REAL TIME COMMUNICATION SYSTEM POWERED BY AI FOR SPECIALLY ABLED

IBM-Project-40539-1660630937

Project ID:PNT2022TMID33994

PROJECT REPORT

Submitted by

Brintha.T-TeamLead (960219104036)

Arockia Sweety.V-TeamMember1(960219104021)

Babys Dharaniya.S.B-TeamMember2(960219104028)

GodshaniRibisha.L-TeamMember3(960219104044)

in partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING IN

COMPUTER SCIENCE & ENGINEERING

ARUNACHALA COLLEGE OF ENGINEERING FOR WOMEN

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ANNA UNIVERSITY: CHENNAI 600025

1. INTRODUCTION

1.1 ProjectOverview:

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

1.2 Purpose:

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

2. LITERATURE SURVEY

REAL TIME COMMUNICATION SYSTEM POWERED BY AI FOR SPECIALLY ABLED

2.1 Existing Solution:

Title: Artificial Intelligence enabled virtual sixth sense application for the disabled

Author Name: Aditya Sharma, Aditya Vats, Shiv Shankar Dash

Year of publishing: January 2020

Description:

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. The mobile app even works for object detection cases in offline mode. What we want to achieve using this is to make a better world for the people suffering from disabilities as well as an educational end for people with cognitive disabilities using our app. The current implementation deals 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.

Title: Sign Language Recognition System for People with Disability using Machine Learning and Image Processing

Author: Bayan Mohammed Saleh, Reem Ibrahim AI-Beshr, Muhammad Usman Tariq

Year of Publishing: September 2020

Description:

Communication plays a significant role in making the world a better place. Communication creates bonding and relations among the people, whether persona, social, or political views. Most people communicate efficiently without any issues, but many cannot due to disability. They cannot hear or speak, which makes Earth a problematic place to live for them. Even simple basic tasks become difficult for them. Disability is an emotive human condition. It limitsthe individual to a certain level of performance. Being deaf and dumb pushes the subject to oblivion, highly introverted. In a world of inequality, this society needs empowerment. Harnessing technology to improve their welfare is necessary. In a tech era, no one should be limited due to his or her inability. The application of technology should create a platform or a world of equality despite the natural state of humans. On the other hand, technology is the most innovative thing on Earth for every time the clock ticks, researchers, software engineers, programmers, and information technology specialists are always coming up with bright ideas to provide convenience to everyone. This paper shows how artificial intelligence is being used to help people who are unable to do what most people do in their everyday lives. Aligned with communication, D-talk is a system that allows people who are unable to talk and hear be fully understood and for them to learn their language easier and also for the people that would interact and communicate with them. This system provides detailed hand gestures that show the interpretation at the bottom so that everyone can understand them. This research allows the readers to learn the system and what it can do to people who are struggling with what they are not capable of and will provide the technical terms on how the system works.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. The mobile app even works for object detection cases in offline mode.What we want to achieve using this is to make a better world for the people suffering from disabilities as well as an educational end for people with cognitive disabilities using our app. The current implementation deals with object recognition and text to speech and a speechto 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.

Title: An AI software to communicate with deaf and mute in real time

Author: Bhargav D V,

Year of publishing: September 27th, 2021

Description:

The software will assist them in establishing a two-way communication channel even with unimpaired people who have never studied sign language**.** The software, christened DnD Mate, does not only translate sign language into text and speech, but also translates speech into sign language, all in real time and as quick as the person speaks. Currently, there are no applications/software that facilitates a two-way communication channel. This easy-to-use innovative digital translator works with your device's in-built cameras, reads hand and facial gestures by the deaf and mute user and translates them into text and speech. That is not all! The software will also translate your voice or text input into sign language. 'The software is based on a Deep Learning model and can work both offline and online. While in the offline mode, the deaf and mute person can communicate with you on the same device in real time; in the online mode, you can converse sitting in far off places as well, just like you talk to anyone over a video call.

2.2 REFERENCES:

[1] Mukesh Kumar Makwana, " Sign Language Recognition", M.Tech thesis, Indian Institute of Science, Bangalore

[2] Pigou, Lionel, et al. "Sign language recognition using convolutional neural networks." Workshop at the European Conference on Computer Vision. Springer International Publishing, 2014.

[3] Escalera, Sergio, et al. "Chalearn looking at people challenge 2014: Dataset and results." Workshop at the European Conference on Computer Vision. Springer International Publishing, 2014.

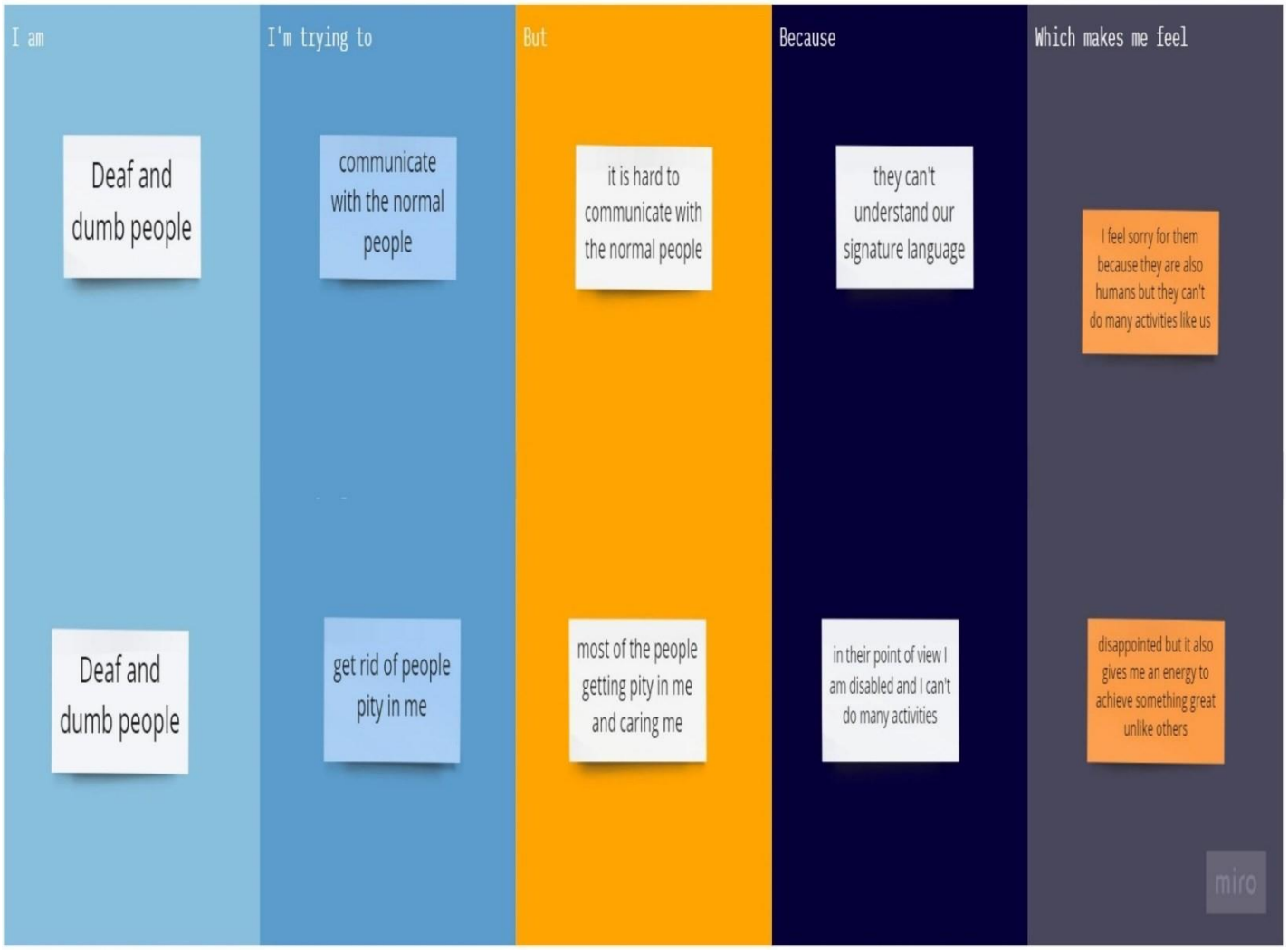
[4] Kuznetsova Alina, Laura Leal-Taix, and Bodo Rosenhahn. "Real-time sign language recognition using a consumer depth camera." Proceedings of the IEEE International Conference on Computer Vision Workshops. 2013.

[5] J. -. Lementec and P. Bajcsy, "Recognition of arm gestures using multiple orientation sensors: gesture classification", Proceedings. The 7 th International IEEE Conference on Intelligent Transportation Systems (IEEE Cat. No.04TH8749), Washington, WA, USA, 2004, pp. 965-970.doi: 10.1109/ITSC.2004.1399037.

[6] S. Hussain, R. Saxena, X. Han, J. A. Khan, and H. Shin, "Hand gesture recognition using deep learning", 2017 International SoC Design Conference (ISOCC) , Seoul, pp. 1-6.

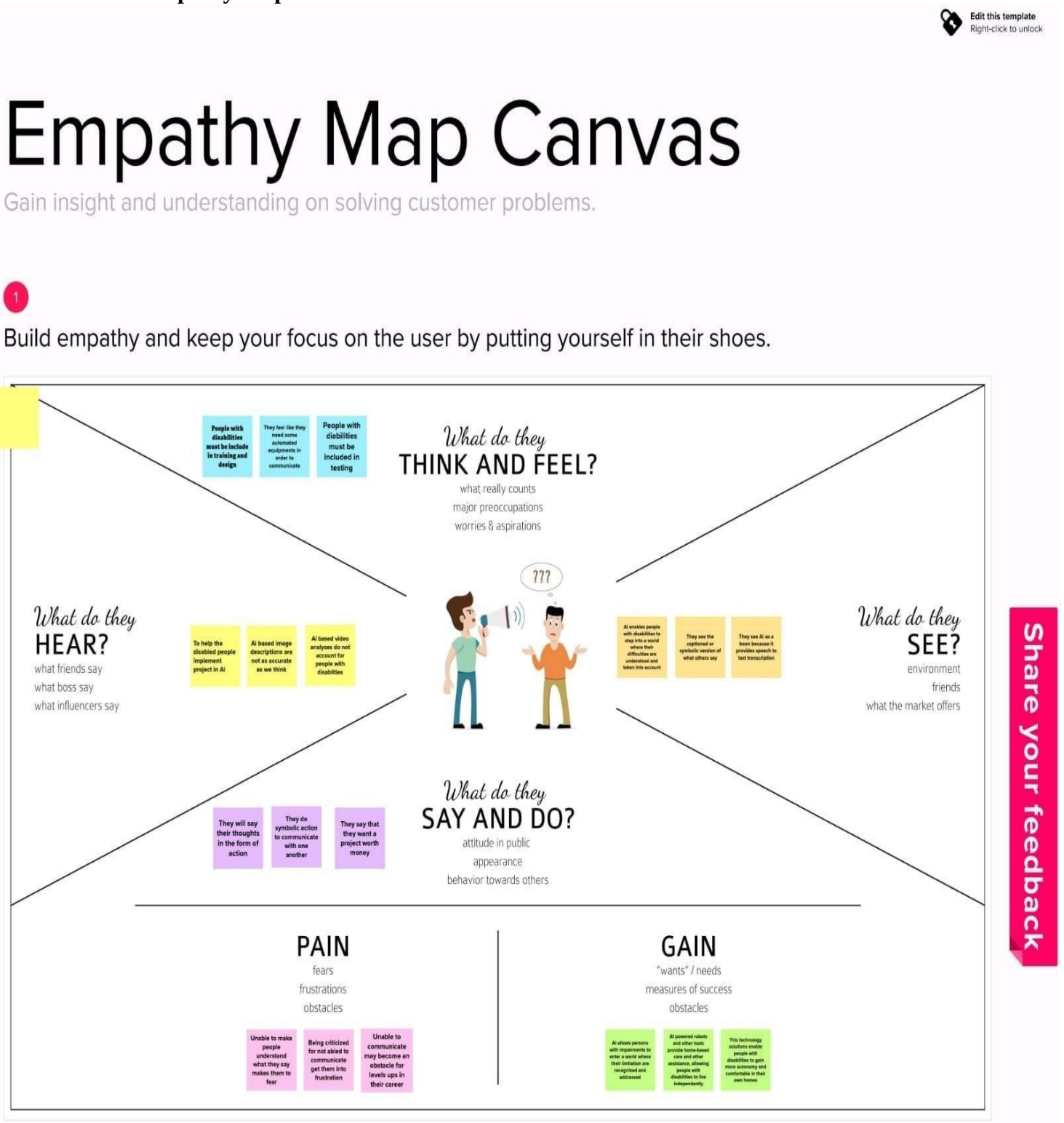
[7] T. Yamashita and T. Watasue, "Hand posture recognition based on bottom-up structured deep convolutional neural network with curriculum learning", 2014 IEEE International Conference on Image Processing (ICIP) , Paris, 2014, pp. 853-857.
[8] Pei Xum “A real time hand gesture recognition and human computer interaction,” Dept. of Electrical and Computer Engineering, University of Minnesota, 2017, pp. 1-8.
[9] B. Liao, J. Li, Z. Ju and G. Ouyang, "Hand Gesture Recognition with Generalized Hough Transform and DC-CNN Using Realsense," 2018 Eighth International Conference on Information Science and Technology (ICIST) , Cordoba, 2018, pp. 84-90.
[10] http://nicolas.burrus.name/index.php/Research/KinectCalibration

2.3 Problem Statement Defintion:

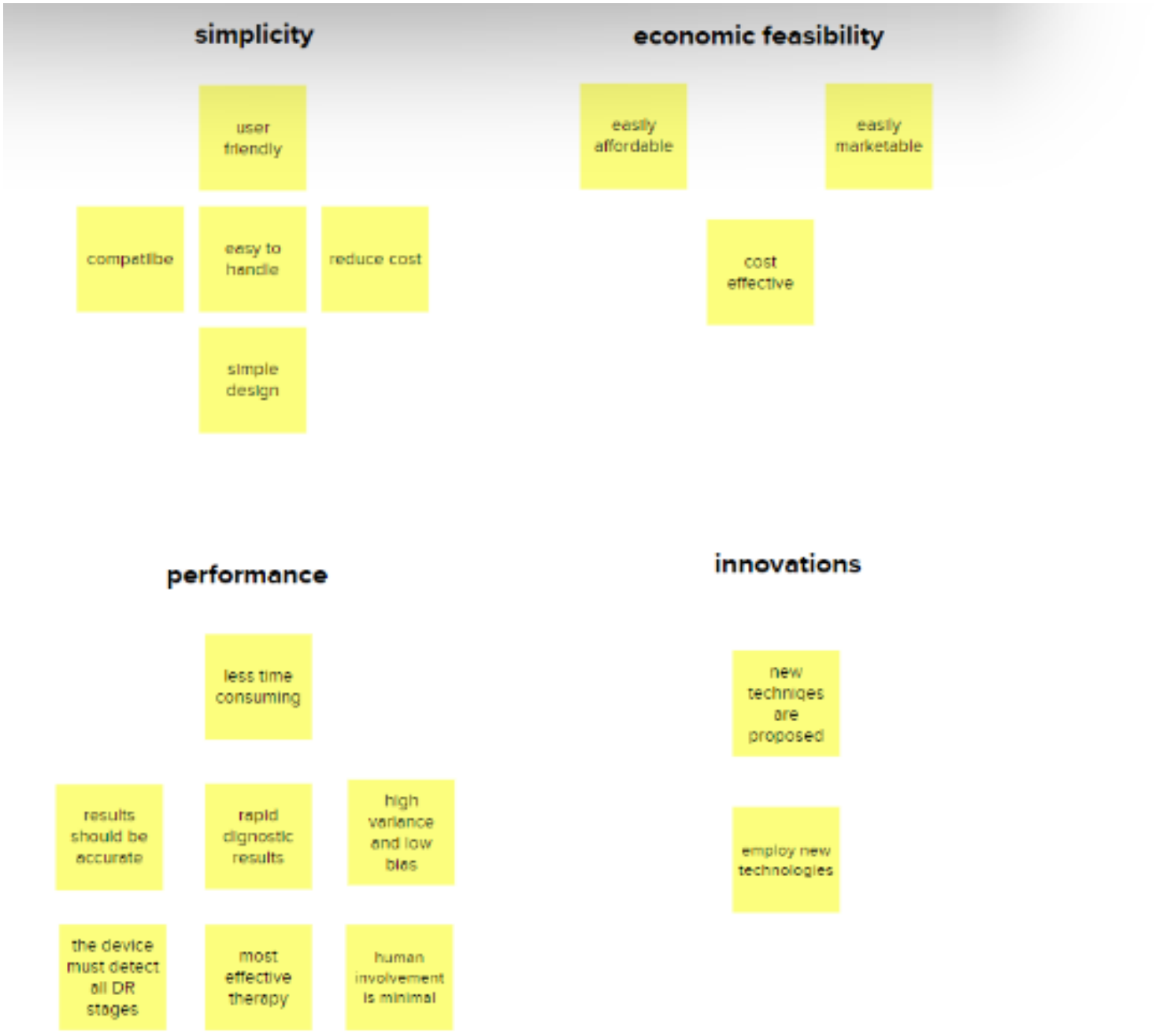


3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas:



3.2 Ideation & Brainstorming:



Brintha T

Easily marketable	Easier to repair	Easily Affordable
Should employ new technologies	Shouldn't cause other complications during diagnosis process	Compatible
Easy to handle the testing equipment	Rapid diagnostic results	Should reach every diabetic patient

sweety v

high variance and low bias	early detection	high resolution In order to diagnose diabetic retinopathy
high specificity and sensitivity	most effective therapy	reduce the risk
stronger robustness and excellent performance	not requiring hand-crafted feature extraction	new techniques are proposed

Ribisha L

simple in design	easy accessability	performance should be of high level
lower demand for human resources	safe screening	it should be within easy reach of patients
clinical report could be automatically generated for each patient	high efficiency	portable

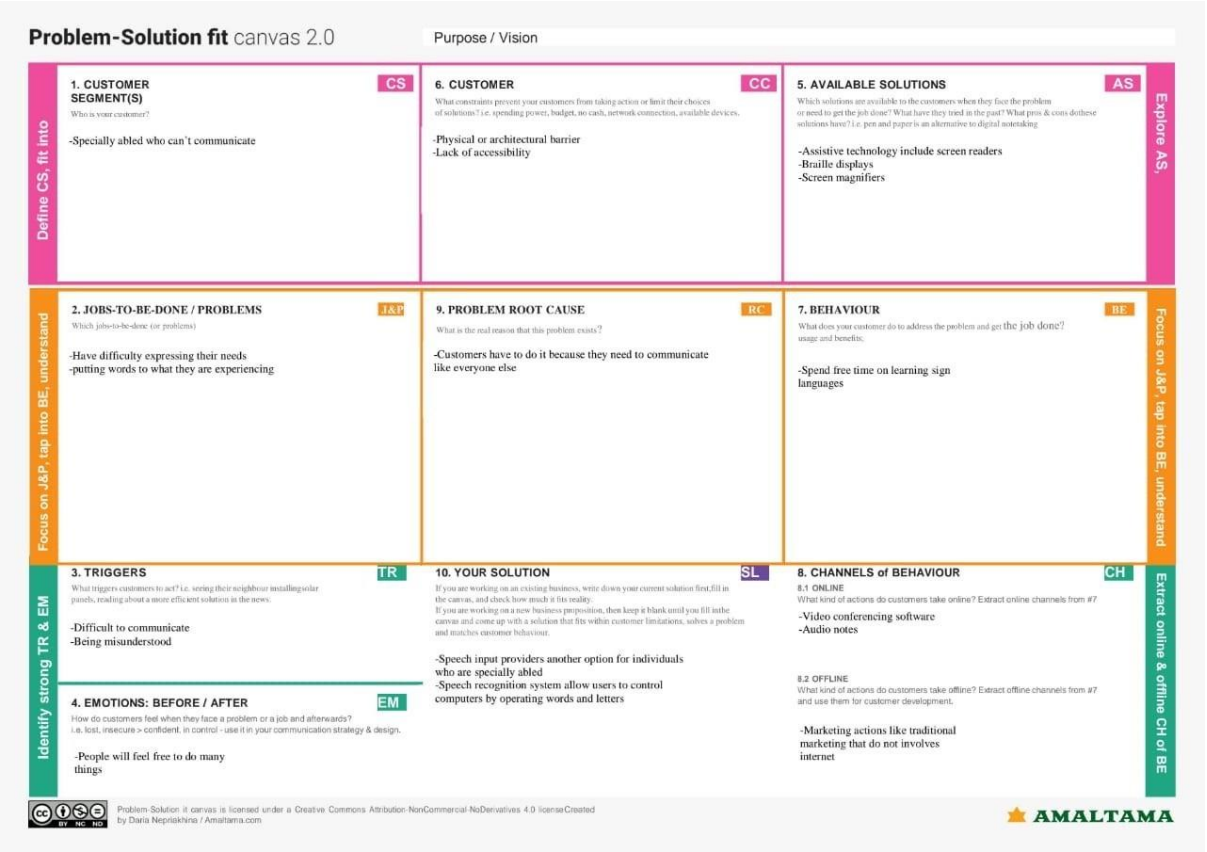
Babys S B

improved quality	results should be accurate	easy to maintain
radiation free	less time consuming	user friendly
the device must detect all DR stages	cost effective	minimal human involvement

3.3 Proposed Solution:

S. No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The project aims to develop a system that converts the sign language into an human hearing voice in the desired language to convey message to normal people, as well as convert speech into understandable sign language for the deaf and dumb.
2.	Idea / Solution description	Activities for all personalities. Music therapy: arousing your emotions Will therapy: enhance your mobility Gardening: no-stress therapy etc…
3.	Novelty / Uniqueness	It unique to develop solutions to many physical and cognitive challenges disabled individuals face at work and in daily life to promote social inclusion for them.
4.	Social Impact / Customer Satisfaction	The social impacts are those consequences of disability that are experienced at the individual, family and community level. They may lack social support and social skills, such as communication, to cope with the disability.
5.	Business Model (Revenue Model)	Freelance writing, web designing, app development, photoediting, online accounting services, tax services are the business model proposed for peoples who are disabled.
6.	Scalability of the Solution	Scalability is a critical concept in disabilities to increasing mobility, enabling independent living, and ensuring equal access to services and realizing operational capabilities.

3.4 Problem Solution Fit:



4. REQUIREMENT ANALYSIS

Solution Requirements (Functional & Non-functional)

4.1 Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Browser	Navigating the local host
FR-2	User Registration	Registration through Gmail

FR-3	User Confirmation	Confirmation via Email
FR-4	Camera	Detects the sign language

4.2 Non-functional Requirements:

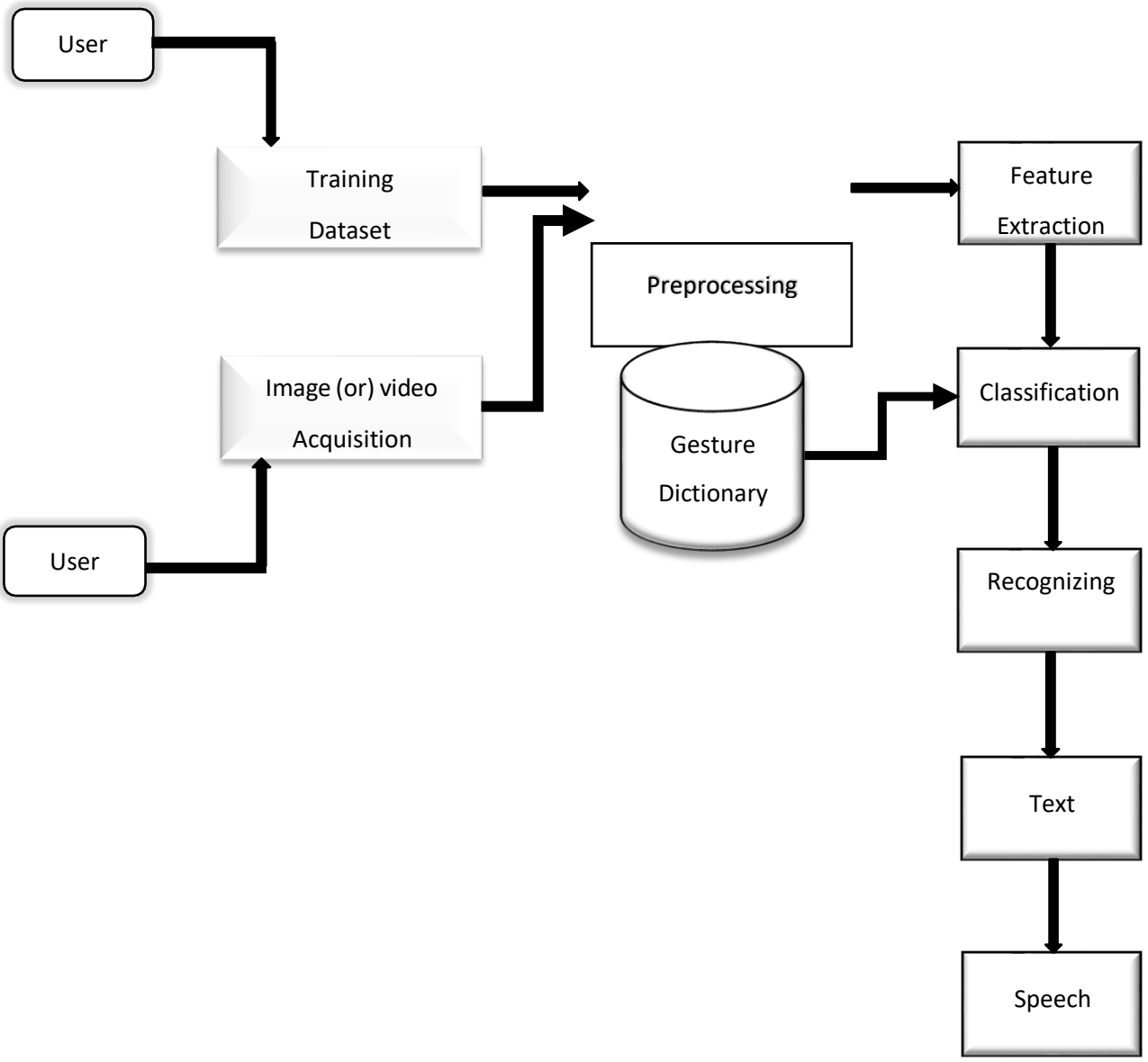
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The degree to which a project satisfies the user requirements. By using our project customers can figure out that their requirements are satisfied.
NFR-2	**Security**	The degree to which a project is secure. Our project can be functional against attacks.
NFR-3	**Reliability**	The degree to which a project provides failure-free operation. Our project works without failures.
NFR-4	**Performance**	The degree to which a project is faster and stable. Our project's processing speed is higher.
NFR-5	**Availability**	The degree to which a project is available whenever a user wants to access. Our project provides better availability when needed.
NFR-6	**Scalability**	The degree to which a project adapts to increased workloads. Our project can be moved from smaller to larger operating system and can be upgraded.

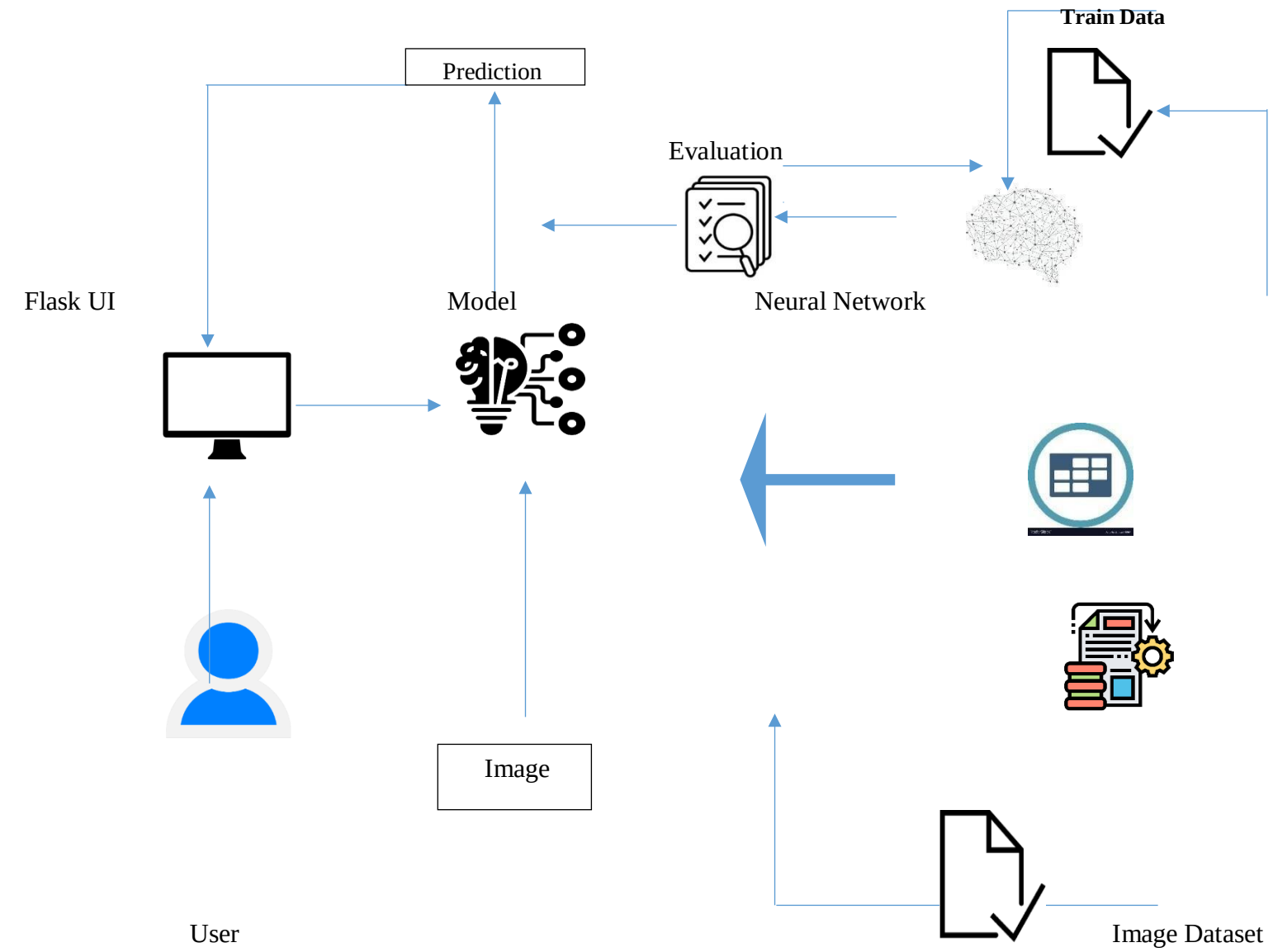
5. PROJECT DESIGN

5.1 Data Flow Diagram:

Data flow diagram (level0)


User
Training Dataset
Image (or) video Acquisition
User
Preprocessing
Gesture Dictionary
Feature Extraction
Classification
Recognizing
Text
Speech

5.1 Solution & Technical Architecture:



5.3 Use Case Stories:

Use Case	Functio nalRequ iremnt(Epic)	User Story Number	User Story/Task	Acceptance Criteria	Priority	Release
Custom er (mobile user, web user, admin)	web cam	USN-1	As a user ,I can start the webcam tocommunicate with people	I can access webcam	high	Sprint-1
	Capture video	USN-2	As a user, I can start the video to capture	I can access video	medium	Sprint-1
	Capture gesture	USN-3	It capture my gesture through video	My gesture is captured	high	Sprint-2
	Transla te gesture	USN-4	It translates gesture into alphabetic letters	My gestureis translated	high	Sprint-2
	Extract features	USN-5	It extract features from preprocessing	My gesture is extracted	low	Sprint-3
	Match features	USN-6	It match my gesture with sign language dataset	My gesture is matched	high	Sprint-3
	Recogni ze gesture	USN-7	Finally ,it gives an exact meaning of my gestures	my gesture video is converted into alphabet in the form of text	high	Sprint-4

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation:

Product Backlog, Sprint Schedule, and Estimation

Sprint	Functional Requirement(Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	Collect Dataset .	9	High	Brintha, Sweety
Sprint-1		USN-2	Image preprocessing	8	Medium	Brintha, Sweety
Sprint-2	Model Building	USN-3	Import the required libraries, add the necessary layers and compile the model	10	High	Babys, Ribisha
Sprint-2		USN-4	Training the image classification model using CNN	7	Medium	Babys, Ribisha
Sprint-3	Trainingand Testing	USN-5	Training the model and testingthe model's performance	9	High	Ribisha, Brintha
Sprint-4	Implementation of the application	USN-6	Converting the input sign language images into English alphabets	8	Medium	Babys, Sweety

6.1 Sprint Delivery Schedule:

Project Tracker, Velocity& Burn down Chart

Sprint	Total Story points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	10	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	10	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	10	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	10	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

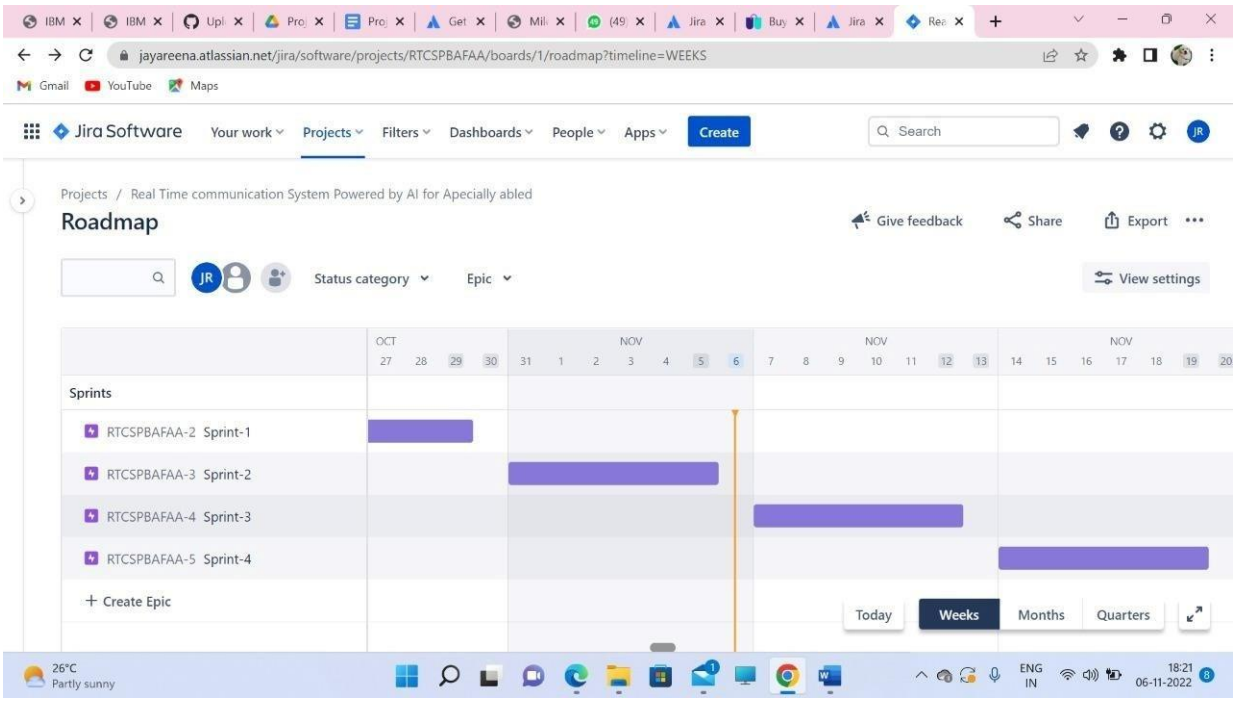
Velocity:

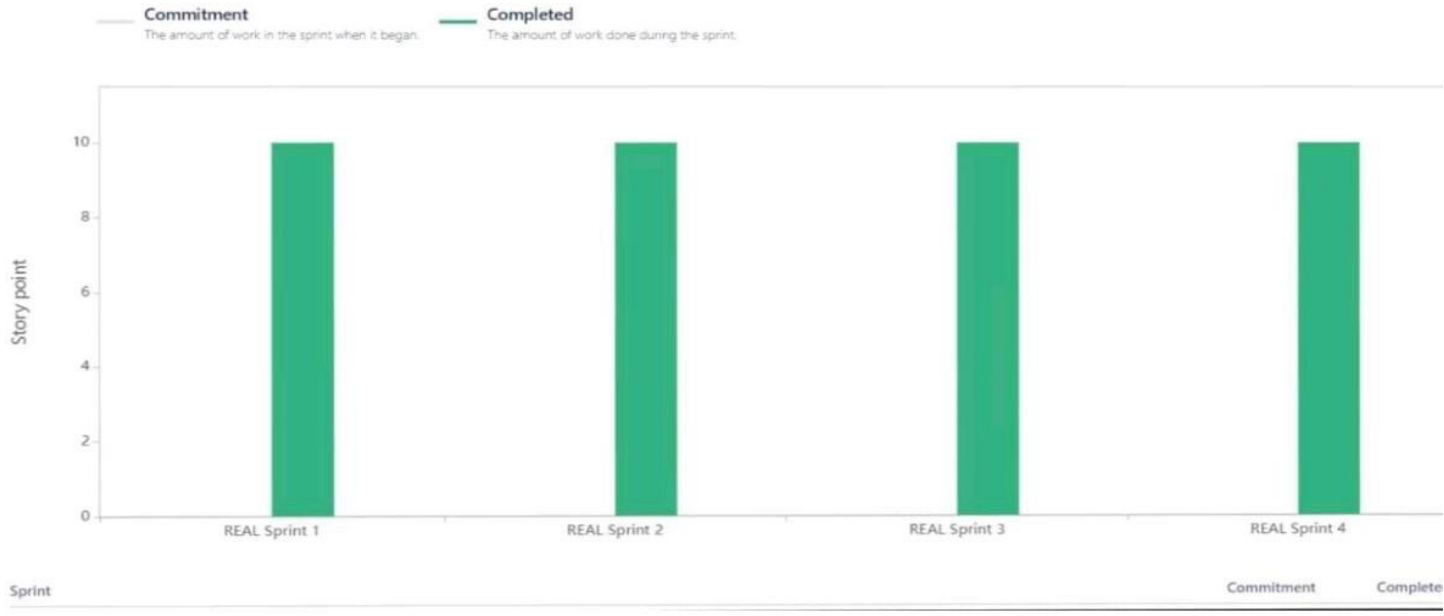
$$AV = \frac{sprint\ duration}{velocity}$$

AV=6/10=0.6

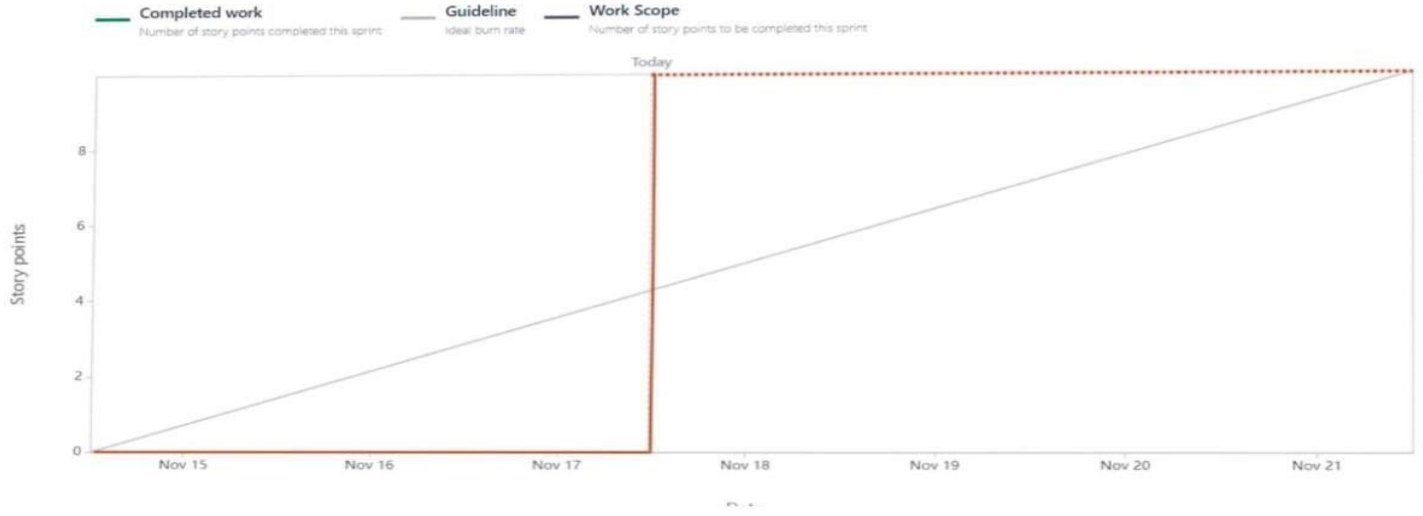
6.2 Reports from JIRA:

Road map, Burndownchart





Date - October 24th, 2022 - October 31st, 2022



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

Importing The Required Model Building Libraries

In [ ]:
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

In [ ]:
```
from keras.models import Sequential, load_model
from keras.layers.core import Dense, Dropout, Activation
from keras.utils import np_utils
```

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 [ ]:
```
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 [ ]:
```
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)
# 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()
```

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
# 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])])

kernel_horizontal = np.array([np.array([2, 2]), np.array([-2, -2])])
print(kernel_horizontal, 'is a kernel for detecting horizontal edges')

kernel_vertical = np.array([np.array([2, -2]), np.array([2, -2])])
print(kernel_vertical, 'is a kernel for detecting vertical edges')
```

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')
plt.show()

# Checking for horizontal and vertical features in image1
print('Horizontal edge confidence score:', apply_kernel(img1,
                                        kernel_horizontal))
print('Vertical edge confidence score:', apply_kernel(img1,
                                        kernel_vertical))
```

In [ ]:
```
# Visualizing img2
plt.imshow(img2)
plt.axis('off')
plt.title('img2')
plt.show()

# Checking for horizontal and vertical features in image2
print('Horizontal edge confidence score:', apply_kernel(img2,
                                        kernel_horizontal))
print('Vertical edge confidence score:', apply_kernel(img2,
                                        kernel_vertical))
```

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)
```

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)
```

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)
# 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)))
```

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'])
```

Compile To The Model

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
# x and y.
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)
```

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 [ ]:
```
def compile_model_results(model, root="./"):

    listing = glob.glob(root + '/models/' + model + '/*/best_pars.pkl')

    dic_list = []
    for file in listing:
        tmp = 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(lr=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
    else:
        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}
```

8. TESTING

8.1 Test cases:

Loading the Dataset & Image Data Generation

In [14]:
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

In [15]:
```
# 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 [25]:
```
# Training Dataset
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
# Testing Dataset
x_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 :  6
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 Model
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
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 [==============================] - 23s 4s/step - loss: 5.1206 - accuracy: 0.1690 - val_loss: 3.6505 - val_accuracy: 0.3119
Epoch 2/10
6/6 [==============================] - 22s 4s/step - loss: 2.3945 - accuracy: 0.3266 - val_loss: 1.5087 - val_accuracy: 0.4991
Epoch 3/10
6/6 [==============================] - 22s 4s/step - loss: 1.4384 - accuracy: 0.4037 - val_loss: 1.0430 - val_accuracy: 0.5836
Epoch 4/10
6/6 [==============================] - 23s 4s/step - loss: 1.0761 - accuracy: 0.6488 - val_loss: 0.7109 - val_accuracy: 0.7955
Epoch 5/10
6/6 [==============================] - 27s 5s/step - loss: 0.7835 - accuracy: 0.7774 - val_loss: 0.4046 - val_accuracy: 0.9501
Epoch 6/10
6/6 [==============================] - 25s 5s/step - loss: 0.5470 - accuracy: 0.8756 - val_loss: 0.2540 - val_accuracy: 0.9752
Epoch 7/10
6/6 [==============================] - 22s 4s/step - loss: 0.4018 - accuracy: 0.9090 - val_loss: 0.1675 - val_accuracy: 0.9799
Epoch 8/10
6/6 [==============================] - 22s 4s/step - loss: 0.2862 - accuracy: 0.9406 - val_loss: 0.1185 - val_accuracy: 0.9847
Epoch 9/10
6/6 [==============================] - 22s 4s/step - loss: 0.2108 - accuracy: 0.9612 - val_loss: 0.0880 - val_accuracy: 0.9863
Epoch 10/10
6/6 [==============================] - 22s 4s/step - loss: 0.1548 - accuracy: 0.9738 - val_loss: 0.0736 - val_accuracy: 0.9843
```

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	8	7	4	4	2
Duplicate	1	0	1	0	0
External	2	1	0	1	1
Fixed	6	1	0	0	10
Not Reproduced	0	0	0	0	1
Skipped	0	0	0	1	1
Won't Fix	0	0	0	1	0
Totals	17	9	5	5	15

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	2	0	0	2
Client Application	7	0	0	7
Security	0	0	0	0
Outsource Shipping	0	0	0	0

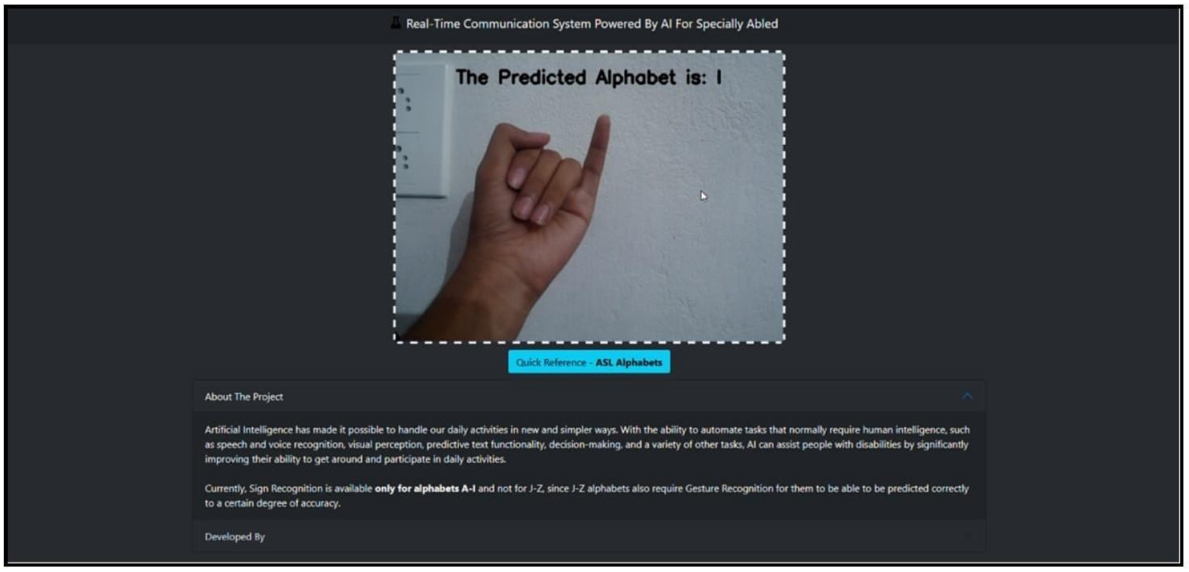
Exception Reporting	5	0	0	6
Final Report Output	6	0	0	5
Version Control	2	0	0	2

9. RESULTS

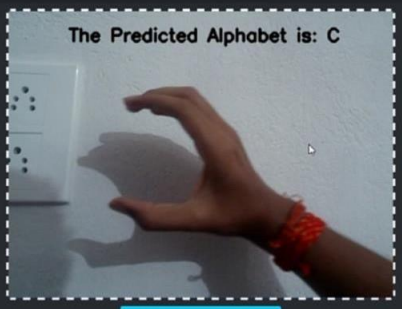
9.1 Performance Metrics:

The proposed procedure was implemented and tested with set of images. The set of 15750 images of Alphabets from “A” to “I” are used for training database and a set of 2250 images of Alphabets from “A” to “I” are used for testing database. Once the gesture is recognise the equivalent Alphabet is shown on the screen.

Some sample images of the output are provided below:



Real-Time Communication System Powered By AI For Specially Abled



Quick Reference - **ASL Alphabets**

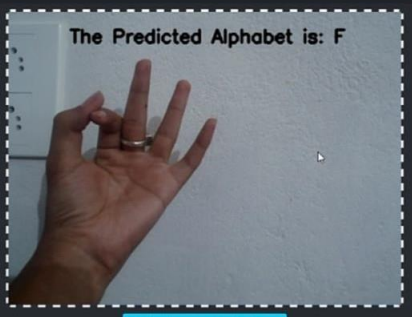
About The Project

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.

Currently, Sign Recognition is available **only for alphabets A-I** 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.

Developed By

Real-Time Communication System Powered By AI For Specially Abled



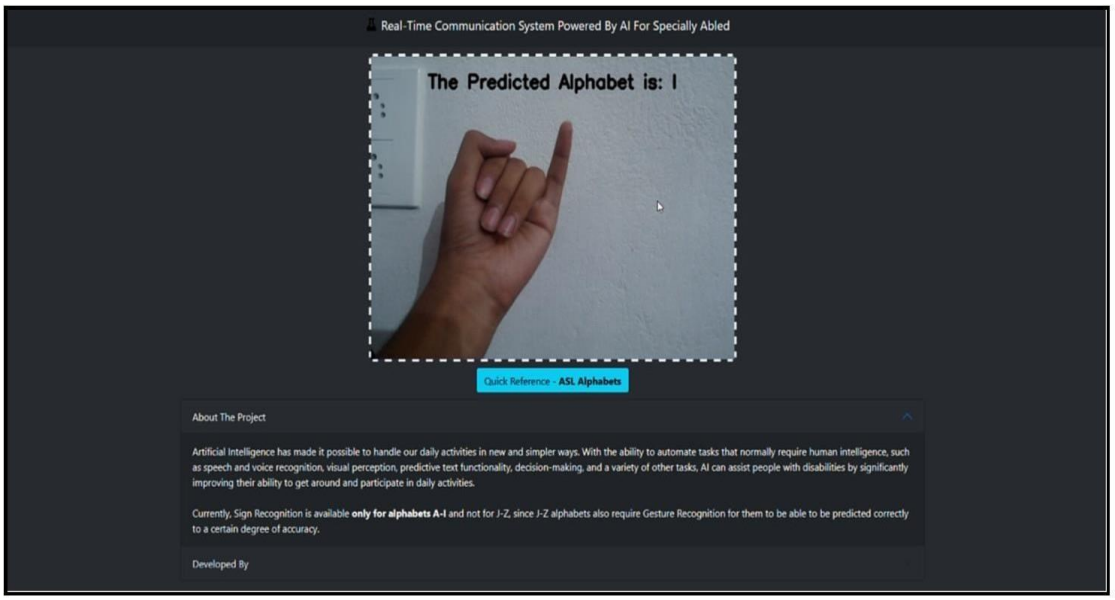
Quick Reference - **ASL Alphabets**

About The Project

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.

Currently, Sign Recognition is available **only for alphabets A-I** 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.

Developed By



10. ADVANTAGES & DISADVANTAGES

Advantages:

1. It is possible to create a mobile application to bridge the communication gap between deaf and dumb persons and the general public.

2. As different sign language standards exist, their dataset can be added, and the user can choose which sign language to read.

Disadvantages:

1. The current model only works from alphabets A to I.

2. In absence of gesture recognition, alphabets from J cannot be identified as they require some kind of gesture input from the user.

3. 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.

11. CONCLUSION

Sign language is a useful tool for facilitating communication between deaf and hearing people. Because it allows for two-way communication, the system aims to bridge the communication gap between deaf people and the rest of society. The proposed methodology translates language into English alphabets that are understandable to humans.

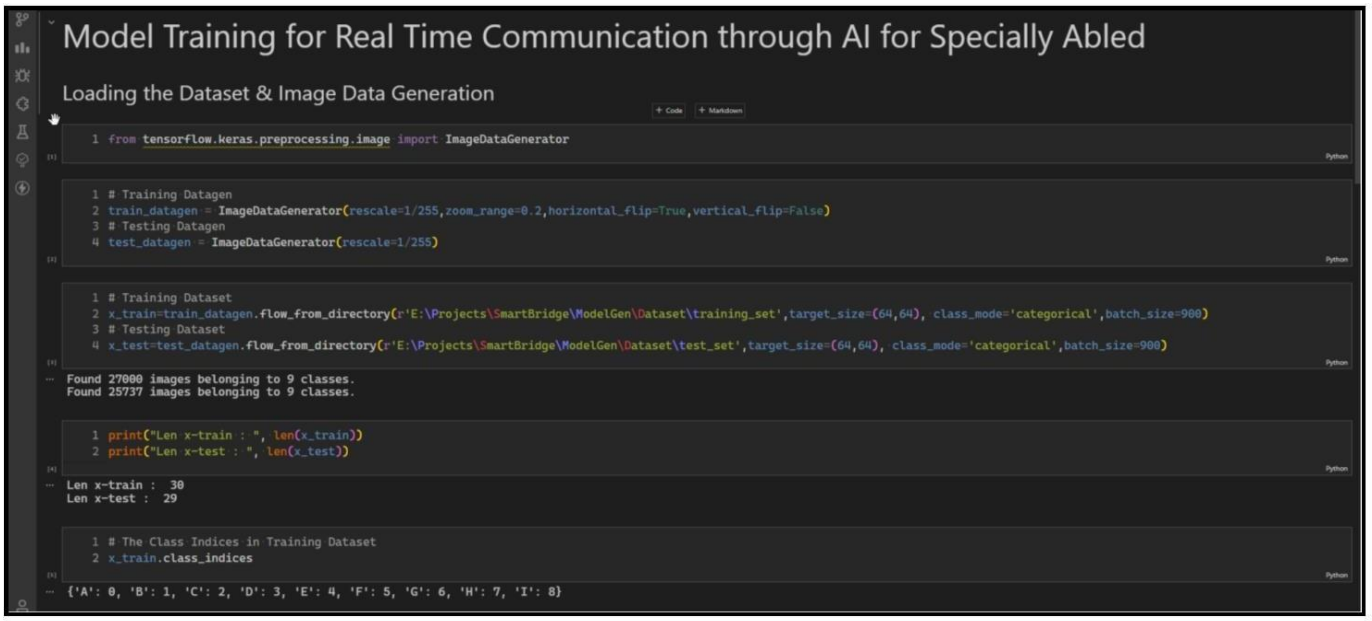
This system sends hand gestures to the model, who recognises them and displays the equivalent Alphabet on the screen. Deaf-mute people can use their hands to perform sign language, which will then be converted into alphabets, thanks to this project.

12. FUTURE SCOPE

Having a technology that can translate hand sign language to its corresponding alphabet is a game changer in the field of communication and Ai for the specially abled people such as deaand dumb. With introduction of gesture recognition, the web app can easily be expanded to recognize letters beyond 'I', digits and other symbols plus gesture recognition can also allow controlling of software/hardware interfaces Having a technology that can translate hand sign language to its corresponding alphabet is a game changer in the field of communication and Ai for the specially abled people such as deaand dumb. With introduction of gesture recognition, the web app can easily be expanded to recognize letters beyond 'I', digits and other symbols plus gesture recognition can also allow

13. APPENDIX

Source code:



```python
# 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))
```

```
C:\Users\Kushagra\AppData\Local\Temp\ipykernel_8892\1042518445.py:2: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use `Model.fit`,
which supports generators.
  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
30/30 [==============================] - 252s 9s/step - loss: 2.1755 - accuracy: 0.1997 - val_loss: 1.9401 - val_accuracy: 0.3477
Epoch 2/10
30/30 [==============================] - 48s 2s/step - loss: 1.7417 - accuracy: 0.4029 - val_loss: 1.4277 - val_accuracy: 0.4825
Epoch 3/10
30/30 [==============================] - 47s 2s/step - loss: 1.3504 - accuracy: 0.5183 - val_loss: 1.1049 - val_accuracy: 0.6162
Epoch 4/10
30/30 [==============================] - 48s 2s/step - loss: 1.0815 - accuracy: 0.6250 - val_loss: 0.8858 - val_accuracy: 0.6947
Epoch 5/10
30/30 [==============================] - 47s 2s/step - loss: 0.8933 - accuracy: 0.6967 - val_loss: 0.7331 - val_accuracy: 0.7595
Epoch 6/10
30/30 [==============================] - 47s 2s/step - loss: 0.7767 - accuracy: 0.7324 - val_loss: 0.6089 - val_accuracy: 0.8044
Epoch 7/10
30/30 [==============================] - 47s 2s/step - loss: 0.6602 - accuracy: 0.7781 - val_loss: 0.5204 - val_accuracy: 0.8304
Epoch 8/10
30/30 [==============================] - 47s 2s/step - loss: 0.6059 - accuracy: 0.7977 - val_loss: 0.4819 - val_accuracy: 0.8374
Epoch 9/10
30/30 [==============================] - 47s 2s/step - loss: 0.5297 - accuracy: 0.8265 - val_loss: 0.4170 - val_accuracy: 0.8636
Epoch 10/10
30/30 [==============================] - 47s 2s/step - loss: 0.4757 - accuracy: 0.8454 - val_loss: 0.3898 - val_accuracy: 0.8692

<keras.callbacks.History at 0x185f72850f0>
```

Saving the Model

```python
model.save('asl_model_84_54.h5')
# Current accuracy is 0.8454
```

Downloading From IBM

Connecting to IBM Cloud Storage to Get Model from Deployment

```python
from ibm_watson_machine_learning import APIClient
wml_credentials = {
    "url": "https://us-south.ml.cloud.ibm.com",
    "apikey": "mNVF7E95G-awR213njShjlGiUfN-1SpPq-ko8Wx7na1-"
}

client = APIClient(wml_credentials)
```

```python
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'])
```

```python
space_uid = guid_from_space_name(client, 'communication_model_deployment')
print("Space UID : ", space_uid)
```

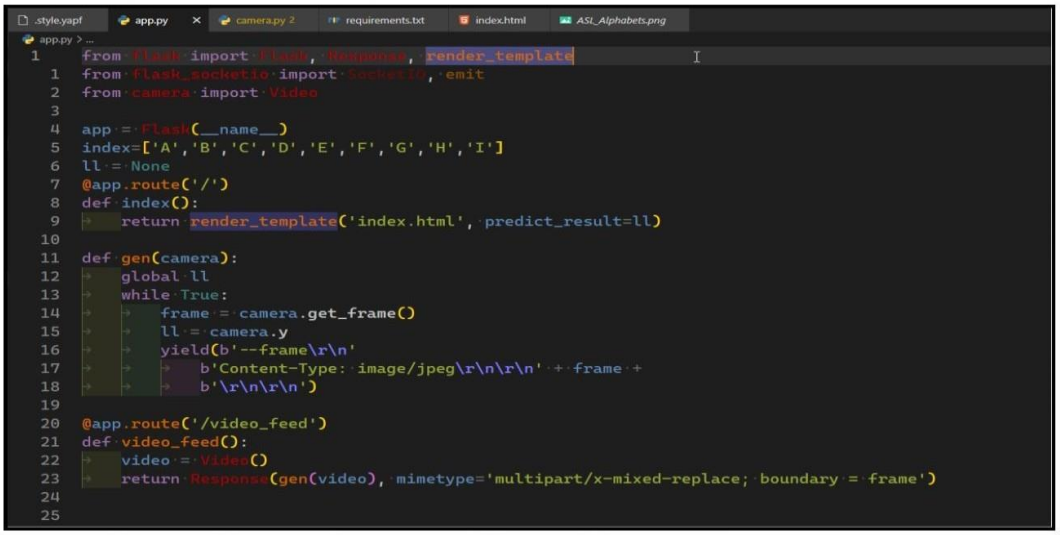
```
Space UID :  21c15ae0-ee26-497d-b615-eb30ef2e16fe
```

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

```
'SUCCESS'
```

```python
client.repository.download("cefca265-2301-4620-897a-9c80d6ff7f1a","IBM_Model_Download.tar.gz")
```

```
Successfully saved model content to file: 'IBM_Model_Download.tar.gz'
'e:\\Projects\\SmartBridge\\ModelGen/IBM_Model_Download.tar.gz'
```

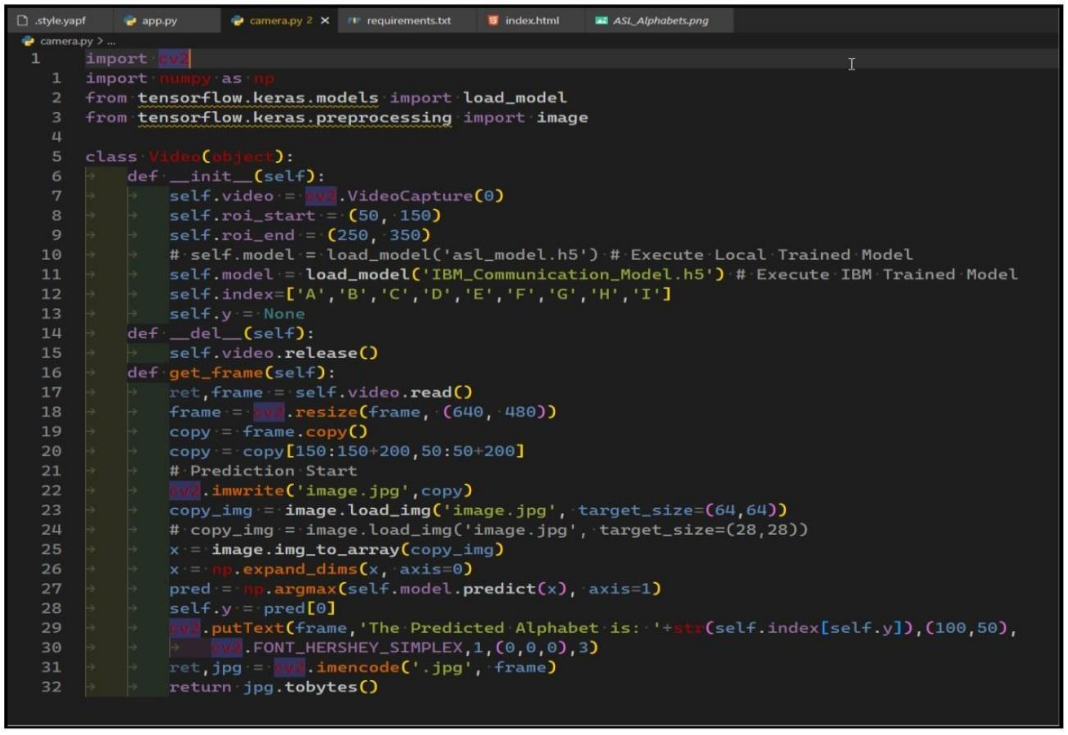


```python
from flask import Flask, Response, render_template
from flask_socketio import SocketIO, emit
from camera import Video

app = Flask(__name__)
index=['A','B','C','D','E','F','G','H','I']
ll = None
@app.route('/')
def index():
    return render_template('index.html', predict_result=ll)

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

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



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

class Video(object):
    def __init__(self):
        self.video = cv2.VideoCapture(0)
        self.roi_start = (50, 150)
        self.roi_end = (250, 350)
        # self.model = load_model('asl_model.h5') # Execute Local Trained Model
        self.model = load_model('IBM_Communication_Model.h5') # Execute IBM Trained Model
        self.index=['A','B','C','D','E','F','G','H','I']
        self.y = None
    def __del__(self):
        self.video.release()
    def get_frame(self):
        ret,frame = self.video.read()
        frame = cv2.resize(frame, (640, 480))
        copy = frame.copy()
        copy = copy[150:150+200,50:50+200]
        # Prediction Start
        cv2.imwrite('image.jpg',copy)
        copy_img = image.load_img('image.jpg', target_size=(64,64))
        # copy_img = image.load_img('image.jpg', target_size=(28,28))
        x = image.img_to_array(copy_img)
        x = np.expand_dims(x, axis=0)
        pred = np.argmax(self.model.predict(x), axis=1)
        self.y = pred[0]
        cv2.putText(frame,'The Predicted Alphabet is: '+str(self.index[self.y]),(100,50),
            cv2.FONT_HERSHEY_SIMPLEX,1,(0,0,0),3)
        ret,jpg = cv2.imencode('.jpg', frame)
        return jpg.tobytes()
```

GitHub link:

https://github.com/IBM-EPBL/IBM-Project-40539-1660630937