In [1]:

```
import cv2 as cv
import os
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'
```

In [2]:

```
img = cv.imread("dataset/vijai/v6.jpeg")
```

In [3]:

```
img = cv.cvtColor(img, cv.COLOR_BGR2RGB)
plt.imshow(img) # RGB
```

Out[3]:

```
<matplotlib.image.AxesImage at 0x163a30450>
```

In [4]:

```
from mtcnn.mtcnn import MTCNN

detector = MTCNN()
results = detector.detect_faces(img)
```

```
1/1 [==============================] - 0s 99ms/step
1/1 [==============================] - 0s 43ms/step
1/1 [==============================] - 0s 13ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
10/10 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 53ms/step
```

In [5]:

```
results
```

Out[5]:

```
[{'box': [262, 123, 263, 364],
  'confidence': 0.9997991919517517,
  'keypoints': {'left_eye': (368, 239),
   'right_eye': (483, 253),
   'nose': (438, 280),
   'mouth_left': (361, 367),
   'mouth_right': (466, 378)}}]
```

In [6]:

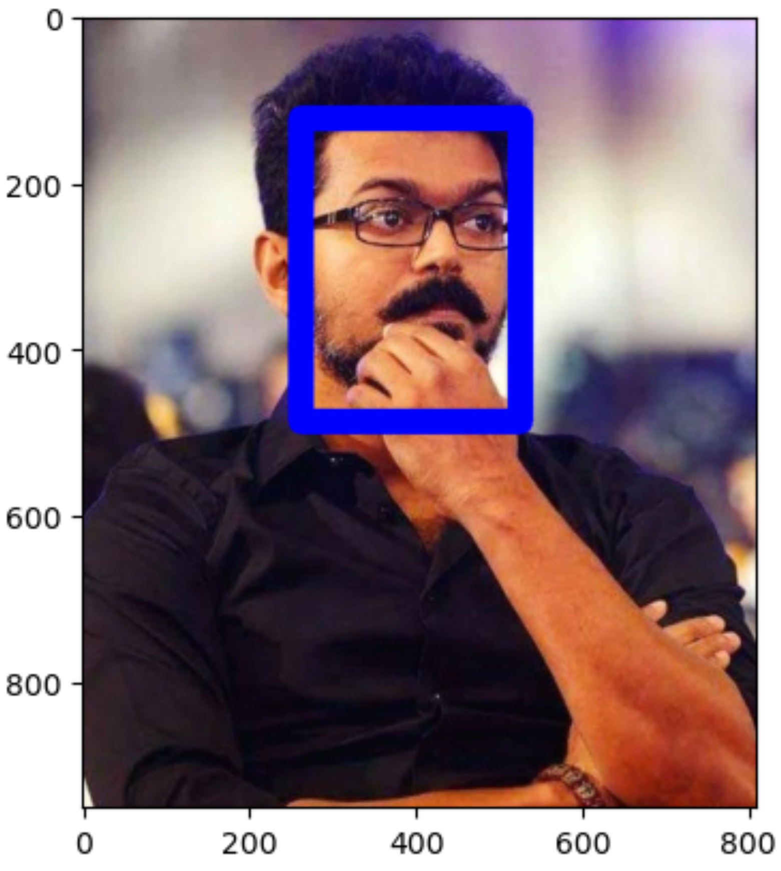
```
x,y,w,h = results[0]['box']
```

In [7]:

```
img = cv.rectangle(img, (x,y), (x+w, y+h), (0,0,255), 30)
plt.imshow(img)
```

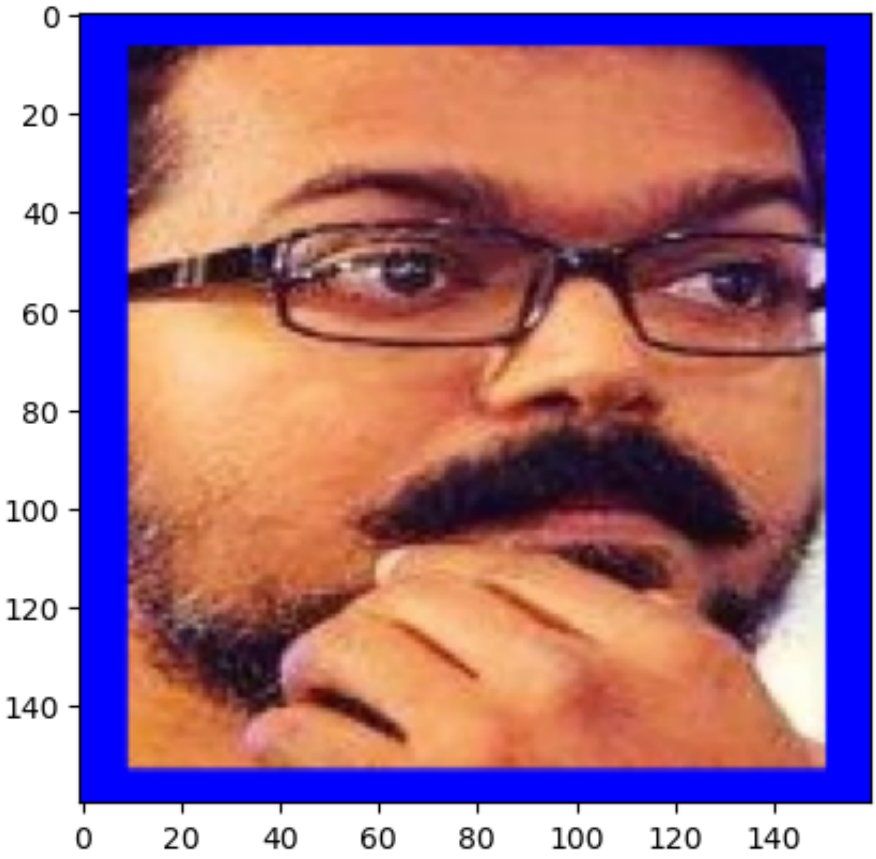
Out[7]:

```
<matplotlib.image.AxesImage at 0x16bfa72d0>
```



In [8]:
```
my_face = img[y:y+h, x:x+w]
#Facenet takes as input 160x160
my_face = cv.resize(my_face, (160,160))
plt.imshow(my_face)
```

Out[8]:
```
<matplotlib.image.AxesImage at 0x16bfddf10>
```



In [9]:
```
my_face
```

```
Out[9]: array([[[  0,   0, 255],
                [  0,   0, 255],
                [  0,   0, 255],
                ...,
                [  0,   0, 255],
                [  0,   0, 255],
                [  0,   0, 255]],

               [[  0,   0, 255],
                [  0,   0, 255],
                [  0,   0, 255],
                ...,
                [  0,   0, 255],
                [  0,   0, 255],
                [  0,   0, 255]],

               [[  0,   0, 255],
                [  0,   0, 255],
                [  0,   0, 255],
                ...,
                [  0,   0, 255],
                [  0,   0, 255],
                [  0,   0, 255]],

               ...,

               [[  0,   0, 255],
                [  0,   0, 255],
                [  0,   0, 255],
                ...,
                [  0,   0, 255],
                [  0,   0, 255],
                [  0,   0, 255]],

               [[  0,   0, 255],
                [  0,   0, 255],
                [  0,   0, 255],
                ...,
                [  0,   0, 255],
                [  0,   0, 255],
                [  0,   0, 255]],

               [[  0,   0, 255],
                [  0,   0, 255],
                [  0,   0, 255],
                ...,
                [  0,   0, 255],
                [  0,   0, 255],
                [  0,   0, 255]]], dtype=uint8)
```

In [10]:

```
class FACELOADING:
    def __init__(self, directory):
        self.directory = directory
        self.target_size = (160,160)
        self.X = []
        self.Y = []
        self.detector = MTCNN()


    def extract_face(self, filename):
        img = cv.imread(filename)
        img = cv.cvtColor(img, cv.COLOR_BGR2RGB)
        x,y,w,h = self.detector.detect_faces(img)[0]['box']
        x,y = abs(x), abs(y)
        face = img[y:y+h, x:x+w]
        face_arr = cv.resize(face, self.target_size)
        return face_arr


    def load_faces(self, dir):
        FACES = []
        for im_name in os.listdir(dir):
            try:
                path = dir + im_name
                single_face = self.extract_face(path)
                FACES.append(single_face)
            except Exception as e:
                pass
        return FACES

    def load_classes(self):
        for sub_dir in os.listdir(self.directory):
            path = self.directory +'/'+ sub_dir+'/'
            FACES = self.load_faces(path)
            labels = [sub_dir for _ in range(len(FACES))]
            print(f"Loaded successfully: {len(labels)}")
            self.X.extend(FACES)
            self.Y.extend(labels)

        return np.asarray(self.X), np.asarray(self.Y)


    def plot_images(self):
        plt.figure(figsize=(18,16))
        for num,image in enumerate(self.X):
            ncols = 3
            nrows = len(self.Y)//ncols + 1
            plt.subplot(nrows,ncols,num+1)
            plt.imshow(image)
            plt.axis('off')
```

In [11]:

```
from mtcnn.mtcnn import MTCNN

detector = MTCNN()
faceloading = FACELOADING("dataset")
```

In [12]:

```
import os

file_path = 'dataset/.DS_Store'

if os.path.exists(file_path):
    os.remove(file_path)
```

In [13]:

```
X, Y = faceloading.load_classes()
```

```
1/1 [==============================] - 0s 53ms/step
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
WARNING:tensorflow:5 out of the last 19 calls to <function Model.make_pre
dict_function.<locals>.predict_function at 0x174fbcf40> triggered tf.func
tion retracing. Tracing is expensive and the excessive number of tracings
could be due to (1) creating @tf.function repeatedly in a loop, (2) passi
ng tensors with different shapes, (3) passing Python objects instead of t
ensors. For (1), please define your @tf.function outside of the loop. For
(2), @tf.function has reduce_retracing=True option that can avoid unneces
sary retracing. For (3), please refer to https://www.tensorflow.org/guide
/function#controlling_retracing and https://www.tensorflow.org/api_docs/p
ython/tf/function for  more details.
4/4 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 55ms/step
1/1 [==============================] - 0s 27ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 14ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
10/10 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 328ms/step
1/1 [==============================] - 0s 129ms/step
1/1 [==============================] - 0s 77ms/step
1/1 [==============================] - 0s 42ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 16ms/step
1/1 [==============================] - 0s 13ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
```

```
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
163/163 [==============================] - 0s 3ms/step
3/3 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
2/2 [==============================] - 0s 4ms/step
1/1 [==============================] - 0s 14ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 13ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
3/3 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 14ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 13ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
23/23 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 15ms/step
1/1 [==============================] - 0s 26ms/step
1/1 [==============================] - 0s 17ms/step
1/1 [==============================] - 0s 13ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
9/9 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 41ms/step
1/1 [==============================] - 0s 27ms/step
1/1 [==============================] - 0s 16ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
```

```
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
6/6 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 14ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
6/6 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 12ms/step
Loaded successfully: 9
1/1 [==============================] - 0s 25ms/step
1/1 [==============================] - 0s 17ms/step
1/1 [==============================] - 0s 13ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
10/10 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 20ms/step
1/1 [==============================] - 0s 15ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
8/8 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
3/3 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 16ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 11ms/step
```

```
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
3/3 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 10ms/step
2/2 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 13ms/step
1/1 [==============================] - 0s 15ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
3/3 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
4/4 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
2/2 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 26ms/step
1/1 [==============================] - 0s 18ms/step
1/1 [==============================] - 0s 13ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
```

```
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
11/11 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
3/3 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 11ms/step
Loaded successfully: 10
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
2/2 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 29ms/step
1/1 [==============================] - 0s 20ms/step
1/1 [==============================] - 0s 14ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
5/5 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 20ms/step
1/1 [==============================] - 0s 15ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
5/5 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 18ms/step
1/1 [==============================] - 0s 14ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
```

```
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
12/12 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 30ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 14ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
50/50 [==============================] - 0s 3ms/step
4/4 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 20ms/step
1/1 [==============================] - 0s 13ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
8/8 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 13ms/step
1/1 [==============================] - 0s 16ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 7ms/step
5/5 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 19ms/step
1/1 [==============================] - 0s 14ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
8/8 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 11ms/step
Loaded successfully: 8
1/1 [==============================] - 0s 12ms/step
```

```
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
4/4 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
3/3 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 93ms/step
1/1 [==============================] - 0s 48ms/step
1/1 [==============================] - 0s 27ms/step
1/1 [==============================] - 0s 19ms/step
1/1 [==============================] - 0s 14ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
13/13 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
3/3 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 41ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 17ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 10ms/step
```

```
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
18/18 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 36ms/step
1/1 [==============================] - 0s 20ms/step
1/1 [==============================] - 0s 14ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
5/5 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
2/2 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 53ms/step
1/1 [==============================] - 0s 28ms/step
1/1 [==============================] - 0s 19ms/step
1/1 [==============================] - 0s 14ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
15/15 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 12ms/step
Loaded successfully: 10
1/1 [==============================] - 0s 115ms/step
```

```
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 15ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
15/15 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 70ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 16ms/step
1/1 [==============================] - 0s 13ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
17/17 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 13ms/step
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 15ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 9ms/step
4/4 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 25ms/step
1/1 [==============================] - 0s 16ms/step
1/1 [==============================] - 0s 13ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
4/4 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 9ms/step
```

```
1/1 [==============================] - 0s 39ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 16ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
6/6 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 15ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
5/5 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 8ms/step
2/2 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 14ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
3/3 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 14ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
3/3 [==============================] - 0s 3ms/step
1/1 [==============================] - 0s 10ms/step
```

```
1/1 [==============================] - 0s 39ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 17ms/step
1/1 [==============================] - 0s 13ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 7ms/step
1/1 [==============================] - 0s 7ms/step
5/5 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 9ms/step
Loaded successfully: 10
```

In [14]:

```
plt.figure(figsize=(16,12))
for num,image in enumerate(X):
    ncols = 3
    nrows = len(Y)//ncols + 1
    plt.subplot(nrows,ncols,num+1)
    plt.imshow(image)
    plt.axis('off')
```

In [15]:

```
from keras_facenet import FaceNet
embedder = FaceNet()

def get_embedding(face_img):
    face_img = face_img.astype('float32') # 3D(160x160x3)
    face_img = np.expand_dims(face_img, axis=0)
    # 4D (Nonex160x160x3)
    yhat= embedder.embeddings(face_img)
    return yhat[0] # 512D image (1x1x512)
```

In [16]:

```
EMBEDDED_X = []

for img in X:
    EMBEDDED_X.append(get_embedding(img))

EMBEDDED_X = np.asarray(EMBEDDED_X)
```

```
1/1 [==============================] - 1s 750ms/step
1/1 [==============================] - 0s 36ms/step
1/1 [==============================] - 0s 39ms/step
1/1 [==============================] - 0s 36ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 34ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 34ms/step
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 34ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 34ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 34ms/step
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 32ms/step
```

In [17]:

```python
np.savez_compressed('faces_embeddings_done_4classes.npz', EMBEDDED_X, Y)
```

In [18]:

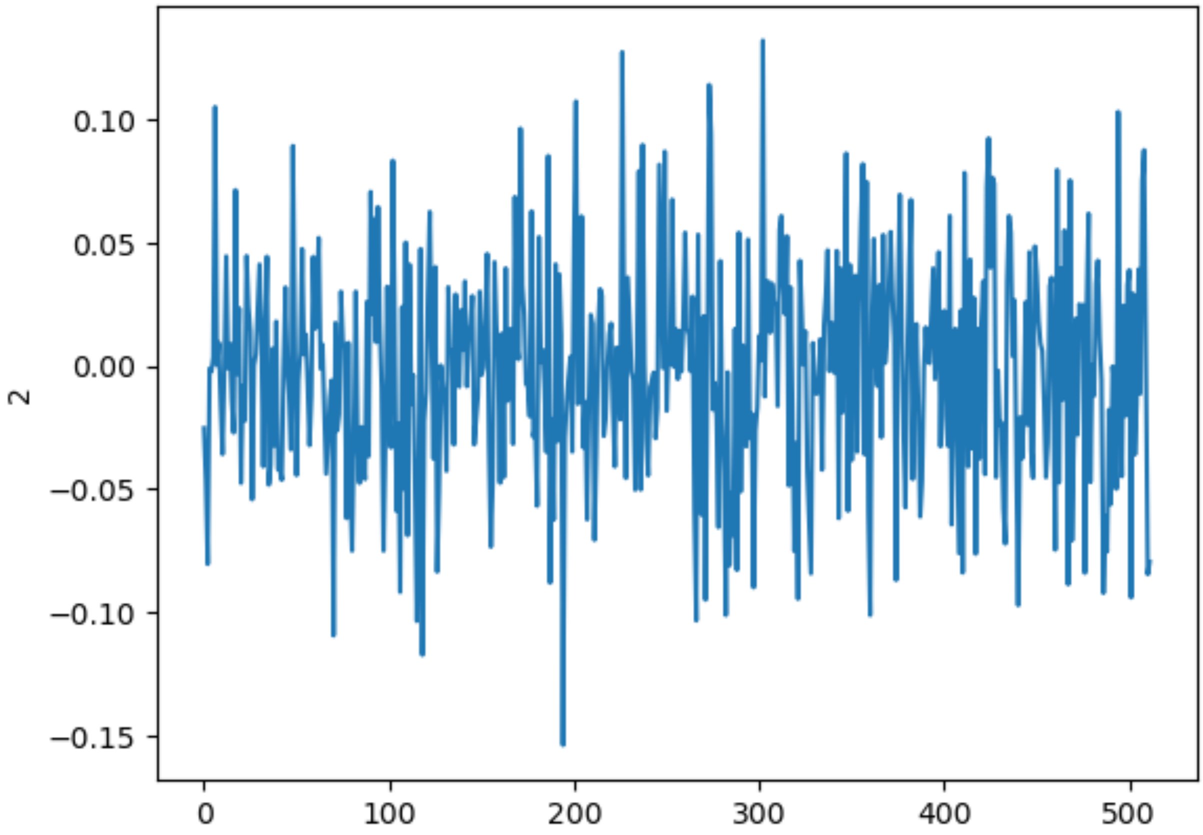
```python
from sklearn.preprocessing import LabelEncoder

encoder = LabelEncoder()
encoder.fit(Y)
Y = encoder.transform(Y)
```

In [19]:

```
plt.plot(EMBEDDED_X[0])
plt.ylabel(Y[0])
```

Out[19]:

```
Text(0, 0.5, '2')
```



In [20]:

```
y
```

Out[20]:

```
123
```

In [21]:

```
from sklearn.model_selection import train_test_split

X_train, X_test, Y_train, Y_test = train_test_split(EMBEDDED_X, Y, shuffl
```

In [22]:

```
from sklearn.svm import SVC
model = SVC(kernel='linear', probability=True)
model.fit(X_train, Y_train)
```

Out[22]:

```
▾ SVC
SVC(kernel='linear', probability=True)
```

In [23]:

```
ypreds_train = model.predict(X_train)
ypreds_test = model.predict(X_test)
```

In [24]:

```
from sklearn.metrics import accuracy_score

accuracy_score(Y_train, ypreds_train)
```

```
Out[24]: 1.0
```

```
In [25]: accuracy_score(Y_test,ypreds_test)
```

```
Out[25]: 1.0
```

```
In [26]: t_im = cv.imread("a10.webp")
         t_im = cv.cvtColor(t_im, cv.COLOR_BGR2RGB)
         x,y,w,h = detector.detect_faces(t_im)[0]['box']
```

```
1/1 [==============================] - 0s 56ms/step
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 9ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
1/1 [==============================] - 0s 8ms/step
4/4 [==============================] - 0s 2ms/step
1/1 [==============================] - 0s 53ms/step
```

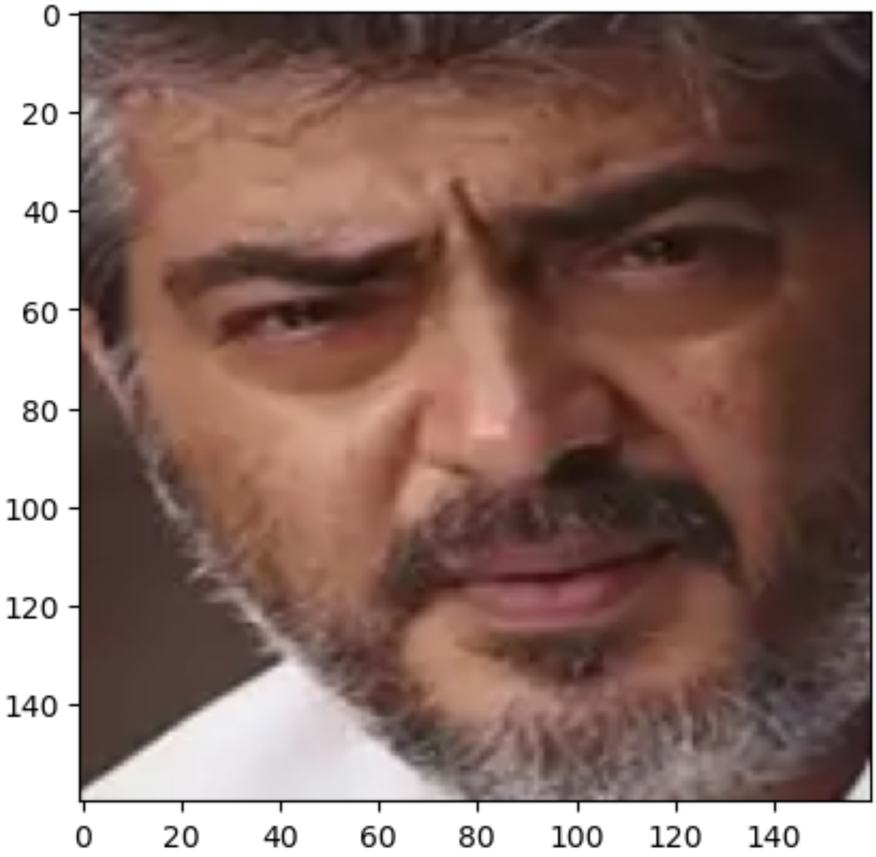
```
In [27]: t_im = t_im[y:y+h, x:x+w]
         t_im = cv.resize(t_im, (160,160))
         test_im = get_embedding(t_im)
```

```
1/1 [==============================] - 0s 41ms/step
```

```
In [28]: test_im = [test_im]
         ypreds = model.predict(test_im)
```

```
In [29]: plt.imshow(t_im)
         encoder.inverse_transform(ypreds)[0]
```

```
Out[29]: 'ajith'
```



```
In [ ]:
```