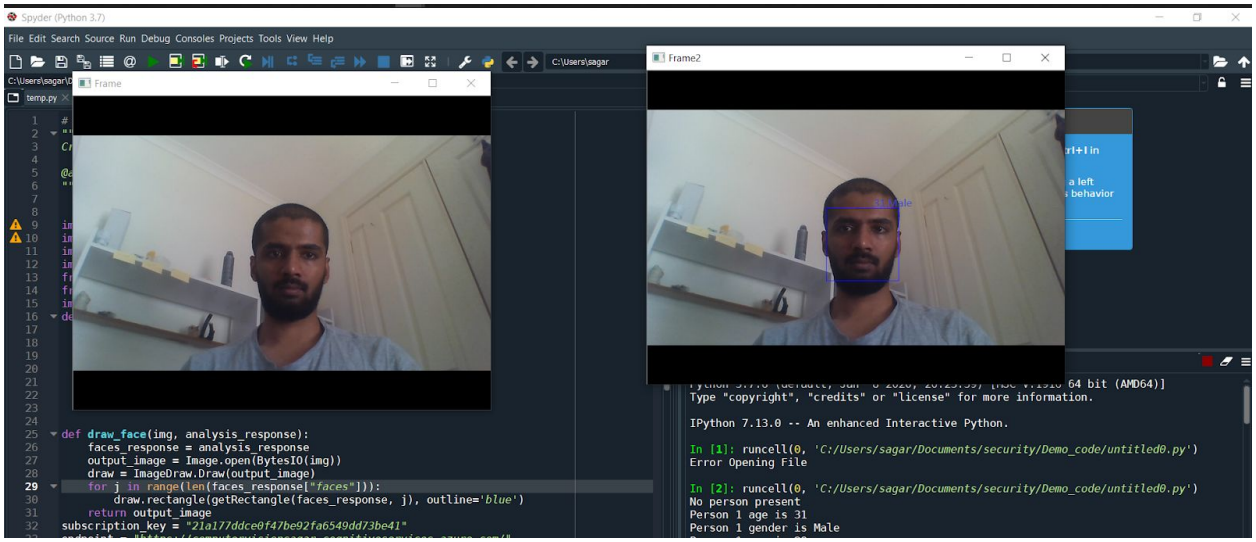
Task 5.3: Real-Time face detection



Code:

```
import os
import sys
import requests
import numpy as np
from io import BytesIO
from PIL import Image, ImageDraw
import cv2
def getRectangle(faceDictionary, k):
    rect = faceDictionary["faces"][k]['faceRectangle']
    left = rect['left']
    top = rect['top']
    bottom = left + rect['height']
    right = top + rect['width']
    return ((left, top), (bottom, right))


def draw_face(img, analysis_response):
    faces_response = analysis_response
    output_image = Image.open(BytesIO(img))
    draw = ImageDraw.Draw(output_image)
    for j in range(len(faces_response["faces"])):
        draw.rectangle(getRectangle(faces_response, j), outline='blue')
    return output_image
subscription_key = "21a177ddce0f47be92fa6549dd73be41"
endpoint = "https://computervisionsagar.cognitiveservices.azure.com/"
```

```
analyze_url = endpoint + "vision/v2.1/analyze"

cap = cv2.VideoCapture(0)
if not cap.isOpened():
   print("Error Opening File")
while cap.isOpened():
   ret, frame = cap.read()
   if ret:
      cv2.imshow('Frame', frame)
      cv2.waitKey(1000)
      retval, frame2 = cv2.imencode('.bmp', frame)
      image_data = frame2.tobytes()
      headers = {'Ocp-Apim-Subscription-Key': subscription_key,
                 'Content-Type': 'application/octet-stream'}

      params = {'visualFeatures': 'Faces'}
      response = requests.post(analyze_url, headers=headers, params=params,
data=image_data)
      response.raise_for_status()

      analysis = response.json()
      if analysis["faces"] == []:
         print("No person present")
      else:
         image_age = []
         image_gender = []
         for i in range(len(analysis["faces"])):
            image_age.append(analysis["faces"][i]['age'])
            image_gender.append(analysis["faces"][i]['gender'])
         for i in range(len(image_age)):
            print("Person {} age is {}".format((i + 1), image_age[i]))
            print("Person {} gender is {}".format((i + 1), image_gender[i]))

         image = draw_face(image_data, analysis)
         image=image.convert('RGB')
         open_cv_image = np.array(image)
         open_cv_image = open_cv_image[:, :, ::-1].copy()
         cv2.imshow('Frame2', open_cv_image)

      if cv2.waitKey(25) & 0xFF == ord('q'):
         break
```

```
    else:
        break
cap.release()
cv2.destroyAllWindows()
```