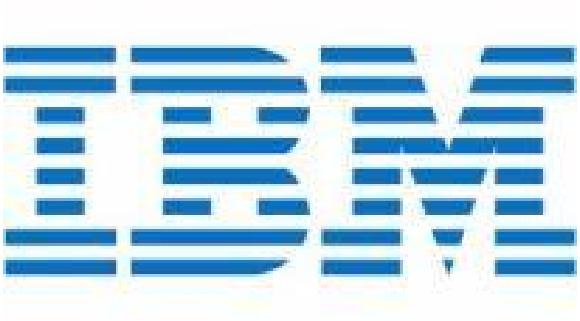
PROJECT REPORT



REAL-TIME COMMUNICATION SYSTEM POWERED BY AI FOR SPECIALLY ABLED

Team ID: PNT2022TMID03466

TEAM MEMBERS:

THENNARASU M [TL] - 212219220055
FAZIL MOHAMED M [TM1] - 212219220008
ROHIT N [TM2] - 212219220043
ANISH GEORGE [TM3] - 212219223001

ABSTRACT:

One of the most precious gifts of nature to the human race is the ability to express itself by responding to the events that occur in its environment. Every normal person sees, hears, and then reacts to the situations by expressing himself. But there are some less lucky ones who are deprived of this precious gift. Such people, especially deaf and mute, rely on some sort of gesture language to communicate their feelings to others. The deaf, dumb and the blind follow similar problems when it comes to the use of computers. In the era of advanced technologies, where computers, laptops and other processor-based devices are an integral part of everyday life, efforts must be made to make the disabilities in life more independent. Our goal is to design a human computer interface system that can accurately identify the language of the deaf and dumb. With the use of image processing and artificial intelligence, many techniques and algorithms have been developed in this area. Each character speech recognition system is trained to recognize the characters and convert them into the required pattern. The proposed system aims to give speech speechless, a real-time character language is captured as a series of images, and it is processed and then converted into speech and text.

1. INTRODUCTION:

1.1. 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. This enables deaf and dumb people to convey their information using signs which get converted to human-understandable language as output.

1.2. Purpose:

Dumb people are usually face some problems on normal communication with other people in society. It has been observed that they sometimes find it difficult to interact with normal people with their gestures. Because people with hearing problems or deaf people cannot speak like normal people, they have to depend on a kind of visual communication in most cases. To overcome these problems, we have proposed a system that uses cameras to capture and convert videos of hand gestures from dumb people who turn into speech for understanding normal people. The primary application for addressing the sign language is the improvement of the sign language. Computer recognition of the sign language is an important research problem for communication with the hearing impaired. The system proposed to develop and build an intelligent system that uses image processing, machine learning and artificial intelligence concepts to make visual inputs of hand gestures of sign language and to create an easily recognizable form of outputs.

2. LITERATURE SURVEY:

2.1. Existing Problem:

A. Two Way Communicator between Deaf and Dumb People and Normal People. [1]

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.

B. Sign Language Recognition System to aid Deaf-dumb People Using PCA. [2]

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

C. Sign Language to Text and Vice Versa Recognition using Computer Vision in Marathi. [3]

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.

D. Sign Language Learning based on Android for Deaf and Speech Impaired People. [4]

This research makes an Android-based application that can directly interpret Sign language presented by deaf people in written language. Translation process Starts with the detection of hands with OpenCV and translation of and signals The K-NN classification. Tutorial features added in this application with the goal to train intensively to guide the user when using the sign language.

2.2. References:

[1] Prof. P.G. Ahire, K.B. Tilekary,T.A. Jawake, P.B. Warale,“Two Way Communicator between Deaf and Dumb Peopleand Normal People", 978-1-4799-6892-3/15 31.00 c 2015 IEEE.

[2] Shreyashi Narayan Sawant, "Sign Language recognition System to aid Deaf- dumb People Using PCA", IJCSET ISSN : 2229-3345 Vol. 5 No. 05 May 2014.

[3] Amitkumar Shinde, Ramesh Kagalkar, ”Sign Language to Text and Vice Versa Recognition using Computer Vision in Marathi”, International Journal of Computer Applications (0975 – 8887) National Conference on Advances in Computing (NCAC 2015)

Setiawardhana, Rizky Yuniar Hakkun, Achmad Baharuddin, "Sign Language Learning based on Android For Deaf and Speech Impaired People", 978-1-4673-9345-4/15/31.00 c 2015 IEEE.

[5] W. Sureshkumar and R. Rama, “Chomsky hierarchy control on isotonic array P systems,” International Journal of Pattern Recognition and Artificial Intelligence, vol. 30, no. 2, pp. 1650004–1650249, 2016.

[6] L. Zhou, G. Yang, Y. Yin, L. Yang, and K. Wang, “Finger vein recognition based on stable and discriminative superpixels,” International Journal of Pattern Recognition & Artificial Intelligence, vol. 30, no. 6, pp. 1650015–1650079, 2016.

[7] Y. Zhang-Jing, H. Pu, and Z. Fan-Long, “Center-based line neighborhood discriminant embedding algorithm and its application to face recognition,” Pattern Recognition & Artificial Intelligence, vol. 21, no. 3, p. 65, 2015.

[8] D. Verma and S. Dubey, “Fuzzy brain storm optimization and adaptive thresholding for multimodal vein-based recognition system,” International Journal of Pattern Recognition & Artificial Intelligence, vol. 13, no. 5, pp. 49–61, 2016.

[9] F. Tian, Y. Gao, Z. Fang, and J. Gu, “Automatic coronary artery segmentation algorithm based on deep learning and digital image processing,” Applied Intelligence, vol. 51, no. 12, pp. 8881–8895, 2021.

[10] E. Giacomin, T. Greenberg-Toledo, S. Kvatinsky, and P.E. Gaillardon, “A robust digital RRAM-based convolutional block for low-power image processing and learning applications,” IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 66, no. 2, pp. 643–654, 2019.

[11] Z. Che and X. Zhuang, “Digital affine shear filter banks with 2layer structure and their applications in image processing,” IEEE Transactions on Image Processing, vol. 27, no. 8, pp. 3931–3941, 2018.

[12] L. Zhang, L. Zhang, and L. Zhang, “Application research of digital media image processing technology based on wavelet transform,” EURASIP Journal on Image and Video Processing, vol. 2018, no. 1, 2018.

[13] R. Mouhcine, A. Mustapha, and M. Zouhir, “Recognition of cursive Arabic handwritten text using embedded training based on HMMs,” Journal of Electrical Systems & Information Technology, vol. 32, no. 1, p. 455, 2017.

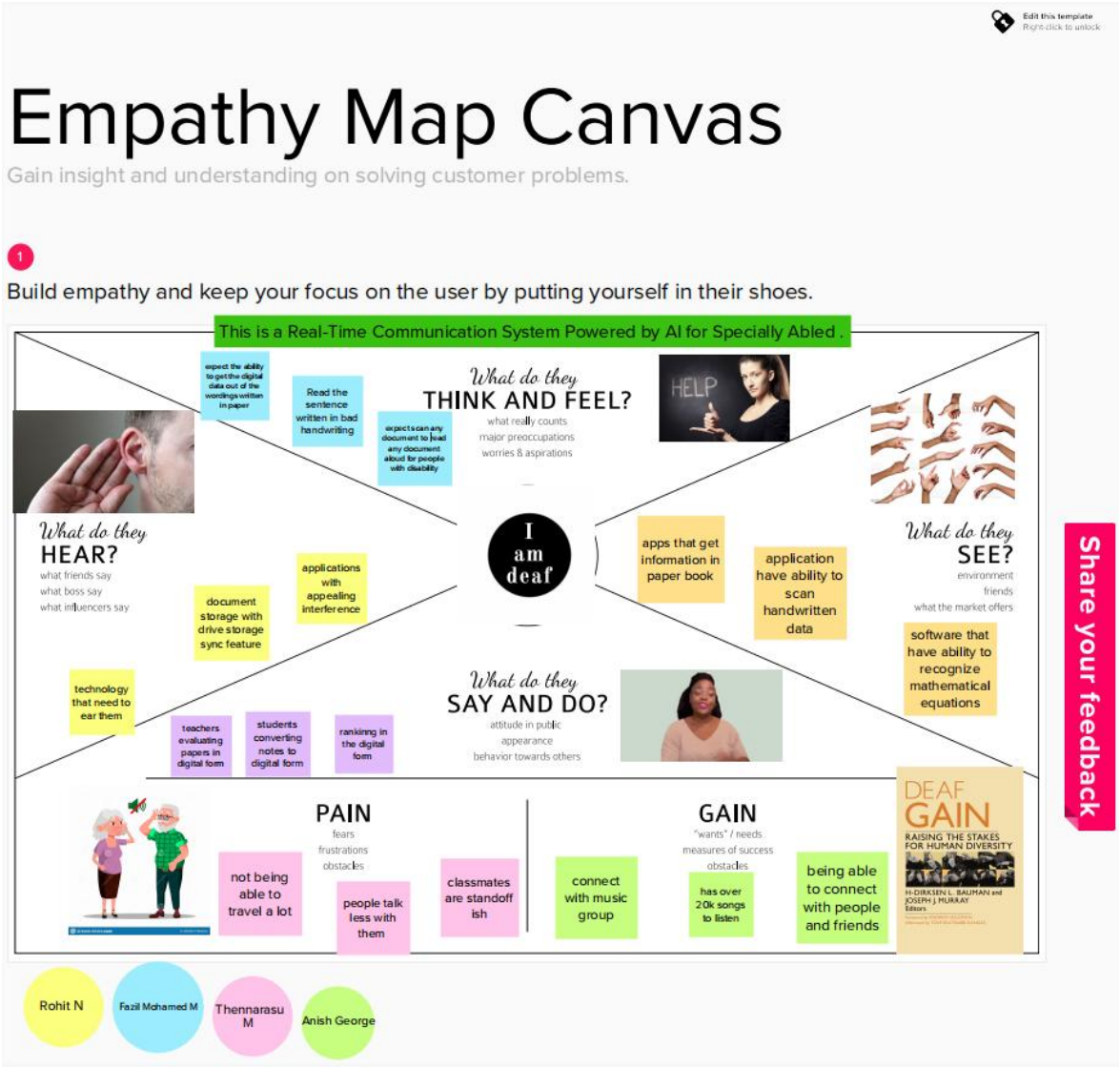
2.3. Proposed solution:

This paper describes the system that overcomes the problem faced by the speech and hearing impaired. The objectives of the research are as follow:

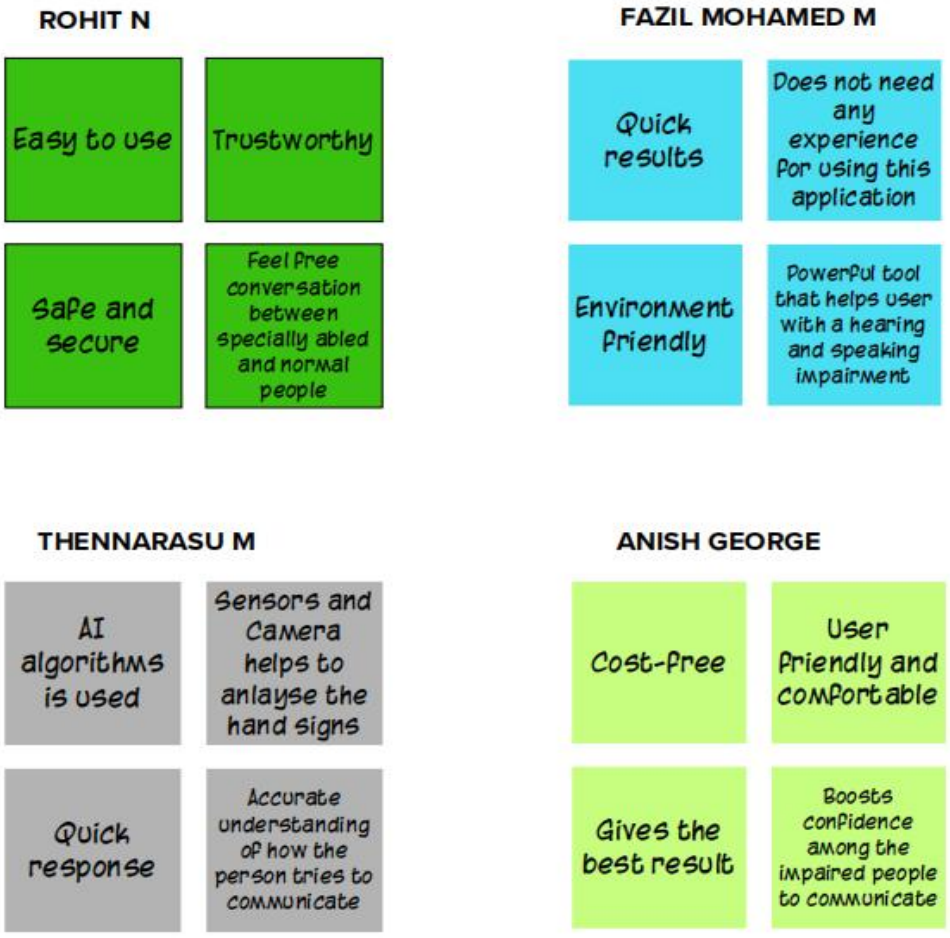
1. To design and develop a system which lowers the communication gap between speech hearing impaired and normal world.
2. To build a communication system that enables communications between deaf-dumb person and a normal person.
3. A convolution neural network is being used to develop a model that is trained on various hand movements. This model is used to create an app. This programme allows deaf and hard of hearing persons to communicate using signs that are then translated into human readable text.

3. IDEATION & PROPOSED SYSTEM

3.1. Empathy Map Canvas:



3.2. Ideation & Brainstorm:



3.3. Proposed Solution:

Project Design Phase-I
Proposed Solution

Date	19 September 2022
Team ID	PNT2022TMID03466
Project Name	Real-Time Communication System Powered by AI for Specially Abled
Maximum Marks	2 Marks

Proposed Solution :

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	An application for deaf and dumb people to convey their information using signs which get converted to human-understandable language and speech in Artificial Intelligence
2.	Idea / Solution description	By using Voice Conversion System with Hand Gesture Recognition and translation will be very useful to have a proper conversation
3.	Novelty / Uniqueness	We are using a convolution neural network to create a model that is trained on different hand gestures and an app is built for the use this mode
4.	Social Impact / Customer Satisfaction	Communicating with others and being connected in the society and remove accessibility barriers
5.	Business Model (Revenue Model)	By Using: Better communication with the disabled and Financial By Without Using: Can't Communicate and leads to loneliness
6.	Scalability of the Solution	Enhance people with disabilities to step into a world where their are facing difficulties in communication

3.4. Problem Solution Fit:

Project Design Phase-I-Solution Fit Canvas

Project Title : Real-Time Communication System Powered by AI for Specially Abled

Team ID : PNT2022TMID03466

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

- Normal People, Who needs to communicate with specially abled.
- Deaf People
- Dumb People

6. CUSTOMER CONSTRAINTS CC

Artificial Intelligence technology solutions, discover how accessibility for people can be enhanced.

5. AVAILABLE SOLUTIONS AS

- Voice Conversion system with hand gestures recognition and translation will be very useful to have a proper conversation between a normal people and an impaired person in any language.
- An app is built which enables deaf and dumb people to convey their information using signs.

Explore AS, differentiate

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

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

To predict the delay of the flight

- Input is given as Hand gestures image, which undergoes image preprocessing and voice recognition.
- In Neural network, the hand gestures are trained by Convolution Neural Network.

9. PROBLEM ROOT CAUSE RC

- Communication between deaf-mute and a normal people has always been a challenging task.
- It is very difficult for mute people to convey their message to normal people.
- Normal people are not trained on hand sign language, and in emergency times it is very difficult.

7. BEHAVIOUR BE

What do to address the problem

- Artificial Intelligence model that converts sign language into a speech that can be understood by normal people.
- An application built for efficient usage.

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

Identify strong TR & EM

3. TRIGGERS TR

- The benefit of the 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.

4. EMOTIONS: BEFORE / AFTER EM

BEFORE: Normal people are not aware of hand signs and Communication is difficult between the normal people and specially abled.

AFTER: Communication is good and efficient between the normal people and specially abled and Normal people can learn sign language.

10. YOUR SOLUTION SL

- An application for deaf and dumb people to convey their information using signs.
- By using voice conversion system with hand gesture recognition and translation will be very useful to have a proper conversation.

8.CHANNELS of BEHAVIOR CH

8.1 ONLINE

- A simple and beautiful user interface is used and supports different languages.
- Accurate prediction and used speech to text & text to speech.

8.2 OFFLINE

- Communication is made between the normal people and specially abled.
- Normal people used to learn sign language.

4. REQUIREMENT ANALYSIS

4.1. Functional Requirement:

Date	22 October 2022
Team ID	PNT2022TMID03466
Project Name	Project – Real-Time Communication System Powered by AI for Specially Abled
Maximum Marks	4 Marks

Functional Requirements:
Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Requirement	Converting sign language into speech that can be understand by normal people using an application.
FR-2	User Registration	Manual Sign up using the application or Gmail.
FR-3	User Confirmation	OTP authentication through phone messages, email, notices, paper and confirmation.
FR-4	Product Implementation	Install the dataset to recognise and translate hand gestures and voice for the real-time communication by using the application.
FR-5	Payment Option	Bank transfer, Debit cards, UPI method, if pro version required.
FR-6	Product Feedback	Through the application, phone conversation and Gmail.

4.2. Non-Functional Requirement:

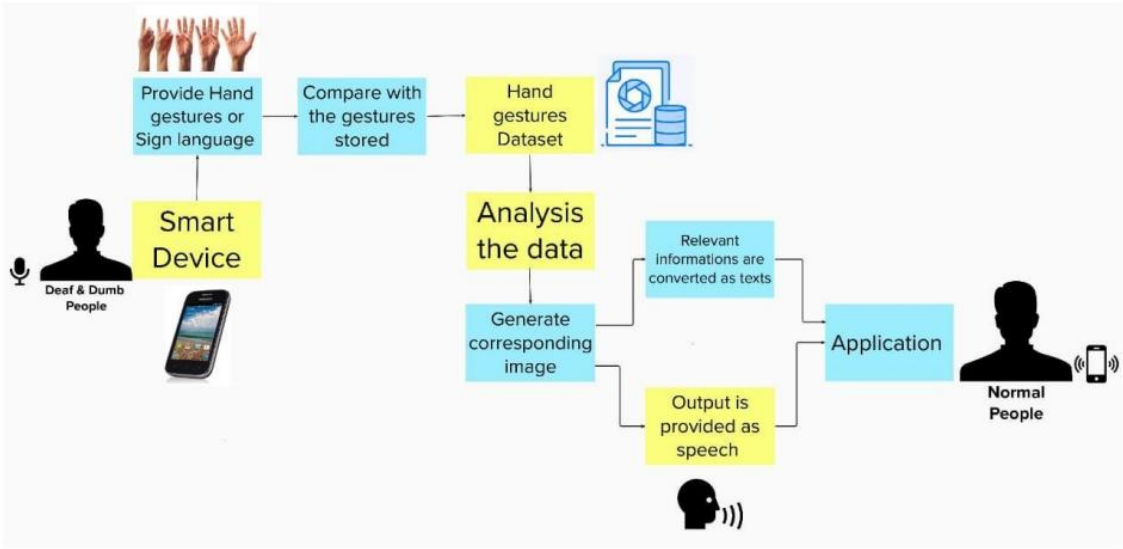
Non-functional Requirements:
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	It is used to describe the application and easy to access the application with the guidelines.
NFR-2	**Security**	It ensures the security of the application by building a firewall and two step verification support. Accessed only by authorised person by given user ID and password or OTP verification.
NFR-3	**Reliability**	To maintain the application conditions and update the version of the application. System update and software update are possible to increase various features and durability based on technology.
NFR-4	**Performance**	This application collects the datasets of hand gestures to provide accurate prediction. Using this method, we can communicate easily at anytime. This application is user friendly and can be accessed by both specially abled and normal people.
NFR-5	**Availability**	Depending on the requirements of the user, all required functions will be offered. When the user requests any features, the features are made available in places where users like to know about it.
NFR-6	**Scalability**	As based on application, real-time communication is accessed on a compatible devices. The application is based on voice conversion system, hand gesture recognition and translation.

5. PROJECT DESIGN:

5.1. Data Flow Diagram:

Data Flow Diagram:



5.2. Solution & Technical Architecture:

Technical Architecture:

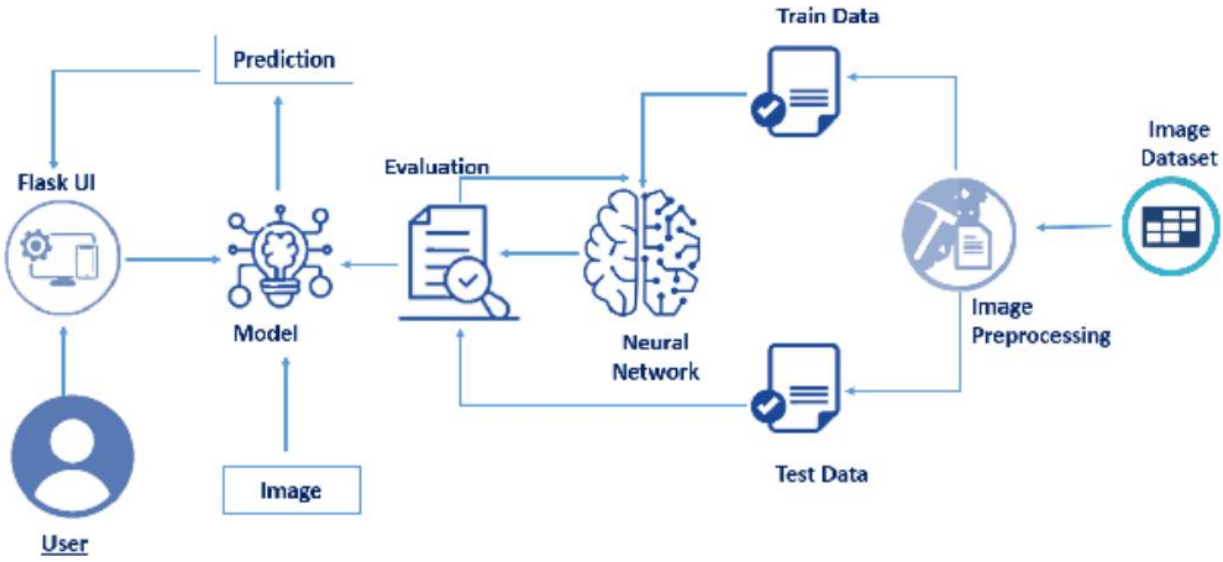


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Application Logic-1	Logic for a process in the application	Java / Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson STT service
4.	Application Logic-3	Logic for a process in the application	IBM Watson Assistant
5.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local Filesystem
8.	External API-1	Purpose of External API used in the application	IBM Weather API, etc.
9.	External API-2	Purpose of External API used in the application	Aadhar API, etc.
10.	Machine Learning Model	Purpose of Machine Learning Model	Object Recognition Model, etc.
11.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	Local, Cloud Foundry, Kubernetes, etc.

5.3. User Stories:

User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Register with the users information.	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 in the application.	High	Sprint-1
Customer (Deaf people)	To communicate with people using signs.	USN-2	As a user, I can see my application and made changes in any browser and register to it.	I can login and see my account in the application anywhere at anytime.	High	Sprint-1
Customer (Dumb people)	To communicate with people easily and efficiently.	USN-3	As a user, I can see my application and made changes in any browsers and register to it.	I can login and see my account in the application anywhere.	High	Sprint-1
Customer (Normal people)	User needs to communicate with specially abled people.	USN-4	As a user, I can register for the application by entering my email, password, and confirmation is made.	I can login and see my account.	Medium	Sprint-2
Customer (Learner of Sign language)	User needs to be aware and learn about sign language.	USN-5	As a user, I can create my account in the application with my email and password, to get knowledge about sign languages.	I can create my account and access the dashboard in the application.	High	Sprint-1
Customer (Web user)	They want the update on the application condition.	USN-6	As a user, I can register for the application by entering my email, password, and confirming my password. To get details about real-time communication.	I can able to use any browser to access the application from anywhere, to know anything about real-time communication.	High	Sprint-1
Customer Care Executive	They want to help people by sending application conditions.	USN-7	As a user, I can receive a message from the administration about conditions of application of real-time communication.	I will analyse and send SMS to the people.	High	Sprint-1

6. CODING & SOLUTIONING

Building a simple CNN model with Conv2D, Maxpool, Conv2D, maxpool trend

```
from random import uniform
from keras.layers.normalization import BatchNormalization


input_size = 28
filter_size = 3
num_filter = 8
maxpool_size = 2
batch_size = 16
epochs = 10
steps_per_epoch = 24720/batch_size

import keras
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D
from keras.layers import Dropout, Flatten, Dense


model = Sequential()
model.add(Conv2D(16, (filter_size,filter_size),
                 input_shape= (input_size,input_size,3),
                 activation ='relu',
                 padding='same'))

model.add(MaxPooling2D(pool_size=(maxpool_size, maxpool_size),strides=1))


model.add(Conv2D(32, (filter_size,filter_size),
                 activation='relu',
                 padding='valid'))


model.add(MaxPooling2D(pool_size=(maxpool_size, maxpool_size),strides=2))


model.add(Flatten())
model.add(Dense(120, activation='relu'))
model.add(Dense(24,activation='softmax'))
```

Now we try a nother model using Conv2D, pool,Conv2D,pool, Conv2D, pool with dropout and BatchNormalization

```
input_size = 28
filter_size = 3
num_filter = 8
maxpool_size = 2
batch_size = 16
epochs = 10
steps_per_epoch = 24720/batch_size

import keras
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D
from keras.layers import Dropout, Flatten, Dense


model = Sequential()
model.add(Conv2D(16, (filter_size,filter_size),
                 input_shape= (input_size,input_size,3),
                 activation ='relu',
                 padding='same'))
model.add(MaxPooling2D(pool_size=(maxpool_size, maxpool_size),strides=2))


model.add(Conv2D(32, (filter_size,filter_size),
                 input_shape= (input_size,input_size,3),
                 activation ='relu',
                 padding='same'))
model.add(MaxPooling2D(pool_size=(maxpool_size, maxpool_size),strides=2))
model.add(Dropout(uniform(0, 1)))

model.add(Conv2D(64, (filter_size,filter_size),
                 activation='relu',
                 padding='valid'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(maxpool_size, maxpool_size),strides=2))
model.add(Dropout(uniform(0, 1)))

model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(24,activation='softmax'))
```

```
METRICS = [ 'accuracy']#, 'precision','recall']


model.compile( optimizer= keras.optimizers.Adam(lr=0.001),loss='categorical_crossentropy',metrics=METRICS)

#
model.summary()
```

we will use EarlyStopping

```
from keras.callbacks import EarlyStopping

es = EarlyStopping(monitor='val_loss', mode='max', verbose=1, patience=2)

history = model.fit_generator(
training_set,
steps_per_epoch= steps_per_epoch,
epochs= epochs,
validation_data=validation_set,
validation_steps= 2735  // batch_size,
callbacks=[es]
)
```

```
testing_data_generator = ImageDataGenerator(rescale =1/255)

testing_set = training_data_generator.flow_from_directory('../input/handsignimages/Test',
                                                          target_size =(28, 28),
                                                          batch_size = 16 ,
                                                          class_mode ='categorical')
```

```
score = model.evaluate_generator(testing_set, steps= len(testing_set))
for idx, metric in enumerate(model.metrics_names):
    print(metric, score[idx])
```

6.1. Model Building:

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
```

```
#Creating the model
model=Sequential()
#Adding the layers
model.add(Convolution2D(32,(3,3), input_shape=(64,64,1), activation = 'relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())

#adding hidden layers
model.add(Dense(400, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dense(100, activation='relu'))

#Adding the output layer
model.add(Dense(9, activation='softmax'))
```

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

```
model.fit_generator(x_train, steps_per_epoch=30, epochs=10, validation_data=x_test,validation_steps=50)
```

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

6.2. Data Scheme:

Explore the data

Plot the images to better understand the kind of data we are working with

```
## plot the image from class C as example to plot the images

from matplotlib import pyplot as plt
import os
import random


_, _, sign_images = next(os.walk('../input/handsignimages/Train/C/'))

### prepare a 4x4 plot (total of 16 images)
fig, ax = plt.subplots(3, 2, figsize=(10,10))

### randomly select and plot an image
for idx, img in enumerate(random.sample(sign_images, 6)):
    img_read = plt.imread('../input/handsignimages/Train/C/'+img)
    ax[int(idx/2), idx%2].imshow(img_read)
    ax[int(idx/2), idx%2].axis('off')
    ax[int(idx/2), idx%2].set_title('Train/C'+img)
plt.show()
```

Pre-Process Data

we read the training and validation dataset using ImageDataGenerator

```
from keras.preprocessing.image import ImageDataGenerator

training_data_generator = ImageDataGenerator(rescale =1/255, validation_split=0.1)

training_set = training_data_generator.flow_from_directory('../input/handsignimages/Train', target_size =(28, 28), batch_size = 16 , class_mode ='cate

validation_set = training_data_generator.flow_from_directory('../input/handsignimages/Train', target_size =(28, 28), batch_size = 16 , class_mode ='ca

print(training_set)
print(training_set)
```

```
Found 24720 images belonging to 24 classes.
Found 2735 images belonging to 24 classes.
```

DATA AUGMENTATION

```
from 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)
```

```
x_train = train_datagen.flow_from_directory("/content/Dataset/training_set", target_size=(64,64),batch_size=100,
                                            class_mode='categorical', color_mode ="grayscale")
```

```
Found 15750 images belonging to 9 classes.
```

```
x_test = test_datagen.flow_from_directory("/content/Dataset/test_set", target_size=(64,64),batch_size=100,
                                          class_mode='categorical', color_mode ="grayscale")
```

```
Found 2250 images belonging to 9 classes.
```

```
len(x_train)
```

```
158
```

```
len(x_test)
```

```
23
```

```
x_train.class_indices
```

```
{'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}
```

7. TESTING

Testing The Model.

IMPORT LIBRARIES

```
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 cv2
```

unzip the file

```
!unzip '/content/drive/MyDrive/IBMPROJECT/conversation engine for deaf and dumb.zip'
```

```
Streaming output truncated to the last 5000 lines.
 extracting: Dataset/training_set/G/1225.png
 extracting: Dataset/training_set/G/1226.png
 extracting: Dataset/training_set/G/1227.png
 extracting: Dataset/training_set/G/1228.png
 extracting: Dataset/training_set/G/1229.png
  inflating: Dataset/training_set/G/123.png
 extracting: Dataset/training_set/G/1230.png
 extracting: Dataset/training_set/G/1231.png
```

TRAIN AND TEST THE DATA

```
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)
```

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

```
Found 15750 images belonging to 9 classes.
```

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

```
Found 2250 images belonging to 9 classes.
```

7.1. TEST THE MODEL

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

```
model = load_model('/content/Real_time.h5')
```

```
img = image.load_img('/content/Dataset/test_set/H/107.png',target_size = (100,100))
img
```

```
from skimage.transform import resize
def detect(frame):
    img=image.img_to_array(frame)
    img = resize(img,(64,64,1))
    img = np.expand_dims(img,axis=0)
    pred=np.argmax(model.predict(img))
    op=['A','B','C','D','E','F','G','H','I']
    print("THE PREDICTED LETTER IS ",op[pred])
```

```
img=image.load_img("/content/Dataset/test_set/H/107.png")
detect(img)
```

```
1/1 [==============================] - 0s 28ms/step
THE PREDICTED LETTER IS  H
```

8. RESULT

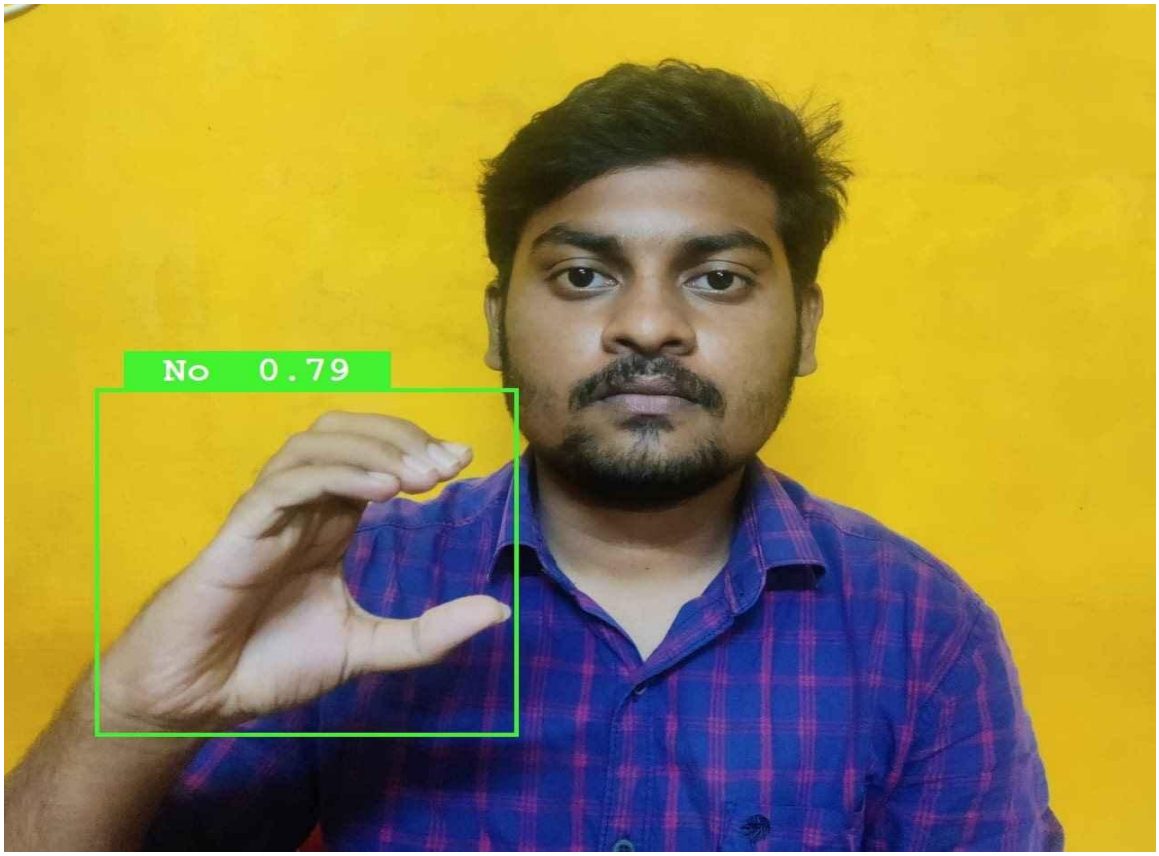
Output Screenshot:

Our model recognize the following sign languages:

I LOVE YOU

HELLO

NO



THANK YOU

YES

9. ADVANTAGES & DISADVANTAGES:

Advantage:

[1] Integrate the deaf and mute community into the conventional world..
[2] Smoothen the day-to-day interaction of deaf and mute people.
[3] Recognize the diverse range of sign languages present around the globe.
[4] Provide a feasible platform for the conventional world to interact with the deaf and mute community.
[5] Eliminate discrimination because of the perceptable communication gap.

Disadvantage:

[1] High Cost.
[2] Accuracy may vary.
[3] Highly depend on the composition of the images.
[4] As the quantity/quality of images in the dataset is low, the accuracy is not great, but that can easily be improved by change in dataset.

10. CONCLUSION

The proposed communication system between Deaf and Dumb people and ordinary people are aiming for it when bridging the communication gap between two societies. Several work is done earlier in this area, but this paper adds in complete two - sided communication in an efficient manner because the system is implemented as one Handy mobile application. So, it really serves its needs in all aspects. The above strategies prove to be efficient In terms of time and accuracy. Further improvements can be done in the implementation of the communicator with other sign language such as American Sign Language, Accent recognition for different accents throughout Globe, recognition of emotions in sign language and language Translation.

11. FUTURE SCOPE:

- Proposed systems scope is related with education of dumb peoples. Dumb people faces many problems when normal person could not understand their language. They were facing communication gap with normal peoples.
- For communication between deaf person and a second person, a mediator is required to translate sign language of deaf person. But a mediator is required to know the sign language used by deaf person. But this is not always possible since there are multiple sign languages for multiple languages. So to understand all sign languages, Hand gestures of deaf peoples by normal peoples this system is proposed. System gives output in the form of sound.
- Upgrade datasets and training to recognize combination of static gestures.
- Interpretation of facial expression by the real-time system.
- Inclusion of different types of sign languages.
- Integrating more words into the vocabulary of the system.

12. APPENDIX

1.]INSTALLING THE KERAS ,INSTALLING THE TENSORFLOW

```
!pip install Keras==2.2.4 !pip install tensorflow==2.7
```

2.]IMPORTING LIBRARIES TO BUILD MODEL.

```
#library to train the model
import keras
import tensorflow


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

3.]IMPORTING LIBRARIES FOR IMAGE AUGMENTATION.

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

4.]ADDING STREAMING_BODY_OBJECT FOR DATASET.ZIP

```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
```

5.]UNZIPPING THE DATASET

```
from io import BytesIO
import zipfile
unzip=zipfile.ZipFile(BytesIO(streaming_body_6.read()),'r')
file_paths=unzip.namelist()
for path in file_paths:
    unzip.extract(path)
```

6.]TRAINING AND TESTING IMAGES UNDER CLASSES

```
x_train=train_datagen.flow_from_directory("/home/wsuser/work/Dataset/training_set",target_size=(64,64),class_mode="categorical",batch_size=25)
```

```
Found 15750 images belonging to 9 classes.
```

```
x_test=test_datagen.flow_from_directory("/home/wsuser/work/Dataset/test_set",target_size=(64,64),
class_mode="categorical" , batch_size=25)
```

```
Found 2250 images belonging to 9 classes.
```

7.]TOTAL CLASSES UNDER TRAINING AND TESTING.

```
x_train.class_indices
```

```
{'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}
```

```
x_test.class_indices
```

```
{'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}
```

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

```
test_datagen=ImageDataGenerator(rescale=1./255)
```

8.]MODEL BUILDING USING CNN

```
model=Sequential()
```

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

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

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

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 32)        896

 max_pooling2d (MaxPooling2D  (None, 31, 31, 32)       0
 )

 flatten (Flatten)           (None, 30752)             0

=================================================================
Total params: 896
Trainable params: 896
Non-trainable params: 0
_________________________________________________________________
```

9.]ADDING LAYERS FOR MODEL TRAINING.

HIDDEN LAYERS

```
model.add(Dense(units = 300, activation='relu'))
#model.add(Dense(unit = 150,init = "uniform" activation='softmax'))
```

OUTPUT LAYERS

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

10.]OPTIMIZING THE MODEL

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

```
len(x_train)
```

```
630
```

```
len(x_test)
```

```
90
```

11.]FITTING THE MODEL

```
### model.fit_generator(x_train,steps_per_epoch=len(x_train),validation_data=x_test,validation_steps=len(x_test),epochs=10)
# Fitting the Model Generator
model.fit_generator(x_train,steps_per_epoch=630,epochs=10,validation_data=x_test,validation_steps=90)
#model.fit(x_train, epochs=100, verbose=1)
```

12.]SAVING THE MODEL

```
ls
```

```
Dataset/
```

```
pwd
```

```
'/home/wsuser/work'
```

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

```
Dataset.h5
```

13.]CONVERTING ZIP FILE TO TAR FILE FOR LOCAL USE.

```
#converting the model to tar
!tar -zcvf image.Classification.model_new.tgz Dataset.h5
```

```
Dataset.h5
```

```
ls -1
```

```
Dataset/
Dataset.h5
image.Classification.model_new.tgz
test_set/
training_set/
```

14.]INSTALLING WATSON MACHINE LEARNING CLIENT SOFTWARE

```
#installing the machine learning repository
!pip install watson_machine_learning_client --upgrade
```

```
Collecting watson_machine_learning_client
  Downloading watson_machine_learning_client-1.0.391-py3-none-any.whl (538 kB)
     |████████████████████████████████| 538 kB 18.5 MB/s eta 0:00:01
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson_machine_learning_client) (0.3.3)
Requirement already satisfied: tqdm in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson_machine_learning_client) (4.62.3)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson_machine_learning_client) (1.26.7)
Requirement already satisfied: boto3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson_machine_learning_client) (1.18.21)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson_machine_learning_client) (0.8.9)
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson_machine_learning_client) (2022.9.24)
Requirement already satisfied: pandas in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson_machine_learning_client) (1.3.4)
Requirement already satisfied: ibm-cos-sdk in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson_machine_learning_client) (2.11.0)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson_machine_learning_client) (2.26.0)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3->watson_machine_learning_c
lient) (0.10.0)
Requirement already satisfied: botocore<1.22.0,>=1.21.21 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3->watson_machine_learnin
g_client) (1.21.41)
```

```
Requirement already satisfied: numpy>=1.17.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas->watson_machine_learning_client)
(1.20.3)
Installing collected packages: watson-machine-learning-client
Successfully installed watson-machine-learning-client-1.0.391
```

15.]IMPORTING APICLIENT FOR DEPLOYING.

```
from ibm_watson_machine_learning import APIClient
url_credentials = {
    "url": "https://us-south.ml.cloud.ibm.com",
    #"apikey": "sqLVTXSP3nnAKfzJ1rKRKCpNzS_XZ8_HXa9FRwV7BvOP"
    "apikey":  "yVlgJh_0MVtYQmrWl9PAa6M60YXRYSkm0BXYZjlfnmrz"
}
client = APIClient(url_credentials)
```

```
client = APIClient(url_credentials)
client
```

16.]CREATING API_CLIENT SPACE ID.

```
def guid_from_space_name(client, space_name):
    space = client.spaces.get_details()
    return(next(item for item in space['resources'] if item['entity']['name'] == space_name)['metadata']['id'])
```

```
space_uid = guid_from_space_name(client, 'newspace')
print("space UID = " + space_uid)
```

```
space UID = 26031c6a-3567-437f-9ccb-d8ca0f32a42f
```

```
client.set.default_space(space_uid)
```

```
'SUCCESS'
```

17.]STORING THE MODEL_ID FOR DATASET.H5

```
#store the model
model_details = client.repository.store_model(model='image-Classification-model_new.tgz',meta_props={
    client.repository.ModelMetaNames.NAME: "CNN",
    client.repository.ModelMetaNames.TYPE: "keras_2.2.4",
    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_spec_uid
})
model_id = client.repository.get_model_uid(model_details)
```

18.]DOWNLOADING THE TAR FILE ON CLIENT REPOSITORY

```
client.repository.download(model_id, 'my_model.tar.gz')
```

19.]TEST THE MODEL

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

20.]LOADING THE DATASET

```
#Load the model
model=load_model('Dataset.h5')
```

21.]ADDING STREAMING_BODY FOR TEST IMAGE.

22.]TESTING ON SEVERAL TESTING IMAGES

```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
```

23.]IBM DEPLOYMENT

```
!pip install watson-machine-learning-client
```

```
from ibm_watson_machine learning import APIClient
wml_credentials={
"url":"https://us-south.ml.cloud.ibm.com",
"apikey":"x91CJTUTrrIfLvrXsKf8yLyI1KHb3JV0Y7Qrwy1zilb2"
}
client=APIClient(wml_credentials)
```

CLIENT

```
def guid_space_name(client,animal_deploy):
space-client.spaces.get_details()
return(next(item for item in space[' resources'] if iten['entity']['name']= animal_deploy)["metadata']['id'])
```

```
space_uid-guid_space_name(client,'animal_deploy")
print("Space UID "+space_uid)
```

```
client.set.default_space(space_uid)
```

```
client,software specifications.list(200)
```

```
software_space_uid=client.software_specifications.get_uid_by_name('tensorflow_rt22.1-py3.9¹)
```

```
software_space_uid
```

```
model_details=client.repository.store_model(model='Dataset.tgz',meta_props={
```