Final Code For Flask Application:

* Make Sure that all the Packages are installed
* This code has been saved and executed successfully using Pycharm.
* This code belongs to team-PNT2022TMID0172.

from flask import Flask,render\_template,Response, request

import cv2

from cvzone.HandTrackingModule import HandDetector

from cvzone.ClassificationModule import Classifier

import numpy as np

import math

import pyttsx3

import keyboard

app=Flask(\_\_name\_\_)

cap = cv2.VideoCapture(0)

detector = HandDetector(maxHands=1)

offset = 20

imgSize = 300

str=""

# classifier = Classifier("A2i.h5", "labelsa2j.txt")

classifier = Classifier("Models/keras\_model.h5", "Models/labels.txt")

labels={0:"A", 1:"B", 2:"C" ,3:"D",4:"E",5:"F", 6:"G",7:"H",8:"I",9: "J", 10:"K", 11:"L", 12:"M", 13:"N", 14:"O", 15:"P",16:"Q",17:"R",18:"S",19:"T",20:"U",

21:"V",22:"W",23:"X",24:"Y",25:"Z"}

def function(img):

success, frame = cap.read()

imgoutput = frame.copy()

hands, frame = detector.findHands(frame)

return frame

def generate\_frames():

#str=""

global str

while True:

#labels = {0: "A", 1: "B", 2: "C"}

## read the camera frame

success, frame = cap.read()

if not success:

break

else:

success, frame = cap.read()

imgOutput = frame.copy()

hands, frame = detector.findHands(imgOutput)

if hands:

hand = hands[0]

x, y, w, h = hand['bbox']

imgWhite = np.ones((imgSize, imgSize, 3), np.uint8) \* 255

imgCrop = frame[y - offset:y + h + offset, x - offset:x + w + offset]

imgCropShape = imgCrop.shape

aspectRatio = h / w

if aspectRatio > 1:

k = imgSize / h

wCal = math.ceil(k \* w)

imgResize = cv2.resize(imgCrop, (wCal, imgSize))

imgResizeShape = imgResize.shape

wGap = math.ceil((imgSize - wCal) / 2)

imgWhite[:, wGap:wCal + wGap] = imgResize

prediction, index = classifier.getPrediction(imgWhite, draw=False)

#print(prediction, index)

#print(labels[index])

if keyboard.is\_pressed('s') :

str +=labels[index]

cv2.putText(imgOutput, str, (10, 30), cv2.FONT\_HERSHEY\_SIMPLEX, 1, (255, 255, 0), 3)

if keyboard.is\_pressed('a'):

str+=" "

cv2.putText(imgOutput, str, (10, 30), cv2.FONT\_HERSHEY\_SIMPLEX, 1, (255, 255, 0), 3)

if keyboard.is\_pressed('d'):

str = str[:-1]

cv2.putText(imgOutput, str, (10, 30), cv2.FONT\_HERSHEY\_SIMPLEX, 1, (255, 255, 0), 3)

if keyboard.is\_pressed('w'):

str=""

cv2.putText(imgOutput, str, (10, 30), cv2.FONT\_HERSHEY\_SIMPLEX, 1, (255, 255, 0), 3)

else:

k = imgSize / w

hCal = math.ceil(k \* h)

imgResize = cv2.resize(imgCrop, (imgSize, hCal))

imgResizeShape = imgResize.shape

hGap = math.ceil((imgSize - hCal) / 2)

imgWhite[hGap:hCal + hGap, :] = imgResize

prediction, index = classifier.getPrediction(imgWhite, draw=False)

#print(prediction, index)

#print(labels[index])

if keyboard.is\_pressed('s') :

str += labels[index]

cv2.putText(imgOutput, str, (10, 30), cv2.FONT\_HERSHEY\_SIMPLEX, 1, (255, 255, 0), 3)

if keyboard.is\_pressed('a'):

str += " "

cv2.putText(imgOutput, str, (10, 30), cv2.FONT\_HERSHEY\_SIMPLEX, 1, (255, 255, 0), 3)

if keyboard.is\_pressed('d'):

str = str[:-1]

cv2.putText(imgOutput, str, (10, 30), cv2.FONT\_HERSHEY\_SIMPLEX, 1, (255, 255, 0), 3)

if keyboard.is\_pressed('w'):

str=""

cv2.putText(imgOutput, str, (10, 30), cv2.FONT\_HERSHEY\_SIMPLEX, 1, (255, 255, 0), 3)

cv2.rectangle(imgOutput, (x - offset, y - offset - 50),

(x - offset + 90, y - offset - 50 + 50), (255, 0, 255), cv2.FILLED)

cv2.putText(imgOutput, labels[index], (x, y - 26), cv2.FONT\_HERSHEY\_COMPLEX, 1.7, (255, 255, 255), 2)

cv2.rectangle(imgOutput, (x - offset, y - offset),

(x + w + offset, y + h + offset), (255, 0, 255), 4)

cv2.putText(imgOutput, str, (10, 30), cv2.FONT\_HERSHEY\_SIMPLEX, 1, (255, 255, 0), 3)

ret,buffer=cv2.imencode('.jpg',imgOutput)

imgOutput=buffer.tobytes()

yield(b'--frame\r\n'

b'Content-Type: image/jpeg\r\n\r\n' + imgOutput + b'\r\n')

return render\_template("index.html", pred=str)

@app.route('/predict',methods=['POST','GET'])

def predictions():

return render\_template("index.html", pred=str)

# return generate\_frames()

@app.route('/stop',methods=['POST','GET'])

def stopping():

count = 0

while True:

## read the camera frame

success,frame=cap.read()

if not success:

return "The text is converted into voice.Restart the app again to start predicting.Thank you!!!!!!!!"

break

# if count==1:

# return "Exceeded"

break

else:

#cap.release()

#print("The Recorded String is:", str)

text2speech = pyttsx3.init()

newVoiceRate = 125

text2speech.setProperty('rate', newVoiceRate)

text2speech.say(str)

text2speech.runAndWait()

return render\_template('index.html')

@app.route('/')

def index():

return render\_template('index.html')

@app.route('/video')

def video():

return Response(generate\_frames(),mimetype='multipart/x-mixed-replace; boundary=frame')

#Team-Sajith,Stanley,Sachin,Harish

if \_\_name\_\_=="\_\_main\_\_":

app.run(debug=True)