Here is our azure code :

```
!pip install azure-cognitiveservices-vision-customvision

from azure.cognitiveservices.vision.customvision.training import
CustomVisionTrainingClient
from azure.cognitiveservices.vision.customvision.prediction import
CustomVisionPredictionClient
from azure.cognitiveservices.vision.customvision.training.models import
ImageFileCreateBatch, ImageFileCreateEntry, Region
from msrest.authentication import ApiKeyCredentials
import time

#Big data set
project_id = "55ea2262-3a09-4d16-9559-8171daf00c25"
ENDPOINT = "https://southcentralus.api.cognitive.microsoft.com/"
training_key = "77de383cbf434ad993114c8836fccfc8"
prediction_key = "77de383cbf434ad993114c8836fccfc8"
prediction_resource_id = "/subscriptions/67514f87-fc06-4ea6-b14a-ab01c46c9
edc/resourceGroups/Test/providers/Microsoft.CognitiveServices/accounts/Tra
sh"

credentials = ApiKeyCredentials(in_headers={"Training-key": training_key})
trainer = CustomVisionTrainingClient(ENDPOINT, credentials)
prediction_credentials = ApiKeyCredentials(in_headers={"Prediction-key":
prediction_key})
predictor = CustomVisionPredictionClient(ENDPOINT, prediction_credentials)

project = trainer.get_project(project_id, custom_headers=None, raw=False)
project.id

iterations = trainer.get_iterations(project_id, custom_headers=None,
raw=False)
iterations[1].id

iteration = trainer.get_iteration(project.id, iterations[1].id)

iteration.status

import requests
import datetime
import os
import math
from PIL import Image, ImageDraw
import cv2
import numpy as np
import glob
import pandas as pd

url =
'https://player.vimeo.com/play/2532528922?s=535577674_1623629041_a6a680be0
78134b2b5834f7324d027d2&sid=64e0197edb8eb6b81b62b2bb2bf2c933f5ade0e1162361
8241&oauth2_token_id=&download=1'
r = requests.get(url, allow_redirects=True)

open('video.mp4', 'wb').write(r.content)

def convertFpsToTime(Fps):
 seconds = Fps /30
 return str(datetime.timedelta(seconds=seconds))

!mkdir buffers   #Folder to store input frames

!mkdir images   #Folder to store output frames

#Function to process each input frame
def countImg(count):
  font = cv2.FONT_HERSHEY_SIMPLEX
  with open("buffer" +str(count)+ ".jpg", mode="rb") as test_data:   #Impo
rting input frame
    results = predictor.detect_image_with_no_store(project.id, "Iteration1
",test_data)  #Sending it to the model
  image = cv2.imread("buffer" +str(count)+ ".jpg")   #Reading the input fr
ame
  fish=0
  for i in results.predictions:
    if i.probability >pred and i.tag_name in tags:
      #Calculating bounding box size
      box = i.bounding_box
      h,w,_ = image.shape
      start_point =(  math.floor( box.left * w)  , math.floor(box.top * h)
 )
      end_point =     math.floor( (box.left * w )+ box.width* w) , math.fl
oor( (box.top * h) + box.height* h )
      color = (255, 0, 0)
      thickness = 2
      #Drawing bounding box
      image = cv2.rectangle(image,start_point,end_point,color, thickness)
  cv2.imwrite("../images/" +str(count)+ ".jpg" ,image )   #Exporting outpu
t frame

cd buffers

#Cutting the video into frames
cap = cv2.VideoCapture("../video.mp4")  #Importing the video
count=0
# TRACKER INITIALIZATION
success, frame = cap.read()

while(cap.isOpened()):
    ret, img = cap.read()
    if ret == False :
      break
    cv2.imwrite("buffer" +str(count)+ ".jpg" ,img )  #Storing input frame
    print("read",count)
    countImg(count)  #Pass the frame to be processed
    count +=1

#Exporting the images to create the final video
img_array = []
scale_percent = 60 # percent of original size
for i in range(count):
   filename=f'images/{i}.jpg'
   img = cv2.imread(filename)
   width = int(img.shape[1] * scale_percent / 100)
   height = int(img.shape[0] * scale_percent / 100)
   dim = (width, height)
   resized = cv2.resize(img, dim, interpolation = cv2.INTER_AREA)
   img_array.append(resized)
out = cv2.VideoWriter('project2.mp4',cv2.VideoWriter_fourcc(*'DIVX'), 30,
dim)
for i in range(len(img_array)):
   out.write(img_array[i])
out.release()

!gsutil -q -m cp project.mp4 /content/drive/MyDrive/Azure1
```