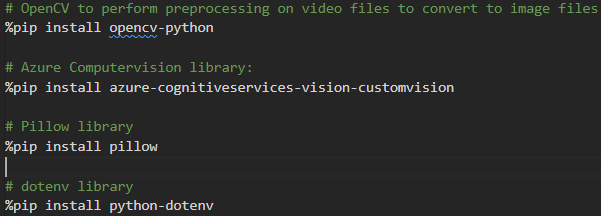
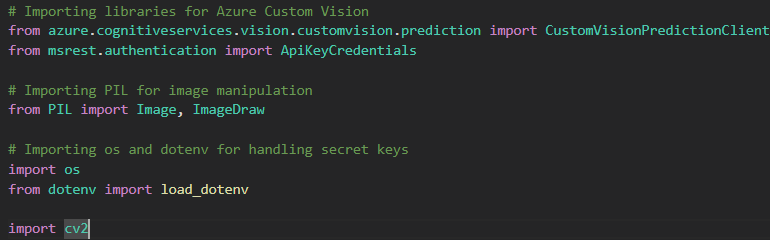
# **Task 5 D: Custom vision**

## Prerequisites

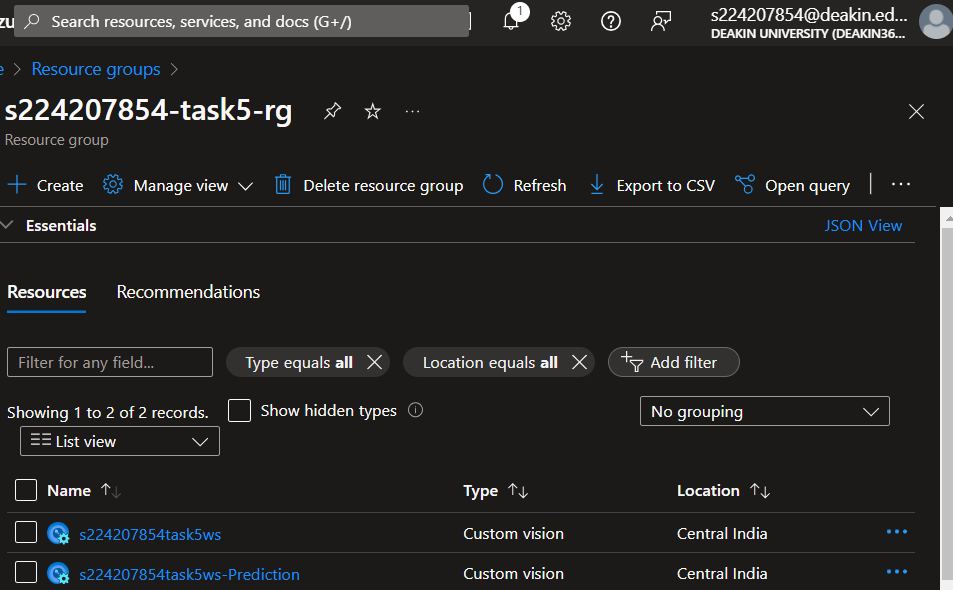
* Install libraries that will be required for this task



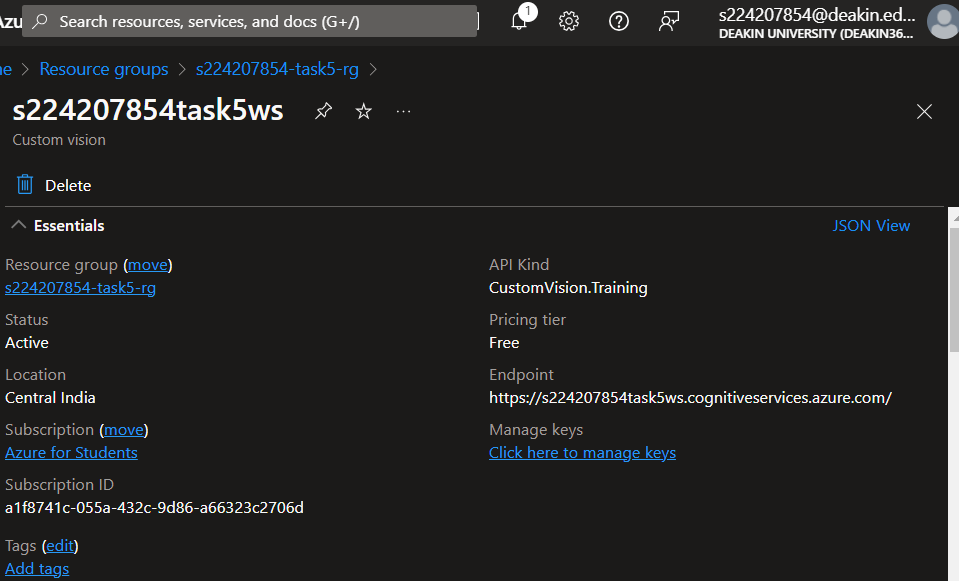
* Import libraries



* Setting Azure Custom Vision in Azure portal
  + Resource Group set-up:



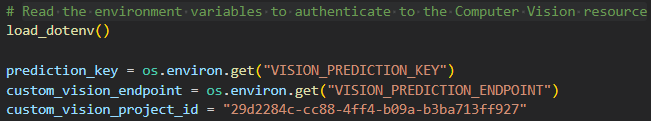
* + Creating custom vision service in Azure AI service



## Azure Custom Vision Notebook

In our notebook we will be using azure custom vision client to analyze different images and videos as input and then detecting the objects in videos.

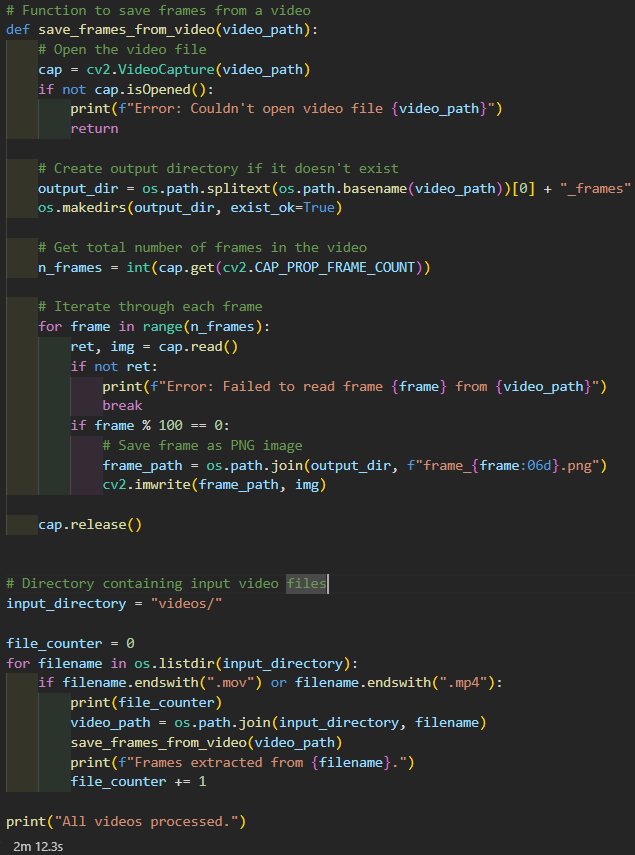
Reading the environment variables to authenticate to the Computer Vision resource.



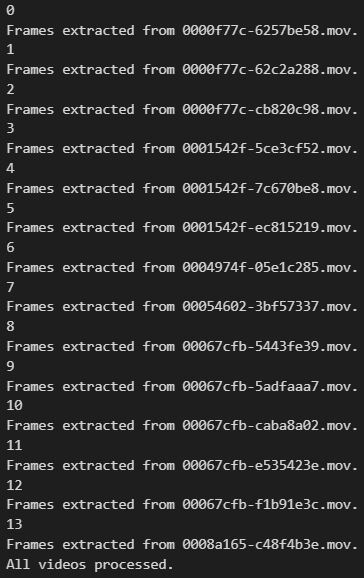
We're going to use a tool called OpenCV to change videos into pictures. First, we need to do something called "object detection." This means finding things in the video. Here's how we do it:

* Using OpenCV: This is a tool we'll use to work with videos.
* Converting the Video: We'll use something called cv2 VideoCapture to take the video and turn it into lots of pictures.
* Going through the Frames: We'll look at each picture, one by one, and change it into its own separate picture. Then, we can study each picture on its own.

This code uses two modules called OpenCV and OS. It's made to take frames out of video files. The function called "save\_frames\_from\_video" does a few things. First, it opens a video file. Then, it makes a new folder where it will put the frames. After that, it counts all the frames in the video. Next, it goes through each frame. It saves every 100th frame as a picture in the new folder.



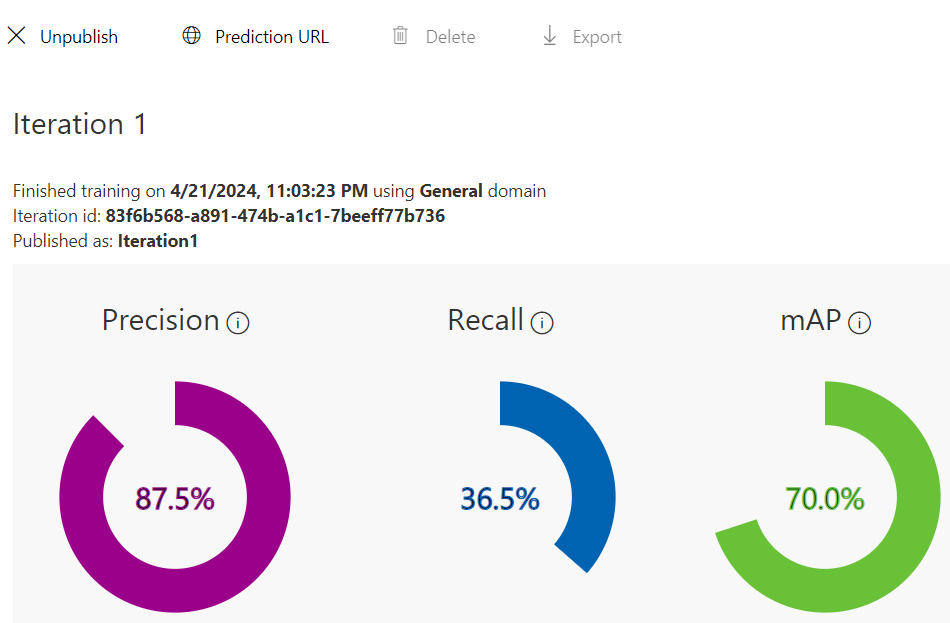
Output:



## Uploading images for Model Training

First, after getting pictures from the video, I put them into a project on Azure Custom Vision to teach a computer model.

* Tagging: I looked at each picture and put labels on them like "car", "person", "big vehicle" and more.
* Teaching the Model: I uploaded the labeled pictures to Azure Custom Vision. The computer uses these to learn how to find and name things in the pictures. This helps it to understand what's in the video frames better.



We're doing well in recognizing objects, with an accuracy of 87.5%. But we're not so good at finding all the objects, with recall rate 36.5% of them being detected. Overall, our performance is good, with a mean average precision(mAP) of 70%.

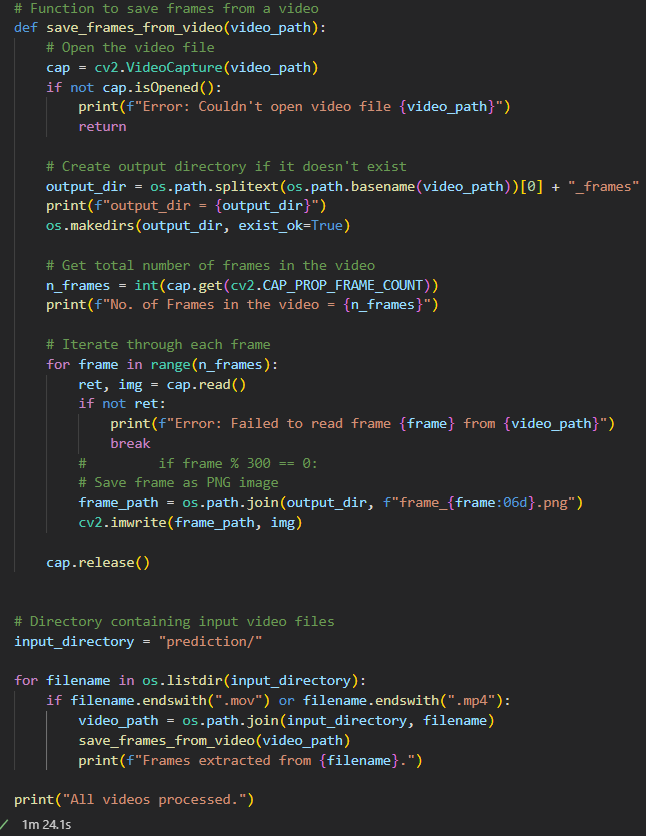
## Converting Test Video to Image Frame

I again defined the function save\_frames\_from\_video that extracts frames from a video file in the prediction/ directory and saves them as PNG images inside a directory with suffix \_frames.



## Predicting Object Tags Using Azure Custom Vision SDK

In this section, I utilize the Azure Custom Vision SDK to predict object tags within images extracted from the a test video. The provided code provides the necessary credentials and parameters for prediction which includes the prediction key, endpoint, model name, and threshold for confidence



## Converting the Video back to MP4 with Bounding Box

