TABLE OF CONTENT

FINTECH FOR STOCK PRICING

ABSTRACT

Communication plays a critical role for people and is regarded as a skill in life. Having this important aspect of life and surroundings in mind, we present our project article, which focuses primarily on supporting patients with pain or silent speech. Our research work leads to improved contact with the deaf and the mute. Each sign language uses sign patterns visually conveyed to express the true meaning. The combination of hand gestures and/or motions of arm and body is called Sign Language and the Dictionary. It is the combination of hands and facial expressions. Our program project is able to understand signals in sign language. These symbols may be used to interact with hearing aids. Our article suggests a program that allows common people to interact effectively with others that are hard to understand. In this case, we are implementing the Indian Sign Language (ISL) method by using a microphone and a camera. Translation of the voice into Indian sign language system by the ISL translation system is possible. The ISL translation framework uses a microphone to get pictures (from ordinary people) or continuous video clips, which the application interprets.

LIST OF FIGURES

LIST OF ABBREVIATIONS

API	-	Application Program Interface
CSS	-	Cascading Style Sheets
GIF	-	Graphics Interchange Format
GUI	-	Graphical User Interface
HTML	-	Hypertext Markup Language
MVC	-	Model View Controller
OOAD	-	Object Oriented Analysis and Design
RAM	-	Random Access Memory
SQL	-	Structured Query Language
UX	-	User Experience

INTRODUCTION

CHAPTER 1

1. INTRODUTION

In recent times stock market predictions is gaining more attention, maybe due to the fact that if the trend of the market is successfully predicted the investors may be better guided. The profits gained by investing and trading in the stock market greatly depends on the predictability. If there is a system that can consistently predict the direction of the dynamic stock market will enable the users of the system to make informed decisions. More over the predicted trends of the market will help the regulators of the market in taking corrective measures.

1.1. AIM AND OBJECTIVE

The aim of a Speech to Sign Language Translator is to bridge the communication gap between hearing-impaired individuals who use sign language and those who communicate verbally.

1.2. SIGN LANGUAGE

Sign language is a language that consists of signs made with hands and other movements, facial expressions and postures of body, which is primarily used by people who are deaf or hard hearing peoples that they can easily express their thoughts or can easily communicate with other people. Sign language is very important a far the deaf people are concerned for their emotional, social and linguistic growth. First language for the deaf people is sign language which get proceeded bilingually with the education of national sign language as well as national written or spoken language. There are different communities of deaf people all around the world therefore the sign language for these communities will be different. The different sign languages used by different communities are: America uses American Sign Language, Britain sign language is used by Britain, similarly, India uses Indian sign language etc. For expressing thoughts and communicating with each other. Manual communication and body language is used by Indian sign language to convey thoughts, feelings and ideas. ISL is classified into two classes: manual and non-manual signs. One handed and two handed are part of manual sign where the information is being conveyed by the signer using his/ her hands to make the sign.

1.3. MOTIVATION

The motivation behind developing a Speech to Sign Language Translator is to address communication barriers between individuals who use spoken language and those who are hearing-impaired. **Barrier Reduction:** The primary motivation is to reduce communication barriers faced by the hearing-impaired community. **Real-Time Translation:** The system aims to convert spoken language (audio) into sign language (visual gestures) in real time, allowing seamless communication. The system provides a user-friendly environment, making it accessible for both speech-impaired individuals and those who interact with them. By converting audio messages into sign language, the system empowers deaf individuals to participate in various activities, access information, and communicate effectively.

1.4. OBJECTIVES

The aim of a Speech to Sign Language Translator is to bridge the communication gap between hearing-impaired individuals who use sign language and those who communicate verbally. **Communication Enhancement:** The primary objective is to develop a system that allows deaf people to communicate more effectively with others. **Real-time Translation:** The system should convert spoken language (audio) into sign language (visual gestures) in real time. **User-Friendly Interface:** The system should be user-friendly, making it accessible to both speech-impaired individuals and those who interact with them.

1.5. EXISTING SYSTEM

Current solutions: Since technology evolves at a dizzying speed, humans make smart ideas every year to help themselves and those who are disabled. We want to make it simpler for deaf people to interact with each other, so we designed a language interpreter that quickly transforms audio to sign language. For the deaf, sign language is their sole way of communicating. People who are physically disabled use sign language to express their emotions to others. It's difficult to communicate because ordinary people struggle to master the specific sign language. Because sign language comprises of a wide range of hand motions and gestures, acquiring the necessary precision at a reasonable cost has proven to be a monumental undertaking. We already have physical software and hardware that can convert audio to sign language. As a result, we're upgrading the product using the processing of natural languages. The word library may be expanded to encompass the great majority of English terms that are often used. Speech to text - to - speech and language processing may be enhanced using various NLP methods.

1.6. PROPOSED SYSTEM

Linear Regression is a linear approach to modeling the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables). The case of one explanatory variable is called simple linear regression.

Advantages

- Space complexity is very low it just needs to save the weights at the end of training. Hence, it's a high latency algorithm.
- Its very simple to understand
- Good interpretability

Feature importance is generated at the time model building. With the help of hyperparameter lamba, you can handle features selection hence we can achieve dimensionality reduction

LITERATURE SURVEY

CHAPTER 2

2. LITERATURE SURVEY

2.1. Increasing adaptability of a speech into sign language translation system

Authors: López Ludeña; San Segundo; Morcillo, López.

Year: 2018

The authors have designed a speech–sign translation system for Spanish Si Language (SSL) using a speech recognizer, a natural language translator, and a 3D avatar animation module. The automatic speech recognizer (ASR), converts natural speech into a sequence of words. It uses an HMM (Hidden Markov Model)-based system able to recognize continuous speech. It is based on Hidden Semi-Markov Models (HSMMs) in which the user can choose the voice gender (female or male), the emotion type (happy, sad, angry, surprise, and fear) and the Emotional.

2.2. Real Time Sign Language Recognition Framework

Authors: Tewari, Soni Singh, Turlapati, Bhuva

Year: 2021

In this paper, a two-way communication system is suggested in the paper but the authors are only able to convert 26 alphabets and three characters with an accuracy rate of 99.78% using CNN models. The authors only suggest that future work must be conducted in the field of natural language processing to convert speech into sign language.

2.3. Design and Implementation of an Intelligent System to translate Arabic text to Arabic sign language

Authors: Jamil

Year: 2021

In this paper, the authors summarized converting Arabic text to Arabic Sign Language by using uses various text parsing and word processing techniques. The system successfully converts Arabic text to Arabic sign language with 87% efficiency and then shows the corresponding sign language animated.

2.4. Speech-to-sign language translation system for Spanish

Authors: R. San Segundo, R. Barra, R. Córdoba, L.F. d'Haro, F. Fernández, J. Ferreiros, J.M.Lucas,J. Macías-Guarasa, J.M. Montero

Authors: R. San Segundo, R. Barra, R. Córdoba, L.F. d'Haro, F. Fernández, J. Ferreiros, J.M.Lucas,J. Macías-Guarasa, J.M. Montero

Year: 2019

The first module, the speech recognizer, converts natural speech into a sequence of words (text).The natural language translation module converts a word sequence into a sign sequence. For this module, the paper presents two proposals. The first one consists of a rulebased translation strategy, where a set of translation rules (defined by an expert) guides the translation process. The second alternative is based on a statistical translation approach where parallel corpora are used for training language and translation models.

SYSTEM SPECIFICATION

CHAPTER 3

3. SYSTEM SPECIFICATION

3.1. HARDWARE REQUIREMENT

➢ **PROCESSOR**	**:**	PENTIUM IV, 2.4 GHZ.
➢ **RAM**	**:**	8 GB
➢ **MAIN MEMORY**	**:**	8 GB RAM
➢ **PROCESSING SPEED**	**:**	600 MHZ
➢ **HARD DISK DRIVE**	**:**	1 TB
➢ **KEYBOARD**	**:**	104 KEYS

3.2. SOFTWARE REQUREMENT

➢ **FRONT END**	**:**	HTML, CSS
➢ **BACK END**	**:**	PYTHON
➢ **OPERATING SYSTEM**	**:**	WINDOWS 10

PROJECT IMPLEMENTATION

CHAPTER 4

4. PROJECT IMPLEMENTATION

4.1. MODULE DECRIPTION

The implementation of this project is divided into following steps

- Data Preprocessing
- Feature Selection
- Building and Training Model

4.1.1.DATA PREPROCESSING:

The entries are present in the dataset. The null values are removed using df = df.dropna() where df is the data frame. The categorical attributes (Date,High,Low,Close,Adj value) are converted into numeric using Label Encoder. The date attribute is splitted into new attributes like total which can be used as feature for the model.

4.1.2. FEATURE SELECTION:

It used to select the different languages given and it translates the languages in the real time and these can be easily read or understand even by common people , so it provides great transparency to everyone so nothing is hided or modified.

4.1.3. BUILDING AND TRAINING MODEL:

After feature selection location and month attribute are used for training. The dataset is divided into pair of xtrain ,ytrain and xtest, y test. The algorithms model is imported form skleran. Building model is done using model. Fit (xtrain, ytrain).

4.2. ARCHITECTURE

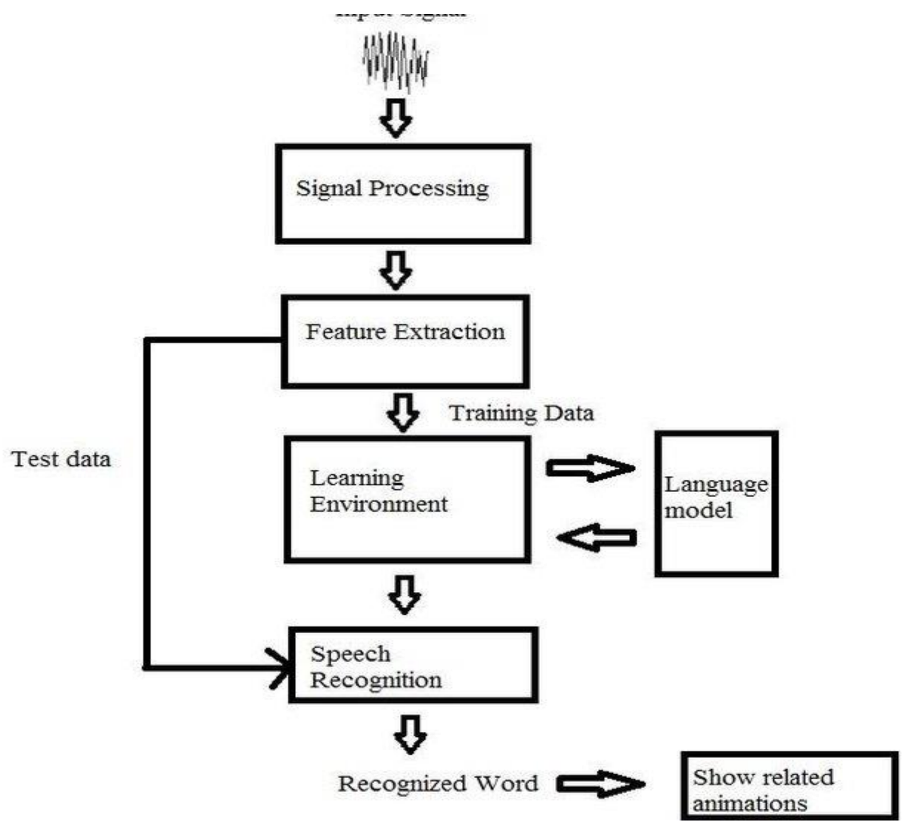


Fig 4.1 Data Flow Diagram

SYSTEM DESIGN

CHAPTER 5

5. SYSTEM DESIGN

5.1. GENERAL

System design refers to the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. It involves creating a blueprint or plan that outlines how the system will be structured, organized, and implemented. System design is a crucial phase in software development or engineering, as it lays the foundation for the development process and guides the implementation and integration of the system. It encompasses both the technical and functional aspects of a system, considering factors like scalability, reliability, performance, security, and maintainability.

5.2. SYSTEM ARCHITECTURE

A system architecture is a conceptual model that outlines a system's structure, behaviour, and additional viewpoints. An architectural description is a formal description and representation of a system that is arranged in a way that allows for reasoning about the system's structures and actions.

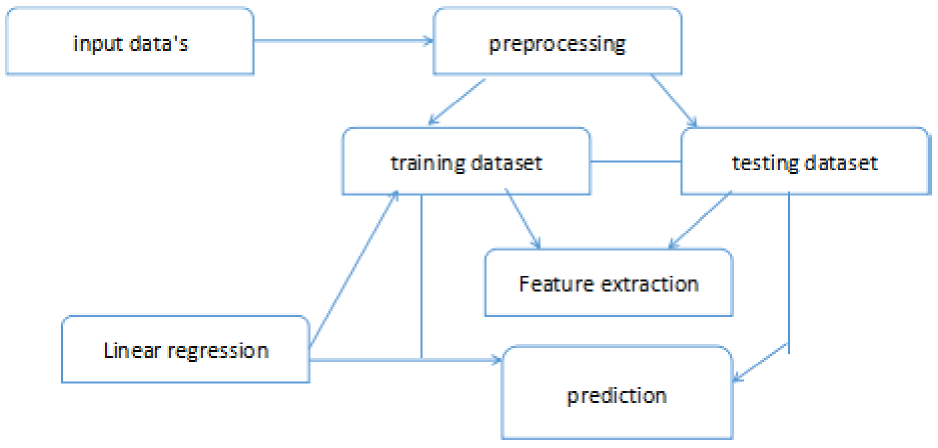


FIG 5.1 SYSTEM ARCHITECTURE

SOFTWARE SPECIFICATION

CHAPTER 6

6. SOFTWARE SPECIFICATION

6.1 GENERAL

Software Specification is a list of Software, packages, libraries required to develop a project. Software Specification contains the deep description about the project. Software specifications helps to ensure that all stakeholders, including developers, designers, testers, and clients, have a common understanding of the software's purpose and features.

6.2 PYTHON TECHNOLOGY

Python is an interpreted, object- oriented programming language similar to PERL, that has gained popularity because of its clear syntax and readability. Python is said to be relatively easy to learn and portable, meaning its statements can be interpreted in a number of operating systems, including UNIX- based systems, Mac OS, MS- DOS, OS/2, and various versions of Microsoft Windows 98. Python was created by Guido van Rossum, a former resident of the Netherlands, whose favourite comedy group at the time was Monty Python's Flying Circus. The source code is freely available and open for modification and reuse. Python has a significant number of users.

A notable feature of Python is its indenting of source statements to make the code easier to read. Python offers dynamic data type, ready- made class, and interfaces to many system calls and libraries. It can be extended, using the C or C++language.

Python can be used as the script in Microsoft's Active Server Page (ASP) technology. The scoreboard system for the Melbourne (Australia) Cricket Ground is written in Python. Z Object Publishing Environment, a popular Web application server, is also written in the Python language's

6.2.1 PYTHON PLATFORM

Apart from Windows, Linux and MacOS, CPython implementation runs on 21 different platforms. IronPython is a .NET framework based Python implementation and it is cabable of running in both Windows, Linux and in other environments where .NET framework is available.

6.2.2 PYTHON LIBRARY

Machine Learning, as the name suggests, is the science of programming a computer by which they are able to learn from different kinds of data. A more general definition given by Arthur Samuel is –“Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed.” They are typically used to solve various types of life problems.

In the older days, people used to perform Machine Learning tasks by manually coding all the algorithms and mathematical and statistical formula. This made the process time consuming, tedious and inefficient. But in the modern days, it is become very much easy and efficient compared to the olden days by various python libraries, frameworks, and modules. Today, Python is one of the most popular programming languages for this task and it has replaced many languages in the industry, one of the reason is its vast collection of libraries. Python libraries that used in Machine Learning are:

- Numpy
- Scipy
- Scikit- learn
- TensorFlow

- Speech-recognition

6.2.2.1 NumPy

NumPy is a very popular python library for large multi- dimensional array and matrix processing, with the help of a large collection of high- level mathematical functions. It is very useful for fundamental scientific computations in Machine Learning. It is particularly useful for linear algebra, Fourier transform, and random number capabilities. High- end libraries like TensorFlow uses NumPy internally for manipulation of Tensors.

6.2.2.2 SciPy

SciPy is a very popular library among Machine Learning enthusiasts as it contains different modules for optimization, linear algebra, integration and statistics. There is a difference between the SciPy library and the SciPy stack. The SciPy is one of the core packages that make up the SciPy stack. SciPy is also very useful for image manipulation.

6.2.2.3 Scikit

Scikit- learn is one of the most popular ML libraries for classical ML algorithms. It is built on top of two basic Python libraries, viz., NumPy and SciPy. Scikit- learn supports most of the supervised and unsupervised learning algorithms. Scikit- learn can also be used for data- mining and data- analysis, which makes it a great tool who is starting out with ML.

6.2.2.4 TensorFlow

TensorFlow is a very popular open- source library for high performance numerical computation developed by the Google Brain team in Google. As the name suggests, Tensorflow is a framework that involves defining and running computations involving tensors. It can train and run deep neural networks that can be used to develop several AI applications. TensorFlow is widely used in the field of deep learning research and application.

6.2.2.5 Speech-recognition

The Speech Recognition module, often referred to as SpeechRecognition, is a library that allows Python developers to convert spoken language into text by utilizing various speech recognition engines and APIs. It supports multiple services like Google Web Speech API, Microsoft Bing Voice Recognition, IBM Speech to Text, and others.

SOFTWARE TESTING

CHAPTER 7

7. SOFTWARE TESTING

7.1 GENERAL

Software testing is a crucial process in software development that involves evaluating the quality, functionality, and performance of a software system. It is performed to identify defects, bugs, or any discrepancies between the expected and actual behavior of the software. The goal of software testing is to ensure that the software meets the specified requirements, functions as intended, and provides a satisfactory user experience. Software testing encompasses various techniques and approaches to validate different aspects of the software, including its functionality, usability, reliability, performance, security, and compatibility. It involves the execution of test cases, the comparison of actual results with expected results, and the analysis of any discrepancies or failures.

7.2 TYPES OF TESTING

7.2.1 UNIT TESTING

Testing individual units or components of the software to verify their correctness and behavior in isolation. Unit testing is an essential part of ensuring the reliability and functionality of the mouse application. The purpose of unit testing is to verify the behavior and correctness of individual units or components of the application. By thoroughly testing each unit, such as the speech recognition we can identify any defects or issues early in the development process. The test plan outlines the objectives, scope, and approach for unit testing, while the test cases specify the expected outcomes for different scenarios and eye movement. During test execution, we set up the testing environment, execute the unit tests, and record the results. Test coverage is measured to determine the extent to which the code has been tested. Any failures or issues encountered during testing are documented and reported clearly. Integration testing is also considered to ensure the smooth integration of the virtual with other components. Test maintenance and regression testing are performed to keep the unit tests up to date as the application evolves. Overall, unit testing plays a crucial role in identifying and resolving issues early, contributing to the overall quality and reliability of the eye controlled mouse application.

7.2.2 FUNCTIONAL TESTING

Functional testing for a speech recognition involves testing the system as a whole to ensure it functions correctly and meets the specified requirements. It focuses on validating the functionality and behavior of the eye controlled mouse in real-world scenarios.

1. SPEECH RECOGNITION FUNCTIONALITY:

   Speech-recognition is verified as they get the input through audio and convert them into text.

2. SPEECH TO SIGN CONVERSION:

   Based on the pretrained data, the speech gets converted into sign languages ad each letter is recognized for their sign language

3. USER INTERFACE TESTING:

   Tested the user interface elements and interactions. Verified that the graphical user interface displays the voice stream correctly, overlays over the speech, and responds appropriately to user inputs and settings adjustments.

7.2.3 SYSTEM TESTING

System testing is a critical phase in the software testing process that focuses on evaluating the behavior and performance of a complete software system. It involves testing the integrated components of the system as a whole to ensure that it meets the specified requirements and functions correctly in a real-world environment. The goal of system testing is to verify the system's compliance with user expectations, assess its functionality, and identify any defects or issues before its deployment. Set up the hardware and software components of the mouse system. This may involve connecting a mic or sensor to your computer, installing speech-recognition libraries, and configuring your development environment. This involves creating a user interface that displays sign language on your screen and programming the speech-recognition to respond to your voice. Tested the accuracy and responsiveness of the sign language by giving voice in different directions and observing how it responds.

7.2.4 PERFORMANCE TESTING

Performance testing for accuracy is crucial for evaluating the performance of Automatic Speech Recognition (ASR) systems, including Speech-to-Text (STT) systems. However, it’s essential to recognize that accuracy scores can vary significantly based on use cases, audio recording quality, and acoustic conditions. The industry standard method for comparison is the Word Error Rate (WER). WER measures the difference between the recognized transcription and the ground truth (expected transcription). Lower WER indicates better accuracy. Pre-trained models, fine-tuned with labeled speech data, can achieve strong performance in downstream speech recognition tasks. Big corporations have pre-trained encoders on vast amounts of unlabelled audio data, making them a good starting point for ASR systems. Consider examining pivotal aspects related to phoneme recognition, such as vowel and consonant recognition, acoustic-phonetic cues, contextual effects, feature extraction methods, classification techniques, and performance metrics. To calculate accuracy, you’ll need a ground truth file (usually in .txt or .csv format). This file contains the correct or expected transcriptions for comparison. If you don’t have a ground truth file, consider downloading the transcription in a text format for evaluation.

7.2.5 INTEGRATION TESTING

Integration testing for Speech-to-Sign Recognition Verify that the speech recognition module correctly transcribes audio input into text. You can use sample audio clips or synthetic speech data for testing. Ensure that the transcription accurately captures spoken language nuances, accents, and variations. The NLP module converts English sentences (from speech or text) into Indian Sign Language (ISL) sentences. Test the NLP module by providing a range of English sentences and validating the corresponding ISL translations. Consider edge cases, such as complex sentences or idiomatic expressions. The final step involves generating sign language animations using 3D avatars or other visual representations. Verify that the animations correspond correctly to the ISL sentences produced by the NLP module. Test different signs, gestures, and facial expressions to ensure accurate representation. Conduct end-to-end tests that simulate real-world scenarios. Combine speech recognition, NLP, and animation modules to validate the entire pipeline. Use both synthetic and real audio inputs to cover a wide range of use cases. Evaluate the system's response time during integration. Measure latency between speech input and sign language animation output. Consider load testing to assess performance under heavy usage.

7.2.6 ACCEPTANCE TESTING

Acceptance testing for Speech-to-Sign Recognition plays a crucial role in ensuring the system’s readiness for real-world usage. Verify that the system correctly recognizes spoken language and translates it into sign language. Test various scenarios, including different accents, speech speeds, and vocabulary. Ensure that the system handles common phrases, commands, and conversational language accurately. Is the interface intuitive for both hearing-impaired users and those interacting with them? Can users easily switch between speech input and sign language output? Are there clear instructions for using the system?
Gather feedback from potential end-users to identify any usability issues. Compare the system’s output to ground truth sign language translations. Use metrics like Word Error Rate (WER) to quantify accuracy. Test with diverse signers to account for variations in signing styles. Involve actual users (both hearing-impaired and non-signing users) in the testing process.Collect feedback on usability, accuracy, and overall satisfaction.Address any issues identified during UAT.

CODING

CHAPTER 8

8. CODING

8.1EX.HTML

```
<!DOCTYPE html>

<html lang="en">

<div id="bg">

<head>

  <div id="title">

  <title>SPEECH TO SIGN LANGUAGE TRANSLATOR</title>

  <p class="fonting"><b>SPEECH TO SIGN LANGUAGE
        TRANSLATOR</b></p>

  <link rel="stylesheet" href="style.css">

  <script
        src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></
        script>

  <title>Speech to Symbol</title>

</div>

</head>

<body>
```

```
<div  id="lang-translation-input">
  <label for="translatedLang" class="custom-label">Choose translation
       language:</label>
      <select name="translatedLang" id="translatedLang">
         <option value="en" disabled selected hidden>Select</option>
         <option value="as">Assamese</option>
         <option value="bn">Bengali</option>
         <option value="gu">Gujarati</option>
         <option value="hi">Hindi</option>
         <option value="kn">Kannada</option>
         <option value="ml">Malayalam</option>
         <option value="mr">Marathi</option>
         <option value="ne">Nepali</option>
         <option value="or">Odia/Oriya</option>
         <option value="pa">Punjabi</option>
         <option value="sa">Sanskrit</option>
         <option value="te">Telugu</option>
         <option value="ur">Urdu</option>
      </select>

</div>

<button id="startRecording">Start Recording</button>

<button id="stopRecording" disabled>Stop Recording</button>


<p class="spacer" class="status">Status: <span id="instructions">Press the
      'Start Recording' button and speak in English.</span>

</p>


<p class="label">Current Speech:</p>

<div id="resultText">&nbsp;</div>

<div id="resultTextTrans">&nbsp;</div>

<div id="resultSymbol">&nbsp;</div>


<p class="label completeSpeech">Complete Speech:</p>

<div id="resultTextTot">&nbsp;</div>


<div id="resultSymbolTot">&nbsp;</div>
```

```
  <div id="visits-data">


  </div>

  <script src="script.js"></script>




 </div>


</body>


        </html>
```

8.2 STYLE.CSS

```
*{
  margin: 0;
  padding: 0;
  font-family: sans-serif;
}



#bg{
  padding-top: 100px;
  padding-left: 40px;
  width: 100%;
  height: 100vh;
  background-image: url(b.jpg);
  background-size: cover;
  background-position: center;
  position: relative;
}
```

```
.title{

  text-size-adjust: 20px;

}

.fonting{

  font-size: 25px;

  text-align: center;

  color:  #ffffffec;"



}


.custom-label {



  font-size: 20px;

}





.spacer {

  margin-top: 20px;

  margin-bottom: 20px;

  font-size: 20px;
  color:  #ffffffbf;"

}

select {
  border: 2px solid #fff;
  background-color: transparent;
  width: 3cm;
  height: 7mm;
  border-radius: 3.5mm;
  f
  font-size: 4mm;

}


.lang-translation-input{
 padding-top: 100px;
 width: 90%;
 margin-top: 100px;
 display: flex;

 align-items: center;

 justify-content: space-between;

 position: relative;

 z-index: 10;


}



button{

  padding: 8px 25px;

  background: transparent;

  outline: none;

  border: 2px solid #fff;

  border-radius: 20px;

  collor: #ffff;

  font-size: 16px;

  font-weight: bold;

}
```

```
.ASL-letter {
  width: 40px;
}

.status {
  font-style: italic;
  font-size: 80%;
}

#resultText {
  padding: 4px 6px;
  width: 15cm;
  border: 3px solid #ffffff;
  border-radius: 20px;
  margin-bottom: 8px;
}

#resultTextTot {
  padding: 4px 6px ;

  width: 15cm;

  border: 3px solid #ffffff;

  border-radius: 20px;

  margin-bottom: 8px;

}


#resultTextTrans  {

  padding: 4px 6px;

  width: 15cm;

  border: 3px solid #ffffff;

  border-radius: 20px;

  margin-bottom: 8px;

}




.label {

  font-size: 16px;

  font-weight: bold;
  color: #ca1a1acf;

  text-transform: uppercase;

}


.tagline {

  font-size: 90%;

}


#lang-translation-input {

  margin-bottom: 1rem;

  font-size: 90%;

}
```

8.3. MAIN.PY

```
import numpy as np
import cv2
import os
import PIL
from PIL import ImageTk
import PIL.Image
import speech_recognition as sr
import pyttsx3
from itertools import count
import string
from tkinter import *
import time
try:
    import Tkinter as tk
except:
    import tkinter as tk
import numpy as np
image_x, image_y = 64,64
from keras.models import load_model
classifier = load_model('model.h5')
def give_char():
       import numpy as np
       from keras.preprocessing import image
       test_image = image.load_img('tmp1.png', target_size=(64, 64))
       test_image = image.img_to_array(test_image)
       test_image = np.expand_dims(test_image, axis = 0)
       result = classifier.predict(test_image)
       print(result)
       chars="ABCDEFGHIJKMNOPQRSTUVWXYZ"
       indx=  np.argmax(result[0])
       print(indx)
       return(chars[indx])

def check_sim(i,file_map):
       for item in file_map:
           for word in file_map[item]:
               if(i==word):
                   return 1,item
       return -1,""

op_dest="./filtered_data/"
alpha_dest="./alphabet/"
dirListing = os.listdir(op_dest)
editFiles = []
for item in dirListing:
       if ".webp" in item:
           editFiles.append(item)

file_map={}
for i in editFiles:
       tmp=i.replace(".webp","")
       #print(tmp)
       tmp=tmp.split()
       file_map[i]=tmp

def func(a):
       all_frames=[]
       final= PIL.Image.new('RGB', (380, 260))
       words=a.split()
       for i in words:
           flag,sim=check_sim(i,file_map)
           if(flag==-1):
               for j in i:
                   print(j)
                   im = PIL.Image.open(alpha_dest+str(j).lower()+"_small.gif")
                   frameCnt = im.n_frames
                   for frame_cnt in range(frameCnt):
                       im.seek(frame_cnt)
                       im.save("tmp.png")
                       img = cv2.imread("tmp.png")
                       img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
                       img = cv2.resize(img, (380,260))
                       im_arr = PIL.Image.fromarray(img)
                       for itr in range(15):
                           all_frames.append(im_arr)
           else:
               print(sim)
               im = PIL.Image.open(op_dest+sim)
               im.info.pop('background', None)
               im.save('tmp.gif', 'gif', save_all=True)
               im = PIL.Image.open("tmp.gif")
               frameCnt = im.n_frames
               for frame_cnt in range(frameCnt):
                   im.seek(frame_cnt)
                   im.save("tmp.png")
                   img = cv2.imread("tmp.png")
                   img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
                   img = cv2.resize(img, (380,260))
                   im_arr = PIL.Image.fromarray(img)
                   all_frames.append(im_arr)
       final.save("out.gif", save_all=True, append_images=all_frames, duration=100,
loop=0)
       return all_frames

img_counter = 0
img_text=''
class Tk_Manage(tk.Tk):
       def __init__(self, *args, **kwargs):
           tk.Tk.__init__(self, *args, **kwargs)
           container = tk.Frame(self)
           container.pack(side="top", fill="both", expand = True)
           container.grid_rowconfigure(0, weight=1)
           container.grid_columnconfigure(0, weight=1)
           self.frames = {}
           for F in (StartPage, VtoS, StoV):
               frame = F(container, self)
               self.frames[F] = frame
               frame.grid(row=0, column=0, sticky="nsew")
           self.show_frame(StartPage)

       def show_frame(self, cont):
           frame = self.frames[cont]
           frame.tkraise()



class StartPage(tk.Frame):

       def __init__(self, parent, controller):
           tk.Frame.__init__(self,parent)
           label = tk.Label(self, text="Voice to Sign", font=("Verdana", 12))
           label.pack(pady=10,padx=10)
           button = tk.Button(self, text="Voice to Sign",command=lambda:
controller.show_frame(VtoS))

           button2.pack()
           load = PIL.Image.open("Voice to Sign.png")
           load = load.resize((620, 450))
           render = ImageTk.PhotoImage(load)
           img = Label(self, image=render)
           img.image = render
           img.place(x=100, y=200)
```

```
class VtoS(tk.Frame):
    def __init__(self, parent, controller):
        cnt=0
        gif_frames=[]
        inputtxt=None
        tk.Frame.__init__(self, parent)
        label = tk.Label(self, text="Voice to Sign", font=("Verdana", 12))
        label.pack(pady=10,padx=10)
        gif_box = tk.Label(self)

        button1 = tk.Button(self, text="Back to Home",command=lambda:
controller.show_frame(StartPage))
        button1.pack()

        def gif_stream():
            global cnt
            global gif_frames
            if(cnt==len(gif_frames)):
                return
            img = gif_frames[cnt]
            cnt+=1
            imgtk = ImageTk.PhotoImage(image=img)
            gif_box.imgtk = imgtk
            gif_box.configure(image=imgtk)
            gif_box.after(50, gif_stream)
        def hear_voice():
```

```
        global inputtxt
        store = sr.Recognizer()
        with sr.Microphone() as s:
            audio_input = store.record(s, duration=10)
            try:
                text_output = store.recognize_google(audio_input)
                inputtxt.insert(END, text_output)
            except:
                print("Error Hearing Voice")
                inputtxt.insert(END, '')
    def Take_input():
        INPUT = inputtxt.get("1.0", "end-1c")
        print(INPUT)
        global gif_frames
        gif_frames=func(INPUT)
        global cnt
        cnt=0
        gif_stream()
        gif_box.place(x=400,y=160)

    l = tk.Label(self,text = "Enter Text or Voice:")
    l1 = tk.Label(self,text = "OR")
    inputtxt = tk.Text(self, height = 4,width = 25)
    voice_button= tk.Button(self,height = 2,width = 20, text="Record
Voice",command=lambda: hear_voice())
    voice_button.place(x=50,y=180)
    Display = tk.Button(self, height = 2,width = 20,text ="Convert",command =
lambda:Take_input())
        l.place(x=50, y=160)
        l1.place(x=115, y=230)
        inputtxt.place(x=50, y=250)
        Display.pack()




app = Tk_Manage()
app.geometry("800x750")
app.mainloop()
```

8.4. DATA.PY

```
import cv2
import os



directory= 'SignImage48x48/'
print(os.getcwd())

if not os.path.exists(directory):
    os.mkdir(directory)
if not os.path.exists(f'{directory}/blank'):
    os.mkdir(f'{directory}/blank')



for i in range(65,91):
    letter  = chr(i)
    if not os.path.exists(f'{directory}/{letter}'):
        os.mkdir(f'{directory}/{letter}')





import os
import cv2
cap=cv2.VideoCapture(0)
```

```
while True:
    _,frame=cap.read()
    count = {
            'a': len(os.listdir(directory+"/A")),
            'b': len(os.listdir(directory+"/B")),
            'c': len(os.listdir(directory+"/C")),
            'd': len(os.listdir(directory+"/D")),
            'e': len(os.listdir(directory+"/E")),
            'f': len(os.listdir(directory+"/F")),
            'g': len(os.listdir(directory+"/G")),
            'h': len(os.listdir(directory+"/H")),
            'i': len(os.listdir(directory+"/I")),
            'j': len(os.listdir(directory+"/J")),
            'k': len(os.listdir(directory+"/K")),
            'l': len(os.listdir(directory+"/L")),
            'm': len(os.listdir(directory+"/M")),
            'n': len(os.listdir(directory+"/N")),
            'o': len(os.listdir(directory+"/O")),
            'p': len(os.listdir(directory+"/P")),
            'q': len(os.listdir(directory+"/Q")),
            'r': len(os.listdir(directory+"/R")),
            's': len(os.listdir(directory+"/S")),
            't': len(os.listdir(directory+"/T")),
            'u': len(os.listdir(directory+"/U")),
            'v': len(os.listdir(directory+"/V")),
            'w': len(os.listdir(directory+"/W")),
            'x': len(os.listdir(directory+"/X")),
            'y': len(os.listdir(directory+"/Y")),
            'z': len(os.listdir(directory+"/Z")),
            'blank': len(os.listdir(directory+"/blank"))
            }

    row = frame.shape[1]
    col = frame.shape[0]
    cv2.rectangle(frame,(0,40),(300,300),(255,255,255),2)
    cv2.imshow("data",frame)
    frame=frame[40:300,0:300]
    cv2.imshow("ROI",frame)
    frame = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    frame = cv2.resize(frame,(48,48))
    interrupt = cv2.waitKey(10)
    if interrupt & 0xFF == ord('a'):
       cv2.imwrite(os.path.join(directory+'A/'+str(count['a']))+'.jpg',frame)
    if interrupt & 0xFF == ord('b'):
       cv2.imwrite(os.path.join(directory+'B/'+str(count['b']))+'.jpg',frame)
    if interrupt & 0xFF == ord('c'):
       cv2.imwrite(os.path.join(directory+'C/'+str(count['c']))+'.jpg',frame)
    if interrupt & 0xFF == ord('d'):
       cv2.imwrite(os.path.join(directory+'D/'+str(count['d']))+'.jpg',frame)
    if interrupt & 0xFF == ord('e'):
       cv2.imwrite(os.path.join(directory+'E/'+str(count['e']))+'.jpg',frame)
    if interrupt & 0xFF == ord('f'):
       cv2.imwrite(os.path.join(directory+'F/'+str(count['f']))+'.jpg',frame)
    if interrupt & 0xFF == ord('g'):
        cv2.imwrite(os.path.join(directory+'G/'+str(count['g']))+'.jpg',frame)
    if interrupt & 0xFF == ord('h'):
        cv2.imwrite(os.path.join(directory+'H/'+str(count['h']))+'.jpg',frame)
    if interrupt & 0xFF == ord('i'):
        cv2.imwrite(os.path.join(directory+'I/'+str(count['i']))+'.jpg',frame)
    if interrupt & 0xFF == ord('j'):
        cv2.imwrite(os.path.join(directory+'J/'+str(count['j']))+'.jpg',frame)
    if interrupt & 0xFF == ord('k'):
        cv2.imwrite(os.path.join(directory+'K/'+str(count['k']))+'.jpg',frame)
    if interrupt & 0xFF == ord('l'):
        cv2.imwrite(os.path.join(directory+'L/'+str(count['l']))+'.jpg',frame)
    if interrupt & 0xFF == ord('m'):
        cv2.imwrite(os.path.join(directory+'M/'+str(count['m']))+'.jpg',frame)
    if interrupt & 0xFF == ord('n'):
        cv2.imwrite(os.path.join(directory+'N/'+str(count['n']))+'.jpg',frame)
    if interrupt & 0xFF == ord('o'):
        cv2.imwrite(os.path.join(directory+'O/'+str(count['o']))+'.jpg',frame)
    if interrupt & 0xFF == ord('p'):
        cv2.imwrite(os.path.join(directory+'P/'+str(count['p']))+'.jpg',frame)
    if interrupt & 0xFF == ord('q'):
        cv2.imwrite(os.path.join(directory+'Q/'+str(count['q']))+'.jpg',frame)
    if interrupt & 0xFF == ord('r'):
        cv2.imwrite(os.path.join(directory+'R/'+str(count['r']))+'.jpg',frame)
    if interrupt & 0xFF == ord('s'):
        cv2.imwrite(os.path.join(directory+'S/'+str(count['s']))+'.jpg',frame)
    if interrupt & 0xFF == ord('t'):
        cv2.imwrite(os.path.join(directory+'T/'+str(count['t']))+'.jpg',frame)
```

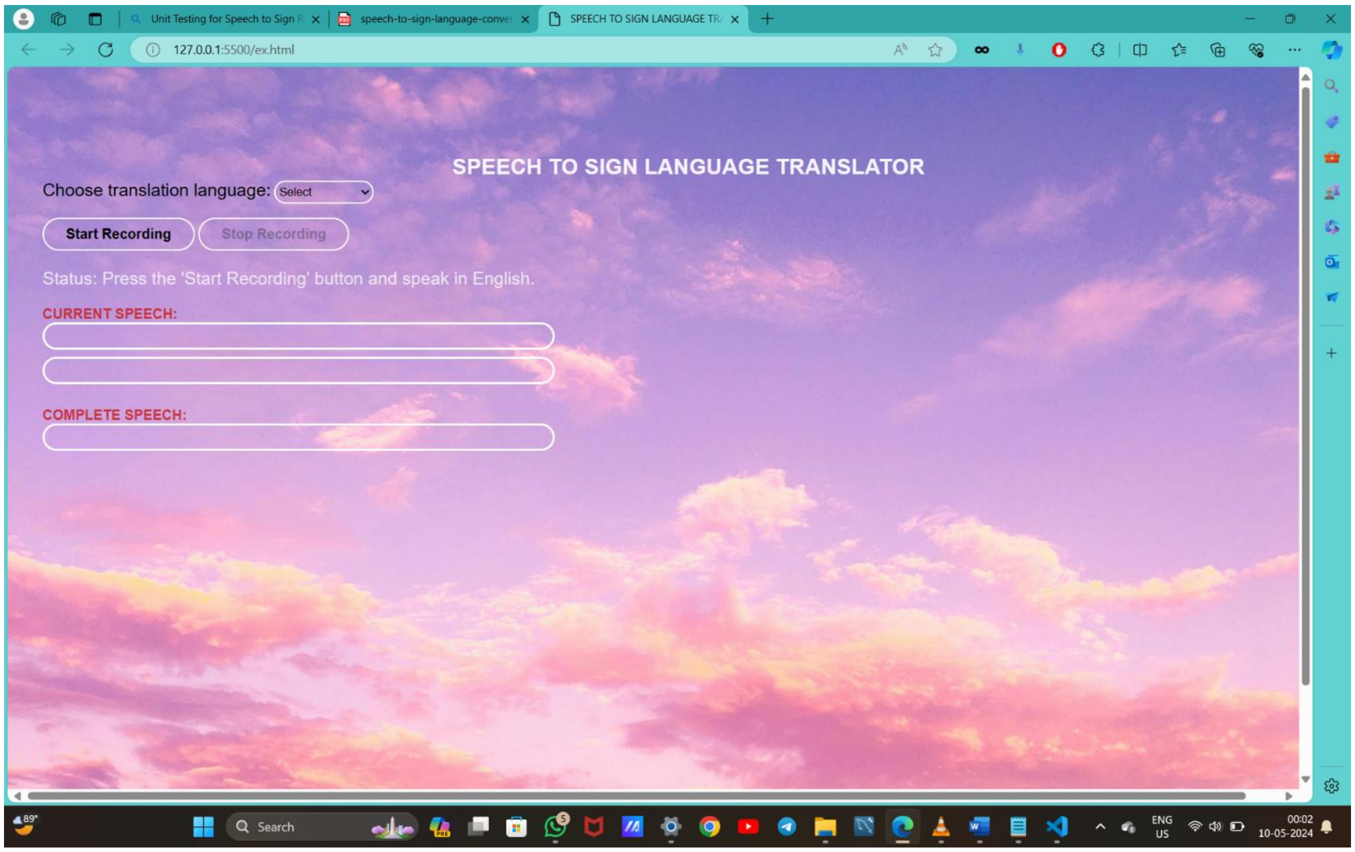
```
if interrupt & 0xFF == ord('u'):
    cv2.imwrite(os.path.join(directory+'U/'+str(count['u']))+'.jpg',frame)
if interrupt & 0xFF == ord('v'):
    cv2.imwrite(os.path.join(directory+'V/'+str(count['v']))+'.jpg',frame)
if interrupt & 0xFF == ord('w'):
    cv2.imwrite(os.path.join(directory+'W/'+str(count['w']))+'.jpg',frame)
if interrupt & 0xFF == ord('x'):
    cv2.imwrite(os.path.join(directory+'X/'+str(count['x']))+'.jpg',frame)
if interrupt & 0xFF == ord('y'):
    cv2.imwrite(os.path.join(directory+'Y/'+str(count['y']))+'.jpg',frame)
if interrupt & 0xFF == ord('z'):
    cv2.imwrite(os.path.join(directory+'Z/'+str(count['z']))+'.jpg',frame)
if interrupt & 0xFF == ord('.'):
    cv2.imwrite(os.path.join(directory+'blank/' + str(count['blank']))+ '.jpg',frame)
```

SCREENSHOT

CHAPTER 9

9. SCREENSHOT

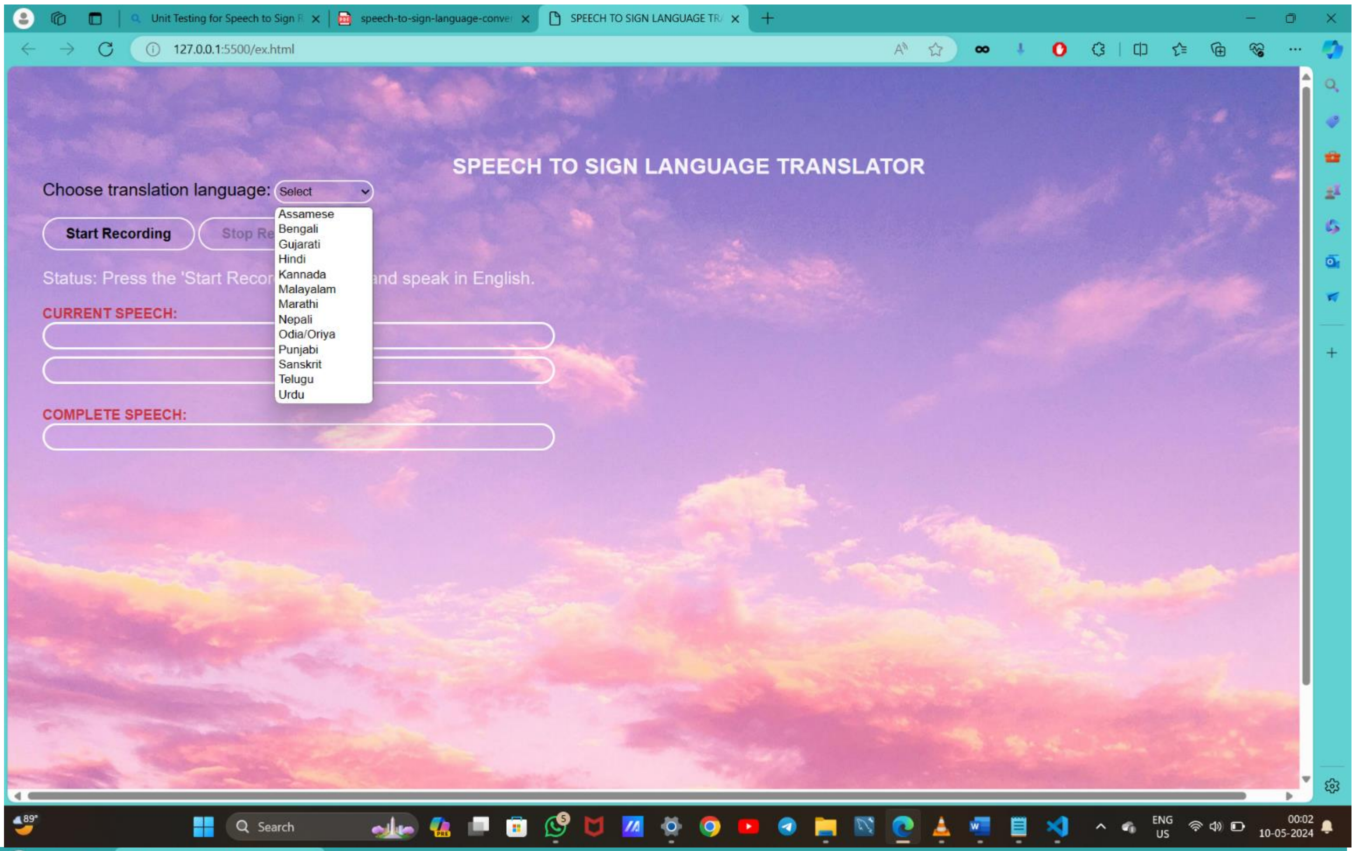
9.1. OUTPUT



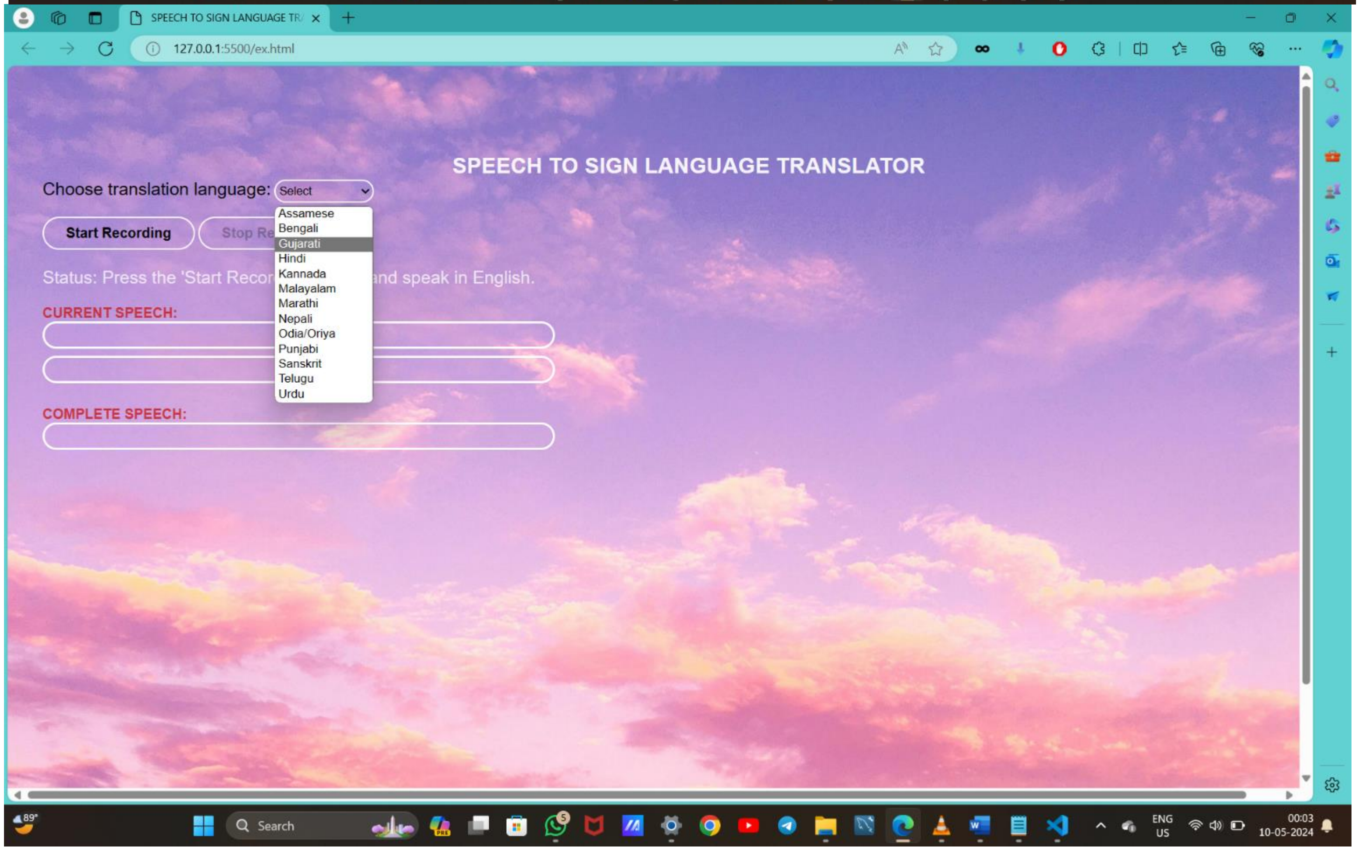
SPEECH TO SIGN LANGUAGE TRANSLATOR

Choose translation language: Select

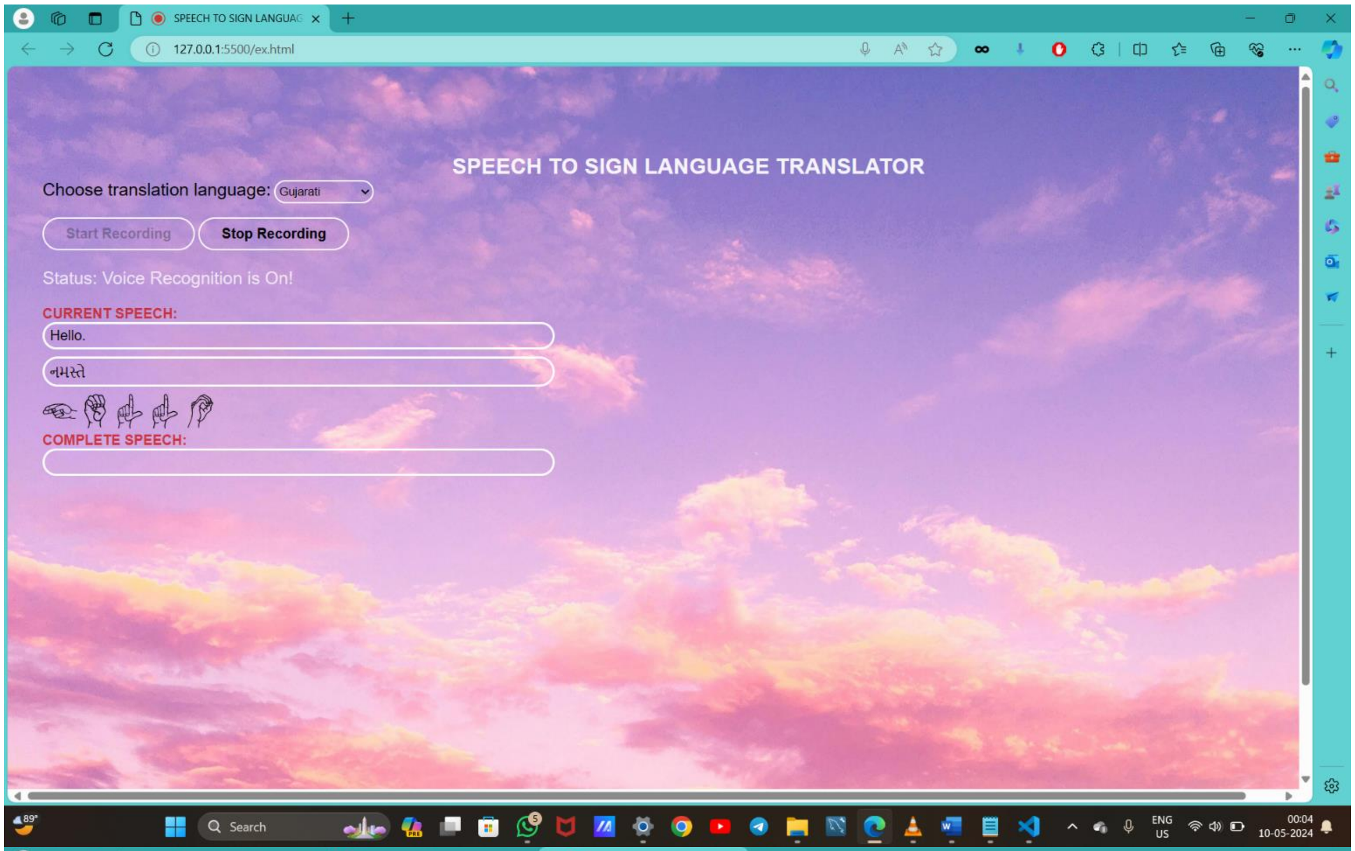
Status: Press the 'Start Recording' button and speak in English.



Unit Testing for Speech to Sign
speech-to-sign-language-conver
SPEECH TO SIGN LANGUAGE TR
127.0.0.1:5500/ex.html
SPEECH TO SIGN LANGUAGE TRANSLATOR
Choose translation language:
Select
Assamese
Bengali
Gujarati
Hindi
Kannada
Malayalam
Marathi
Nepali
Odia/Oriya
Punjabi
Sanskrit
Telugu
Urdu
Start Recording
Stop Re
Status: Press the 'Start Recor … nd speak in English.
CURRENT SPEECH:
COMPLETE SPEECH:



SPEECH TO SIGN LANGUAGE TR
127.0.0.1:5500/ex.html
SPEECH TO SIGN LANGUAGE TRANSLATOR
Choose translation language:
Select
Assamese
Bengali
Gujarati
Hindi
Kannada
Malayalam
Marathi
Nepali
Odia/Oriya
Punjabi
Sanskrit
Telugu
Urdu
Start Recording
Stop Re
Status: Press the 'Start Recor … nd speak in English.
CURRENT SPEECH:
COMPLETE SPEECH:



Choose translation language: Gujarati

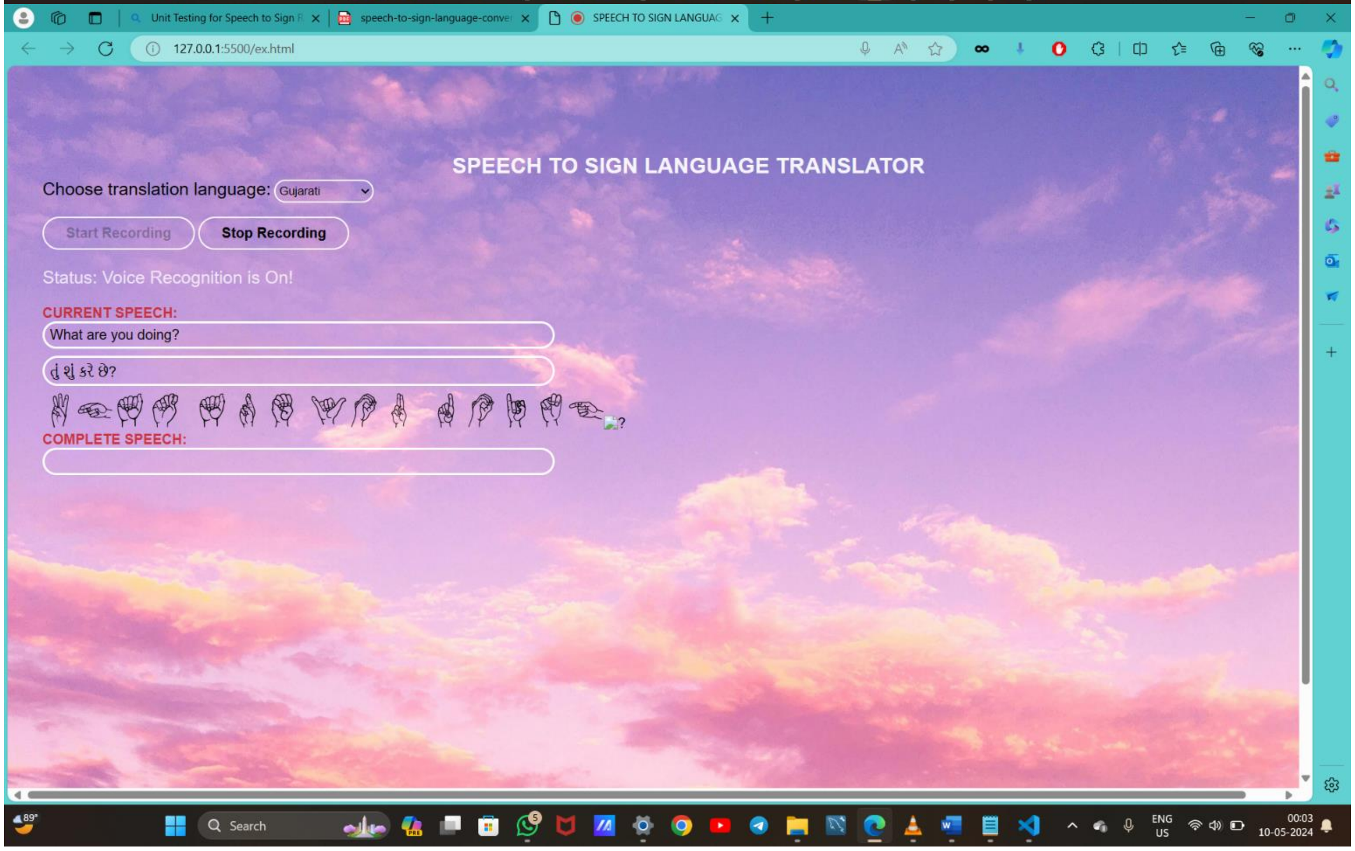
Status: Voice Recognition is On!

CURRENT SPEECH:

Hello.

નમસ્તે

COMPLETE SPEECH:



Choose translation language: Gujarati

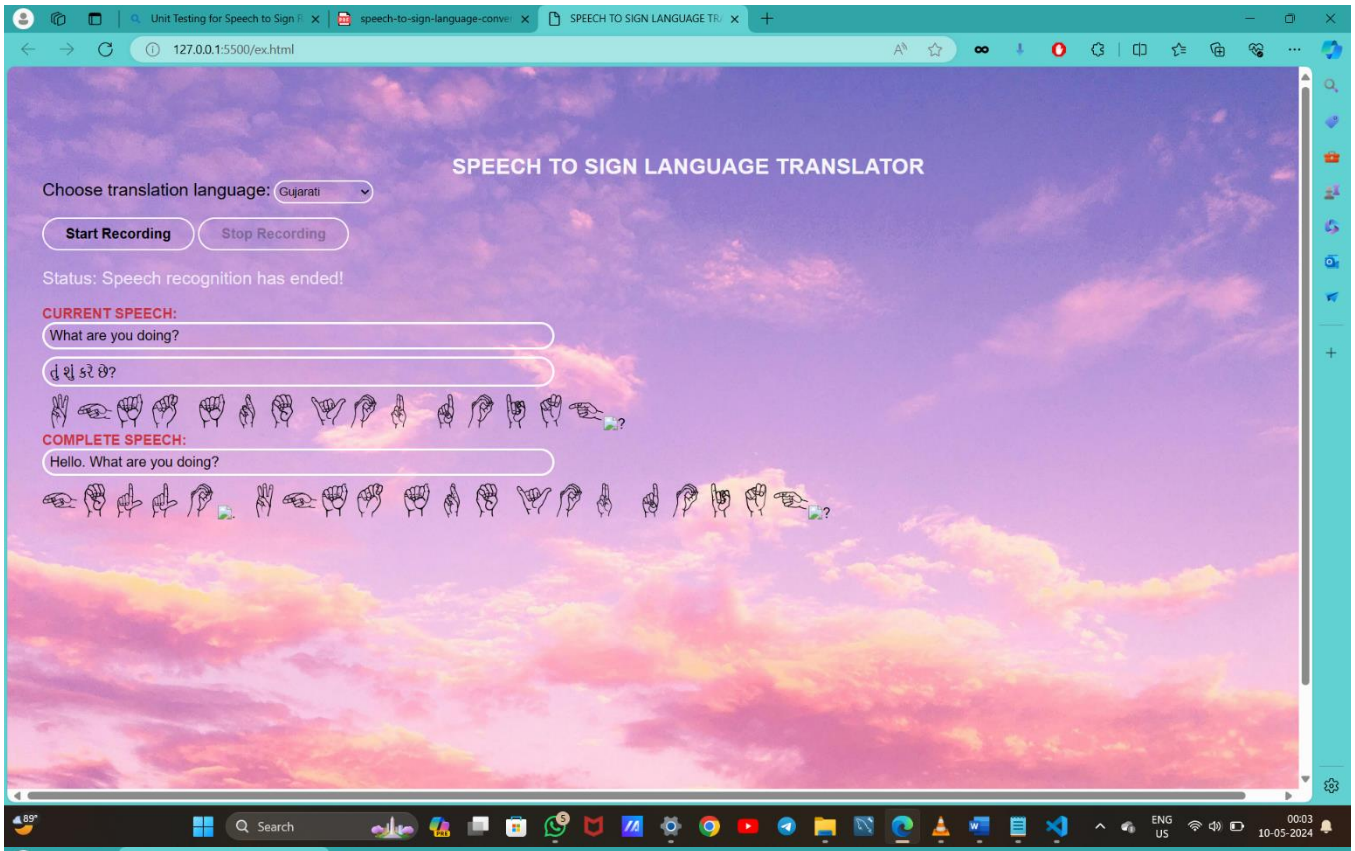
Status: Voice Recognition is On!

CURRENT SPEECH:

What are you doing?

તું શું કરે છે?

COMPLETE SPEECH:



SPEECH TO SIGN LANGUAGE TRANSLATOR

Choose translation language: Gujarati

Start Recording | Stop Recording

Status: Speech recognition has ended!

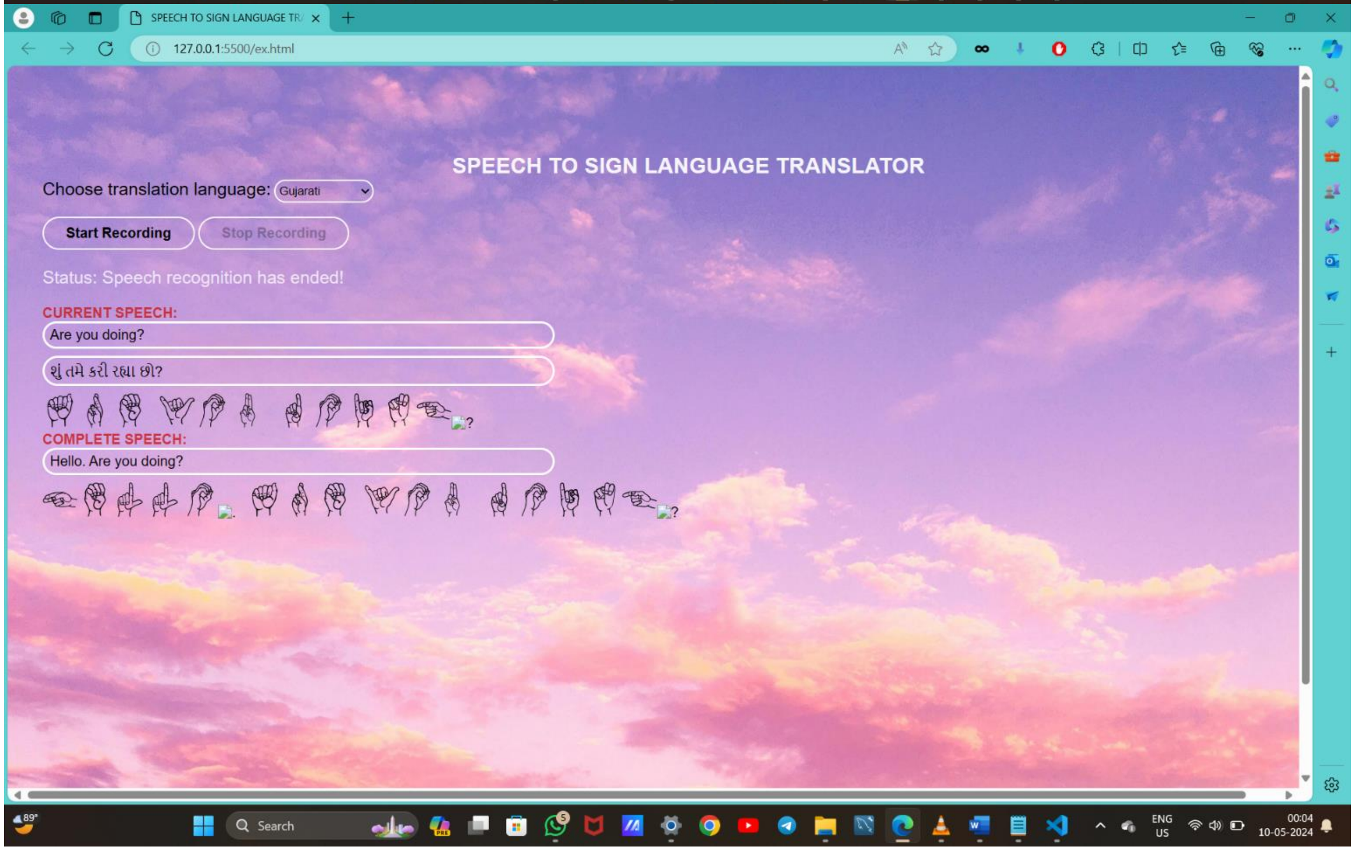
CURRENT SPEECH:

What are you doing?

તું શું કરે છે?

COMPLETE SPEECH:

Hello. What are you doing?



SPEECH TO SIGN LANGUAGE TRANSLATOR

Choose translation language: Gujarati

Start Recording | Stop Recording

Status: Speech recognition has ended!

CURRENT SPEECH:

Are you doing?

શું તમે કરી રહ્યા છો?

COMPLETE SPEECH:

Hello. Are you doing?

FUTURE ENHANCEMENT

CHAPTER 10

10. FUTURE ENHANCEMENT

Future scope of this project will involve adding more parameters and factors like the multiple languages and animated model etc. The more the parameters are taken into account more will be the accuracy. The algorithms can also be applied for analyzing the contents of public comments and thus determine patterns/relationships between the customer and the corporate employee.

CONCLUSION

CHAPTER 11

11. CONCLUSION

Our model successfully converts the whole audio input sentence or word into text using speech-to-text API then using the semantics of tongue. Processing to breakdown the text into smaller understandable pieces which need Machine Learning as a neighborhood. Finally, Data sets of predefined signing are used because the input in order that the software can use AI to display the converted audio into the signing for more development on this track are often done because the ISL dictionary remains small and wishes to grow eventually on process performance.

REFERENCES

CHAPTER 12

12. REFERENCES

[1] Palaz, Dimitri, Mathew Magimai Doss, and Ronan Collobert. ”Convo- lutional neural networks-based continuous speech recognition using raw speech signal.” In 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4295-4299. IEEE, 2015.

[2] Abdel-Hamid, Ossama, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. ”Convolutional neural networks for speech recognition.” IEEE/ACM Transactions on audio, speech, and language processing 22, no. 10 (2014): 1533-1545.

[3] Noda, Kuniaki, Yuki Yamaguchi, Kazuhiro Nakadai, Hiroshi G. Okuno, and Tetsuya Ogata. ”Audio-visual speech recognition using deep learn- ing.” Applied Intelligence 42, no. 4 (2015): 722-737.

[4] Park, Sunchan, Yongwon Jeong, and Hyung Soon Kim. ”Multiresolution CNN for reverberant speech recognition.” In 2017 20th Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment (O- COCOSDA), pp. 1-4. IEEE, 2017.

[5] HY, Vani, and Anusuya MA. ”A Neuro Fuzzy Classifier with Linguistic Hedges for Speech Recognition.” EAI Endorsed Transactions on Internet of Things 5, no. 20 (2020).

[6] Damodar, Navya, H. Y. Vani, and M. A. Anusuya. ”Voice emotion recognition using CNN and decision tree.” Int J Innov Technol Expl Eng 8, no. 12 (2019): 4245-4249.

[7] https://www.kaggle.com/uwrfkaggler/ravdess-emotional speech-audio.

[8] Deng, Li, Geoffrey Hinton, and Brian Kingsbury. ”New types of deep neural network learning for speech recognition and related applications: An overview.” In 2013 IEEE international conference on acoustics, speech and signal processing, pp. 8599-8603. IEEE, 2013.