing.\n\n— Rollo May",
font=("Times",19,"italic"),fg="white",background="#222222")
l50.place(x=20,y=350)

l51=Label(ft_v,image=img10,background="#222222")
l51.place(x=80,y=60)

l55=Label(ct_v,text="Select your input language : ",font="Times 19",fg="black",bg="white")
l55.place(x=580,y=95)

combo3=customtkinter.CTkComboBox(ct_v,values=lang_list,font=("Robot",15),
button_color="#28869D",button_hover_color="cyan",bg_color="white",text_color="black",
width=250)

```

```
combo3.place(x=880,y=100)
```

```
combo3.set("english")
```

```
t3=Text(ft_v,font="Robot 16",bg="light blue",relief="groove",wrap="word")
```

```
t3.place(x=540,y=143,height=200,width=660)
```

```
scrollbar3=Scrollbar(t3,width=20,cursor="hand2")
```

```
scrollbar3.pack(side="right",fill="y")
```

```
scrollbar3.configure(command=t3.yview)
```

```
t3.configure(yscrollcommand=scrollbar3.set)
```

```
l57=Label(ct_v,text="Select your output language : ",font="Times 19",fg="black",bg="white")
```

```
l57.place(x=580,y=495)
```

```
combo4=customtkinter.CTkComboBox(ct_v,values=lang_list,font=("Robot",15),
```

```
    button_color="#28869D",button_hover_color="cyan",bg_color="white",text_color="black",  
    width=250)
```

```
combo4.place(x=880,y=500)
```

```
combo4.set("english")
```

```
b36=customtkinter.CTkButton(ct_v,text="Reset",font=("Robot",20),
```

```
    fg_color="#28869D",text_color="white",hover_color="red",
```

```
    border_width=0,height=45,command=reset_tv)
```

```
b36.place(x=800,y=410)
```

```
l59=Label(ct_v,text="Want to translate into text ? click here",bg="white",
```

```
    font=("Times",19,"italic"))
```

```
l59.place(x=620,y=780)
```

```
# l60=Label(ct_v,text="Click here to play the translated voice",bg="white",
```

```
#    font=("Times",20,"italic"))
```

```
# l60.place(x=620,y=660)
```

```
img26=PhotoImage(file="playbutton.png")
```

```
b38=Button(ct_v,image=img26,bg="white",relief="flat",command=clicked)
```

```
b38.place(x=950,y=600)
```

```
b37=Button(ct_v,image=img15,bg="white",relief="flat",
```

```
    command=lambda:[reset_tv(),ft_v.place_forget(),ft_t.place(x=0,y=0,width=1920,height=1920)])
```

```
b37.place(x=1040,y=770)
```

```
##### hand #####
```

```
chand=Canvas(fhand,height=1080,width=1920,background="white",relief="ridge")
```

```
chand.place(x=0,y=0)
```

```
b33=customtkinter.CTkButton(chand,text="Home",font=("Times",25,"underline"),border_width=2,
```

```
width=120,hover_color="grey",fg_color="white",text_color="black",border_color="white",corner_radius=20,
```

```
    command=lambda:[fhand.place_forget(),fmain.place(x=0,y=0,height=1080,width=1920)])
```

```
b33.place(x=300,y=80)
```

```
l52=Label(chand,text=" ",height=1080,width=40,background="#222222")
```

```
l52.place(x=0,y=0)
```

```
b34=customtkinter.CTkButton(chand,text="Help",font=("times",20,"underline"),bg_color="#222222",
```

```

        fg_color="#222222",hover_color="black",text_color="white",command=clicked)
b34.place(x=80,y=700)

b35=customtkinter.CTkButton(chand,text="About",font=("times",20,"underline"),bg_color="#222
222",
        fg_color="#222222",hover_color="black",text_color="white",command=clicked)
b35.place(x=80,y=750)

l53=Label(chand,text="\n\"The ability to \nsimplify means to \neliminate the \nunecessary so
that\nthe necessary may\nspeak.\\\"\\n\\n— Hans Hofmann",
        font=("Times",19,"italic"),fg="white",background="#222222")
l53.place(x=38,y=350)

l54=Label(chand,image=img10,background="#222222")
l54.place(x=80,y=60)

fstartup.place(x=0,y=0,height=1080,width=1920)

#test_net()

root.mainloop()

```

## REFERENCE

- [1] T. Shanableh and K. Assaleh, “Telescopic vector composition and polar accumulated motion residuals for feature extraction in Arabic Sign Language recognition,” *EURASIP J. Image Video Process.*, vol. 2007, p. 10, 2007, Art. no. 87929, doi: 10.1155/2007/87929.
- [2] G. Latif, N. Mohammad, J. Alghazo, R. AlKhalaf, and R. AlKhalaf, “ArASL: Arabic alphabets sign language dataset,” *Data Brief*, vol. 23, Apr. 2019, Art. no. 103777, doi: 10.1016/j.dib.2019.103777.
- [3] S. M. Shohieb, H. K. Elminir, and A. M. Riad, “Signs World Atlas; a benchmark Arabic sign language database,” *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 27, pp. 68–76, Jan. 2015, doi: 10.1016/j.jksuci.2014.03.011.
- [4] M. Alfonse, A. Ali, A. S. Elons, N. L. Badr, and M. Aboul-Ela, “Arabic sign language benchmark database for different heterogeneous sensors,” in *Proc. 5th Int. Conf. Inf. Commun. Technol. Accessibility (ICTA)*, Dec. 2016, pp. 1–9, doi: 10.1109/ICTA.2015.7426902.
- [5] M. Mohandes, S. I. Quadri, and M. Deriche, “Arabic sign language recognition an image-based approach,” in *Proc. 21st Int. Conf. Adv. Inf. Netw. Appl. Workshops (AINAW)* May 2007, pp. 272–276, doi: 10.1109/AINAW.2007.98.
- [6] M. El Badawy, A. S. Elons, H. Sheded, and M. F. Tolba, “A proposed hybrid sensor architecture for Arabic Sign Language recognition,” in *Intelligent Systems*. Cham, Switzerland: Springer, 2015, pp. 721–730.
- [7] Z. Zafrulla, H. Brashear, T. Starner, H. Hamilton, and P. Presti, “American sign language recognition with the Kinect,” in *Proc. 13th Int. Conf. Multimodal Interfaces*, 2011, pp. 279–286, doi: 10.1145/2070481.2070532.



- [8] S. Aliyu, M. Mohandes, M. Deriche, and S. Badran, “Arabic sign language recognition using the Microsoft Kinect,” in Proc. 13th Int. MultiConf. Syst., Signals Devices (SSD), Mar. 2016, pp. 301–306, doi: 10.1109/SSD.2016.7473753.
- [9] M. Ahmed, M. Idrees, Z. Ul Abideen, R. Mumtaz, and S. Khalique, “Deaf talk using 3D animated sign language: A sign language interpreter using Microsoft’s Kinect v2,” in Proc. SAI Comput. Conf. (SAI), Jul. 2016, pp. 330–335, doi: 10.1109/SAI.2016.7556002.
- [10] F. Soares, J. S. Esteves, V. Carvalho, C. Moreira, and P. Lourenco, “Sign language learning using the hangman videogame,” in Proc. Int. Congr. Ultra Mod. Telecommun. Control Syst. Workshops, Oct. 2015, pp. 231–234, doi: 10.1109/ICUMT.2015.7382433.
- [11] M. A. Almasre and H. Al-Nuaim, “Recognizing Arabic sign language gestures using depth sensors and a KSVM classifier,” in Proc. 8<sup>th</sup> Comput. Sci. Electron. Eng. Conf., Sep. 2017, pp. 146–151, doi: 10.1109/CEEC.2016.7835904.
- [12] P. Kumar, H. Gauba, P. P. Roy, and D. P. Dogra, “A multimodal framework for sensor based sign language recognition,” *Neurocomputing*, vol. 259, pp. 21–38, Oct. 2017, doi: 10.1016/j.neucom.2016.08.132.