FINAL REPORT

REAL-TIME COMMUNICATION SYSTEM POWERED BY AI FOR SPECIALLY ABLED

SUBMITTED BY

TEAM ID : PNT2022TMID47348

NAME	REG. NO
KAMALESH S	830119104014
SELVARAJ M	830119104041
SRIDHAR V	830119104046
MAGENDRAN P	830119104302
NIMAL SOLAI RAJHA SP	830119104303

Industry Mentor(s) Name : Divya

13. APPENDIX

Source Code

GitHub & Project Demo Link

1.INTRODUCTION

1.1 Project Overview

In our society, we have people with disabilities. The technology is developing day by day but no significant developments are undertaken for the betterment of these people. Communications between deaf-mute and a normal person has always been a challenging task. It is very difficult for mute people to convey their message to normal people. Since normal people are not trained on hand sign language. In emergency times conveying their message is very difficult. The human hand has remained a popular choice to convey information in situations where other forms like speech cannot be used. Voice Conversion System with Hand Gesture Recognition and translation will be very useful to have a proper conversation between a normal person and an impaired person in any language.

The project aims to develop a system that converts the sign language into a human hearing voice in the desired language to convey a message to normal people, as well as convert speech into understandable sign language for the deaf and dumb. We are making use of a convolution neural network to create a model that is trained on different hand gestures. An app is built which uses this model. This app enables deaf and dumb people to convey their information using signs which get converted to human-understandable language and speech is given as output.

1.2 Purpose

The main purpose of this research is to enhance the communication of the disabled community. The authors of this chapter propose an enhanced interpersonal-human interaction for people with special needs, especially those with physical and communication disabilities. The proposed model comprises of automated real time behaviour monitoring, designed and implemented with the ubiquitous and affordable concept in mind to suit the underprivileged. In this chapter, the authors present the prototype which encapsulates an automated facial expression recognition system for monitoring the disabled, equipped with a feature to send Short Messaging System (SMS) for notification purposes. The authors adapted the Viola-Jones face detection algorithm at the face detection stage and implemented template matching technique for the expression classification and recognition stage. They tested their model with a

few users and achieved satisfactory results.

2. LITERATURE SURVEY

2.1 Existing problem

This system consists mainly of two modules, the first module is Indian Sign Language (ISL) gestures from real-time video and mapping it with human-Understandable speech. Accordingly, the second module is the natural language as Input and card with equivalent Indian Sign Language animated gestures.

This paper presents design and implementation of real-time sign language recognition system, to 26 gestures from the Indian sign language .

In this system edge detection algorithm is used to recognize the input character image gray scale and recognition of the edges of the hand gesture. The system is able to handle the different input records images of alphabets, words, sentences, and translates them in text and vice versa. The system is designed to translate the Marathi sign language to text

2.2 References

1. Aditi Kalsh, N.S. Garewal , “Sign Language Recognition for Deaf & Dumb”, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 9, September 2013.

2.Beifang Yi, Frederick C. Harris Jr., Ling Wang, and Yusong Yan, “Real-Time Natural Hand Gestures,” IEEE journal on Computing in Science and Engineering, 2005

3. A. Chaudhary, J.L. Raheja, K. Das, S. Raheja, "A Survey on Hand Gesture Recognition in context of Soft Computing", Published as Book Chapter in “Advanced Computing” CCIS, Springer Berlin Heidelberg, Vol. 133, 2011, pp. 46-55

4. Ibrahim, P., and Srinivasa, R. “Automated speech recognition approach to continuous cue symbols generation”, International journal of power control signal and computation, Vol. 18, No. 8,pp. 434- 520,2007

5. J.L. Raheja, A. Chaudhary, S. Maheshwari, “Automatic Gesture Pointing Location Detection”, Optik: International Journal for Light and Electron Optics, Elsevier, Vol. 125, Issue 3, 2014, pp. 993-996.

6. Yuan Yao, and Yun Fu, “Contour Model based Hand-Gesture Recognition Using Kinect Sensor,” IEEE Transactions on Circuits and Systems for Video Technology, 2013

2.3 Problem Statement Definition

Communication is the only medium by which we can share our thoughts or convey the message but for a person with disability (deaf and dumb) faces difficulty in communication with normal person. Generally dumb people use sign language for communication but they find difficulty in communicating with others who don’t understand sign language

3. IDEATION & PROPOSED SOLUTION

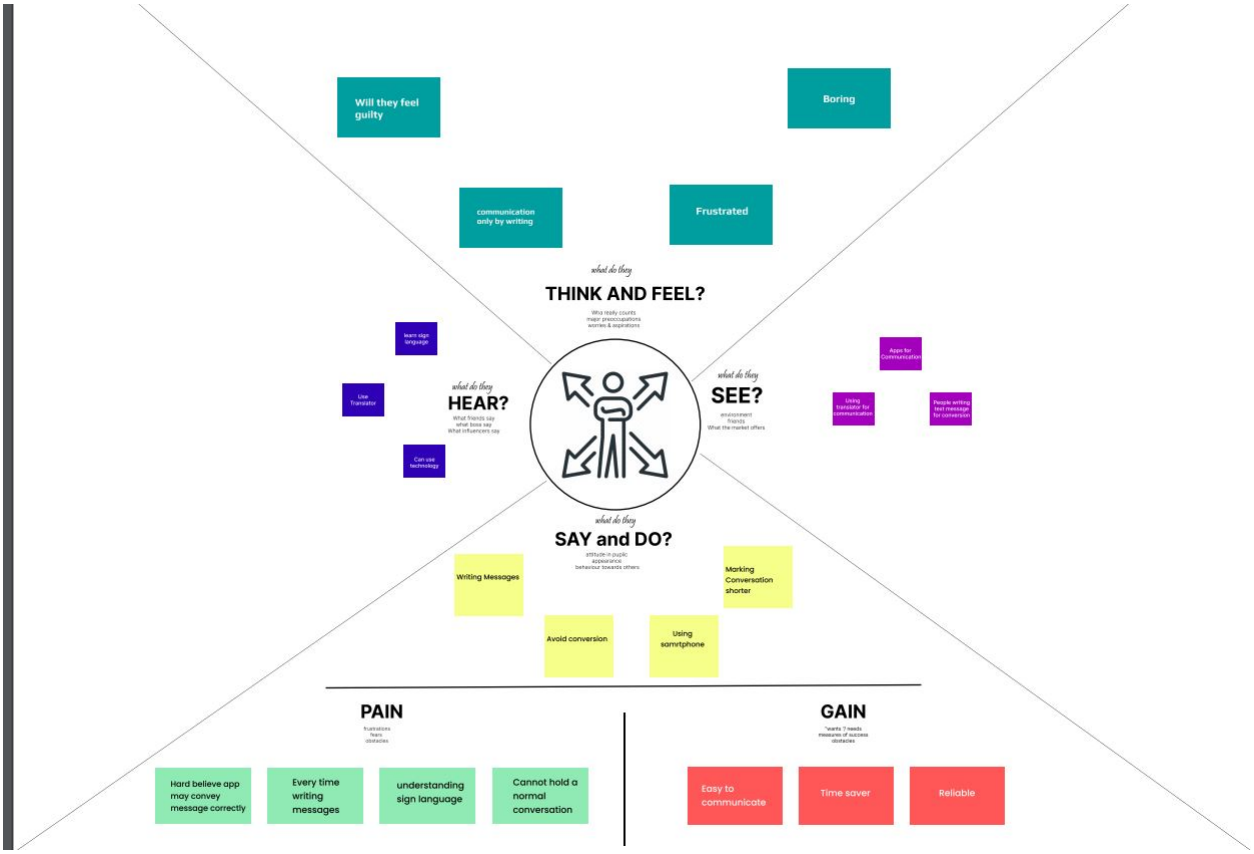


Figure.3.1: Empathy map for normal user

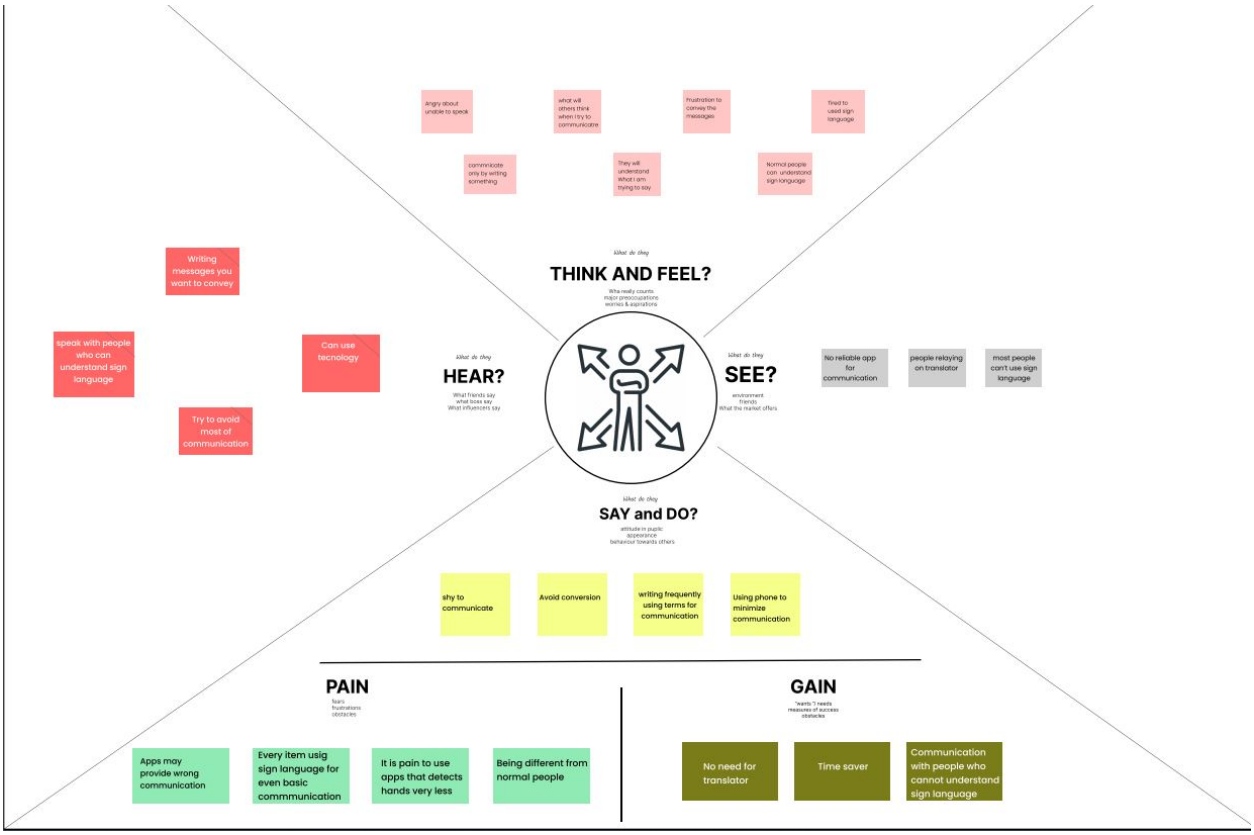


Figure 3.2 : Empathy map for abled person

3.2 Ideation & Brainstorming

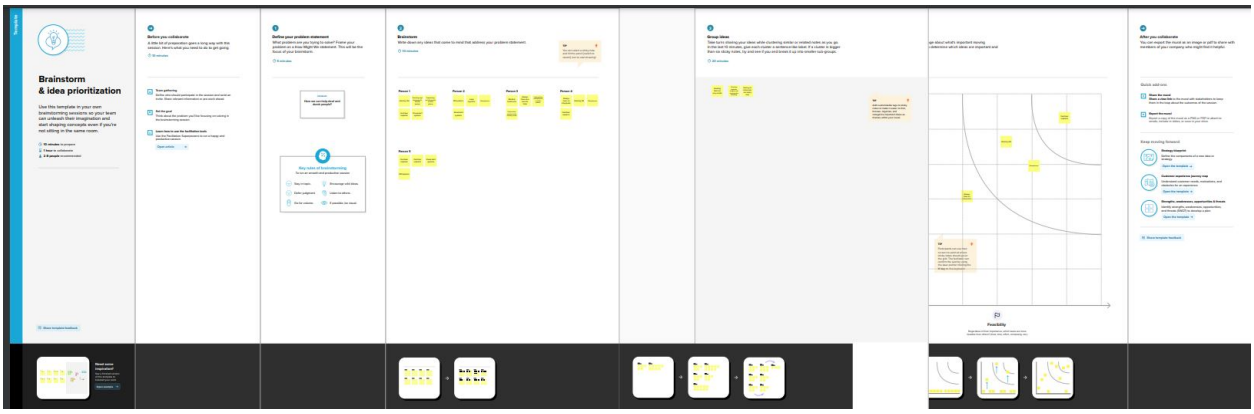
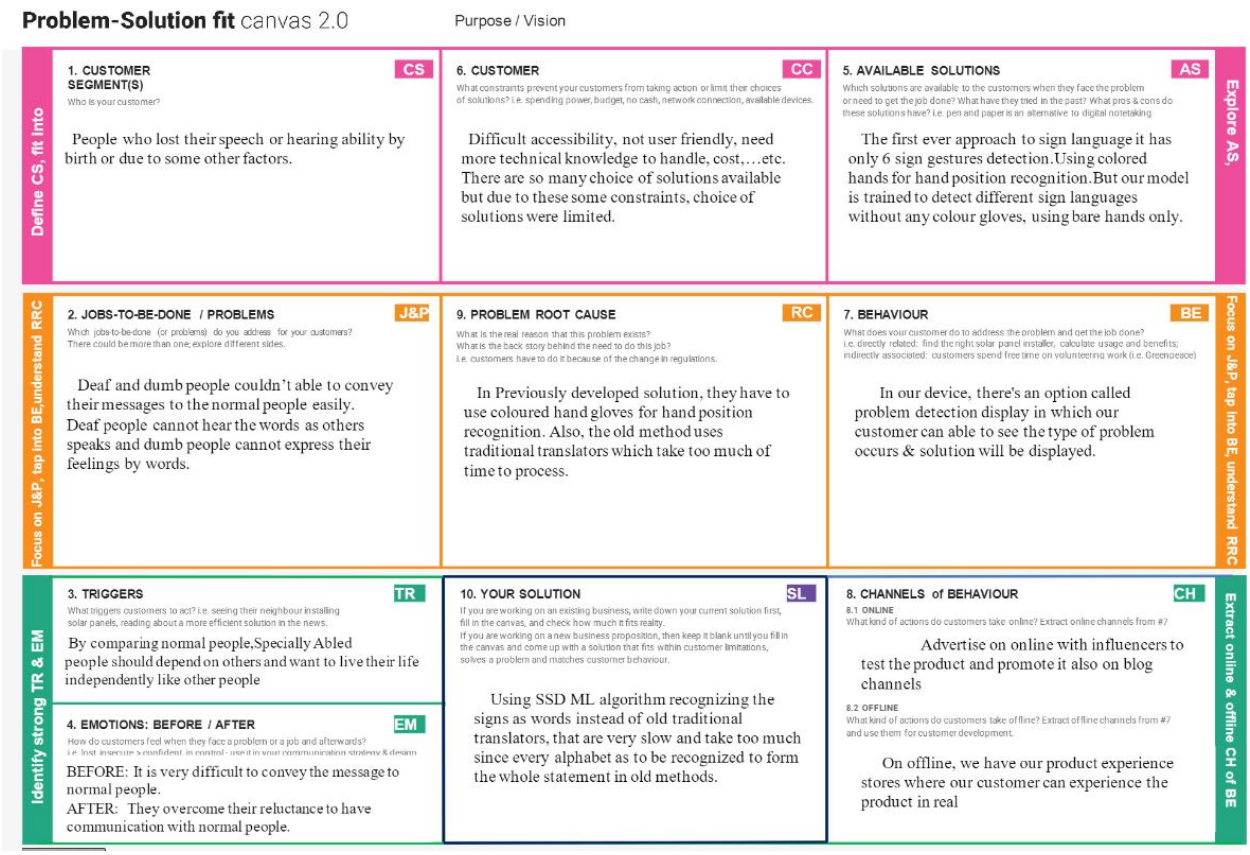


Figure 3.3 : Brainstroming

3.3 Proposed Solution

S.No	Parameter	Description
1	Problem Statement (Problem to be solved)	Deaf and dumb people couldn't able to communicate with the normal people easily.
2	Idea/Solution description	A real time ML based system is built for the real time sign language detection with a Tensor Flow object detection.
3	Novelty/Uniqueness	This model using SSD ML algorithm recognizing the signs as words instead of old traditional translators, that are very slow and take too much since every alphabet as to be recognized to form the whole statement in old methods.
4	Social Impact/Customer satisfaction	It drastically reduce communication difference gap between normal people and specially abled people with the help of AI. So they can live their life independently.
5	Business Model (Revenue Model)	We use freemium business revenue model for making revenue. In our device, we give most of the basic features for free of charge but they have to pay if they need more advanced features.
6	Scalability of the Solution	The model which is TensorFlow model that has been used can be replaced with another model as well. The same system can be implemented for different sign languages by substituting the dataset.

3.4 Problem Solution fit



4. REQUIREMENT ANALYSIS

4.1 Functional requirement

REQUIREMENTS	FUNCTIONAL REQUIREMENTS
Objective	Most people communicate efficiently without any issues, but many cannot due to disability.
Focus	The hand gesture recognition system consists of three major parts: palm detection, hand tracking, and trajectory recognition

Documentation	In Ideation phase
End case	This aimed at evaluating and comparing the methods used in the sign recognition systems, classification methods used and identifies the most promising approach for this project.
Essentially	Web camera is essential for capturing image.
Origin type	Artificial intelligence that was being developed can identify errors on hand gesture matches and will stop as a default. It will generate corresponding gestures that allow every user to read and be able to understand what the gesture means

Testing	Feature extraction depends on the application. On D-talk, finger status, skin color, alignments of the finger, and the palm position are taken into consideration. After features extracted, they sent to training and testing classification algorithms to reach the output.

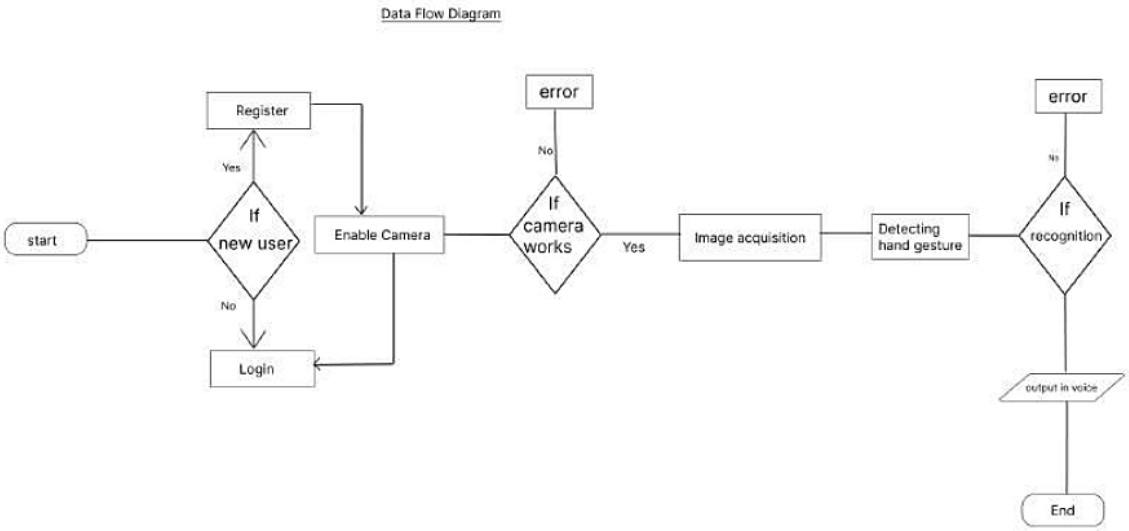
4.2 Non-Functional requirements

NFR No.	Non-Functional Requirement	Description
NFR-1	Usability	The user will have access to all the resources present in that website.
NFR-2	Security	User information is protected.

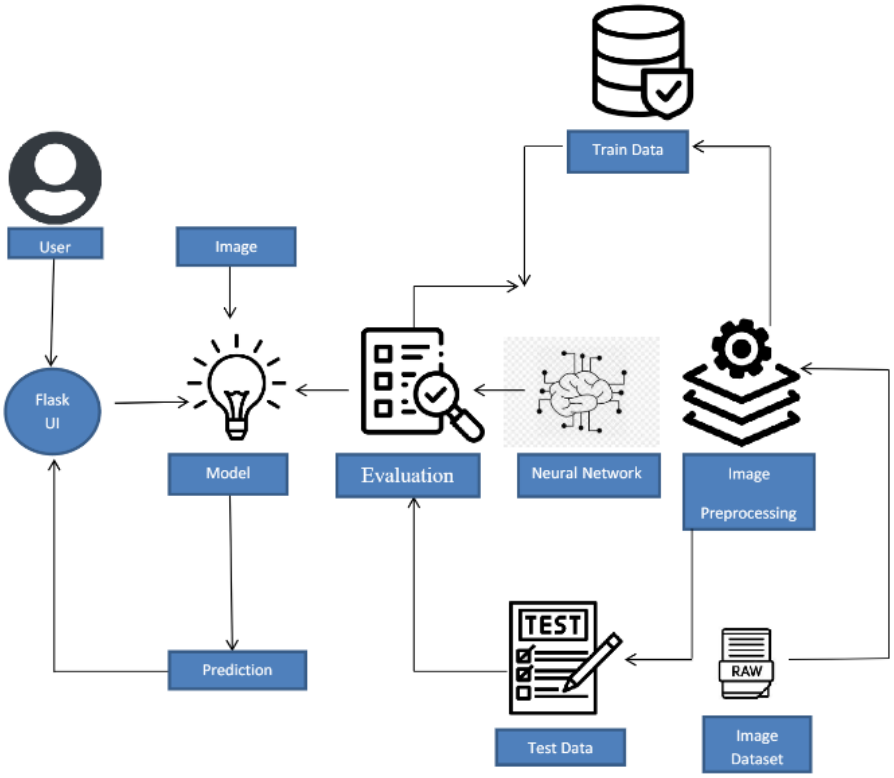
NFR-3	Reliability	It offers accurate results.
NFR-4	Performance	The web application makes use of light weight model hence the result will be accurate and fas
NFR-5	Availability	The web application can be accessed 24/7 from anywhere when connected to the internet
NFR-6	Scalability	The trained ML model can provide accurate results whenever the size of the dataset and the number of users is extended

5. PROJECT DESIGN

5.1 Data Flow Diagrams



5.2 Solution & Technical Architecture



5.3 User Stories

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer	Uploading the real time data.	USN-1	The user will be presented with two options. 1. Speech to sign language conversion.	They can access the portal	High	Sprint 1

			2. Sign language to speech conversion			
		USN-2	Language selection	They can access the portal	Low	Sprint-1
		USN-3	The deaf-mute person will choose the speech to sign language conversion which would take them into a portal that collects the real time data (sign language recognition) and converts it into speech simultaneously.	Video processing	High	Sprint 2
		USN-4	Emotion detection	Video processing	Medium	Sprint-1
		USN-5	Normal person would choose speech to sign language which would take them into a portal	Video and audio processing	High	Sprint-1

			where their speech is converted into sign language simultaneously.			

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	Medium	Magendran, Sridhar
Sprint-1	Login	USN-2	As a user, I can log into the application by entering email & password	1	High	Magendran, Sridhar
Sprint-1	Data Collection	USN-3	Gathering the information from variousresources	1	Medium	Nimal solai raja,Kamlesh,Selvaraj
Sprint-1	Data Preprocessing	USN-4	To Convert and clean the raw data	2	High	Kamalesh, Magendran, Nimal solai raja, Selvaraj, Sridhar
Sprint-2	Model Building	USN-5	Using cleaned dataset,	2	High	Kamalesh, Magendran, Nimal solai

			Model can be build using ML Algorithm			raja, Selvaraj, Sridhar
Sprint-2		USN-6	Training the classificati on mode	1	High	Magendran, Selvaraj
Sprint-3	Application Building	USN-7	Building Python code and run the application	1	Medium	Kamalesh, Magendran, Nimal solai raja, Selvaraj, Sridhar
Sprint-3		USN-8	Predicted Result	1	Medium	Kamalesh, Magendran, Nimal solai raja, Selvaraj, Sridhar
Sprint-4	Implementa tion of the application and deployment on cloud	USN-9	Deployed on IBM Cloud	2	High	Kamalesh, Magendran, Nimal solai raja, Selvaraj, 6.2 Sprint Delivery ScheduleSri dhar

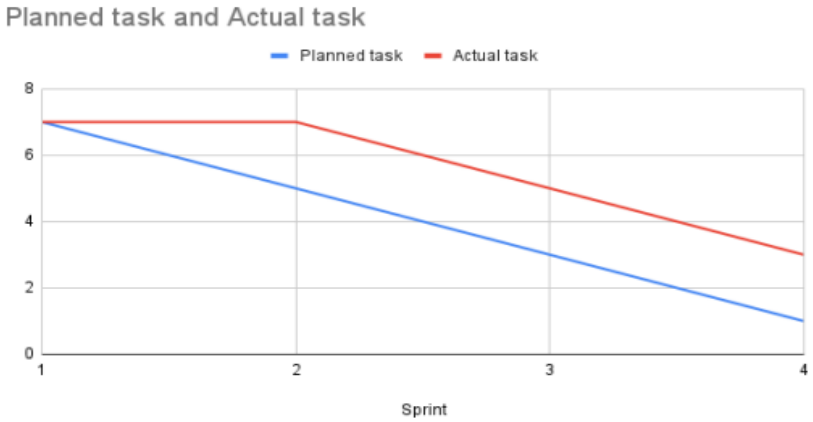
6.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End)	Sprint Release Date (Actual)
Sprint 1	20	6 Days	30 Oct 2022	05 Nov 2022	20	05 Oct 2022

Sprint 1	20	6 Days	06 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-3	20	5 Days	12 Nov 2022	17 Nov 2022	20	17 Nov 2022
Sprint-4	20	2 Days	17 Nov 2022	19 Nov 2022	20	19 Nov 2022

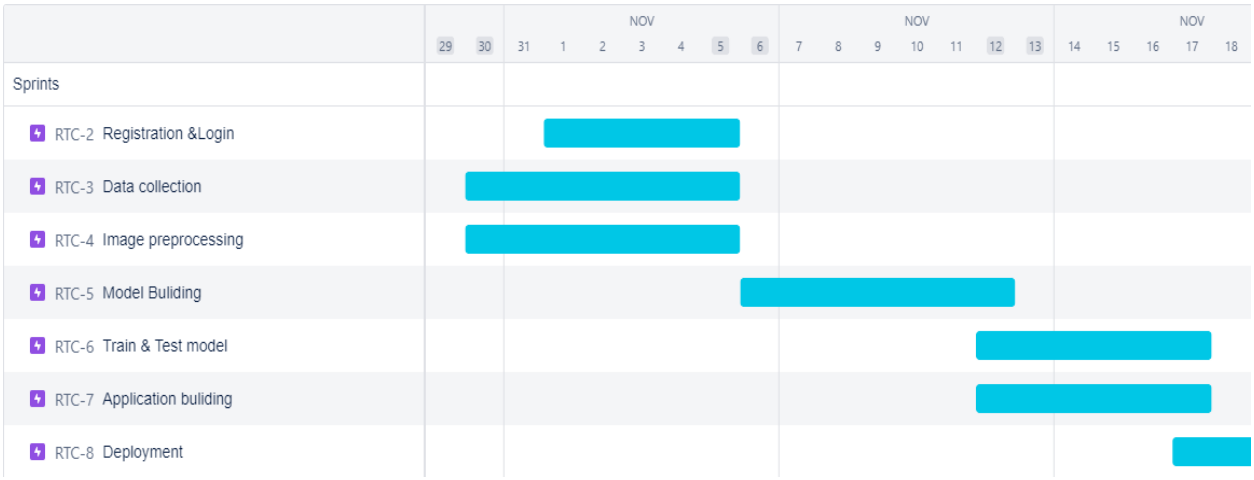
Velocity:

We have 6-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day).



AV = 20/6 = 3.33

6.3 Reports from JIRA



7. CODING & SOLUTIONING

7.1 Libraries to be installed

1. **pip install open-cv**
2. **pip install flask**
3. **pip install tensorflow**
4. **pip install keras**
5. **pip install gTTS**
6. **pip install scikit-image**
7. **pip install numpy**
8. **pip install pyttsx3**

7.2 Real time sign to speech

Sign language is generally used by the people who are unable to speak, for communication. Most people will not be able to understand the Universal Sign Language (unless they have learnt it) and due to this lack of knowledge about the language, it is very difficult for them to communicate with mute people. A device that helps to bridge a gap between mute persons and other people forms the crux of this project. Our system makes use of a model build using CNN that is capable of detection sign languages real time.

7.3 Facial Emotion Detection

Our system makes use of the FER model. Facial Emotion Recognition (commonly known as FER) is one of the most researched fields of computer vision till date and is still in continuous evaluation and improvement. The model is a convolutional neural network with weights saved to HDF5 file in the data folder relative to the module's path. It can be overridden by injecting it into the FER() constructor during instantiation with the emotion_model parameter.

7.4 Language Customization

Google Translate is a free multilingual machine translation service. It can translate the Website's text content from one language to another. It offers a huge list of languages to translate and has an efficient, reliable and easy way to translate the webpage in whatever language the user wants. It supports over 100 languages. Use this website translator to convert webpages into your choice of language.

7.5 Real time speech to text

With the Web Speech API, we can recognize speech using JavaScript. It is super easy to recognize speech in a browser using JavaScript and then getting the text from the speech to use as user input. We use the SpeechRecognition object to convert the speech into text and then display the text on the screen. Our system is capable of doing this over real time. It is capable of recognizing any languages in which the user is trying to communicate. But the support for this API is limited to the Chrome browser only. So if you are viewing this example in some other browser, the live example below might not work.

8. TESTING

8.1 Test Cases

- Verify if user can see the options when user clicks the URL
- Verify if the UI elements are getting displayed properly
- Verify if the user can choose any languages
- Verify if the user is getting redirected to the sign to speech page
- Verify if the application can convert the sign to speech
- Verify if the user can exit the sign to speech page
- Verify if the user is getting redirected to the speech to sign page
- Verify if the UI elements are being displayed
- Verify if the application can convert speech to text on clicking voice to text button.

8.2 User Acceptance Testing

1. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Desig	11	7	4	2	24
Duplicate	1	0	2	0	3
External	2	3	2	1	8
Fixed	10	5	3	14	32
Not Reproduced	0	0	0	1	1
Skipped	0	0	1	1	2
Won't Fix	1	0	0	0	1
Totals	25	15	13	18	71

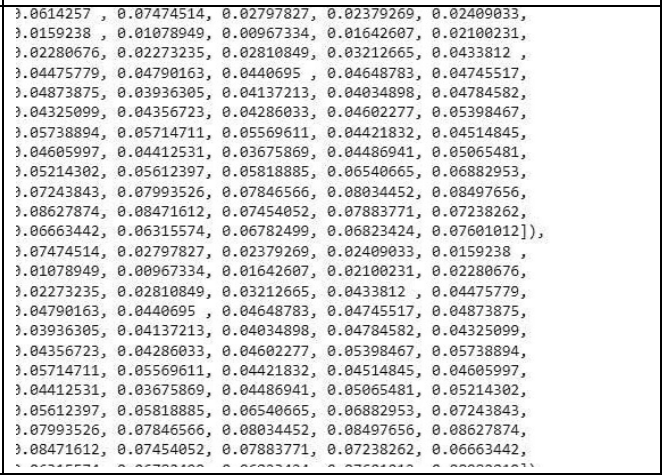
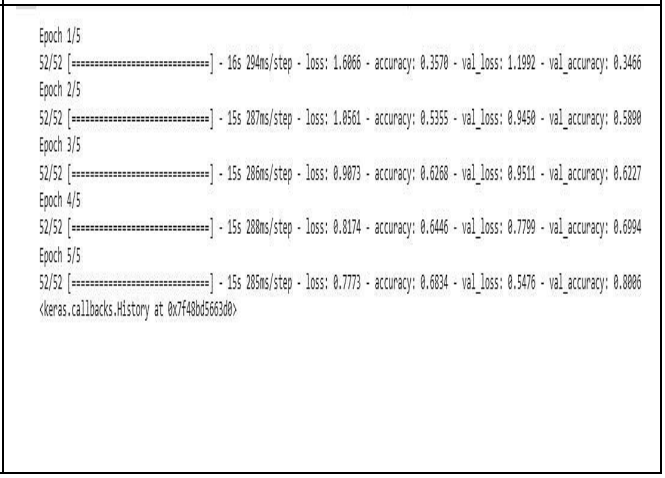
2. Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	15	0	0	15

Security	2	0	0	2
Outsource Shipping	2	0	0	2
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Contro	2	0	0	2

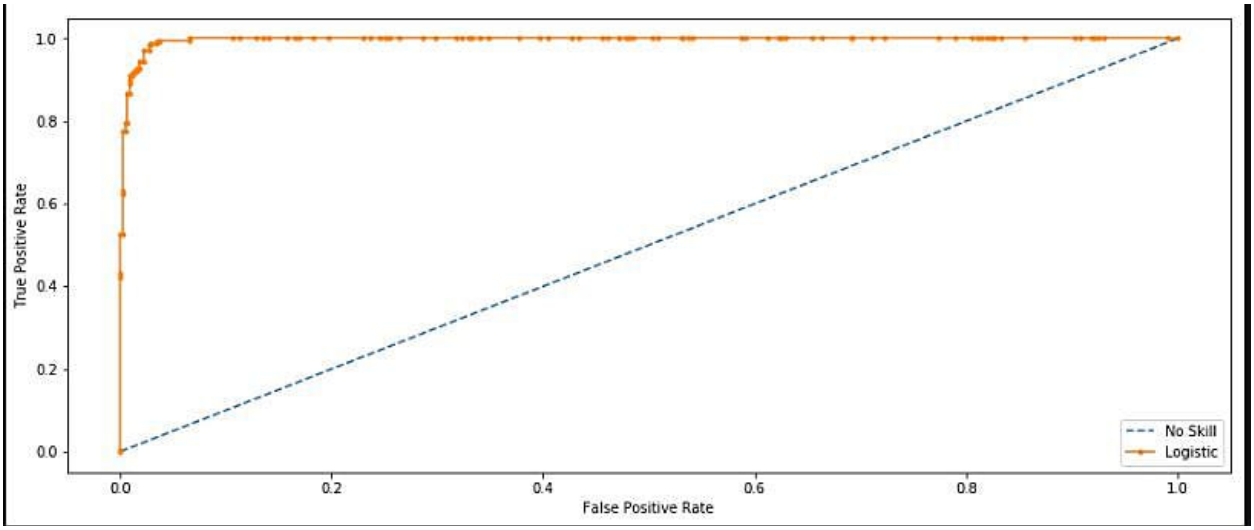
8.3 Performance Testing

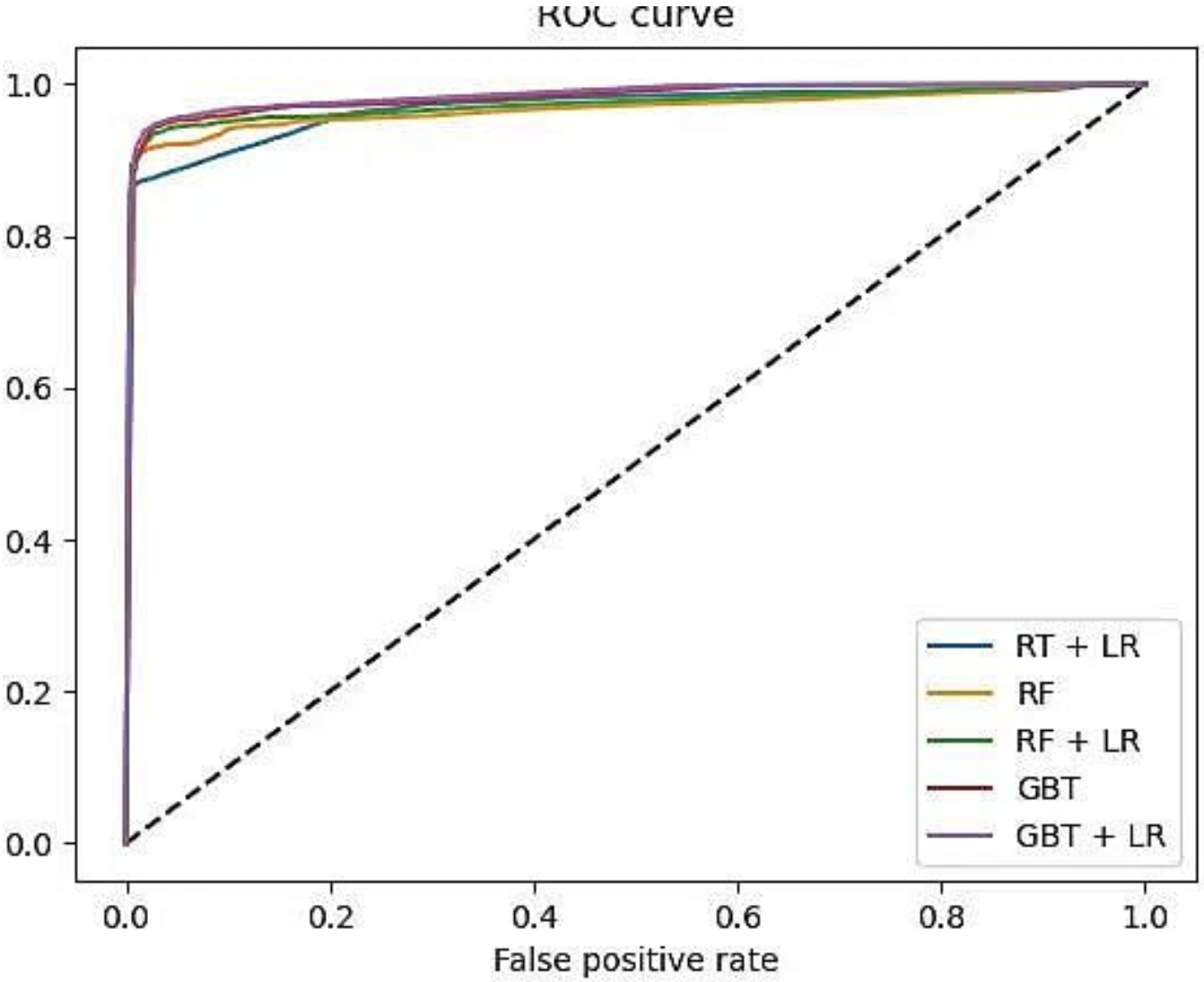
S.NO	Parameters	Values	Screenshot
1	Model Summary	Total params: 1,103,721 Trainable params: 1,103,721 Non-trainable params: 0	
2	Accuracy	Training Accuracy - 0.9998 Validation Accuracy -0.9976	

9. Results

Performance Metrics

The following images can be studied to understand the performance metrics of our system





Source: Eugenio Zuccarelli

10. Advantages and Disadvantages

Advantages:

- Real time sign to speech detection.
- Model provides good accuracy
- Real time facial emotion detection.
- Language Customization.
- Real time speech to text conversion.
- Data privacy

Disadvantages:

- At times the website may lag.
- Model is not tested on a wide set of data set, having all the signs.
- Sign language customization feature is not available.
- User cannot take notes while using the app.
- User cannot make calls using the app.
- Speech recognition works only on google chrome.

11. CONCLUSION

Communication is crucial for self-expression. Additionally, it meets one's necessities. Effective communication is necessary for career advancement. Effective communication skills can make your personal life easier and improve your interactions with others by facilitating mutual understanding. A system that translates speech into acceptable sign language for the deaf and dumb has been developed as part of our project. It also translates sign language into a human hearing voice to communicate with average people. A convolution neural network has been used to build a model that is trained on various hand motions. Utilizing this concept, an app is created. Through the use of signs that are translated into speech and human-understandable English, this software aids deaf and dumb individuals to communicate easily.

12. Future Scope

The following are the features that can be added in our application:

- The accuracy of the model shall be increased.
- Customization of languages shall be added.
- Users shall be allowed to write notes while on call
- Customization of signs can also be added as a feature
- A communication app can be built with the same set of features. The user can choose the appropriate mode (speech to sign or sign to speech) and accordingly the real time detection would take place on both the end users' application

13. Appendix

Source Code

Model Building

```
import cv2 import
os os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'
import numpy as np
 from keras.models
import Sequential
import matplotlib.pyplot as plt
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.layers
import Conv2D, MaxPool2D
from keras_preprocessing.image
 import ImageDataGenerator

test_path = 'Dataset/test_set'
train_path='Dataset/training_set'
train=ImageDataGenerator(rescale=1./255,zoom_range=0.2,shear_
range=0.2,horizontal_flip=True)
test=ImageDataGenerator(rescale=1./255)
train_batches=train.flow_from_directory(directory=train_path,
target_size=(64,64),class_mode=
'categorical',
batch_size=300,shuffle=True,color_mode="grayscale")

test_batches = test.flow_from_directory(directory=test_path,
target_size=(64,64),                class_mode='categorical',
batch_size=300, shuffle=True,color_mode="grayscale")
```

```
model = Sequential()
model.add(Conv2D(32,kernel_size=(3,3),activation='relu',input_shape=(64,64,1))) model.add(MaxPool2D(pool_size=(2,2)))
model.add(Conv2D(512, (3, 3), padding="valid"))
model.add(MaxPool2D(pool_size=(2,2)))
model.add(Conv2D(32, (3, 3), padding="same"))
model.add(MaxPool2D(pool_size=(2,2)))
model.add(Flatten())
model.add(Dense(512,activation ="relu"))
model.add(Dense(9,activation ="softmax"))
model.compile(optimizer='adam',
loss='categorical_crossentropy', metrics=['accuracy'])
history = model.fit(train_batches,
batch_size=32,validation_data=test_batches,epochs=25)
model.save('model.h5')

Model Testing
import keras.models immport load_model
import cv2
import numpu as np
import os
as.environ[TF-CP_MIN_LOG_LEVEL']='2'
val=['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
model=load_,odel('aslpng.h5')
from skimage.transform import resize
def detect(frame):
      img= resize(frame,(64,64,1))
      img= np.expand_dims(img,axis=0))
if(np.max(img)>1);
      img=img/255.0
      predict_x=model.predict(img)
     print(predict_x)
     predtict=np.argmax(predict_x,axis=1)
     x=predict[0]
     print(val[x])
frame=cv2.imread(r"c/C:\Users\GCE\Desktop\ibm\Realtime_Communication_System_For_Specially_Abled\Dataset\test_set\B\1.png
data=detect(frame)
```

```
27
28
29 Flask App Building
30 import numpy as np
31 import os
32 import math
33  import cv2
34  from fer import FER
35  import pyttsx3
36 from keras.models
37 import model_from_json os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'
38  from keras.models import load_model
39 from flask import Flask, render_template, Response, request
40 import tensorflow as tf
41 from cvzone.HandTrackingModule import HandDetector
42 from skimage.transform import resize
43 facecascade=cv2.CascadeClassifier("haarcascade_frontalface_default
   .xml") graph=tf.compat.v1.get_default_graph()
44 writer=None
45 model=load_model('model.h5')
46 font = cv2.FONT_HERSHEY_SIMPLEX
47 vals=['A','B','C','D','E','F','G','H','I']
48 emotion_detector = FER(mtcnn=True)
49 app=Flask( name ,template_folder="template")
50 print("Accessing video stream")
51 app.static_folder = 'static'
52 vs=cv2.VideoCapture(0)
53 detector=HandDetector(maxHands=1)
54  pred="" def SpeakText(command):
55 engine = pyttsx3.init()
56 engine.say(command)
57 engine.runAndWait()
58  def generate_frames():
59           while (vs.isOpened()):
60                   success, frame = vs.read() hands,
61                   frame=detector.findHands(frame)
62                                   dominant_emotion,  emotion_score  =
   emotion_detector.top_emotion(frame)
63 if not success:
```

```
64              break:
65   else:
66      if hands :
67         hand=hands[0]
68
69 x,y,w,h=hand['bbox']
70 imgCrop=frame[y-20:y+h+20,x-20:x+w+20]
71 black=np.ones((300,300,3), np.uint8)*0
72  ishape=imgCrop.shape
73 if h/w>1:
74 k=300/h
75 wcal=math.ceil(k*w)
76 imgresize=cv2.resize(imgCrop,(wcal,300))
77 irshape=imgresize.shape
78 wgap=math.ceil((300-wcal)/2)
1   black[:,wgap:wcal+wgap]=imgresiz
2  else:
3   k=300/w
4  hcal=math.ceil(k*h)
5  imgresize=cv2.resize(imgCrop,(300,hcal))
6  irshape=imgresize.shape hgap=math.ceil((300-hcal)/2)
7  black[hgap:hcal+hgap,:]=imgresize
8  img=resize(black,(64,64,1))
9  img=np.expand_dims(img,axis=0)
10 if(np.max(img)>1):
11 img = img/255.0
12 predict_x=model.predict(img)
13 classes_x=np.argmax(predict_x,axis=1)
14 x=classes_x[0]
15 SpeakText(vals[x])
16 dominant_emotion=str(dominant_emotion)
17  if(dominant_emotion!=""):
18      value=vals[x] +" "+ dominant_emotion
19 else:
20                                                    value=vals[x]
   cv2.putText(frame,value,(x+20,y+20),cv2.FONT_HERSHEY_SIMPLEX,
   1,(255, 255, 150),2,cv2.LINE_AA)
21 ret, buffer = cv2.imencode('.jpg', frame)
22  frame = buffer.tobytes()
23  yield (b'--frame\r\n' b'Content-Type: image/jpeg\r\n\r\n' + frame
```

```
    + b'\r\n')
@app.route('/')
def index():
    return render_template('index.html');
    @app.route('/sign_to_speech')
     def sign_to_speech();
          return render_template('sign_to_speech.html')
     @app.route('speech_to_sign')
          def speech_to_sign():
             return render_template('speech_to_sign.html')
@app.route('/video',methods=['GET', 'POST'])
def video():
return Response(generate_frames(),
mimetype='multipart/x-mixed-replace; boundary=frame')
if ( name == " main "):
app.run(debug=True)


HTML Files

index.html

<html>
<head>
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<meta    name="viewport"    content="width=device-width,    initial-
scale=1">
<script
src="http://translate.google.com/translate_a/element.js?cb=loadGoo
gleTranslate"
></script>
<script>
function loadGoogleTranslate()
{
new google.translate.TranslateElement("google_element")
}
</script>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min
```

```
   .js"></script>
27 <script
28 src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.
   min.js"></scri
29 pt>
30 <link rel="stylesheet"
31 href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstra
   p.min.css">
32 <style>
33 .row {
34 display: flex;
35 }
36 .col {
37 flex: 50%;
38 }
39 *,*:after,*:before{
40 -webkit-box-sizing: border-box;
41 -moz-box-sizing: border-box;
42 -ms-box-sizing: border-box;
43 box-sizing: border-box;
44 }
45 body{
46 font-family: arial;
47 font-size: 16px;
48 margin: 0;
49 color: #000;
50 display: flex;
51 min-height: 100vh;
52 }
53 .voice_to_text{
54 width: 600px;
55 text-align: center;
56 }
57 h1{
58 color: #000000;
59 font-size: 50px;
60 }
61 #convert_text{
62 width: 100%;
63 height: 200px;
```

```
border-radius: 10px;
resize: none;
padding: 10px;
font-size: 20px;
margin-bottom: 10px;
}
button{
align-items: center;
justify-content: center;

padding: 12px 20px;
background: #0ea4da;
border: 0;
color: #fff;
font-size: 18px;
cursor: pointer;
border-radius: 5px;
}
</style>
</head>
<body>
<div class="container">
<div class="row">
<div class="col">
<img src="https://img.freepik.com/free-vector/sign-language-alphabet-hand⊠drawn-style_23-2147872270.jpg?w=2000" style="width:50%;"/>
</div>
<div class="col">
<div class="voice_to_text">
<div class="text_center" id="google_element">
</div>
<h1>Voice to text converter</h1>
<textarea name="" id="convert_text"></textarea>
<button id="click_to_record" class="btn-primary">Voice to Text</button><br/>
<a href="/">
<button class="btn btn-danger btn-lg">Exit</button>
</a>
</div>
```

```
</div>
</div>
</div>
<script                type="text/javascript"                src="{{
url_for('static',filename='javascript/script.js') }}">
</script>
</body>
</html> Sign_to_speech.html
<html>
<head>
<style>
img{
display: block;
margin-left: auto;
margin-right: auto;
}
</style>
<script>
function loadGoogleTranslate()
{
new google.translate.TranslateElement("google_element")
}
</script>
<script
src="http://translate.google.com/translate_a/element.js?cb=loadGoo
gleTranslate"
></script>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min
.js"></script>
<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.
min.js"></scri pt>
<link                                          rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstra
p.min.css"/>
</head>
<body>
<h1>Sign to speech</h1>
<div>
```

```
<div class="text-center" id="google_element"></div>
<img src="{{ url_for('video') }}" width="50%" height="50%"/>
<br/>
<div class="text-center">
<a href="/">
<button class="btn btn-danger btn-lg" >Exit</button>
</a></div>
</div>
</body>
</html>
CSS Files
Index.css
@import
   url("https://fonts.googleapis.com/css2?family=Oxygen:wght@400;700&
   family= Roboto:wght@300;900&display=swap");
* {
box-sizing: border-box;
padding: 0;
margin: 0;
}
:root {
--black: #000;
--white: #fff;
--hover: #000;
}
.main {
position: relative;
height: 100vh; width: 100%;
display: flex;
align-items: center;
 justify-content: center;
 }
 .inside {
 position: relative;
 height: 60%;
 width: 50%;
 background: rgba(255,255,255,0.9);
 border-radius: 30px;
 /* border: 5px solid var(--black); */
display: flex;
```

```
11 align-items: center;
12 justify-content: space-evenly;
13 -webkit-box-shadow: 12px 12px 17px 1px rgba(0, 0, 0, 0.59);
14 -moz-box-shadow: 12px 12px 17px 1px rgba(0, 0, 0, 0.59);
15 box-shadow: 12px 12px 17px 1px rgba(0, 0, 0, 0.59);
16 }
17 .wrapper { position: relative;
18 height: 75%;
19 width: 30%;
20 display: flex;
21 align-items: center;
22 justify-content: space-evenly;
23 flex-direction: column;
24 }
25 .Head {
26 position: relative;
27 font-size: 3rem;
28 text-transform: uppercase;
29 font-family: "Roboto", sans-serif;
30 font-weight: 900;
31 display: flex;
32 align-items: center;
33 justify-content: center;
34 flex-direction: column;
35 height: 30%;
36 }
37 .Head h1 {
1  font-size: 3rem;
2  }
3  .Head span { position: relative;
4  height: 5px;
5  width: 60%;
6  background: var(--black);
7  }
8  .box {
9  position: relative;
10 font-family: "Oxygen", sans-serif;
11 font-weight: 700;
12 border: 2px solid var(--black);
13 border-radius: 1.5rem;
```

```
text-decoration: none;
overflow: hidden;
cursor: pointer;
z-index: 1;
}
.box1 {
padding: 0.8rem 2rem;
}
.box2 {
padding: 0.8rem 1.5rem;
}
.box:hover {
color: var(--white);
background: var(--hover);
}
Javascript Files
Script.js
click_to_record.addEventListener('click',function(){
var speech = true;
const SpeechRecognition = window.speechRecognition || window.webkitSpeechRecognition;
const recognition =new SpeechRecognition();
recognition.interimResults = true;
recognition.addEventListener('result', e => {
const transcript = Array.from(e.results)
.map(result => result[0])
.map(result => result.transcript)
.join('')
document.getElementById("convert_text").innerHTML = transcript;
console.log(transcript);
});
if (speech == true) {
recognition.start();
}
})
```

Output

13.2 Github and Demo Link:

https://github.com/IBM-EPBL/IBM-Project-37034-1660299814

Demo video
https://drive.google.com/file/d/1yZWcMwuyT91wNDCUb6NYuQYSwOsbOHQH/view?usp=sharing