In [13]:
```
# Importing of Libraries
import boto3
import csv

# Create client
client  = boto3.client('rekognition',
                       aws_access_key_id = "AKIA3S3TYBBZMYEBGUNI",
                       aws_secret_access_key = "DILsSW4kXd8RoOHeAXTfyGzfsuSqCIZyW
                                                region_name = 'us-east-2'
                       )

def create_collection(collection_id):

    #Create a collection
    print('Creating collection:' + collection_id)

    #Using inbuilt function within rekognition client
    response=client.create_collection(CollectionId=collection_id)

    #Printing the collection details, save the printed output in a text file.
    print('Collection ARN: ' + response['CollectionArn'])
    print('Status code: ' + str(response['StatusCode']))
    print('Done...')

def main():
    collection_id='tiet-student' #Assign Collection ID Name
    create_collection(collection_id) # Creation of Collection ID

if __name__ == "__main__":
    main()
```

```
Creating collection:tiet-student
---------------------------------------------------------------------------
ResourceAlreadyExistsException                Traceback (most recent call last)
<ipython-input-13-05d37b50fe88> in <module>
     28
     29 if __name__ == "__main__":
---> 30     main()

<ipython-input-13-05d37b50fe88> in main()
     25 def main():
     26     collection_id='tiet-student' #Assign Collection ID Name
---> 27     create_collection(collection_id) # Creation of Collection ID
     28
     29 if __name__ == "__main__":

<ipython-input-13-05d37b50fe88> in create_collection(collection_id)
     16
     17     #Using inbuilt function within rekognition client
---> 18     response=client.create_collection(CollectionId=collection_id)
     19
     20     #Printing the collection details, save the printed output in a te
xt file.

~\anaconda3\lib\site-packages\botocore\client.py in _api_call(self, *args, **
kwargs)
```

```
    355                     "%s() only accepts keyword arguments." % py_opera
tion_name)
    356             # The "self" in this scope is referring to the BaseClien
t.
--> 357             return self._make_api_call(operation_name, kwargs)
    358
    359         _api_call.__name__ = str(py_operation_name)

~\anaconda3\lib\site-packages\botocore\client.py in _make_api_call(self, oper
ation_name, api_params)
    674             error_code = parsed_response.get("Error", {}).get("Code")
    675             error_class = self.exceptions.from_code(error_code)
--> 676             raise error_class(parsed_response, operation_name)
    677         else:
    678             return parsed_response

ResourceAlreadyExistsException: An error occurred (ResourceAlreadyExistsExcep
tion) when calling the CreateCollection operation: The collection id: tiet-st
udent already exists
```

In [9]:
```
# Defining a function to add faces to the collection
def add_faces_to_collection(bucket,photo,collection_id):

    #here, we have used MaxFaces as 1, so make sure you use only portrait images
    #so that you can be sure which face has been detected and put into the collec
    response = client.index_faces(CollectionId=collection_id,
                                  Image={'S3Object':{'Bucket':bucket,'Name':photo}}
                                  ExternalImageId=photo,
                                  MaxFaces=1,
                                  QualityFilter="AUTO",
                                  DetectionAttributes=['ALL'])


    print ('Results for ' + photo)
    print('Faces indexed:')
    for faceRecord in response['FaceRecords']:
         print('  Face ID: ' + faceRecord['Face']['FaceId'])
         print('  External Id:' + faceRecord['Face']["ExternalImageId"])
         print('  Location: {}'.format(faceRecord['Face']['BoundingBox']))


    print('Faces not indexed:')
    for unindexedFace in response['UnindexedFaces']:
        print(' Location: {}'.format(unindexedFace['FaceDetail']['BoundingBox']))
        print(' Reasons:')
        for reason in unindexedFace['Reasons']:
            print('    ' + reason)
    return len(response['FaceRecords'])

# Defining a main function
def main():
    bucket = 'tiet-student'  #Your Bucket Name
    collection_id='tiet-student'  #Your Collection Name you created in the last s
    #List the names of all the photos you want to put in the colletion
    #these are the filepaths of the images in AWS S3
    # give them names in such a way that removing the last 4 characters of filena
    # ".jpg", we can get to know the name of person and thus create folders by th
    photos = ["sehwag.jpg","Ganguly.jpg","kapildev.jpg"]

    for photo in photos:
        indexed_faces_count=add_faces_to_collection(bucket, photo, collection_id)
        print("Faces indexed count: " + str(indexed_faces_count))


if __name__ == "__main__":
    main()
```

```
Results for sehwag.jpg
Faces indexed:
  Face ID: f770d2a5-931a-4f2a-9dbe-75b4b1c9636f
  External Id:sehwag.jpg
  Location: {'Width': 0.3840073347091675, 'Height': 0.4242989122867584, 'Lef
t': 0.44988515973091125, 'Top': 0.1449604630470276}
Faces not indexed:
Faces indexed count: 1
Results for Ganguly.jpg
```

```
Faces indexed:
  Face ID: be599684-9331-4347-92ad-61340717842f
  External Id:Ganguly.jpg
  Location: {'Width': 0.2629672586917877, 'Height': 0.5258278846740723, 'Lef
t': 0.3263614773750305, 'Top': 0.18856088817119598}
Faces not indexed:
Faces indexed count: 1
Results for kapildev.jpg
Faces indexed:
  Face ID: 0d684cc8-956b-4051-86cd-39d049161ded
  External Id:kapildev.jpg
  Location: {'Width': 0.19932161271572113, 'Height': 0.35589325428009033, 'Le
ft': 0.40642550587654114, 'Top': 0.14667260646820068}
Faces not indexed:
Faces indexed count: 1
```

In [10]:
```
# Defining function to list the faces
def list_faces_in_collection(collection_id):

    maxResults=2
    faces_count=0
    tokens=True
   #using built in function of rekognition
    response=client.list_faces(CollectionId=collection_id,
                               MaxResults=maxResults)

    print('Faces in collection : ' + collection_id)
    while tokens:
        faces=response['Faces']
        #to print details of each face in the collection
        for face in faces:
            print("Face Id     : " + face["FaceId"]) #The id by which Rekognitior
            print("External Id : " + face["ExternalImageId"]) #The name by which
            faces_count+=1

        if 'NextToken' in response:
            nextToken=response['NextToken']
            response=client.list_faces(CollectionId=collection_id,
                                       NextToken=nextToken,MaxResults=maxResults)
        else:
            tokens=False
    return faces_count #returns the total number of faces found in collection

def main():
    bucket = 'tiet-student' # Replace with your bucket name
    collection_id='tiet-student' # Replace with your collection id

    faces_count=list_faces_in_collection(collection_id)
    print("faces count: " + str(faces_count))

if __name__ == "__main__":
    main()
```

```
Faces in collection : tiet-student
Face Id     : 0d684cc8-956b-4051-86cd-39d049161ded
External Id : kapildev.jpg
Face Id     : be599684-9331-4347-92ad-61340717842f
External Id : Ganguly.jpg
Face Id     : f770d2a5-931a-4f2a-9dbe-75b4b1c9636f
External Id : sehwag.jpg
faces count: 3
```

In [8]:
```
# Importing of Liraries
import cv2
import boto3
import csv
import datetime
import time
import imutils
import requests

# Enabling the Cv2
video_capture = cv2.VideoCapture(0)
faceCascade = cv2.CascadeClassifier(cv2.data.haarcascades +'haarcascade_frontalfa

# Creation of client to rekognition Service
client  = boto3.client('rekognition',
                       aws_access_key_id = "AKIA3S3TYBBZMYEBGUNI",
                       aws_secret_access_key = "DILsSW4kXd8RoOHeAXTfyGzfsuSqCIZyW
                                               region_name = 'us-east-2')
# Creation of client to S3 Service
s3client  = boto3.client('s3',
                       aws_access_key_id = "AKIA3S3TYBBZMYEBGUNI",
                       aws_secret_access_key = "DILsSW4kXd8RoOHeAXTfyGzfsuSqCIZyW
                                               region_name = 'us-east-1'
                       )

# Declaring global Variables
global name,period,url

# Defining of upload image function to S3
def uploadimage():

    bucket = 'tiet-student' # Replace with your bucket name

    filename = 'test.jpg' # Naming of captured to store in S3
    relative_filename = 'test.jpg'

    s3client.upload_file(filename, bucket, relative_filename)
    print("file Uploaded")

# Comparing of the captures image with S3
def photo():

    bucket = 'tiet-student' # Replace with your bucket name
    collection_id = 'tiet-student'
    fileNames = ['test.jpg']
    threshold = 70 # Threshold limit for the similarity
    maxFaces = 2
    #here max faces is the number of faces it shoudl give as output if more than
    #being rekognized with abover threshold confidence,
    for fileName in fileNames:
        response=client.search_faces_by_image(CollectionId=collection_id,
                                    Image={'S3Object':
                                           {'Bucket':bucket,
                                            'Name':fileName}},
                                    FaceMatchThreshold=threshold,
                                    MaxFaces=maxFaces)
```

```
        faceMatches=response['FaceMatches']
        print ('Matching faces')
        for match in faceMatches:
            print ('FaceId:' + match['Face']['FaceId'])
            print ('External Id:' + match['Face']["ExternalImageId"])
            #Assigning a variable for external id
            name1=match['Face']["ExternalImageId"]
            name=name1.split(".") # Spliting the External id to remove .jpg exter
            name=name[0]
            date=str(datetime.datetime.now())[0:11] # Capturing time
            time=time_1.strftime('%H')
            period = ""
            if(time == '9'):
                period = "Period1"
            elif(time == '10'):
                period = "Period2"
            else:
                period = "Period3"
            # Hitting API Gateway url to send captured image name & period
            url = "https://z8lugn4i15.execute-api.us-east-2.amazonaws.com/attenda
            status = requests.request("GET",url)
            print(status.json())
            print("uploaded to DB")
            print("Student Detected :"+name)
            print ('Similarity: ' + "{:.2f}".format(match['Similarity']) + "%")


# Main function
while True:

    current_time = datetime.datetime.now().strftime("%d-%m-%y  %H-%M-%S ")
    time_1 = dt.datetime.now()
    print("present time:",time_1)
    hr = time_1.strftime('%H')
    sd = time_1.minute;

    ret, frame = video_capture.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = faceCascade.detectMultiScale(
        gray,
            scaleFactor=1.2,
             minNeighbors=5
           # minSize=(30, 30)
        )
    # Draw a rectangle around the faces
    for (x, y, w, h) in faces:
        print (faces.shape)
        cv2.putText(frame, "faces detected: " + str(faces.shape[0]), (50, 30),
                                              cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0,
        cv2.rectangle(frame, (x, y), (x+w+30, y+h+30), (0, 255, 0), 1)
        roi_gray = gray[y:y+h, x:x+w]
        roi_color = frame[y:y+h+30, x:x+w+30]
        imgname = "test.jpg"
        cv2.imwrite(imgname, roi_color)
        uploadimage()
        a =  photo()
```

```
        print(a)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

    cv2.imshow('Video', frame)


video_capture.release()
cv2.destroyAllWindows()
```

```
present time: 2021-03-14 12:45:47.616228
(1, 4)
file Uploaded
Matching faces
FaceId:f770d2a5-931a-4f2a-9dbe-75b4b1c9636f
External Id:sehwag.jpg
{'statusCode': 200, 'body': '"Hello from Lambda!"'}
uploaded to DB
Student Detected :sehwag
Similarity: 100.00%
None
present time: 2021-03-14 12:45:53.969857
(1, 4)
file Uploaded
Matching faces
FaceId:f770d2a5-931a-4f2a-9dbe-75b4b1c9636f
External Id:sehwag.jpg
{'statusCode': 200, 'body': '"Hello from Lambda!"'}
uploaded to DB
```

In [ ]:

In [ ]: