Government college of engineering Sengipatti

Thanjavur

Real-Time Communication System Powered By AI For Specially Abled

TEAM ID: PNT2022TMID47262

TEAM MEMBERS:

1. Dharshini.D	(822719106013)
2. Yashini.K	(822719106049)
3. DurgaDevi.A	(822719106014)
4. SubhaDharshini.G	(822719106043)

5. Anusaya.B.G (822719106004)

CONTENTS

- 1. INTRODUCTION
- 1.1 Abstract
 - 1.2 Project Overview
- 2. LITERATURE SURVEY
- 2.1 Existing problem
- 2.2 References
- 2.3 Problem Statement Definition
- 3. IDEATION & PROPOSED SOLUTION
- 3.1 Empathy Map Canvas
- 3.2 Ideation & Brainstorming
- 3.3 Proposed Solution
- 3.4 Problem Solution fit
- 4. REQUIREMENT ANALYSIS
- 4.1 Functional requirement
- 4.2 Non-Functional requirements
- 5. PROJECT DESIGN
- 5.1 Data Flow Diagrams
- 5.2 Solution & Technical Architecture
- 5.3 User Stories
- 6. PROJECT PLANNING & SCHEDULING
- 6.1 Sprint Planning & Estimation
- 6.2 Sprint Delivery Schedule
- 6.3 Reports from JIRA
- 7. CODING & SOLUTIONING (Explain the features added in the project along withcode)
- 8. TESTING
- 8.1 Test Cases
- 8.2 User Acceptance Testing
- 9. RESULTS
- 9.1 Performance Metrics
- 10.ADVANTAGES & DISADVANTAGES
- 11.CONCLUSION
- 12.FUTURE SCOPE
- 13.APPENDIX
- Source Code & GitHub Link

1. INTRODUCTION

1.1 ABSTRACT

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

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

1.2 PROJECT OVERVIEW

Deaf and dumb people are humans at the deepest psychological level. Many of these people are not even exposed to sign languages and it is observed that it gives a great relief on a psychological level, when they find out about signing to connect themselves with others by expressing their love or emotions. About 5% population in world are suffering from hearing loss. Deaf and dumb people use sign language as their primary means to express their thoughts and ideas to the people around them with different hand and body gestures. There are only about 250 certified sign language interpreters in India for a deaf population of around 7 million. In this work, the design of prototype of an assistive device for Deaf-mute people is presented so as to reduce this communication gap with the normal people. This device is portable and can hang over the neck. This device allows the person to communicate with sign hand postures in order to recognize different gestures based signs. The controller of this assistive device is

developed for processing the images of gestures by employing various imageprocessing techniques and deep learning models to recognize the sign. This sign is converted into speech in real-time using text-to-speech module.

2. LITERATURE SURVEY

2.1 Existing problem:

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.

According to the World Health Organization, the world population experiencing hearing and speech challenges approximates over 466 million people globally [1,2]. With such disability, instead unequally distributed resources, these people are vulnerable to discrimination [3]. The fact that every human being, abled or disabled, is entitled to a good life with equal opportunities calls for affirmative action [4,10]. This society requires attention from all quarters, especially on technological enhancement, to ensure the disabled get a comfortable life [5,16,20]. With the number increasing significantly, something needs to be done. The deaf and dumb are introverts, remaining engraved in their thoughtful world. Communication, which is essential in human life, is challenging. Humans are social beings, and effective communication is necessary [6, 22,24]. The development of technology should, therefore, serve to improve their lives as well [7]. The introduction of an application that can be used by the deaf and dumb will be a great innovation. It will not only make life easier but will as well increase their life opportunities, including employability. The deaf and dumb category must be involved within technology on PC experience as they involved in technology on smartphones. D- talk application provides this experience for them by reading their hand movements and displays a certain function.

2.2 REFRENCES:

- 1. 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.
- 2. Raheja, J. L., Mishra, A., & Chaudhary, A. (2016). Indian sign language recognition using SVM. Pattern Recognition and Image Analysis, 26(2), 434-441.

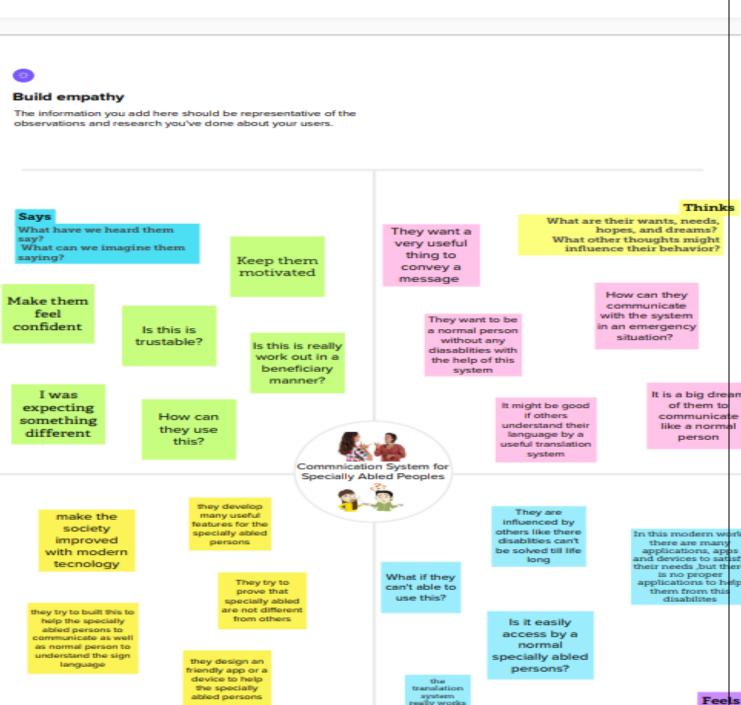
- 3. Jin, C. M., Omar, Z., & Jaward, M. H. (2016, May). A mobile application of American sign language translation via image processing algorithms. In 2016 IEEE Region 10 Symposium (TENSYMP) (pp. 104109). IEEE
- 4. Neiva, D. H., &Zanchettin, C. (2018). Gesture recognition: a review focusing on sign language in a mobile context. Expert Systems with Applications, 103, 159-183.

2.3 Problem Statement Definition:

The image recognition process is a process that enables the input of the sign language into the application for necessary processing [20,31,46]. The process requires a sign to be made in front of the webcam. The computer captures the sign made via the webcam and stores the different images made. Images that come from the camera will be resized, and the resolution will change. The colors will change to grayscale image and then to black and white images while editing the images [25, 33,47]. There several techniques used to extract the image, such as SIFT, SURF, BRISK, and HSV algorithms. Compared to standard algorithms, neural networks can solve somewhat complicated issues at a much easier level about the complexity of algorithms. Neural networks can solve somewhat complicated issues at a much easier level concerning the complexity of algorithms [26, 30]. The neural network builds to mimic human brain neural function but with the mathematical functions [31, 33,38]. The training phase was based on storing the images in the database. The database contained images of hands, both men and women. The training was based on identifying all possible signs that can be made using one hand. For this purpose, 30 different images with different levels of lights and duration were captured and stored in the database. These images were used as training images that will help in making the right decision for the tasks. The database contained over 1000 images of unique hands and signs.

3.IDEATION & PROPOSED SOLUTION

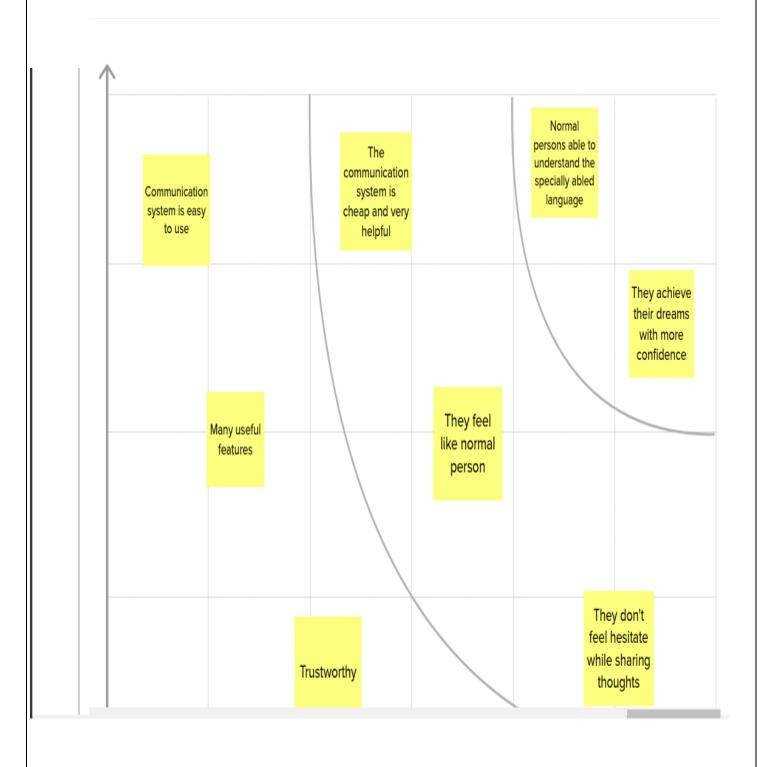
3.1Empathy Map Canvas



What behavior have we observed? What can we imagine them doing?

What are their fears, frustrations and anxieties What other feelings migh influence their behavior

3.2 Ideation & Brainstorming



3.3 Proposed Solution

S. No:	Parameter	Description
1.	Problem Statement (Problem to besolved)	The sixth sense is a multi- platform app for aiding the people in need that is people who are handicapped in the form of lack of speech (dumb), lack of hearing (deaf), lack of sight (blind), lack of judicial power to differentiate between objects (visual agnosia) and people suffering from autism (characterized by great difficulty in communicating and forming relationships with other people and in using language and abstract concepts). Our current implementation of the product is on two platforms, namely, mobile and a web app.

2.	ldea / Solution	The current implementation deals
	description	with object recognition and text
		to speech and a speech to text
		converter. The speech to text
		converter and text to speech
		converter utilized the Web
		Speech API (Application Program
		Interface) for the website and text
		to speech and speech to text
		library for the mobile platform.
		The object recognition wouldn't
		fetch enough use out of a
		website. Hence, it has been
		implemented on the mobile app
		utilizing the Firebase ML toolkit
		and different pre-trained models,
		which are both available offline as
		well as online.

3.	Novelty / Uniqueness	The world does not want just machine to do what they are told but even expect devices to work like us. Machine learning, a subsection of AI, is the hottest technology right now being implemented daily basis and is to be supposed to reach its peak in the next decade. Now, as the world has become a better place by providing everything at ease through technology to humans, we need to utilize the technology for differently-abled people.
4.	Social Impact / Customer Satisfaction	Thus, customer satisfaction has benefits as it helps minimize extra costs, enables industry know their repeat customer better, which could help in improving future service. Higher accuracy could be achieved in the future scope of the implementation through the use of custom models for object detection and text recognition as it could take into account the cases of objects for differently-abled people and work on those only yielding faster and accurate results.
5.	Business Model (Revenue Model)	 The major contribution of the work is: Integration of multiple modules to provide a single application to aid people of different disabilities. All the modules are researched solely rather than have a single source for all. This is the aim of work being performed in this work. An innovative approach for text to speech is implemented to provide a faster and convenient approach for mute to communicate through SAM (Speech Assisted for Mute).

	Scalability of the	The text to speech and Speech to
6.	Solution	Text Engine was based on the
		Google Speech Engine. IT has a
		similar implementation, as
		discussed in the website
		counterpart. All these sections
		have been then integrated into a
		single application to provide a
		single solution for all.

3.4 Problem Solution fit





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
FR- 2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Authentication	Authentication through Facial recognition Authentication through Password authentication protocol
FR- 4	External interfaces	Robots and other tools provide home-based care and other assistance, allowing people with disabilities to live independently
FR- 5	Transaction Processing	More application can use to translate the sign language like D talk in the system
FR- 6	Reporting	There is a growing feeling that we need to do more, to help make the lives of people with disabilities easier
FR- 7	Business rules	Human augmentation and Practical accuracy are responsible for AI business rules.

4.2 NON FUNCTIONAL REQUIREMENTS:

FR No.	Non-Functional Requirement	Description
NFR- 1	Usability	Provides personalised learning experiences tailored to the specific needs of students with disabilities.
NFR- 2	Security	Set the inclusion and exclusion criteria, Report the results in the survey.

NFR-3	Reliability	It setting the pace of the future and helping people in need.
NFR- 4	Performance	Enables people with disabilities to step into a world where their difficulties are understood and taken into account.
NFR- 5	Availability	Technology solutions that mimic humans and use logic from playing chess to solving equations and Machine learning is one of the technologies.
NFR- 6	Scalability	The improvement in the specially abled persons interaction with the environments.

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Milestone Activity Plan.

Use the below template to create product backlog and sprint schedule

Milestone Function (Epic)		Milestone Story Number	Story / Task	
Milestone -1	Data Collection	M1	we're collecting dataset for building our project and creating two folders, one for training and another one for testing.	
Milestone-2	Image preprocessing	M2	Importing image data generator libraries and applying image data generator functionality to train the test set.	
Milestone-3	Model Building	M3	Importing the model building libraries, Initializing the model, Adding Convolution layers, Adding the Pooling layers, Adding the Flatten layers, Adding Dense layers, Compiling the model Fit and Save the model.	
Milestone-4	Testing the model	M4	Import the packages first. Then we save the model and Load the test image, preprocess it and predict it.	
Milestone-5	Application layer	M5	Build the flask application and the HTML pages.	
Milestone-6	Train CNN model	M6	Register for IBM Cloud and train Image Classification Model.	
Milestone-6	Final result	M7	To ensure all the activities and resulting the final output.	

<u>6.2 Sprint Delivery Schedule :</u>

Sprint	Functional Requirement (Epic)	User Story I	Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1		CollectDataset.	9	High	CHETHAN P GOKUL P
Sprint-1		USN-2		Image preprocessing	8	Medium	CHARLES J PUGAZAHANDHI S
Sprint-2	Model Building	USN-3		Import the required libraries, add the necessary layers and compile the model	10	High	CHETHAN P PUGAZAHANDHI
Sprint-2		USN-4		Training the image classification model using CNN	7	Medium	GOKUL P CHARLES J
Sprint-3	Training and Testing	USN-5		Training the model and testing the model's performance	9	High	CHETHAN P GOKUL P
Sprint-4	Implementation o the application	f USN-6		Converting the input sign language images into English alphabets	8	Medium	CHARLES J PUGAZHANDHI S
	Sprint-4 Te	st the model	USN-8	Model testing	8	Medium	Madhumitha

:

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Model Building

Importing The Required Model Building Libraries

```
In [ ]: from keras.models import Sequential, load_model
             from keras.layers.core import Dense, Dropout, Activation
             from keras.utils import np_utils
            train\_datagen = ImageDataGenerator(rescale = 1/255, zoom\_range = 0.2, horizontal\_flip = True, vertical\_flip = False)
             test_datagen = ImageDataGenerator(rescale=1/255)
   In [ ]: # Training Dataset
             x\_train=train\_datagen.flow\_from\_directory(r'/content/drive/MyDrive/Dataset/training\_set', target\_size=(64,64), class\_mode='categorical', batch\_size=900)
             # Testing Dataset
            x\_test=test\_datagen.flow\_from\_directory(r'/content/drive/MyDrive/Dataset/test\_set', target\_size=(64,64), \ class\_mode='categorical', batch\_size=900)
            Found 15760 images belonging to 9 classes.
            Found 2250 images belonging to 9 classes.
   In [ ]:
    print("Len x-train : ", len(x_train))
    print("Len x-test : ", len(x_test))
            Len x-train : 18
           Len x-test : 3
   In [ ]: # The Class Indices in Training Dataset
           x_train.class_indices
   Out[]: {'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}
        Model Creation
In []: # Importing Libraries
         from tensorflow.keras.models import Sequential
         from tensorflow.keras.layers import Convolution2D, MaxPooling2D, Flatten, Dense
In [ ]: dataset = pd.read_csv('E:\Datasets\Mall_Customers.csv')
```

Initializing The Model

In []: # Creating Model

model=Sequential()

```
In [ ]: from tensorflow.keras.preprocessing.image import ImageDataGenerator
 In [ ]: spatial_dropout=0.05
           recurrent_dropout=0.1
 In [ ]: # Training Datagen
           train_datagen = ImageDataGenerator(rescale=1/255,zoom_range=0.2,horizontal_flip=True,vertical_flip=False)
           test_datagen = ImageDataGenerator(rescale=1/255)
 In [ ]: # Training Dataset
           x_train=train_datagen.flow_from_directory(r'/content/drive/MyDrive/Dataset/training_set', target_size=(64,64), class_mode='categorical', batch_size=900)
           # Testing Dataset
          x_test=test_datagen.flow_from_directory(r'/content/drive/MyDrive/Dataset/test_set',target_size=(64,64), class_mode='categorical',batch_size=900)
          Found 15760 images belonging to 9 classes.
          Found 2250 images belonging to 9 classes.
 In [ ]: print("Len x-train : ", len(x_train))
    print("Len x-test : ", len(x_test))
          Len x-train : 18
          Len x-test : 3
 In [ ]: # The Class Indices in Training Dataset
           x_train.class_indices
 Out[ ]: {'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}
        Model Creation
In [ ]: # Importing Libraries
         from tensorflow.keras.models import Sequential
          from tensorflow.keras.layers import Convolution2D, MaxPooling2D, Flatten, Dense
```

Adding The Convolution Layer

```
In [ ]: import numpy as np
                   import matplotlib.pyplot as plt
 In [ ]: from tensorflow.keras.preprocessing.image import ImageDataGenerator
 In [ ]: # Training Datagen
                    train\_datagen = ImageDataGenerator(rescale=1/255, zoom\_range=0.2, horizontal\_flip=True, vertical\_flip=False)
                    # Testing Datagen
                    test_datagen = ImageDataGenerator(rescale=1/255)
 In [ ]: # Training Dataset
                    x_train=train_datagen.flow_from_directory(r'/content/drive/MyDrive/Dataset/training_set',target_size=(64,64), class_mode='categorical',batch_size=900)
                    # Testing Dataset
                   x_test=test_datagen.flow_from_directory(r'/content/drive/MyDrive/Dataset/test_set',target_size=(64,64), class_mode='categorical',batch_size=900)
                  Found 15760 images belonging to 9 classes.
                  Found 2250 images belonging to 9 classes.
 In [ ]: # Let img1 be an image with no features
                    img1 = np.array([np.array([200, 200]), np.array([200, 200])])
                    img2 = np.array([np.array([200, 200]), np.array([0, 0])])
                    img3 = np.array([np.array([200, 0]), np.array([200, 0])])
                    \label{lem:kernel_horizontal = np.array([np.array([2, 2]), np.array([-2, -2])])} kernel\_horizontal = np.array([np.array([2, 2]), np.array([-2, -2])])
                    print(kernel_horizontal, 'is a kernel for detecting horizontal edges')
                   \label{lem:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma:lemma
In [ ]: # We will apply the kernels on the images by
                  # elementwise multiplication followed by summation
def apply_kernel(img, kernel):
                          return np.sum(np.multiply(img, kernel))
                   # Visualizing img1
                  plt.imshow(img1)
                   plt.axis('off'
                   plt.title('img1')
                   # Checking for horizontal and vertical features in image1
                  print('Horizontal edge confidence score:', apply_kernel(img1, kernel_horizontal))
                  In [ ]: # Visualizing img2
                  plt.imshow(img2)
                   plt.axis('off')
                  plt.title('img2')
                   plt.show()
                   # Checking for horizontal and vertical features in imag
                  print('Horizontal edge confidence score:', apply_kernel(img2,
                                                                                                           kernel_horizontal))
                   print('Vertical edge confidence score:', apply_kernel(img2,
In [ ]:
                  # Visualizing img3
                  plt.imshow(img3)
plt.axis('off')
                  plt.title('img3')
                   plt.show()
                  # Checking for horizontal and vertical features in image3
print('Horizontal edge confidence score:', apply_kernel(img3,
                  kernel_horizontal))
print('Vertical edge confidence score:', apply_kernel(img3,
                                                                                                           kernel_vertical))
```

```
In [ ]:
    print("Len x-train : ", len(x_train))
    print("Len x-test : ", len(x_test))
        Len x-train : 18
        Len x-test : 3
In [ ]: # The Class Indices in Training Dataset
         x_train.class_indices
Out[]: {'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}
         Model Creation
In [ ]: # Importing Libraries
          from tensorflow.keras.models import Sequential
         from tensorflow.keras.layers import Convolution2D,MaxPooling2D,Flatten,Dense
In [ ]: # Creating Model
         model=Sequential()
In [ ]: # Adding Layers
         model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3)))
         Adding The Pooling Layer
In [ ]: from tensorflow.keras.preprocessing.image import ImageDataGenerator
In [ ]: import numpy as np
         from keras.models import Sequential
from keras.layers import MaxPooling2D
In [ ]: # define input image
          image = np.array([[2, 2, 7, 3],
                                           [9, 4, 6, 1],
                                           [8, 5, 2, 4],
                                           [3, 1, 2, 6]])
          image = image.reshape(1, 4, 4, 1)
In [ ]: # define model containing just a single max pooling layer
          model = Sequential(
                  [MaxPooling2D(pool_size = 2, strides = 2)])
          # generate pooled output
          output = model.predict(image)
In [ ]: # print output image
          output = np.squeeze(output)
          print(output)
          train_datagen = ImageDataGenerator(rescale=1/255,zoom_range=0.2,horizontal_flip=True,vertical_flip=False)
          # Testing Datagen
          test_datagen = ImageDataGenerator(rescale=1/255)
```

```
In []: # Training Dataset
          x_train=train_datagen.flow_from_directory(r'/content/drive/MyDrive/Dataset/training_set', target_size=(64,64), class_mode='categorical',batch_size=900)
          # Testing Dataset
          x\_test=test\_datagen.flow\_from\_directory(r'/content/drive/MyDrive/Dataset/test\_set', target\_size=(64,64), \ class\_mode='categorical', batch\_size=900)
         Found 15760 images belonging to 9 classes. Found 2250 images belonging to 9 classes.
In [ ]: print("Len x-train : ", len(x_train))
    print("Len x-test : ", len(x_test))
         Len x-train : 18
         Len x-test : 3
In [ ]: # The Class Indices in Training Dataset
         x_train.class_indices
Out[]: {'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}
         Model Creation
In [ ]: # Importing Libraries
          from tensorflow.keras.models import Sequential
          from tensorflow.keras.layers import Convolution2D, MaxPooling2D, Flatten, Dense
In [ ]: # Creating Model
          model=Sequential()
In [ ]: # Adding Layers
          model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3)))
In [ ]: model.add(MaxPooling2D(pool_size=(2,2)))
          Adding The Flatten Layer
 In [ ]: # importing numpy as np
           import numpy as np
 In [ ]: # declare flatten np gfg = np.array([[6, 9, 12], [8, 5, 2], [18, 21, 24]])
           # using array.flatten() method
flat_gfg = gfg.flatten(order='A')
           print(flat_gfg)
  In [ ]: from tensorflow.keras.preprocessing.image import ImageDataGenerator
  In [ ]: # Training Datagen
           train_datagen = ImageDataGenerator(rescale=1/255,zoom_range=0.2,horizontal_flip=True,vertical_flip=False)
# Testing Datagen
           test_datagen = ImageDataGenerator(rescale=1/255)
  In [ ]: # Training Dataset
           x_train=train_datagen.flow_from_directory(r'/content/drive/MyDrive/Dataset/training_set', target_size=(64,64), class_mode='categorical',batch_size=900)
           x_test=test_datagen.flow_from_directory(r'/content/drive/MyDrive/Dataset/test_set',target_size=(64,64), class_mode='categorical',batch_size=900)
           Found 15760 images belonging to 9 classes.
          Found 2250 images belonging to 9 classes.
 In [ ]: print("Len x-train : ", len(x_train))
    print("Len x-test : ", len(x_test))
           Len x-train : 18
          Len x-test : 3
```

```
In [ ]: # The Class Indices in Training Dataset
          x_train.class_indices
Out[]: {'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}
         Model Creation
In [ ]: model = Sequential()
          for i, feat in enumerate(args.conv_f):
              if i==0:
                   \verb|model-add(Conv2D(feat, input\_shape=x[0].shape, kernel\_size=3, padding = 'same', use\_bias=False)||
              else:
                  model.add(Conv2D(feat, kernel_size=3, padding = 'same',use_bias=False))
model.add(BatchNormalization())
                   model.add(LeakyReLU(alpha=args.conv_act))
                   model.add(Conv2D(feat, kernel_size=3, padding = 'same',use_bias=False))
                   model.add(BatchNormalization())
                   model.add(LeakyReLU(alpha=args.conv_act))
                   model.add(Dropout(args.conv_do[i]))
In [ ]: model.add(Flatten())
          #Input code here
          denseArgs = {'use_bias':False}
          for i,feat in enumerate(args.dense_f):
    model.add(Dense(feat,**denseArgs))
              model.add(BatchNormalization())
              model.add(LeakyReLU(alpha=args.dense_act))
              model.add(Dropout(args.dense_do[i]))
          model.add(Dense(1))
In [ ]: # Importing Libraries
          from tensorflow.keras.models import Sequential
          \textbf{from} \ \texttt{tensorflow}. \texttt{keras.layers} \ \textbf{import} \ \texttt{Convolution2D}, \texttt{MaxPooling2D}, \texttt{Flatten}, \texttt{Dense}
In [ ]: # Importing Libraries
          from tensorflow.keras.models import Sequential
          from tensorflow.keras.layers import Convolution2D,MaxPooling2D,Flatten,Dense
In [ ]: # Creating Model
          model=Sequential()
In [ ]: # Adding Layers
          model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3)))
In [ ]: model.add(MaxPooling2D(pool_size=(2,2)))
In [ ]: model.add(Flatten())
In [ ]: # Adding Dense Layers
          model.add(Dense(300,activation='relu'))
          model.add(Dense(150,activation='relu'))
          model.add(Dense(9,activation='softmax'))
```

Adding The Dense Layers

```
In [ ]: from tensorflow.keras.preprocessing.image import ImageDataGenerator
In [ ]: model.add(Dense(units=512, activation='relu'))
          model.add(Dense(units=9, activation='softmax'))
In [ ]:
         print("Adding dense layer on top")
model.add(layers.Flatten())
          model.add(layers.Dense(64, activation='relu'))
          model.add(layers.Dense(10))
In [ ]: print("Complete architecture of the model")
          model.summary()
In [ ]: # Training Datagen
         train_datagen = ImageDataGenerator(rescale=1/255,zoom_range=0.2,horizontal_flip=True,vertical_flip=False)
          test_datagen = ImageDataGenerator(rescale=1/255)
In [ ]: # Training Dataset
          x_traintrain_datagen.flow_from_directory(r'/content/drive/MyDrive/Dataset/training_set',target_size=(64,64), class_mode='categorical',batch_size=900'
         x_test=test_datagen.flow_from_directory(r'/content/drive/MyDrive/Dataset/test_set',target_size=(64,64), class_mode='categorical',batch_size=900)
         Found 15760 images belonging to 9 classes.
         Found 2250 images belonging to 9 classes.
In [ ]: print("Len x-train : ", len(x_train))
    print("Len x-test : ", len(x_test))
         Len x-train : 18
         Len x-test : 3
In [ ]: # The Class Indices in Training Dataset
         x_train.class_indices
Out[]: {'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}
        Model Creation
In [ ]: # Importing Libraries
         from tensorflow.keras.models import Sequential
         from tensorflow.keras.layers import Convolution2D,MaxPooling2D,Flatten,Dense
In [ ]: # Creating Model
         model=Sequential()
In [ ]: # Adding Layers
         model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3)))
In [ ]: model.add(MaxPooling2D(pool_size=(2,2)))
In [ ]: # Adding Dense Layers
         model.add(Dense(300,activation='relu'))
         model.add(Dense(150.activation='relu'))
         model.add(Dense(9,activation='softmax'))
In [ ]: # Compiling the Model
         model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
```

```
In [ ]: from tensorflow.keras.preprocessing.image
             import ImageDataGenerator
In [ ]: model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
In [ ]: # creating sample sourcecode to multiply two variables
              srcCode = 'x = 10\ny = 20\nmul = x * y\nprint("mul =", mul)'
              # Converting above source code to an executable
execCode = compile(srcCode, 'mulstring', 'exec')
              # Running the executable code. exec(execCode)
             # Training Datagen
train_datagen = ImageDataGenerator(rescale=1/255,zoom_range=0.2,horizontal_flip=True,vertical_flip=False)
# Testing Datagen
test_datagen = ImageDataGenerator(rescale=1/255)
In [ ]:
             # Training Dataset
x_train=train_datagen.flow_from_directory(r'/content/drive/MyDrive/Dataset/training_set',target_size=(64,64), class_mode='categorical',batch_size=900)
             # lesting Dataset

** Iesting Dataset

** Lest-test_datagen.flow_from_directory(r'/content/drive/MyDrive/Dataset/test_set', target_size=(64,64), class_mode='categorical',batch_size=900)
            Found 15760 images belonging to 9 classes. Found 2250 images belonging to 9 classes.
In [ ]: def compile_model_results(model, root="./"):
                    listing = glob.glob(root + '/models/' + model + '/*/best_pars.pkl')
                    dic_list = []
for file in listing:
                                  hyper_parameters load(file)
                         dic_list.append(tmp.to_dictionary())
                    df = pd.DataFrame(dic_list)
df['diff'] = df.test_F1 - df.forecast_F1
df['pci'] = abs(df.test_F1 - df.forecast_F1)
                    if not os.path.exists(root + '/figures/' + model ):
    os.makedirs(root + '/figures/' + model )
                    df.to_csv(root + '/figures/' + model + '/results.csv', index=False)
                    return df
In [ ]: # Set optimizer loss and metrics
    opt = Adam(1r=args.initial_lr, beta_1=0.99, beta_2=0.999, decay=1e-6)
    if args.net.find('caps') != -1:
        metrics = {'out_seg': dice_hard}
                    else:
metrics = [dice_hard]
                    loss, loss_weighting = get_loss(root=args.data_root_dir, split=args.split_num, net=args.net, recon_wei=args.recon_wei, choice=args.loss)
                    # If using CPU or single GPU
if args.gpus <= 1:
    uncomp_model.compile(optimizer=opt, loss=loss, loss_weights=loss_weighting, metrics=metrics)
    return uncomp_model
# If using multiple GPUs</pre>
                          e:
with tf.device("/cpu:0"):
uncomp_model.compile(optimizer=opt, loss=loss, loss_weights=loss_weighting, metrics=metrics)
model = multi_gpu_model(uncomp_model, gpus=args.gpus)
model._setattr__('callback_model', uncomp_model)
model.compile(optimizer=opt, loss=loss, loss_weights=loss_weighting, metrics=metrics)
              X = array[:,0:8]
Y = array[:,8]
test_size = 0.33
              seed = 7
X_train, X_test, Y_train, Y_test = model_selection.train_test_split(X, Y, test_size=test_size,
random_state=seed)
In [ ]: print("Len x-train : ", len(x_train))
    print("Len x-test : ", len(x_test))
             Len x-train : 18
Len x-test : 3
In [ ]: # The Class Indices in Training Dataset
x_train.class_indices
Out[ ]: {'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}
```

Model Compilation

```
In [ ]:
# Importing Libraries
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Convolution2D,MaxPooling2D,Flatten,Dense
In [ ]: # Creating Mode
                model=Sequential()
In [ ]: # Adding Layers
model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3)))
In []:
    # Adding Dense Layers
    model.add(Dense(300,activation='relu'))
    model.add(Dense(150,activation='relu'))
    model.add(Dense(9,activation='softmax'))
In [ ]: # Compiling the Model model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
In [ ]: # reading code from a file
f = open('main.py', 'r')
temp = f.read()
f.close()
                code = compile(temp, 'main.py', 'exec')
exec(code)
             Saving the Model
 In [ ]: model.save('asl_model_84_54.h5')
                Fit And Save The Model
   In [ ]: from tensorflow.keras.preprocessing.image import ImageDataGenerator
   In [ ]: # Training Datager
                 # Train_datagen = ImageDataGenerator(rescale=1/255,zoom_range=0.2,horizontal_flip=True,vertical_flip=False)
# Testing Datagen
test_datagen = ImageDataGenerator(rescale=1/255)
   In [ ]: # Training Dataset

x_train=train_datagen.flow_from_directory(r'/content/drive/MyDrive/Dataset/training_set', target_size=(64,64), class_mode='categorical', batch_size=900)
                  x_test-test_datagen.flow_from_directory(r'/content/drive/MyDrive/Dataset/test_set',target_size=(64,64), class_mode='categorical',batch_size=900)
                Found 15760 images belonging to 9 classes.
Found 2250 images belonging to 9 classes.
   In [ ]:
                  # Save Model Using PickLe
                 import pandas
from sklearn.import model_selection
from sklearn.linear_model import LogisticRegression
import pickle
   In [ ]: url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-
                 url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-
diabetes.data.csv"
names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
dataframe = pandas.read_csv(url, names=names)
array = dataframe.values
X = array[',9:8]
Y = array[',9:8]
test_size = 0.33
seed = 7
                  secd = 7
X_train, X_test, Y_train, Y_test = model_selection.train_test_split(X, Y, test_size=test_size,
random_state=seed)
  In []:
    # Fit the model on training set
    model = LogisticRegression()
    model.fit(X_train, Y_train)
    # save the model to disk
    filename = 'finalized_model.sav'
    pickle.dump(model, open(filename, 'wb'))
                 # Load the model from disk
loaded_model = pickle.load(open(filename, 'rb'))
result = loaded_model.score(X_test, Y_test)
print(result)
```

```
In [ ]: print("Len x-train : ", len(x_train))
    print("Len x-test : ", len(x_test))
      Len x-train : 18
In [ ]: # The Class Indices in Training Dataset
x_train.class_indices
Out[ ]: {'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}
      Model Creation
In [ ]: # Importing Libraries from tensorflow.keras.models import Sequential
      from tensorflow.keras.layers import Convolution2D, MaxPooling2D, Flatten, Dense
In [ ]: # Creating Mode
      model=Sequential()
In [ ]: # Adding Layers
      model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3)))
In [ ]: model.add(MaxPooling2D(pool_size=(2,2)))
In [ ]: model.add(Flatten())
In [ ]: # Adding Dense Layers
      model.add(Dense(300,activation='relu'))
model.add(Dense(150,activation='relu'))
      model.add(Dense(9,activation='softmax'))
In [ ]: # Compiling the Model
      model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
In [ ]: # Fitting the Model Generator
      \verb|model.fit_generator(x_train,steps_per_epoch=len(x_train),epochs=10,validation_data=x_test,validation_steps=len(x_test))|
      /usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future versio
      n. Please use 'Model.fit', which supports generators.
      Epoch 1/10
      Fnoch 2/10
      18/18 [====
                   =============== ] - 90s 5s/step - loss: 0.0040 - accuracy: 0.9995 - val_loss: 0.2074 - val_accuracy: 0.9773
      Epoch 3/10
18/18 [=====
               Epoch 4/10
      18/18 [=====
                Epoch 5/10
      18/18 [====
                    ========== ] - 88s 5s/step - loss: 0.0037 - accuracy: 0.9993 - val_loss: 0.2439 - val_accuracy: 0.9782
      Epoch 6/10
      Epoch 7/10
                18/18 F=====
      Epoch 8/10
      Epoch 9/10
      Saving the Model
In [ ]: model.save('asl_model_84_54.h5')
```

8. TESTING

8.1 Test Cases

Loading the Dataset & Image Data Generation

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
          train_datagen = ImageDataGenerator(rescale=1/255,zoom_range=0.2,horizontal_flip=True,vertical_flip=False)
# Testing Datagen
          test_datagen = ImageDataGenerator(rescale=1/255)
          # Training Datase
          x_train=train_datagen.flow_from_directory(r'C:\Users\india\Desktop\Final_Project\Dataset\test_set',target_size=(64,64), class_mode='categorical',batch
          *** test-test_datagen.flow_from_directory(r'C:\Users\india\Desktop\Final_Project\Dataset\training_set',target_size=(64,64), class_mode='categorical',bat
         Found 4969 images belonging to 9 classes.
         Found 4969 images belonging to 9 classes.
 In [26]:
         print("Len x-train : ", len(x_train))
print("Len x-test : ", len(x_test))
         Len x-train :
Len x-test : 6
In [27]: # The Class Indices in Training Dataset
x_train.class_indices
Out[27]: {'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}
         Model Creation
 In [28]:
         # Importing Libraries
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Convolution2D,MaxPooling2D,Flatten,Dense
 In [29]:
         # Creating Mode
          model=Sequential()
In [30]: # Adding Layers
model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3)))
         model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
         # Adding Hidden Layers
model.add(Dense(300,activation='relu'))
         model.add(Dense(150,activation='relu'))
        # Adding Output Layer
model.add(Dense(9,activation='softmax'))
In [31]: # Compiling the Model
         model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
In [32]: # Fitting the Model Generator
         \verb|model.fit(x_train, steps_per_epoch=len(x_train), epochs=10, validation_data=x_test, validation_steps=len(x_test))|
        Epoch 1/10
        6/6 [=====
Epoch 2/10
6/6 [=====
Epoch 3/10
                       ========= ] - 23s 4s/step - loss: 5.1206 - accuracy: 0.1690 - val_loss: 3.6505 - val_accuracy: 0.3119
                               ======] - 22s 4s/step - loss: 2.3945 - accuracy: 0.3266 - val_loss: 1.5087 - val_accuracy: 0.4991
        6/6 [=====
Epoch 4/10
6/6 [=====
Epoch 5/10
                             :=======] - 22s 4s/step - loss: 1.4384 - accuracy: 0.4037 - val loss: 1.0430 - val accuracy: 0.5836
                             6/6 [==
        Epoch 6/10
6/6 [=====
Epoch 7/10
                             6/6 [=
        Epoch 8/10
        6/6 [=====
Epoch 9/10
                          :=======] - 22s 4s/step - loss: 0.2108 - accuracy: 0.9612 - val loss: 0.0880 - val accuracy: 0.9863
        6/6 [=
        Epoch 10/10
                      Out[32]:
```

8.2 User Acceptance Testing

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [ProductName] project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

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

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal	
By Design	11	2	3	2	18	
Duplicate	1	3	4	0	8	
External	3	5	0	0	8	
Fixed	12	2	5	22	41	
Not Reproduced	0	1	0	0	1	
Skipped	0	0	1	2	3	
Won't Fix	0	4	1	1	7	
Totals	27	17	14	27	86	

3. Test Case Analysis

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

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	8	0	0	8
Client Application	49	0	0	49
Security	4	0	0	4

Outsource Shipping	4	0	0	4
Exception Reporting	11	0	0	11
Final Report Output	2	0	0	2
Version Control	1	0	0	1

9. RESULTS

9.1 Performance Metrics

Technical Mallis Evaluation Marrina								
	Name .	magray (%)	Decigios	the fraction Mercu & Scottag Little Ia	-	Decripton	Renderd Suspine	Strating Secretions
			To not all in continued a facilitating	Number of Tests 1 Marion Ministrative Trial 1 Mari	-	The efficient Militage of Hispaniers		Section Spine - 10 Nove
6115	Technical Training & Assignment	-		Assignment	1.0	Sheets francescope	94	
	Technical Factory & Recognition	1,440	. Particularies the Day & the and of participation residence	Antonomi i Antonomi i	11	Stocks framework Stocks framework		Compress 1 Parts 1 No Scientist Comment 2 Parts 1 No Scientist Compress 2 Parts 1 No Scientist Compress 2 Parts 1 No Scientist
				Augment	1.	Times, temperary		Company 1 Parks 1 No Common
	Associate & Problem Batting 19			4. Separation & Discourse		Aprillant distance of the budget on the 3 emily fits, and, when, when, why of the protoco, below the lamping provider.	Tex	Complete & Parker I, No Colombia
			The house set is measured and droug based on the unique services had your code and in the proposed values. The beautiful stand author the holisting broughteen		1.	Service to be proper pro-	-	
			The sales of the billioning template. 1. Index the Full Comment of the sales of th		-	More ranging a violent is a primary, it is reported.	-	
				1 Selse to Problem Selement		an interesting who pay to dainly any fair, so you see heap that heart, if always, furthering, per principal beauty plate growthing (Yes	Company of Fairs & Navigation
				March March College		Service and product a better and an extremely		
				3. Statistics & Provide blue		Bankering promise a few actions account the control promise account of the control promise	Tex	Complete II, Fache II, No Searcher
-						atorig		
	Regulation and Analysis unless (1991) of Photology		The best of its seed held of its Submissor of Francisco			Credit Loborius (suche) helps to contrated the excitorios of customer	Yes	Congress 4,74444 CARC Statement
		100,		Interview to Requirements (Codemon Asserts March		Andreador and the state of the		
						Analysis for equinosess, proposed from the formation of the steel of the steel re- teriorises to self-equinoses.	Yes	Complete in Partie A, No. 5 (contract
				Requirement Assistant (Possistanti, Spendisme, Technical) / Fine Chara	-			
						Name which of the companion is transmission as he act most and have tray off the dishease. For that and has been able to departed or worker from the product orange, in agriculture.		Comprise is Twice Charleston
				Integration & Building		project, in paginisms		
						Charles of the Contract of		
			The term of the species and some based or the scientists of proper design and the scientists of the sc	Proposed solition Solition Solition Streets		Shall reject the originates of proposed selection, models, soorial or because where our beautiful of	Yes	Complete: 4: Factor 1, No. Common
	Project Design using Design	100	The cases will be assumed and across based on the substance of course based on the substance of the substanc	Francisco d'Oprope Bross Surrey House				
	Floreng	177		-	1			
			3. Problem - Schollen PV In Schollen Architecture	Printer Salata Pr		The a template to high blands, substance with high as distance of substant adoption, not one first report of parting and gas a code, consider of the committee obtains	Yes	Company - 1, Fartac 4, No. Societies
				Spiritus Artifiquian		That you be an included agent of projective	Yes	Corporal -1, False 1, No. Scientific
			The near of the seasons of treet bead or the obtains of technology is finding the progression with the	Suitable Statement		Control to the second second of the second state of the second se	-	Salkander Chair 191 Se
				- in the trystam colours in a 2 chapter would also 2 in 2 chapter		Sergion of collegement with a constraint feet for the		
				Is I hallow per loss & helps consisting		www.m		araconomico de la result
	Technology Toxik	Ph.		Open Busins Pratraments	1.	And in charge of particular fermion is to		The second second sec
					-	Name to harder of ATTs this state is the value	-	1900-1907 7 8040 Marchell 2 (1904) 1 4 6074 - 1 8040 1 6070 - 1 8040 Surger Schillens 4 1
				Total Ingespeer			~	11 MTs 1 Mess
						Creat petro, the solution regressed as black to benefitied and the Track New Hay disjustmented aloud refers assemble to the	-	Byterydect progress - 1
	Project Planning using Agine Methodologist		This name of the assessment of a containing of the desirable points assessing to a containing the desirable points assessing to the purpose from the assessment of the purpose from the assessment of the containing the assessment of the containing the assessment of the containing the containi	Proper Winnesser & Testa		Trave to leader (more), if your localismen, to each reported below, reported on, estimated with the drawn 6-epts, but become	Yes	Services & Select 19-7, Co.
				The State of the S	272	with the drawn & spirit, but becomes		
		n				Special control of the special control of the		Name of Spirit planning markets my
				Sprint Selbody Flan		Special property in an experimental and the later of the appear of applications of the state of the special and the decimal in the special and the decimal in the special and	fee	Minima of Feature Minima of Colors Street and Colors Street St. St. Co. Co. Co. Co. Co. Co. Co. Co. Co. Co
				Project Program Tracking		Conductor of many is made expension. Providence of made. These the programs of all table. Subsections of the	-	hai hanning - 1. Fartai - 1 Garagin
					-	NATURE DR.		
				Name management out for agite planning (e.g. Trade, Jrs. 45.)		(All along of project) date; may appropriate best for the project defines;	*	Markette sand — E-marks Argumentari — E-marks
_				1000 1000				1 - 175 Blacks 1 Balls
	Goling & Nationary 191		This many of the conservated of the following parameters of the conservation of the co	No. of Functional Pasteries Included to the automoti		Assert for halos presented in Historymon	**	1 - 22% Majoring 1 Marin 21 - 22% Majoring 1 Marin 1880, Majoring 7 Admin.
				Code Lagran, Francisco per Terrescotto		States for collection of money	144	Average it lives -1 forward -2
					-			
۴.		m		Witness of Squatters, Sprants Programming Spring Manager Millioness		Savage the distinguish of Republica, Payments Propositions, Carried Mesony Utiliation	**	henge 6 liss 1 lisates 5
				Petrograp & Constitute		France de Calegging & Transdelle Capabille de Recorne	-	Average - 0. Stock - 1. Stockholt - 0
				magation and	1.			oragi cassi cassar s
				Enterior handing		Street the Exception Franchis substitly of the costs developed:	-	hongs it live 1 beater 2
	Anapters laring to		STANT MINNEY	GRT between	-1	Stocker has accomply of 1967 chapted (Dallow Appropris, that DBM congressions, and)	***	the Santhorn E. Partie - 1 Carpet
				OFTimelyi	-	Name of the CART composition of the	760	Notaces : Charles : Company
		200		DITTEMENTOR .		Annual in every studies on the represent	- 5	Na Calenter - C Party - 1 Company
				INT Report Submission	- 1	Assets the report sales than	Yes	Indianamic S. Parker - I. Complete
				Militarios el Tras maragament Carlo	1	Number the ultimated of Toron Day Steel, Agency, London etc.	~	National Charter Company
	Participate and Torontog (19)		The notice of the economic of a count balant of the inflaming is appeared from the count of the first plants. In a special from the count of the cou	Application Performance Menting				
				Applicates Perferenced Medical Date class Personnel class Control class		house the application participation contains substitute or world solid participation substitute to the	Tex	Selection (1975 - 1975
				Commer equipment in				Sealer Wis-1876
				- AV 65				
		~		Motor Fundament Marine				
				Confidence representation of the Confidence of t	1			
110				Rocal		Assess the results partie reports problems autorished by early unity, missing partier reasons reading basis	Year	Sebiliatory 10% 50% George 57% 70%
				Parame Bangler Hill rame				Excelled - 10% - 10%
				NSC Receive Operating Characteristics (union Rft or ROC curve.	1			
				Supposed the Story Marine	-			
				Endity and Engine End Engine and Amount Edwar Residence United and Place Fibers		Access the constitutes shallows been in the	700	Rabin Average: 7 (100) Substitution - (47) - 50% Cent. 1075, 1076 Examina - 50% - 1076

10.ADVANTAGES & DISADVANTAGES

Advantages:

- It is a cost-effective way of getting several people from different locations to attend meetings and conferences.
- It enables employees from across the world to communicate with each other 24×7 and share ideas or solve problems quickly.

Disadvantages:

- Also accuracy depends upon distance between camera and object.
- It takes a lot of time to listen, speak, read, or write to someone.

11. 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. It provides complete two sided communication in an efficient manner between the disabled and the normal person.

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.

12. FUTURE SCOPE

The speech-to-text and text-to-speech technologies helped those people who had difficulties in communicating or expressing their feelings to the normal people.

This reduces the communication gap between the normal people and the specially abled people.

Using image pre-processing and Artificial Intelligence it is easy to understand the context of objects and clearly explains it to the people who use it for communication.

13.APPENDIX

```
SOURCE CODE:
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0,</pre>
shrink-to-fit=no">
  <title>SmartBridge_WebApp_VideoTemplate</title>
  <link rel="stylesheet"</pre>
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css">
  k rel="stylesheet"
href="https://use.fontawesome.com/releases/v5.12.0/css/all.css">
  k rel="stylesheet" href="assets/css/Banner-Heading-Image.css">
  k rel="stylesheet" href="assets/css/Navbar-Centered-Brand.css">
  k rel="stylesheet" href="assets/css/styles.css">
</head>
<body style="background: rgb(39,43,48);">
  <nav class="navbar navbar-light navbar-expand-md py-3" style="background:</pre>
#212529:">
    <div class="container">
       <div></div><a class="navbar-brand d-flex align-items-center"
href="#"><span
           class="bs-icon-sm bs-icon-rounded bs-icon-primary d-flex justify-
content-center align-items-center me-2 bs-icon"><i
              class="fas fa-flask"></i></span><span style="color:
rgb(255,255,255);">Real-Time Communication
           System Powered By AI For Specially Abled</span></a>
       <div></div>
    </div>
  </nav>
  <section>
    <div class="d-flex flex-column justify-content-center align-items-center">
       <div class="d-flex flex-column justify-content-center align-items-center"</pre>
id="div-video-feed"
```

```
style="width: 640px;height: 480px;margin: 10px;min-height:
480px;min-width: 640px;border-radius: 10px;border: 4px dashed
rgb(255,255,255);">
         <img src="{{ url_for('video_feed') }}" style="width: 100%;height:</pre>
100%; color: rgb(255,255,255); text-align: center; font-size: 20px;"
            alt="Camera Access Not Provided!">
       </div>
    </div>
     <div class="d-flex flex-column justify-content-center align-items-center"</pre>
style="margin-bottom: 10px;"><button
         class="btn btn-info" type="button" data-bs-target="#modal-1" data-bs-
toggle="modal">Quick Reference
         -<strong> ASL Alphabets</strong></button></div>
  </section>
  <section>
     <div class="container">
       <div class="accordion text-white" role="tablist" id="accordion-1">
         <div class="accordion-item" style="background: rgb(33,37,41);">
            <h2 class="accordion-header" role="tab"><button class="accordion-
button" data-bs-toggle="collapse"
                 data-bs-target="#accordion-1 .item-1" aria-expanded="true"
                 aria-controls="accordion-1 .item-1"
                 style="background: rgb(39,43,48);color:
rgb(255,255,255);">About The Project</button></h2>
            <div class="accordion-collapse collapse show item-1"</pre>
role="tabpanel" data-bs-parent="#accordion-1">
              <div class="accordion-body">
                 Artificial Intelligence has made it possible to
handle our daily activities
                   in new and simpler ways. With the ability to automate tasks
that normally require human
                   intelligence, such as speech and voice recognition, visual
perception, predictive text
                   functionality, decision-making, and a variety of other tasks, AI
can assist people with
                   disabilities by significantly improving their ability to get
around and participate in
                   daily activities.<br/>
<br/>
Currently, Sign Recognition is
available <strong>only for
```

```
alphabets A-I</strong> and not for J-Z, since J-Z alphabets
also require Gesture
                  Recognition for them to be able to be predicted correctly to a
certain degree of
                  accuracy.
             </div>
           </div>
         </div>
         <div class="accordion-item" style="background: rgb(33,37,41);">
           <h2 class="accordion-header" role="tab"><button class="accordion-
button collapsed"
                data-bs-toggle="collapse" data-bs-target="#accordion-1 .item-2"
aria-expanded="false"
                aria-controls="accordion-1 .item-2"
                style="background: rgb(39,43,48);color:
rgb(231,241,255);">Developed By</button></h2>
           <div class="accordion-collapse collapse item-2" role="tabpanel" data-</pre>
bs-parent="#accordion-1">
             <div class="accordion-body">
                Students at APEC
                  Program.<br/>
<br/>br><1. <strong>KEERTHANA</strong>
420419104029<br>2.
                  <strong>KAVIYA</strong>420419104028<br>>3.
<strong>YUVASHREE</strong> 420419104061<br>>4.
<strong>SUMITHRA</strong> 420419104305<br>
                </div>
           </div>
         </div>
      </div>
    </div>
  </section>
  <div class="modal fade" role="dialog" tabindex="-1" id="modal-1">
    <div class="modal-dialog" role="document">
      <div class="modal-content">
         <div class="modal-header">
           <h4 class="modal-title">American Sign Language -
Alphabets</h4><button type="button"
             class="btn-close" data-bs-dismiss="modal" aria-
label="Close"></button>
```

```
<div class="modal-body"><img src="{{ url_for('static',</pre>
filename='img/ASL_Alphabets.png') }}" width="100%"></div>
         <div class="modal-footer"><button class="btn btn-secondary"</pre>
type="button"
             data-bs-dismiss="modal">Close</button></div>
      </div>
    </div>
  </div>
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
></script>
</body>
</html>
Code:
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()
```

</div>

return Response(gen(video), mimetype='multipart/x-mixed-replace; boundary = frame')

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

Real-Time Communication System Powered By Al For Specially Abled

Loading the Dataset & Image Data Generation

```
In [ ]: from tensorflow.keras.preprocessing.image import ImageDataGenerator
In [ ]: # Training Datagen
    train_datagen = ImageDataGenerator(rescale=1/255,zoom_range=0.2,horizontal_flip=True,vertical_flip=False)
    # Testing Datagen
    test_datagen = ImageDataGenerator(rescale=1/255)
In []: # Training Dataset
x_train=train_datagen.flow_from_directory(r'/content/drive/MyDrive/Dataset/training_set', target_size=(64,64), class_mode='categorical', batch_size=900)
# Training Dataset

# Training Dataset
             Found 15760 images belonging to 9 classes. Found 2250 images belonging to 9 classes.
In [ ]: # The Class Indices in Training Dataset
    x_train.class_indices
Out[]: {'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}
              Model Creation
 In []: # Importing Libraries
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Convolution2D,MaxPooling2D,Flatten,Dense
In [ ]: # Creating Model
    model=Sequential()
 In [ ]: # Adding Layers
model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3)))
 In [ ]: model.add(MaxPooling2D(pool_size=(2,2)))
 In [ ]: model.add(Flatten())
In []: # Adding Dense Layers
    model.add(Dense(300,activation='relu'))
    model.add(Dense(150,activation='relu'))
    model.add(Dense(9,activation='softmax'))
In [ ]: # Compiling the Model
model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
 In [ ]: # Fitting the Model Generator model.fit_generator(x_train,steps_per_epoch=len(x_train),epochs=10,validation_data=x_test,validation_steps=len(x_test))
```

```
Epoch 1/10
18/18 [====
Epoch 2/10
               Epoch 3/10
     tpoch 5/10
18/18 [=========] - 95s 5s/step - loss: 0.0033 - accuracy: 0.9995 - val_loss: 0.3011 - val_accuracy: 0.9764
Epoch 6/10
      18/18 F=
            Epoch 7/10
18/18 [====
               18/18 [====
Epoch 8/10
     18/18 [============] - 95s 5s/step - loss: 0.0021 - accuracy: 0.9997 - val_loss: 0.3332 - val_accuracy: 0.9760 Epoch 9/10
      Epoch 10/10
18/18 [====
                  Out[]:
     Saving the Model
In [ ]: model.save('asl_model_84_54.h5')
     Testing the model
In [ ]:
    import numpy as np
    from tensorflow.keras.models import load_model
    from tensorflow.keras.preprocessing import image
      model=load_model('asl_model_84_54.h5')
      img=image.load_img(r'/content/drive/MyDrive/Dataset/test_set/D/2.png',
    target_size=(64,64))
In [ ]: img
In [ ]: x=image.img_to_array(img)
In [ ]: x.ndim
Out[ ]: 3
In [ ]: x=np.expand_dims(x,axis=0)
In [ ]: x.ndim
Out[ ]: 4
In [ ]: pred=np.argmax(model.predict(x),axis=1)
      1/1 [=====] - 0s 145ms/step
In [ ]: pred
Out[]: array([3])
In [ ]: index=['A','B','C','D','E','F','G','H','I']
    print(index[pred[0]])
      D
      OPEN CV
 In [ ]: import cv2
 In [ ]: img=cv2.imread(r'/content/drive/MyDrive/Dataset/test_set/C/2.png',1)
 In [ ]: img1=cv2.imread(r'/content/drive/MyDrive/Dataset/test_set/B/2.png',0)
 In [ ]: print(img.shape)
      (64, 64, 3)
      from google.colab.patches import cv2_imshow
cv2_imshow(img)
       cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
In [3]: from tensorflow.keras.preprocessing.image import ImageDataGenerator
In [4]: train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
             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 numbor as np
             rrom tensorijow.Keras.preprocessi
import numpy as np
import matplotlib.pyplot as plt
import IPython.display as display
from PIL import Image
import pathlib
In [6]:
    train_datagen = ImageDataGenerator(rescale = 1./255 , shear_range=0.2,
    zoom_range=0.2,horizontal_flip=True)
    test_datagen = ImageDataGenerator(rescale = 1./255)
In [7]: x_train=train_datagen.flow_from_directory('/content/Dataset/training_set', target_size=(64,64), batch_size=300, class_mode='categorical', color_mode="gray"
            Found 15750 images belonging to 9 classes.
In [8]: x_test=train_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.
               Image Preprocessing
               Import ImageDataGenerator Library And Configure It
   In [2]: from tensorflow.keras.preprocessing.image import ImageDataGenerator
  In [3]: # Training Datagen train_datagen = ImageDataGenerator(rescale=1/255,zoom_range=0.2,horizontal_flip=True,vertical_flip=False)
               test_datagen = ImageDataGenerator(rescale=1/255)
  In [4]:
    import tensorflow as tf
    import os
                 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 matplotlib.pyplot as pit
import IPython.display as display
from PIL import Image
import pathlib
```

DEMO LINK:

https://drive.google.com/file/d/1izayOTIhkYqmCMKLXa-Rx47xjo0KSFQq/view?usp=share_link

GITHUB LINK:

IBM-Project-49331-1660818038