Real-Time Communication System Powered By AI For Specially Abled

Team ID:PNT2022TMID49307

TEAM LEAD **:** NIVETHA M S
TEAM MEMBER 1 : NANDHANA G
TEAM MEMBER 2 : PRIYADHARSHINI M
TEAM MEMBER 3 : MOHANRAJ K
TEAM MEMBER 4 : SETHUPATHI A

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.

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.

2.LITERATURE SURVEY

2.1 EXISTING PROBLEM

This project 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.

2.2 REFERENCE

1. Artemov, M., Voronov, V., Voronova, L., Goncharenko, A., & Usachev, V. (2019). Subsystem for Simple Dynamic Gesture Recognition Using 3DCNNLSTM. In Conference of Open Innovations Association, FRUCT (No. 24, pp. 571-577). FRUCT Oy.

2. Gomes, S. L., Rebouças, E. D. S., Neto, E. C., Papa, J. P., de Albuquerque, V. H., Rebouças Filho, P. P., & Tavares, J. M. R. (2017). Embedded real-time speed limit sign recognition using image processing and machine learning techniques. Neural Computing and Applications, 28(1), 573-584.

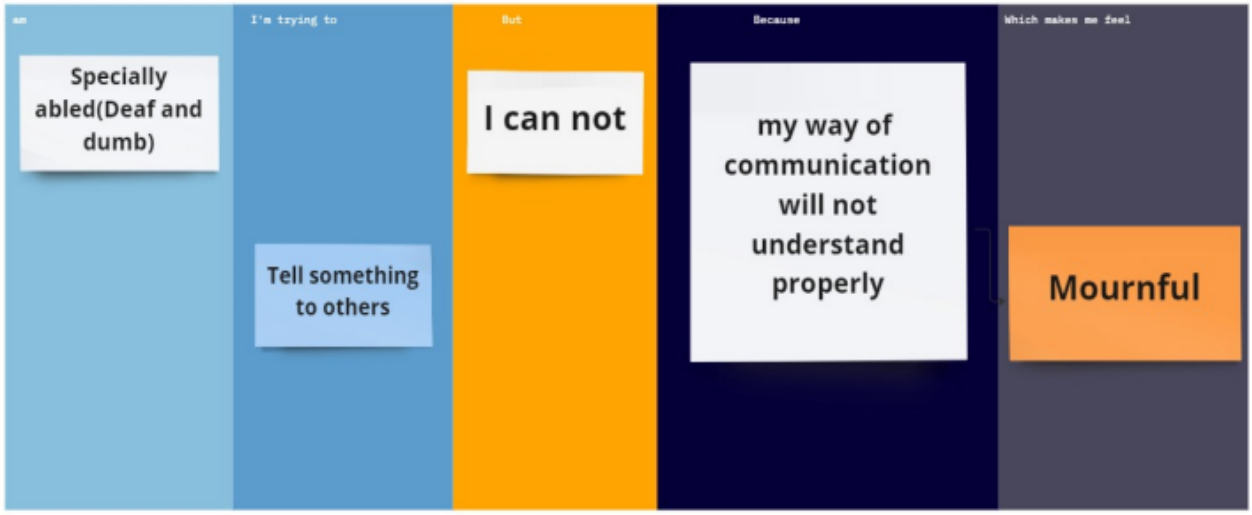
3. Anderson, R., Wiryana, F., Ariesta, M. C., & Kusuma, G. P. (2017). Sign language recognition application systems for deaf-mute people: A review based on input-process-output. Procedia computer science, 116, 441-448

4. Hiele, T. M., Widjaja, A. E., Chen, J. V., & Hariguna, T. (2019). Investigating students' collaborative work to continue to use the social networking site. International Journal of Advanced Trends in Computer Science and Engineering, 8(1.5 Special Issue), 375-386 https://doi.org/10.30534/ijatcse/2019/6181.52019

5. S. Pisupati and M. Ismail, Image Registration Method for Satellite Image Sensing using Feature based Techniques, IJATCSE, Vol. 9. № 1, 2020, pp. 590–593. DOI: https://doi.org/10.30534/ijatcse/2020/82912020.

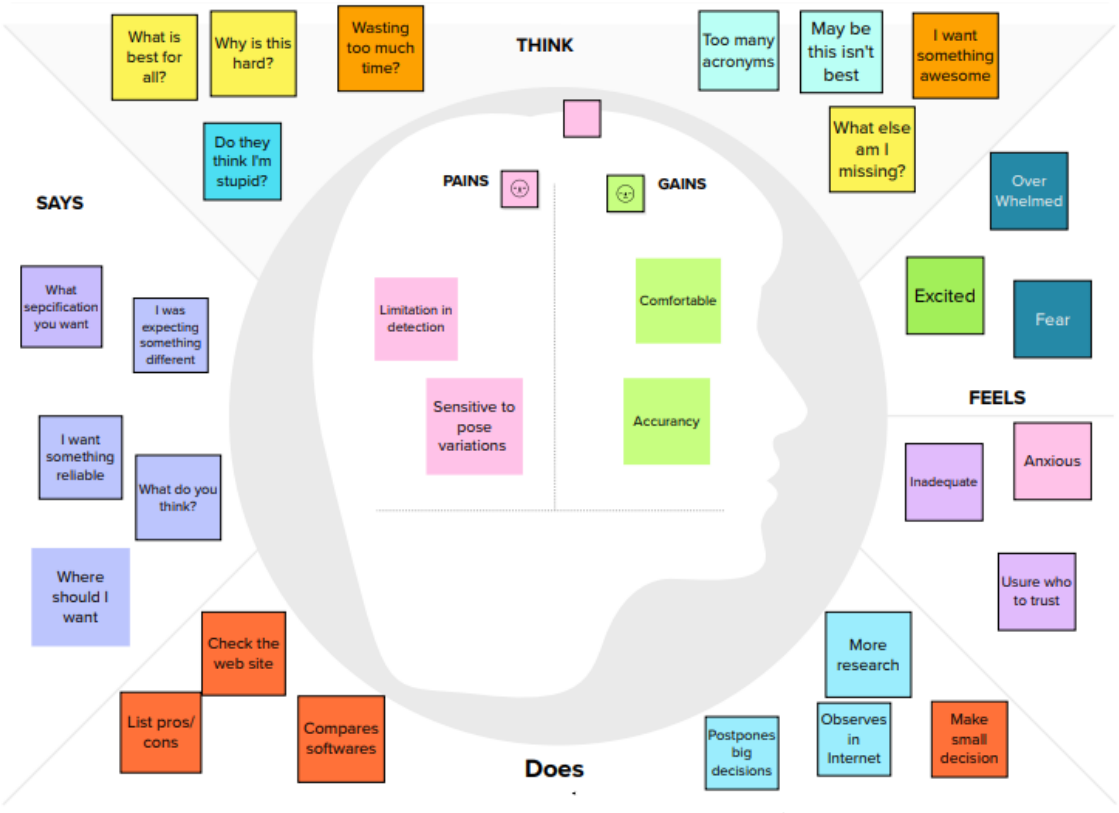
2.3 PROBLEM STATEMENT DEFINITION

Communication between deaf -mute an da normal person has always been a challenging task. This could be solved by our application.

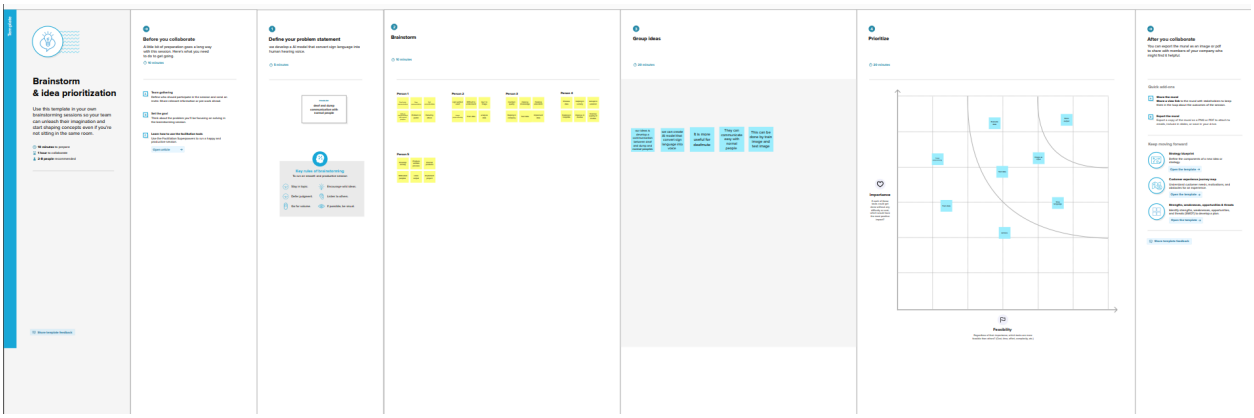


3. IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



3.2 IDEATION & BRAINSTORMING



3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Communication between deaf -mute an da normal person has always been a challenging task. This could be solved by our application.
2.	Idea / Solution description	Converting sing language into human hearing voice.
3.	Novelty / Uniqueness	We are making use of a convolution neural network to create a model that is trained on different hand gestures.
4.	Social Impact / Customer Satisfaction	The main purpose of this application is to make deaf-mute people feel independent an more confident.
5.	Business Model (Revenue Model)	Can generate revenue through direct customer and collaborate with health care sector and generate revenue from their customer.
6.	Scalability of the Solution	This app enables deaf -mute people to convey their information using signs which get converted to human understanding language and speech is given as output.

3.4 PROBLEM SOLUTION FIT

Project Design Phase-I - Solution Fit Template

Project Title: Real Time communication system Powered by AI for specially abled

Team ID: PNT2022TMID49307

Define CS, fit into CC

1.CUSTOMER SEGMENTS CS

Specially abled (deaf and dumb).

6.CUSTOMER CONSTRAINTS CC

Network issues,
User interfaces,
Spending time.

5. AVAILABLE SOLUTIONS AS

Our project convert the sign language to human hearing voice.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

We develop an AI model that convert sign language into a speech that can be understood by normal people.

9. PROBLEM ROOT CAUSE RC

Communication between deaf mute and normal people is challenging task.

For example:
In job interview ,they have trouble to convey interviewer can not understand her language.

7. BEHAVIOUR BE

To find the right application , calculate the efficiency.
To measure the speed of the application .

Focus on J&P, tap into BE, understand RC

3. TRIGGERS TR

Trust

Our model will more efficient to use. Use browse about the model.

4. EMOTIONS: BEFORE / AFTER EM

Before:

They can not communicate with normal people. It's a challenging task.

After:

They can better communication with normal people.

10. YOUR SOLUTION SL

Our solution is simple, better understanding to the normal people about the deaf mute.

8.CHANNELS OF BEHAVIOUR CH

Online :

Extract application from online.

Offline :

Use the application.

4.REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

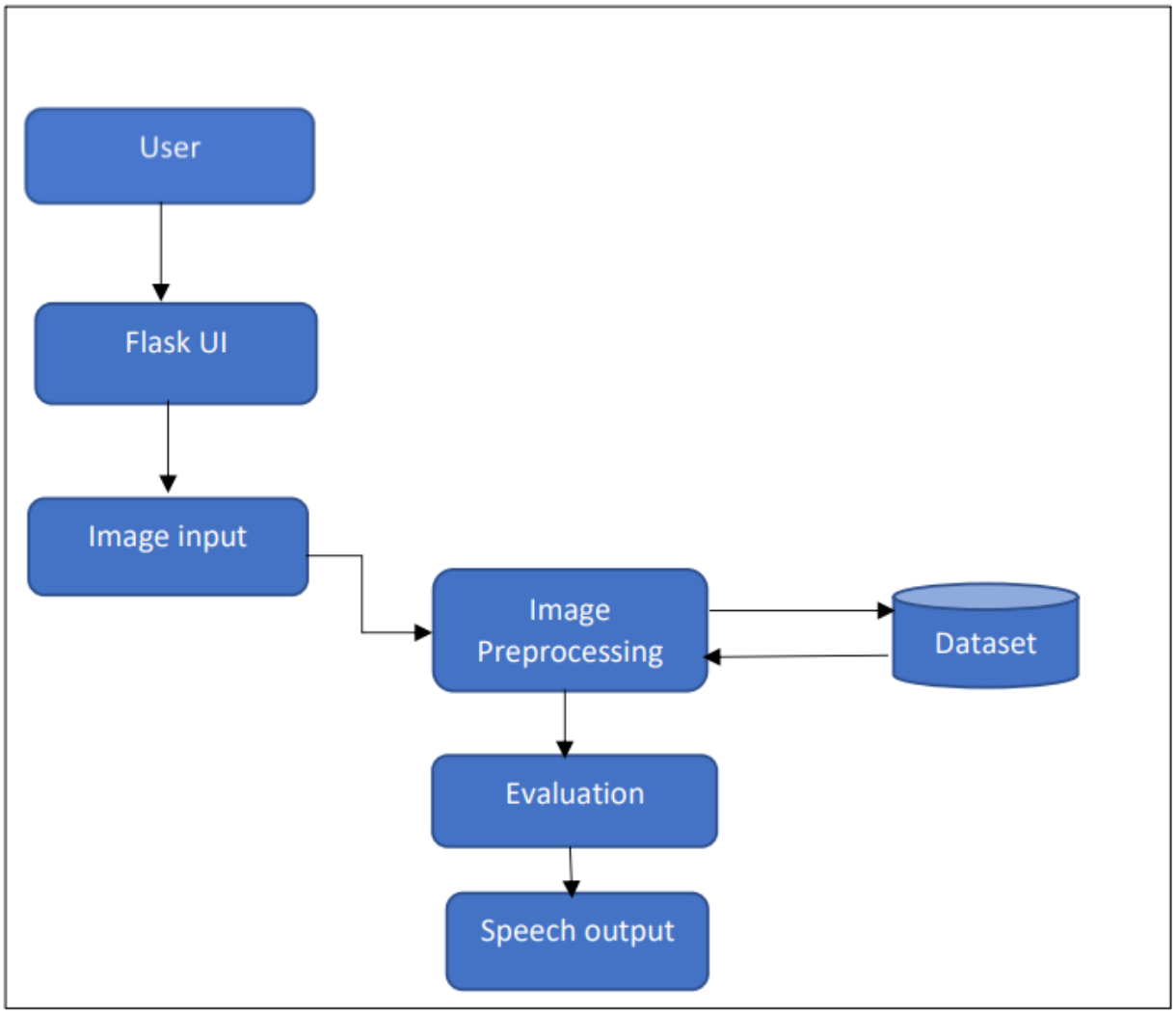
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Camera Enable	Enabling camera on mobile
FR-4	Hand sign	Showing different hand sign
FR-5	Speech output	Hearing voice output

4.2 NON-FUNCTIONAL REQUIREMENT

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	It's is more usable for dead-mute as well as normal people.
NFR-2	**Security**	It describes the tools and strategies that leverage AI to identify, prevent and respond to emerging cyber threats.
NFR-3	**Reliability**	Maintain data privacy and security. Keep the human in the loop.
NFR-4	**Performance**	The more the model's predictions are the same as the true values the higher is the performance of the model.
NFR-5	**Availability**	Operate satisfactorily at a given point in time.
NFR-6	**Scalability**	Data model and infrastructure to operate at the size.

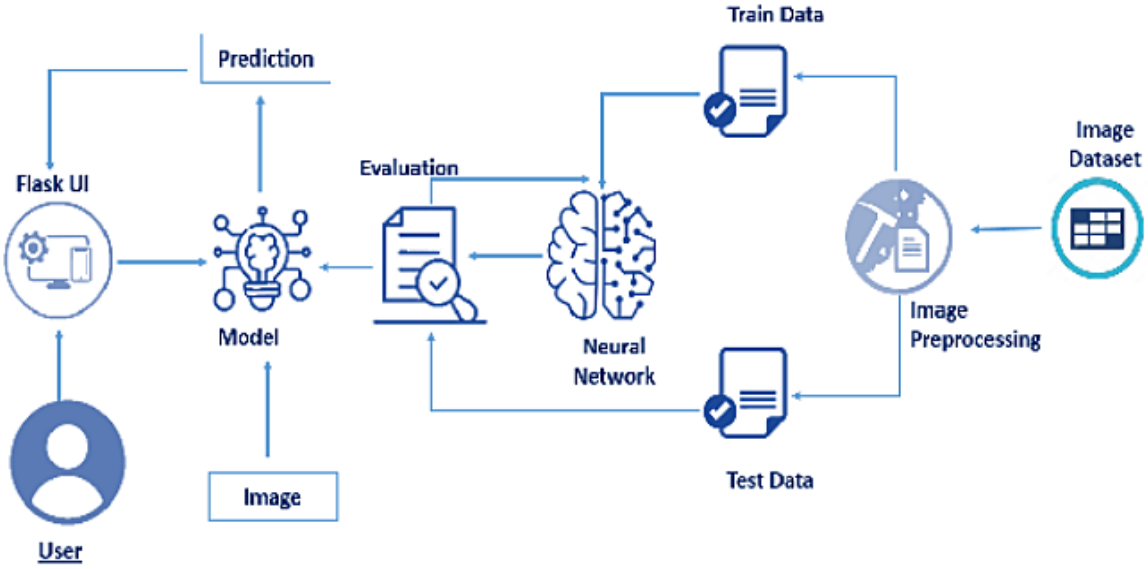
5.PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS



5.2 SOLUTION & TECHNICAL ARCHITECTURE

Technical Architecture



5.3 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
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 account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
	Dashboard					

6.PROJECT PLANNING & SCHEDULING

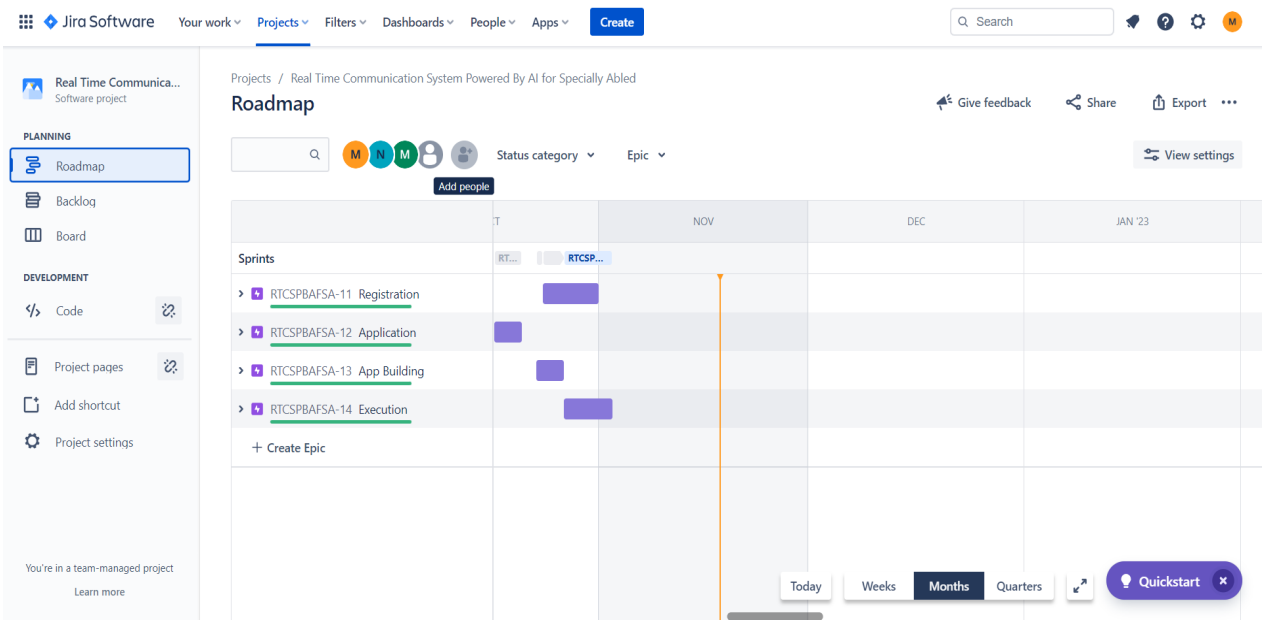
6.1 SPRINT PLANNING & ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Dashboard	USN-6	After enter mail register enter into Dashboard	2	Medium	Nivetha M S, Priyadharshini M, Mohanraj k
Sprint-2	Camera	USN-7	Give a access to camera	3	High	Nivetha M S, Nandhana G, Sethupathi A, Priyadharshini M, Mohanraj k
Sprint-2	Microphone	USN-8	Give a access to Microphone	4	High	Nivetha M S, Nandhana G, Sethupathi A, Priyadharshini M, Mohanraj k
Sprint-3	Hand sign	USN-9	Show hand sign on camera	5	High	Nivetha M S, Nandhana G, Sethupathi A,
Sprint-3	Image	USN-10	Image as input	5	Medium	Nivetha M S, Priyadharshini M, Mohanraj k
Sprint-4	Image Train	USN-11	Train the image using image pre-processing	3	Medium	Nivetha M S, Nandhana G, Sethupathi A, Priyadharshini M, Mohanraj k
Sprint-4	Image Test	USN-12	Test the image using Image pre processing	3	Low	Nivetha M S, Nandhana G, Sethupathi A,
Sprint-4	Evaluate	USN-13	Evaluate image using training and testing	4	High	Nivetha M S, Priyadharshini M, Mohanraj 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	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

6.3 REPORTS FROM JIRA



7. CODING & SOLUTIONING

7.1 FEATURE 1

Image Preprocessing

Import ImageDataGenerator Library and Configure It

```
from keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(rescale =1./255, shear_range=0.2, zoom_range=0.2, horizontal_flip=True)
test_datagen = ImageDataGenerator(rescale =1./255)
```

Apply ImageDataGenerator functionality To Train And Test

```
x_train =train_datagen.flow_from_directory('Dataset/training_set' , target_size=(64,64),batch_size=300,
                                           class_mode= 'categorical', color_mode="grayscale")

x_test = test_datagen.flow_from_directory('Dataset/test_set', target_size=(64,64), batch_size=300,
                                          class_mode='categorical', color_mode="grayscale")
```

Model Building

Import The Required Model Building Libraries

```
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Dropout
from keras.layers import Flatten
```

Initialize The Model

```
model = Sequential()
```

Add The Convolution Layer

```
model.add(Convolution2D(32, (3,3), input_shape=(64,64,1),activation = 'relu'))
```

Add The Pooling Layer

```
model.add(MaxPooling2D(pool_size=(2,2)))
```

Add The Flatten Layer

```
model.add(Flatten())
```

Adding The Dense Layer

```
model.add(Dense(units=512, activation='relu'))
model.add(Dense(units=9, activation='softmax'))
```

Compile The Model

```
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

Fit And Save The Model

```
model.fit_generator(x_train, steps_per_epoch=24, epochs=10, validation_data=x_test, validation_steps=40)

model.save('aslpng1.h5')
```

Load The Test Image, Pre-Process It And Predict

```
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
    prediction =model.predict(img)
    print(prediction)
    prediction = model.predict_classes(img)
    print(prediction)
frame=cv2.imread(r"D:\\Nivetha\\Smart Bridge\\My_project\\conversation engine for deaf and dumb\\Dataset\\test_set\\A\\2.png")
```

7.2 FEATURE 2

```
# import the necessary packages
from flask import Flask,render_template,request
# Flask-It is our framework which we are going to use to run/serve our application.
#request-for accessing file which was uploaded by the user on our application.
import cv2 # opencv library
from tensorflow.keras.models import load_model#to load our trained model
import numpy as np
from gtts import gTTS #to convert text to speech
from skimage.transform import resize
import os
from keras.preprocessing import image
from playsound import playsound
'''
def playaudio(text):
    speech=gTTS(text)
    print(type(speech))
    speech.save("output1.mp3")
    playsound("output1.mp3")
    return
'''
app = Flask(__name__,template_folder="templates") # initializing a flask app
# Loading the model
model=load_model('aslpng1.h5')
print("Loaded model from disk")
vals = ['A', 'B','C','D','E','F','G','H','I']

#app=Flask(__name__,template_folder="templates")
@app.route('/', methods=['GET'])
def index():
    return render_template('home.html')
@app.route('/home', methods=['GET'])
def home():
    return render_template('home.html')
@app.route('/upload', methods=['GET', 'POST'])
def predict():
        # Get a reference to webcam #0 (the default one)
        print("[INFO] starting video stream...")
        vs = cv2.VideoCapture(0)
```

```
	#writer = None
	(W, H) = (None, None)

# loop over frames from the video file stream
	while True:
	  # read the next frame from the file
	  (grabbed, frame) = vs.read()
	  # if the frame was not grabbed, then we have reached the end
	  # of the stream
	  if not grabbed:
	    break
	  # if the frame dimensions are empty, grab them
	  if W is None or H is None:
	    (H, W) = frame.shape[:2]
	  # clone the output frame, then convert it from BGR to RGB
	  # ordering and resize the frame to a fixed 64x64
	  output = frame.copy()
	  #print("apple")
	  img = resize(frame,(64,64,1))
	  img = np.expand_dims(img,axis=0)
	  if(np.max(img)>1):
	    img = img/255.0
	  result = np.argmax(model.predict(img), axis=-1)
	  index=['A', 'B','C','D','E','F','G','H','I']
	  result=str(index[result[0]])
	  #print(result)
	  #result=result.tolist()
	  cv2.putText(output, "It indicates: {}".format(result), (10, 120),
cv2.FONT_HERSHEY_PLAIN,
	        2, (0,255,255), 1)
	  #converts text to speech and plays the audio
	  speech = gTTS(text = result, lang = 'en', slow = False)
	  #speech=gTTS(text)
	  print(type(speech))
	  speech.save("text.mp3")
	  os.system("start text.mp3")
	  cv2.imshow("Output", output)
	  key = cv2.waitKey(1) & 0xFF

	    # if the `q` key was pressed, break from the loop
```

```
            if key == ord("q"):
                break
        # release the file pointers
        print("[INFO] cleaning up...")
        vs.release()
        cv2.destroyAllWindows()
        return render_template("upload.html")

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8000, debug=False)
```

8. TESTING

8.2 TEST CASES

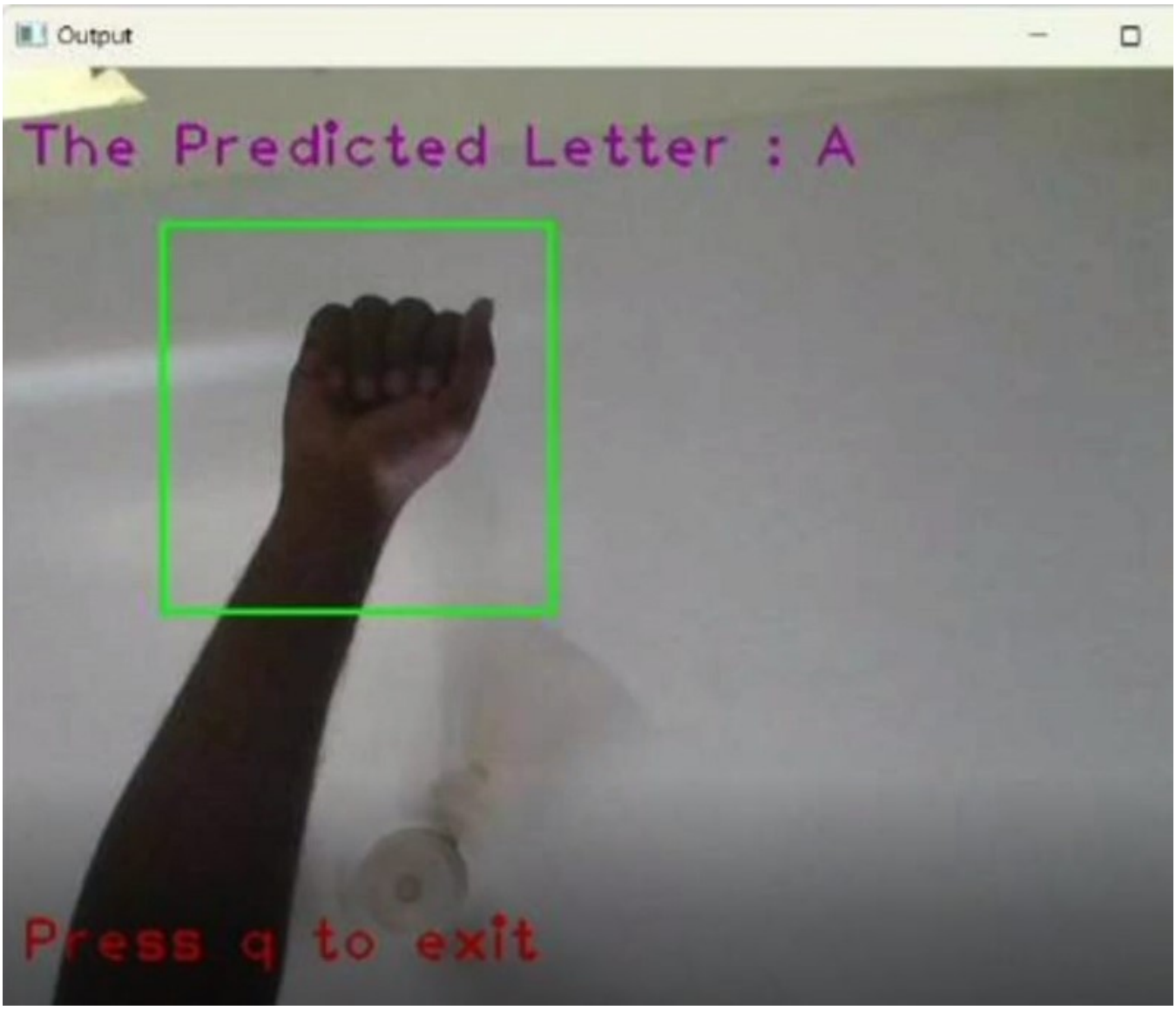
Section	Total Cases	Not Tested	Fail	Pass
Print Engine	6	0	0	6
Client Application	45	0	0	45
Security	6	0	0	6
Outsource Shipping	4	0	0	4
Exception Reporting	5	0	0	5

8.2 USER ACCEPTANCE TESTING

S.No.	Parameter	Values	Screenshot
•	Model Summary	-	An eaducation system obsessed with rote - learning and makes
•	Accuracy	Training Accuracy - Validation Accuracy -	Personalized attention ,senstive treatrment,and care with alot of patience and understanding
3.	Confidence Score (Only Yolo Projects)	Class Detected - Confidence Score -	Trained staff and teachers from all over the world to help the specially abled child

9. RESULTS

9.1 Performance Metrics



10. ADVANTAGES & DISADVANTAGES

ADVANTAGES

1. Speech output allows for increased adativity and emotional interpretation.
2. Get your answer as soon as you ask the question.
3. This app enables deaf and dumb people to converted to human-understandable language an dspeech is given as output.
4. Our project will increase the conversation between the normal people and deaf & dumb people.
5. Usel to have a proper coversation between a normal person and an imparied person in any language.

DISADVANTAGE

1. Poor internet connection can affect the conversation.
2. Too many conversation can lead to overloading.

11. CONCLUSION

who understand sign language may interact with people who are unfamiliar with sign language. Speech interpretation is helpful for sign language non-speakers who want the accompanying hand sign to be understood. Room conditions such as lighting can play a role in predicting the outcome of poor lighting. The light that is either too bright or too dim will result in inaccurate hand segmentation, resulting in inaccurate gesture prediction. The type of inaccuracy can emerge from the user's peripherals, such as poor web camera performance or poor microphone quality.

Our project can change a deaf and dumb people life and they feel Mournful.

12. FUTURE SCOPE

In future work we will implement different day to day life activities with the help of different sensors.

13. APPENDIX

Source Code

Image Preprocessing

Import ImageDataGenerator Library and Configure It

```
from keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(rescale =1./255, shear_range=0.2, zoom_range=0.2, horizontal_flip=True)
test_datagen = ImageDataGenerator(rescale =1./255)
```

Apply ImageDataGenerator functionality To Train And Test

```
x_train =train_datagen.flow_from_directory('Dataset/training_set' , target_size=(64,64),batch_size=300,
                                           class_mode= 'categorical', color_mode="grayscale")

x_test = test_datagen.flow_from_directory('Dataset/test_set', target_size=(64,64), batch_size=300,
                                          class_mode='categorical', color_mode="grayscale")
```

Model Building

Import The Required Model Building Libraries

```
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Dropout
from keras.layers import Flatten
```

Initialize The Model

```
model = Sequential()
```

Add The Convolution Layer

```
model.add(Convolution2D(32, (3,3), input_shape=(64,64,1),activation = 'relu'))
```

Add The Pooling Layer

```
model.add(MaxPooling2D(pool_size=(2,2)))
```

Add The Flatten Layer

```
model.add(Flatten())
```

Adding The Dense Layer

```
model.add(Dense(units=512, activation='relu'))
model.add(Dense(units=9, activation='softmax'))
```

Compile The Model

```
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

Fit And Save The Model

```
model.fit_generator(x_train, steps_per_epoch=24, epochs=10, validation_data=x_test, validation_steps=40)

model.save('aslpng1.h5')
```

Load The Test Image, Pre-Process It And Predict

```
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
    prediction =model.predict(img)
    print(prediction)
    prediction = model.predict_classes(img)
    print(prediction)
frame=cv2.imread(r"D:\\Nivetha\\Smart Bridge\\My_project\\conversation engine for deaf and
```

dumb\\Dataset\\test_set\\A\\2.png")

FLSK APP BUILDING

```
# import the necessary packages
from flask import Flask,render_template,request
# Flask-It is our framework which we are going to use to run/serve our application.
#request-for accessing file which was uploaded by the user on our application.
import cv2 # opencv library
from tensorflow.keras.models import load_model#to load our trained model
import numpy as np
from gtts import gTTS #to convert text to speech
from skimage.transform import resize
import os
from keras.preprocessing import image
from playsound import playsound
'''
def playaudio(text):
    speech=gTTS(text)
    print(type(speech))
    speech.save("output1.mp3")
    playsound("output1.mp3")
    return
'''
app = Flask(__name__,template_folder="templates") # initializing a flask app
# Loading the model
model=load_model('aslpng1.h5')
print("Loaded model from disk")
vals = ['A', 'B','C','D','E','F','G','H','I']

#app=Flask(__name__,template_folder="templates")
@app.route('/', methods=['GET'])
def index():
    return render_template('home.html')
@app.route('/home', methods=['GET'])
def home():
    return render_template('home.html')
@app.route('/upload', methods=['GET', 'POST'])
def predict():
        # Get a reference to webcam #0 (the default one)
        print("[INFO] starting video stream...")
```

```
        vs = cv2.VideoCapture(0)
        #writer = None
        (W, H) = (None, None)

# loop over frames from the video file stream
        while True:
            # read the next frame from the file
            (grabbed, frame) = vs.read()
            # if the frame was not grabbed, then we have reached the end
            # of the stream
            if not grabbed:
                break
            # if the frame dimensions are empty, grab them
            if W is None or H is None:
                (H, W) = frame.shape[:2]
            # clone the output frame, then convert it from BGR to RGB
            # ordering and resize the frame to a fixed 64x64
            output = frame.copy()
            #print("apple")
            img = resize(frame,(64,64,1))
            img = np.expand_dims(img,axis=0)
            if(np.max(img)>1):
                img = img/255.0
            result = np.argmax(model.predict(img), axis=-1)
            index=['A', 'B','C','D','E','F','G','H','I']
            result=str(index[result[0]])
            #print(result)
            #result=result.tolist()
            cv2.putText(output, "It indicates: {}".format(result), (10, 120),
cv2.FONT_HERSHEY_PLAIN,
                        2, (0,255,255), 1)
            #converts text to speech and plays the audio
            speech = gTTS(text = result, lang = 'en', slow = False)
            #speech=gTTS(text)
            print(type(speech))
            speech.save("text.mp3")
            os.system("start text.mp3")
            cv2.imshow("Output", output)
            key = cv2.waitKey(1) & 0xFF
    if key == ord("q"):
```

```
            break
        # release the file pointers
        print("[INFO] cleaning up...")
        vs.release()
        cv2.destroyAllWindows()
        return render_template("upload.html")

if __name__ == '__main__':
      app.run(host='0.0.0.0', port=8000, debug=False)
```

UPLOAD.html

```
<html lang="en">

<head>
   <title>Conversation Engine</title>
   <link href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css" rel="stylesheet">
<style>
.header {         position: relative;
                        top:0;
                        margin:0px;
                        z-index: 1;
                        left: 0px;
                        right: 0px;
                        position: fixed;
                        background-color: #F36262 ;
                        color: white;
                        box-shadow: 0px 8px 4px grey;
                        overflow: hidden;
                        padding-left:20px;
                        font-family: 'Josefin Sans';
                        font-size: 2vw;
                        width: 100%;
                        height:8%;
                        text-align: center;
                }
                .topnav {
  overflow: hidden;
  background-color: #FCAD98;
}
```

```
.topnav-right a {
  float: left;
  color: black;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  font-size: 18px;
}

.topnav-right a:hover {
  background-color: #FCAD98;
  color: black;
}

.topnav-right a.active {
  background-color: #FCAD98;
  color: white;
}

.topnav-right {
  float: right;
  padding-right:100px;
}

body {

  background-color: ;
  background-repeat: no-repeat;
  background-size:cover;
  background-image:
url("https://i.pinimg.com/originals/b2/1d/c6/b21dc69346915015bc4e19bd502f401b.gif");
    background-size: cover;
  background-position: 0px 0px;
  }
  .button {
  background-color: #091425;
  border: none;
  color: white;
  padding: 15px 32px;
```

```
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 12px;
  border-radius: 16px;
}
.button:hover {
  box-shadow: 0 12px 16px 0 rgba(0,0,0,0.24), 0 17px 50px 0 rgba(0,0,0,0.19);
}
form {border: 3px solid #f1f1f1; margin-left:400px;margin-right:400px;}

input[type=text], input[type=password] {
  width: 100%;
  padding: 12px 20px;
  display: inline-block;
  margin-bottom:18px;
  border: 1px solid #ccc;
  box-sizing: border-box;
}

button {
  background-color: #091425;
  color: white;
  padding: 14px 20px;
  margin-bottom:10px;
  border: none;
  cursor: pointer;
  width: 17%;
  border-radius:4px;
  font-family:Montserrat;
}

button:hover {
  opacity: 0.8;
}

.cancelbtn {
  width: auto;
  padding: 10px 18px;
  background-color: #f44336;
```

```
}

.imgcontainer {
  text-align: center;
  margin: 24px 0 12px 0;
}

img.avatar {
  width: 30%;
  border-radius: 50%;
}

.container {
  padding: 16px;
}

span.psw {
  float: right;
  padding-top: 16px;
}

/* Change styles for span and cancel button on extra small screens */
@media screen and (max-width: 300px) {
  span.psw {
     display: block;
     float: none;
  }
  .cancelbtn {
     width: 100%;
  }
}

.home{
          margin:80px;

  width: 84%;
  height: 500px;
  padding-top:10px;
  padding-left: 30px;
```

```
}
.login{
        margin:80px;
        box-sizing: content-box;
  width: 84%;
  height: 420px;
  padding: 30px;
  border: 10px solid blue;
}
.left,.right{
 box-sizing: content-box;
 height: 400px;
 margin:20px;
 border: 10px solid blue;
}

.mySlides {display: none;}
img {vertical-align: middle;}

/* Slideshow container */
.slideshow-container {
  max-width: 1000px;
  position: relative;
  margin: auto;
}

/* Caption text */
.text {
  color: #f2f2f2;
  font-size: 15px;
  padding: 8px 12px;
  position: absolute;
  bottom: 8px;
  width: 100%;
  text-align: center;
}
/* The dots/bullets/indicators */
.dot {
  height: 15px;
  width: 15px;
```

```
  margin: 0 2px;
  background-color: #bbb;
  border-radius: 50%;
  display: inline-block;
  transition: background-color 0.6s ease;
}

.active {
  background-color: #FCAD98;
}

/* Fading animation */
.fade {
  -webkit-animation-name: fade;
  -webkit-animation-duration: 1.5s;
  animation-name: fade;
  animation-duration: 1.5s;
}

@-webkit-keyframes fade {
  from {opacity: .4}
  to {opacity: 1}
}

@keyframes fade {
  from {opacity: .4}
  to {opacity: 1}
}

/* On smaller screens, decrease text size */
@media only screen and (max-width: 300px) {
  .text {font-size: 11px}
}
```

```
.bar
{
margin: 0px;
padding:20px;
background-color:white;
opacity:0.6;
color:black;
font-family:'Roboto',sans-serif;
font-style: italic;
border-radius:20px;
font-size:25px;
}
a
{
color:grey;
float:right;
text-decoration:none;
font-style:normal;
padding-right:20px;
}
a:hover{
background-color:black;
color:white;
border-radius:15px;0
font-size:30px;
padding-left:10px;
}

p
{
color:black;
font-style:italic;
font-size:30px;
}
</style>
</head>

<body style="background-
image:url({{url_for('static',filename='C:\Users\Lenovo\Downloads\HandSign.jpg')}});background-
```

```
position: center;background-repeat: no-repeat;
        background-size: cover;">

<div class="header">
<div style="width:50%;float:left;font-size:2vw;text-align:left;color:black; padding-top:1%;padding-left:5%;">Real Time Communication System for Deaf & Dumb</div>
  <div class="topnav-right"style="padding-top:0.5%;">

    <a  href="/home">Home</a>
    <a class="active" href="/upload">Open Web Cam</a>
  </div>
</div>

</body>
```

GitHub & Project Demo Link

GitHub link

IBM-EPBL/**IBM-Project-44548-1660725237**