PROJECT REPORT

Real-Time Communication System Powered by AI for Specially Abled

submitted by

PNT2022TMID49261

Aheleswaran E - 922319106002

Ragul pandiyan K - 922319106020

Raaghul R - 922319106019

Priyadharsan B - 922319106018

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION

1.1 Pre-Requisites

Machine learning and deep learning play an important role in computer technology and artificial intelligence. With the use of deep learning and machine learning, human effort can be reduced in recognizing, learning, predictions and in many more areas. 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.

1.2 PURPOSE

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.

CHAPTER 2 LITERATURE SURVEY

2.1 EXISTING PROBLEM

Many of us think that artificial intelligence represents an abstract and futuristic notion we only see in sci-fi films with humanoid robots and holograms. However it’s more and more grounded in our reality reaching various fields and categories of people including people with disabilities. Artificial intelligence truly revolutionizes accessibility and inclusion! Thanks to AI technology solutions, people with disabilities can drastically improve their everyday lives. We

had previously seen that smartphones are a powerful tool that help users with a visual impairment. Indeed, many apps enable them to remain autonomous. For example, thanks to Seeing AI, visually impaired people can easily read their mail by placing documents under the smartphone camera. AI technology can apply to any type of disability profile. For instance, people with reduced mobility can control everything at home just by using their voice with a virtual personal assistant such as Amazon Alexa. Let's take a look at AI and how it can enhance accessibility thanks to a few examples of innovative solutions! The future starts now!

2.2 EXISTING SOLUTION

Google Maps: one of the most used GPS apps around the world. Visually impaired people or wheelchair users can prepare their trip in advance and visualize their route and the best means of transportation to use according to their profile. Thanks to the "wheelchair accessible" option, wheelchair users can know where ramps and elevators are located in the city. Plus the feature "accessible places" is useful for them to have more information about the layout of many premises: entrance, parking spots, restrooms, seating arrangements… This feature is also used by people with a visual impairment to find the exact location of a building entrance**.**

Soundscape: an app that describes blind people their surroundings with audio 3D technology. They can easily be aware of the points of interest near them and the intersections. Quite convenient to enjoy the city.

Wheelmap: it lists and maps all accessible public venues (restaurants, shops, cafés…). Even users can add data and information concerning the accessibility level of places.

2.3 PROBLEM STATEMENT DEFINITION

- ❖ 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.
- ❖ That 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.

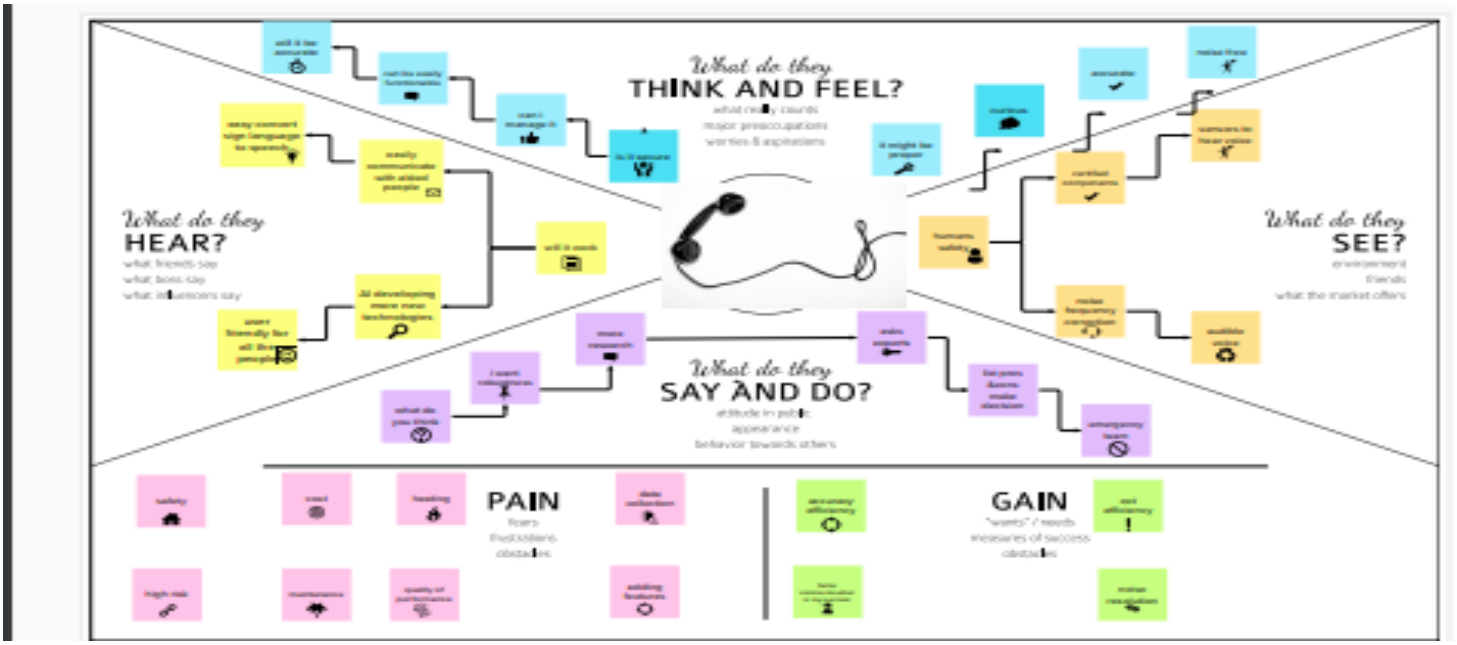
❖ This app enables deaf and dumb people to convey their information using signs which get converted to human-understandable language and speech is given as output.

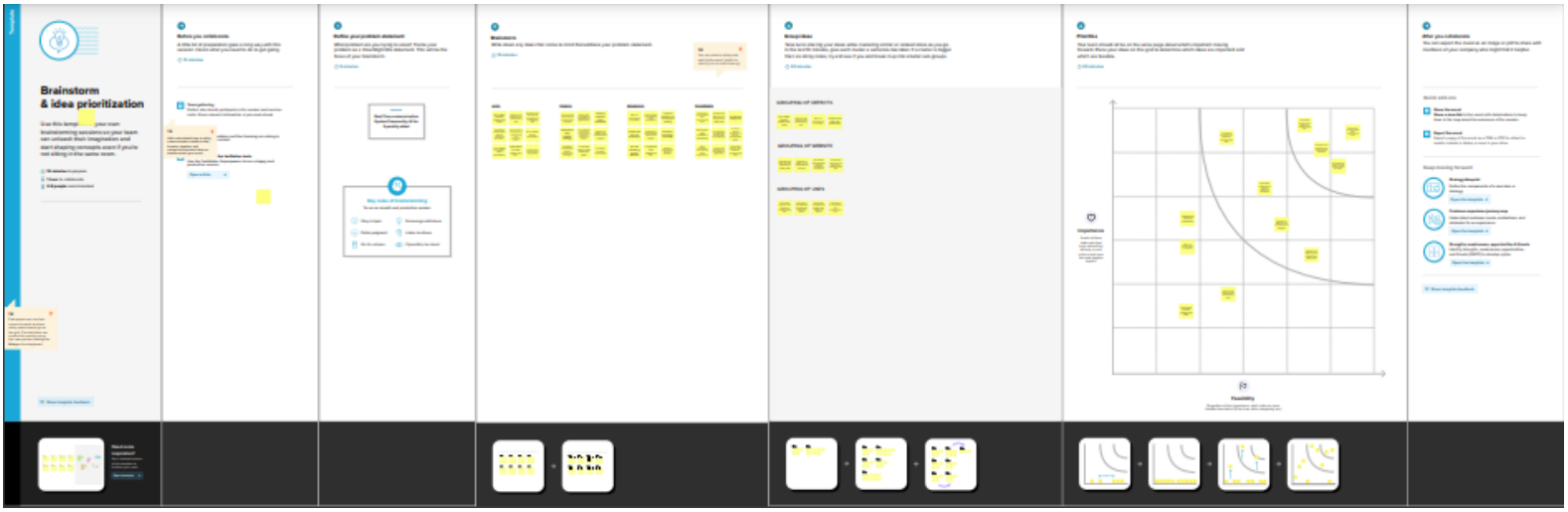
❖ Voice Conversion System with Hand Gesture Recognition and translation will be very useful to have a proper conversation between a normal person and an impaired person in any language.

CHAPTER 3 IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

3.2 IDEATION & BRAINSTORMING





3.3 PROPOSED SOLUTION

IDEA / SOLUTION DESCRIPTION :

❖ That converts the sign language into a human hearing voice.

❖ Convolution Neural Network to create a model that is trained on different hand gestures.

NOVELTY / UNIQUENESS :

❖ 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 by AI . SOCIAL

IMPACT / CUSTOMERS SATISFACTION :

❖ To overcome lack of communication between Mute and Normal People .

❖ It is easy to use in an emergency Situation .

BUSINESS MODEL (FINANCIAL BENEFIT) :

❖ Low power consumption

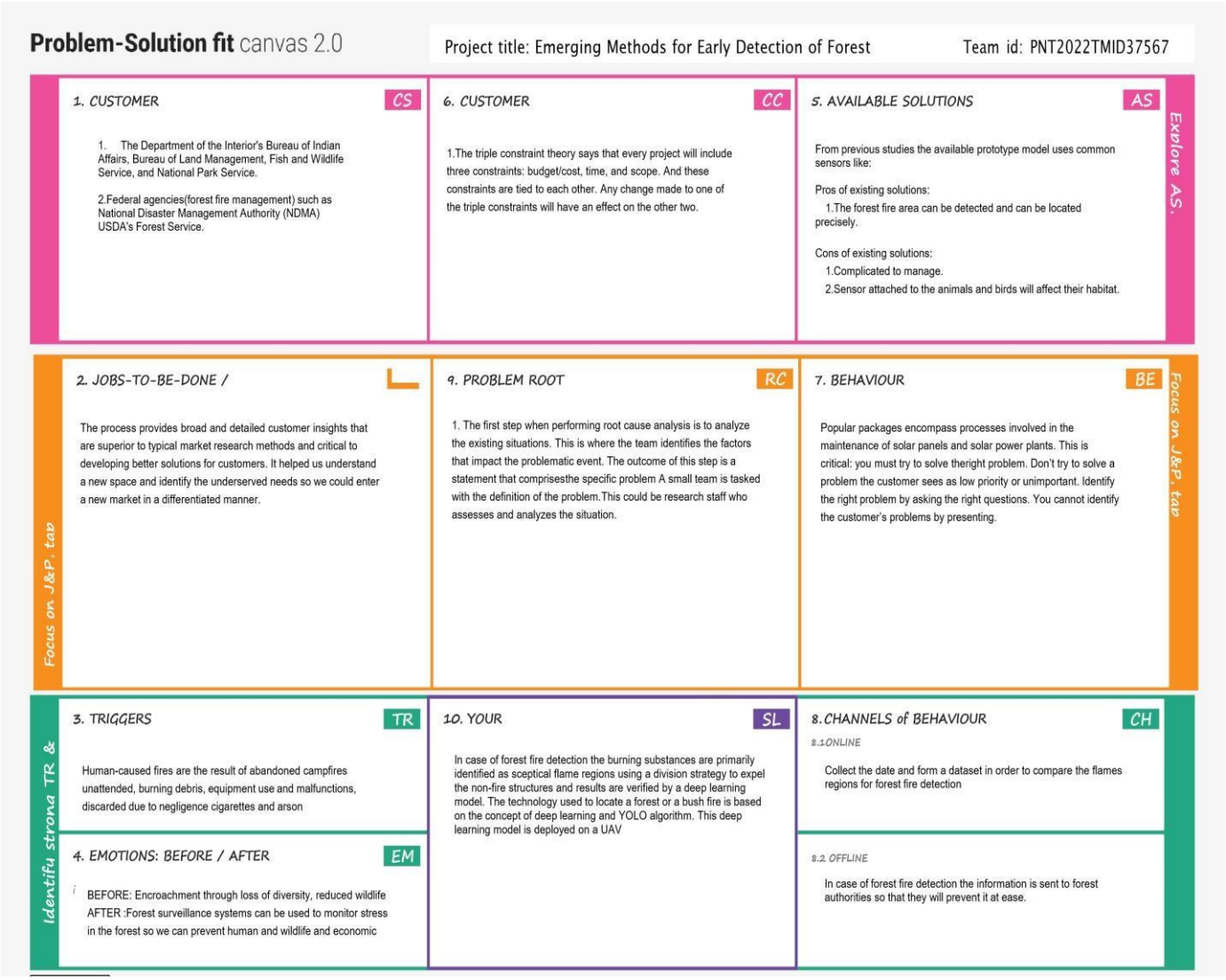
❖ Cost effective

❖ Easy portable

SCALABILITY OF SOLUTION :

❖ Scalable

3.4 PROBLEM SOLUTION FIT



CHAPTER 4 REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

FUNCTIONALREQUIREMENTS:

-Following are the functional requirements of the proposed solution

Sn. No	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
1.	User Registration	Registration through G-mail.
2.	User Confirmation	Confirmation through OTP. Confirmation through mail.
3.	System	• Desktop/ Mobile with good resolution camera. • Provides system access to capture images/ video and other relevant data.
4.	Text conversion	Converts the Sign language into a text using Convolutional Neural Network (CNN) Model.
5.	Sentence Translation	To create sentence(s) by recognizing the signs and pauses in the input video stream.

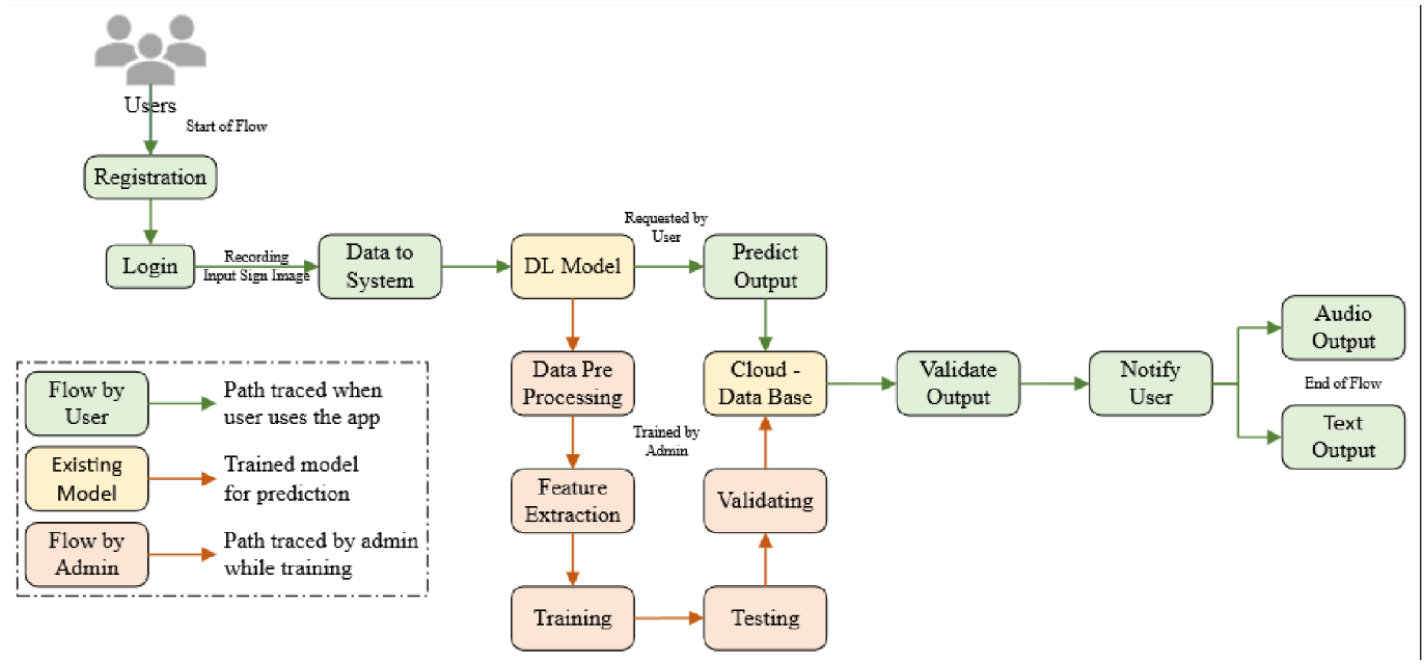
NON-FUNCTIONALREQUIREMENTS:

-Following are the non-functional requirements of the proposed solution.

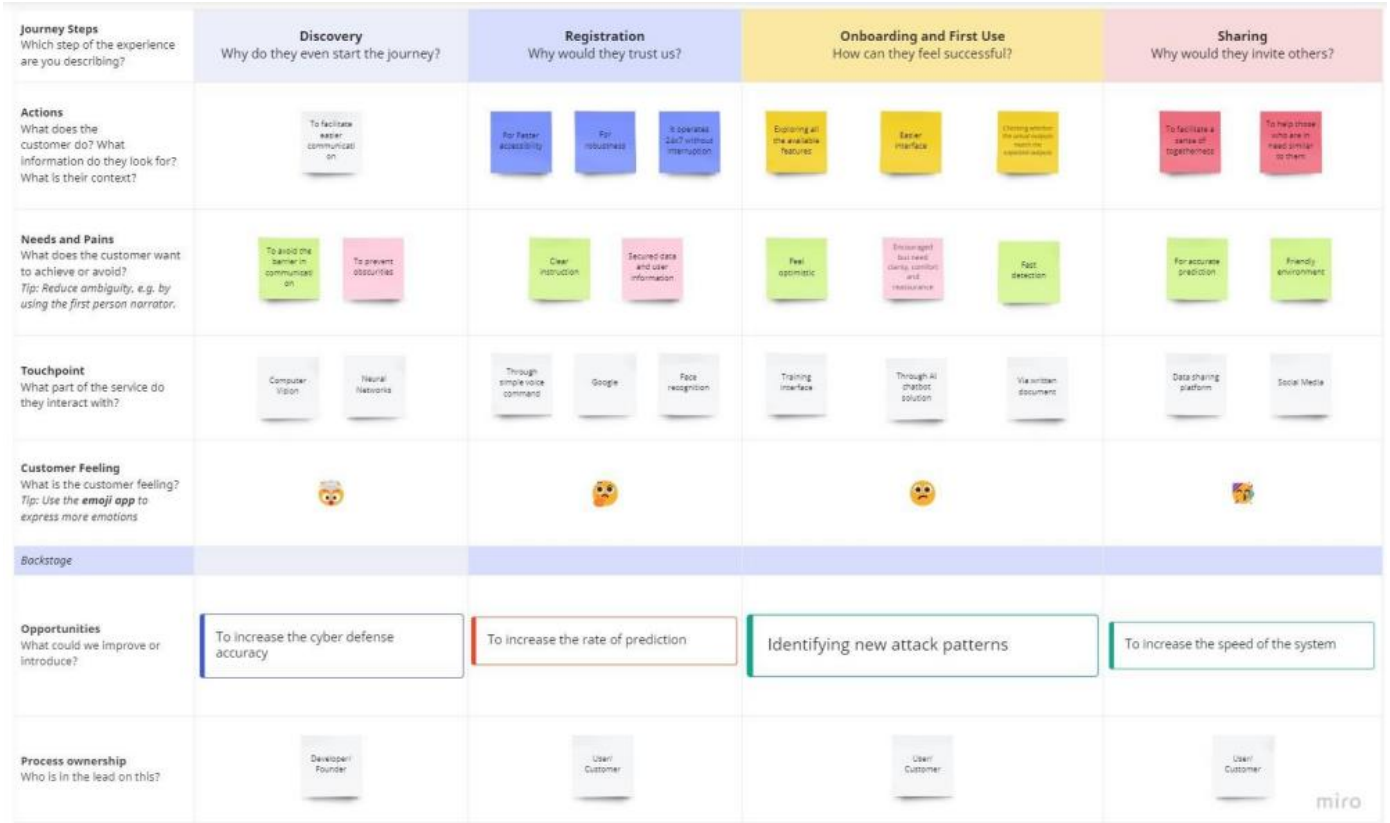
Sn. No.	Non-Functional Requirement	Description
1.	Usability	Deaf-mute people should be able to use the system with ease. The same applies for normal people who get the system's output. The system should have good UI.
2.	Security	Even though the use-case of the system doesn't need any security feature, it must be ensured that the privacy of user data be maintained and handled appropriately.
3.	Reliability	The translation of sign languages should be reliable. The accuracy of the system should be tested extensively to make sure that it is up to the mark
4.	Performance	The processing should be done in considerable time so that the conversation can go on without waiting for the system's output.
5.	Availability	The system should be universally accessible. Since sign language is almost same everywhere, the system can be used across the globe.
6.	Scalability	The system should be scalable to accommodate new features and functionalities and to cater wider range of people in future.

CHAPTER 5 PROJECT DESIGN

Data Flow Diagram



5.2 SOLUTION & TECHNICAL ARCHITECTURE



5.3 USER STORIES

User Stories:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance c
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my acco dashboard
	Authentication	USN-2	As a user, I will receive confirmation email once I have registered for the application.	I can receive confirm click confirm
	Login	USN-3	As a user, I can log into the application by entering email & password.	I am able to get into Dashboard
	Dashboard	USN-4	One place to explore all available features.	I can access my dash
Customer (Web user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my acco dashboard
	Authentication	USN-2	As a user, I will receive confirmation email once I have registered for the application.	I can receive confirm click confirm.
	Login	USN-3	As a user, I can log into the application by entering email & password	I am able to get into Dashboard
	Dashboard	USN-4	One place to explore all available features	I can access my dash
	Upload image	USN-5	As a user, I can upload the sign language image for translating into text format	I can be able to see t text for the sign lang
Administrator	Manage	USN-6	Do-it-yourself service for delivering Everything.	Set of predefined rec must be met to mark story complete

CHAPTER 6 PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a User, I can register for the application by entering my email, password, and confirming my password.	2	High	AHELESWARAN E
Sprint-1	User Confirmation	USN-2	As a User, I will receive confirmation email once I have registered for the application.	1	Medium	PRIYADHARSAN B
Sprint-1	Login	USN-3	As a User, I can log into the application by entering email & password.	2	High	RAAGHUL R
Sprint-2	Interface Sensor	USN-1	A sensor interface is a bridge between a device and any attached sensor. The interface takes data collected by the sensor and outputs it to the attached device.	2	High	RAGULP ANDIYA N K
Sprint-3	Coding (Accessing datasets)	USN-1	Coding is a set of instructions used to manipulate information so that a certain input results in a particular output.	2	High	AH EL ES W AR AN E RA AG HU L R
Sprint-4	Web Application	USN-1	As a user, I will show the current Information of the Forest.	1	Medium	PRI YA DH AR SA N B RA GU LP AN DI YA N K

6.2 SPRINT DELIVERY SCHEDULE

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	27 Oct 2022	20	29 Oct 2022
Sprint-2	20	5 Days	28 Oct 2022	01 Nov 2022	20	04 Nov 2022
Sprint-3	20	8 Days	02 Nov 2022	09 Nov 2022	20	11 Nov 2022

Sprint-4	20	9 Days	10 Nov 2022	18 Nov 2022	20	19 Nov 2022

CHAPTER 7 CODING & SOLUTIONING

```
from flask import Flask, Response, render_template
import cv2


app = Flask(__name__)
cap = cv2.VideoCapture(0)
@app.route('/')
def index():
    return  render_template('index.html')

def generate_frames():
    while True:
        success, frame = cap.read()
        imgOutput=frame.copy()
        yield (b'--frame\r\n'
               b'Content-Type: image/jpeg\r\n\r\n' + imgOutput + b'\r\n')
@app.route('/predict',methods=['POST','GET'])
def predictions():
    #The prediction model code goes here
    #Once the start Button is pressed the prediction model starts
    pass
@app.route('/stop',methods=['POST','GET'])
def stopping():
    #The text to speech code goes here
    #Once the stop button is pressed the text is converted into speech
    pass

@app.route('/video')
def video():
    return Response(generate_frames(),mimetype='multipart/x-mixed-replace; boundary=frame')


if __name__ == '__main__':
    app.run(debug=True)
```

```
import cv2
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import os

class Video(object):
    def __init__(self):
        self.video = cv2.VideoCapture(0)
        self.roi_start = (50, 150)
        self.roi_end = (250, 350)
        #self.model = load_model('asl_model.h5') # Execute Local Trained Model
        self.model = load_model('realtime.h5') # Execute IBM Trained Model
        self.index=['A','B','C','D','E','F','G','H','I']
        self.y = None
    def __del__(self):
        k = cv2.waitKey(1)

        self.video.release()
    def get_frame(self):
        ret,frame = self.video.read()
        frame = cv2.resize(frame,(640,480))
        copy = frame.copy()
        copy = copy[150:150+200,50:50+200]
        # prediction starts
        cv2.imwrite('image.jpg',copy)
        copy_img = image.load_img('image.jpg', target_size=(64,64,3))
        x = image.img_to_array(copy_img)
        x = np.expand_dims(x, axis=0)
        pred = np.argmax(self.model.predict(x), axis=1)
        self.y = pred[0]
        cv2.putText(frame,'The Predicted Alphabet is:
'+str(self.index[self.y]),(100,50),cv2.FONT_HERSHEY_SIMPLEX,1,(0,0,0),3)
        ret,jpg = cv2.imencode('.jpg', frame)
        return jpg.tobytes()
```

CHAPTER 8

TESTING

CHAPTER 8 RESULTS

Final output

```
Import datagenerator to train and test
from tensorflow.keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale = 1./255,shear_range=0.2,zoom_range=
0.2,horizontal_flip=True,vertical_flip=False)
test_datagen = ImageDataGenerator(rescale = 1./255)

import tensorflow as tf
import os
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Conv2D, Flatten, Dropout, MaxPooling2D
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import numpy as np
import matplotlib.pyplot as plt
```

```
import IPython.display as display
from PIL import Image
import pathlib
from keras.preprocessing import image
# and use
# image.ImageDataGenerator()
# image.load_img()
Apply ImageDataGenerator Functionality To Train And Test set
from google.colab import drive
from tensorflow.keras.preprocessing.image import ImageDataGenerator

x_train= train_datagen.flow_from_directory(r"/content/drive/MyDrive/IBM PROJECT/DATA COLLECTION/training_set",target_size=(64,64),class_mode="categorical",batch_size=48)
x_test = test_datagen.flow_from_directory(r"/content/drive/MyDrive/IBM PROJECT/DATA COLLECTION/test_set",target_size= (64,64),class_mode= "categorical",batch_size=48)
x_train.class_indices
x_test.class_indices
MODEL BUILDING
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Convolution2D
from tensorflow.keras.layers import Conv2D, MaxPooling2D
from keras.layers import Dropout
from keras.layers import Flatten
model=Sequential()

model.add(Convolution2D(32,(3,3), input_shape=(64,64,1), activation = 'relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
model.add(Dense( units=512, activation='relu'))
model.add(Dense(units=9, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model.save('Realtime.h5')
```

```
a=len(x_train)
b=len(x_test)
Length of training and testing data
print(a)
print(b)
TEST THE MODEL



from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np
import cv2
img = image.load_img('/content/drive/MyDrive/IBM PROJECT/DATA
COLLECTION/test_set/D/101.png',target_size = (500,500))
img
from skimage.transform import resize
arr=image.img_to_array(frame)
arr = resize(arr,(64,64,1))
arr = np.expand_dims(arr,axis=0)
pred=np.argmax(model.predict(arr))
op=['A','B','C','D','E','F','G','H','I']
print("THE PREDICTED LETTER IS ",op[pred])
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):
    prediction=model.predict(img)
    print(prediction)
    prediction=model.predict_classes(img)
    print(prediction)
arr= image.img_to_array(img)
frame=cv2.imread('/content/drive/MyDrive/IBM PROJECT/DATA COLLECTION/test_set/F/107.png')
```

```
data=detect(frame)
from google.colab.patches import cv2_imshow
cv2_imshow(frame)
cv2.waitKey(0)
cv2.destroyAllWindows()
frame=cv2.imread('/content/drive/MyDrive/IBM PROJECT/DATA COLLECTION/test_set/A/102.png')
data=detect(frame)
from google.colab.patches import cv2_imshow
cv2_imshow(frame)
cv2.waitKey(0)
cv2.destroyAllWindows()

frame=cv2.imread('/content/drive/MyDrive/IBM PROJECT/DATA COLLECTION/test_set/D/108.png')
data=detect(frame)
from google.colab.patches import cv2_imshow
cv2_imshow(frame)
cv2.waitKey(0)
cv2.destroyAllWindows()
print("THE PREDICTED LETTER IS ",op[pred])
```

CHAPTER 10 ADVANTAGES & DISADVANTAGES

ADVANTAGES

The system detects languages into audio that's helps the people to communicate faster and easier .Thats make the difficulties shorter

DISADVANTAGES

Its take much to complete the process

CHAPTER 11 CONCLUSION

Communication plays a significant role in making the world a better place. Communication creates bonding and relations among the people, whether persona, social, or political views. Most people communicate efficiently without any issues, but many cannot due to disability. They cannot hear or speak, which makes Earth a problematic place to live for them. Even simple basic tasks become difficult for them. Disability is an emotive human condition. It limits the individual to a certain level of performance. Being deaf and dumb pushes the subject to oblivion, highly introverted. In a world of inequality, this society needs empowerment. Harnessing technology to improve their welfare is necessary. In a tech era, no one should be limited due to his or her inability. The application of technology should create a platform or a world of equality despite the natural state of humans. On the other hand, technology is the most innovative thing on Earth for every time the clock ticks, researchers, software engineers, programmers, and information technology specialists are always coming up with bright ideas

to provide convenience to everyone. This paper shows how artificial intelligence is being used to help people who are unable to do what most people do in their everyday lives. Aligned with communication, D-talk is a system that allows people who are unable to talk and hear be fully understood and for them to learn their language easier and also for the people that would interact and communicate with them. This system provides detailed hand gestures that show the interpretation at the bottom so that everyone can understand them. This research allows the readers to learn the system and what it can do to people who are struggling with what they are not capable of and will provide the technical terms on how the system works.

APPENDIX

SOURCE CODE

```
<html>

 <head>
   <link rel="stylesheet" href={{ url_for('static', filename='css/style.css') }}>
 </head>
 <body>
```

```
    <h2 class="header">Sign Language TO Speech</h2>
    <div class="video">
        <img src="{{ url_for('video') }}" width="50%">
    </div>
    <div class="container">
    <form action='/predict' method='post'>
    <button type="submit" name="start" value="start" class="button1" >Start</button>
    </form>
  <form action='/stop' method='post'>

    <button type="submit" name="stop" value="stop" class="button2" >Stop</button>
  </form>
  </div>
  <div class="instruction">
    <center>
    <p> <details>
      <summary><b>Instructions to Use</b></summary>
      <p>Once the webcam is <b>ON</b> Click <strong>"START"</strong> to start the predicition
model.<br>
      <br>
      >>Click <strong>"s"</strong> to save the text.<br>
      <br>
      >>Click <strong>"a"</strong> to leave a space.<br>
      <br>
      >>Click <strong>"d"</strong> to delete a character from right to left.<br>
      <br>
      >>Click <strong>"w"</strong> to delete entire text.<br>
      <br>
      >> The Saved text appears on the top left corner of the video Screen<br>
      <br>
      >> Once you are satisfied with the saved text press<b>"STOP"</b> to convert it into speech<br>
      <br>
      >><b>NOTE: The hand must be on the screen to display the text to save,delete or to leave a
space between them.</b>

      </p>
    </details></p>
  </center>
</div>
<br>
</center>
</div>
<p>Feel Free to contact us !!!!!</p>
<center>
<div class="alert info">
  <span class="closebtn">&times;</span>
  <strong>NOTE:</strong> A disturbance free background with good lighting(White-background) is
preferred.

</div>
</center>
<script>
  var close = document.getElementsByClassName("closebtn");
  var i;
```

```
  for (i = 0; i < close.length; i++) {
    close[i].onclick = function(){
      var div = this.parentElement;
      div.style.opacity = "0";
      setTimeout(function(){ div.style.display = "none"; }, 600);
    }
  }
  </script>

  <a class="social-icon"href="https://in.linkedin.com/in/sajith-m-82431721a?trk=profile-badge"
target="_blank">
    <ion-icon name="logo-linkedin"></ion-icon>
  </a>
  <a class="git"href="https://github.com/IBM-EPBL/IBM-Project-2475-1658472446.git" target="_blank">
    <ion-icon name="logo-github"></ion-icon>
  </a>



  <script src="https://platform.linkedin.com/badges/js/profile.js" async defer
type="text/javascript"></script>
  <script type="module"
src="https://unpkg.com/ionicons@5.5.2/dist/ionicons/ionicons.esm.js"></script>
  <script nomodule src="https://unpkg.com/ionicons@5.5.2/dist/ionicons/ionicons.js"></script>




  </body>
</html>
```

```
from flask import Flask, Response, render_template
from camera import Video

app = Flask(__name__)
@app.route('/')
def index():
    return render_template('index.html')

def gen(camera):
    while True:
        frame = camera.get_frame()
        yield(b'--frame\r\n'
            b'Content-Type: image/jpeg\r\n\r\n' + frame +
            b'\r\n\r\n')

@app.route('/video_feed')
def video_feed():
    video = Video()
    return Response(gen(video), mimetype='multipart/x-mixed-replace; boundary = frame')
```

```
if __name__ == '__main__':
    app.run()
```

```
import cv2
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import os

class Video(object):
    def __init__(self):
        self.video = cv2.VideoCapture(0)
        self.roi_start = (50, 150)
        self.roi_end = (250, 350)
        #self.model = load_model('asl_model.h5') # Execute Local Trained Model
        self.model = load_model('realtime.h5') # Execute IBM Trained Model
        self.index=['A','B','C','D','E','F','G','H','I']
        self.y = None
    def __del__(self):
        k = cv2.waitKey(1)

        self.video.release()
    def get_frame(self):
        ret,frame = self.video.read()
        frame = cv2.resize(frame,(640,480))
        copy = frame.copy()
        copy = copy[150:150+200,50:50+200]
        # prediction starts
        cv2.imwrite('image.jpg',copy)
        copy_img = image.load_img('image.jpg', target_size=(64,64,3))
        x = image.img_to_array(copy_img)
        x = np.expand_dims(x, axis=0)
        pred = np.argmax(self.model.predict(x), axis=1)
        self.y = pred[0]
        cv2.putText(frame,'The Predicted Alphabet is:
'+str(self.index[self.y]),(100,50),cv2.FONT_HERSHEY_SIMPLEX,1,(0,0,0),3)
        ret,jpg = cv2.imencode('.jpg', frame)
        return jpg.tobytes()
```

```
import cv2

video = cv2.VideoCapture(0)

while True:
    ret, frame = video.read()
    cv2.imshow("Frame", frame)
    k = cv2.waitKey(1)
    if k == ord('q'):
        break

video.release()
cv2.destroyAllWindows()
```

```css
body{
    background-color: #DED9E2;
    font-family: 'Times New Roman', Times, serif;
}
button{
    border-radius: 20%;
    size: 40px;
    text-align: center;
  padding: 15px 25px;
  font-size: 15px;
  cursor: pointer;
  text-align: center;
  text-decoration: none;
  outline: none;
  color: #fff;
  background-color: #A68CEF;
  border: none;
  border-radius: 15px;
  box-shadow: 0 9px #999;
}
button:hover {background-color: #8410f9}

button:active {
  background-color: #8410f9;
  box-shadow: 0 5px #666;
  transform: translateY(4px);
}
form{
    display: inline;
    text-align: center;
}

.container{
    display: block;
    text-align: center;
    width:175px;
    margin:auto;
    padding-top:10px;
}
h2{
    color: #8410f9;
    font-size: 35px;
    text-decoration: underline;
    text-align: center;
    margin-block-start: 0em !important;
    margin-block-end: 0.5em !important;
}
img{
    display:block;
    margin-left: auto;
    margin-right: auto;
    width:40%;
```

```
    border: 10px solid #C0B5DD;
    border-radius: 10px;
    margin-bottom: 10px;

}
.result{
    text-align: center;
    font-size: 20px;
}
details > summary {
  padding: 20px;
  width: 200px;
  background: rgb(255,218,5);
  background-color: #CF9FFF;
  border: none;
  box-shadow: 1px 1px 2px #bbbbbb;
  cursor: pointer;
}

 details > p {
    background-color: white;
    width: 250px;
    padding: 2em;
    margin:0;
    box-shadow: 1px 1px 2px #bbbbbb;
    font-family:'Nunito', Arial ;
    position:relative;
  }
details>summary:hover {background-color: #8410f9}
.social-icon{
  color: rgb(76, 76, 194);
  transition: color 0.2s;
  font-size: 50px;
  text-decoration: none;
}
.social-icon:hover{
  color:blue
}
.git{
  color: rgb(87, 85, 85);
  transition: color 0.2s;
  font-size: 50px;
}
.git:hover{
  color:black;
}
body {
  background-image: url('https://img.freepik.com/free-vector/festive-blurred-lights_53876-
89104.jpg?w=996&t=st=1668019024~exp=1668019624~hmac=938636c2be5447934cf91cc0fbb1647a77ff8d056c6e940c5
fb05aaa86032700');
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-size: 100% 100%;
}
```

```css
.alert {
  padding: 20px;
  background-color: #f44336;
  color: white;
  opacity: 1;
  transition: opacity 0.6s;
  margin-bottom: 15px;
  width:650px
}
.alert.info {background-color: #368faf;}
.closebtn {
  margin-left: 15px;
  color: white;
  font-weight: bold;
  float: right;
  font-size: 22px;
  line-height: 20px;
  cursor: pointer;
  transition: 0.3s;
}
.closebtn:hover {
  color: black;
}
```

Fire.py (Main file)

```python
from flask import Flask, Response, render_template
import cv2
```

```python
app = Flask(__name__)
cap = cv2.VideoCapture(0)
@app.route('/')
def index():
    return  render_template('index.html')

def generate_frames():
    while True:
        success, frame = cap.read()
        imgOutput=frame.copy()
        yield (b'--frame\r\n'
               b'Content-Type: image/jpeg\r\n\r\n' + imgOutput + b'\r\n')
@app.route('/predict',methods=['POST','GET'])
def predictions():
    #The prediction model code goes here
    #Once the start Button is pressed the prediction model starts
    pass
@app.route('/stop',methods=['POST','GET'])
def stopping():
    #The text to speech code goes here
    #Once the stop button is pressed the text is converted into speech
    pass

@app.route('/video')
def video():
    return Response(generate_frames(),mimetype='multipart/x-mixed-replace; boundary=frame')


if __name__ == '__main__':
    app.run(debug=True)
```

GITHUB

https://github.com/IBM-EPBL/IBM-Project-32032-1660207584