**Create a machine learning model to control the volume of the laptop using mediapipe**

* + HERE WE WILL BE USING ARDUINO INTERFACE.
  + COMPONENTS REQUIRED:
  1. LAPTOP
  2. NET / WIFI CONNECTION

Graphical user interface, website

Description automatically generated

* PROGRAM (app.py)
* import imp
* from tkinter.messagebox import NO
* from handdetector import HandDetector
* import cv2
* import math
* import time
* import numpy as np
* from ctypes import cast,POINTER
* from comtypes import CLSCTX\_ALL
* from pycaw.pycaw import AudioUtilities,IAudioEndpointVolume
* devices=AudioUtilities.GetSpeakers()
* interface=devices.Activate(IAudioEndpointVolume.\_iid\_,CLSCTX\_ALL,None)
* volume=cast(interface,POINTER(IAudioEndpointVolume))
* handDetector=HandDetector(min\_detection\_confidence=0.7)
* cam=cv2.VideoCapture(0)
* oldDistance=0
* while True:
* status,image=cam.read()
* if status:
* result=handDetector.findHandLandMarks(image=image,draw=True)
* print(result)
* if result is not None:
* x1,y1=result[4][1],result[4][2]
* x2,y2=result[8][1],result[8][2]
* distance=math.hypot(x2-x1,y2-y1)
* print(distance)
* if distance!=oldDistance:
* oldDistance=distance
* time.sleep(0.005)
* else:
* pass
* volumeValue=np.interp(oldDistance,[50,250],[-65.25,0.0])
* volume.SetMasterVolumeLevel(volumeValue,None)
* cv2.circle(image,(x1,y1),15,(255,0,255),cv2.FILLED)
* cv2.circle(image,(x2,y2),15,(255,0,255),cv2.FILLED)
* cv2.line(image,(x1,y1),(x2,y2),(255,0,255),3)
* cv2.imshow('result',image)
* #cv2.imshow('hands',image)
* cv2.waitKey(1)
* PROGRAM (handdetector.py)

import mediapipe as mp

import cv2

#detecting hands

mpHands=mp.solutions.hands

mpDraw=mp.solutions.drawing\_utils

class HandDetector:

    def \_\_init\_\_(self,max\_num\_hands=2,min\_detection\_confidence=0.5,min\_tracking\_confidence=0.5):

        self.hands=mpHands.Hands(

            max\_num\_hands=max\_num\_hands,

            min\_detection\_confidence=min\_detection\_confidence,

            min\_tracking\_confidence=min\_tracking\_confidence

        )

    def findHandLandMarks(self,image,handNumber=0,draw=False):

        originalImage=image

        image=cv2.cvtColor(image,cv2.COLOR\_BGR2RGB)

        results=self.hands.process(image)

        landMarkList=[]

        if results.multi\_hand\_landmarks:

            hand=results.multi\_hand\_landmarks[handNumber]

            for id,landMark in enumerate(hand.landmark):

                imgH,imgW,imgC=originalImage.shape

                xPos,yPos=int(landMark.x\*imgW),int(landMark.y\*imgH)

                landMarkList.append([id,xPos,yPos])

            if draw:

                mpDraw.draw\_landmarks(originalImage,hand,mpHands.HAND\_CONNECTIONS)

            return landMarkList

**By Team : AKULA ZAHEER SHA , TAUFEEQ BASHA & QUIZER SHAH**