In [17]:
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing import image
import matplotlib.pyplot as plt
import tensorflow as tf
import numpy as np
import cv2
import os
#image data generator is the package to lable the images & it will automatically l
```

In [18]:
```
import warnings
warnings.filterwarnings
```

Out[18]:
```
<function warnings.filterwarnings(action, message='', category=<class 'Warning'>,
module='', lineno=0, append=False)>
```

In [19]:
```
img = image.load_img(r"C:\Users\gadel\VS Code projects\Emotion Detector\CNN\Testir
```

In [20]:
```
plt.imshow(img)
```

Out[20]:
```
<matplotlib.image.AxesImage at 0x1c50ffafca0>
```

In [21]:
```
i1 = cv2.imread(r"C:\Users\gadel\VS Code projects\Emotion Detector\CNN\Testing\Scr
i1
# 3 dimension metrics are created for the image
# the value ranges from 0-255
```

Out[21]:
```
array([[[0, 0, 0],
        [0, 0, 0],
        [0, 0, 0],
        ...,
        [0, 0, 0],
        [0, 0, 0],
        [0, 0, 0]],

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

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

       ...,

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

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

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

In [22]:
```
i1.shape
# shape of your image height, weight, rgb
```

Out[22]:
```
(1080, 1920, 3)
```

In [23]:
```
train = ImageDataGenerator(rescale = 1/255)
validataion = ImageDataGenerator(rescale = 1/255)
# to scale all the images i need to divide with 255
# we need to resize the image using 200, 200 pixel
```

In [24]:
```
train_dataset = train.flow_from_directory(r"C:\Users\gadel\VS Code projects\Emotic
                                          target_size = (200,200),
                                          batch_size = 3,
                                          class_mode = 'binary')
validataion_dataset = validataion.flow_from_directory(r"C:\Users\gadel\VS Code pro
                                           target_size = (200,200),
                                           batch_size = 3,
                                           class_mode = 'binary')
```

```
Found 7 images belonging to 2 classes.
Found 7 images belonging to 2 classes.
```

In [25]:
```
train_dataset.class_indices
```

Out[25]:
```
{'Happy': 0, 'Sad': 1}
```

In [26]:
```
train_dataset.classes
```

Out[26]:
```
array([0, 0, 0, 0, 1, 1, 1])
```

In [27]:
```
# now we are applying maxpooling

model = tf.keras.models.Sequential([ tf.keras.layers.Conv2D(16,(3,3),activation =
                                    tf.keras.layers.MaxPool2D(2,2), #3 filtr we ap
                                    #
                                    tf.keras.layers.Conv2D(32,(3,3),activation = '
                                    tf.keras.layers.MaxPool2D(2,2),
                                    #
                                    tf.keras.layers.Conv2D(64,(3,3),activation = '
                                    tf.keras.layers.MaxPool2D(2,2),
                                    ##
                                    tf.keras.layers.Flatten(),
                                    ##
                                    tf.keras.layers.Dense(512, activation = 'relu'
                                    #
                                    tf.keras.layers.Dense(1,activation= 'sigmoid')
                                    ]
                                    )
```

In [28]:
```
model.compile(loss='binary_crossentropy',
              optimizer = tf.keras.optimizers.RMSprop(lr = 0.001),
              metrics = ['accuracy']
              )
```

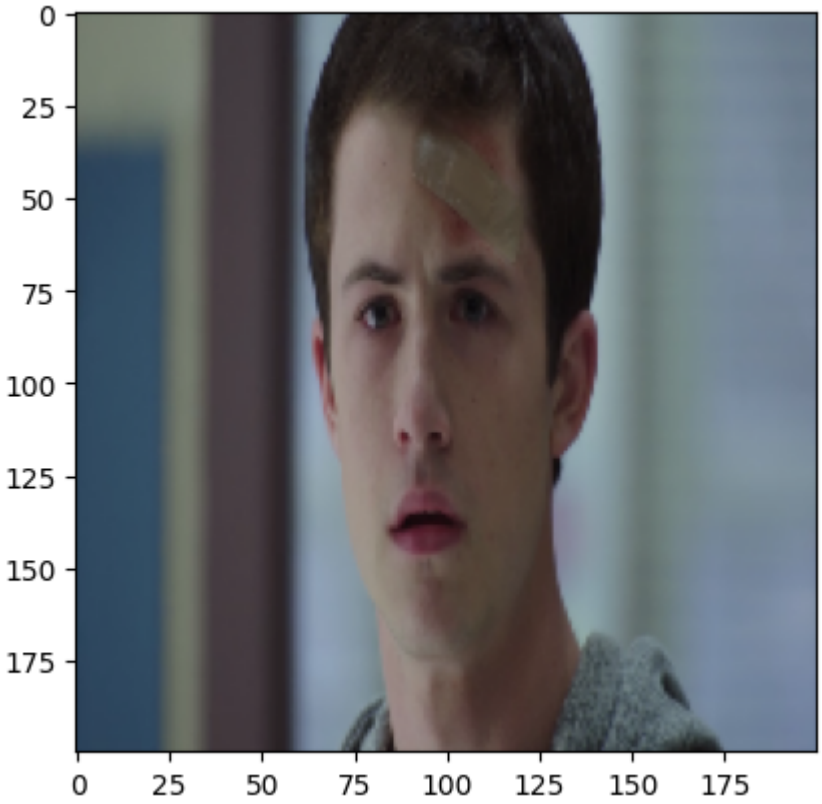
In [29]:
```
model_fit = model.fit(train_dataset,
                      steps_per_epoch = 3,
                      epochs = 10,
                      validation_data = validataion_dataset)
```

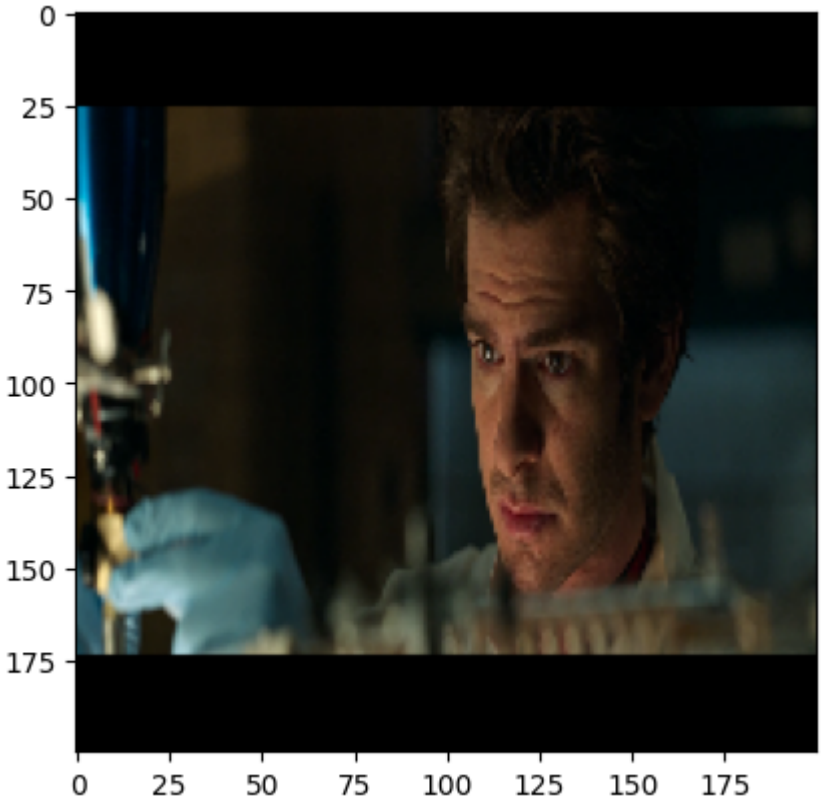
```
Epoch 1/10
3/3 [==============================] - 5s 1s/step - loss: 5.9917 - accuracy: 0.7143
- val_loss: 6.0987 - val_accuracy: 0.5714
Epoch 2/10
3/3 [==============================] - 2s 689ms/step - loss: 1.4254 - accuracy: 0.8
571 - val_loss: 6.8977 - val_accuracy: 0.0000e+00
Epoch 3/10
3/3 [==============================] - 2s 687ms/step - loss: 0.0978 - accuracy: 1.0
000 - val_loss: 8.0969 - val_accuracy: 0.4286
Epoch 4/10
3/3 [==============================] - 2s 661ms/step - loss: 2.4900 - accuracy: 0.4
286 - val_loss: 4.5255 - val_accuracy: 0.2857
Epoch 5/10
3/3 [==============================] - 2s 783ms/step - loss: 0.8447 - accuracy: 0.8
571 - val_loss: 4.5804 - val_accuracy: 0.0000e+00
Epoch 6/10
3/3 [==============================] - 2s 793ms/step - loss: 0.0753 - accuracy: 1.0
000 - val_loss: 5.2270 - val_accuracy: 0.0000e+00
Epoch 7/10
3/3 [==============================] - 2s 676ms/step - loss: 0.0238 - accuracy: 1.0
000 - val_loss: 5.6359 - val_accuracy: 0.0000e+00
Epoch 8/10
3/3 [==============================] - 2s 797ms/step - loss: 0.0211 - accuracy: 1.0
000 - val_loss: 6.1704 - val_accuracy: 0.0000e+00
Epoch 9/10
3/3 [==============================] - 2s 616ms/step - loss: 0.0074 - accuracy: 1.0
000 - val_loss: 6.4930 - val_accuracy: 0.0000e+00
Epoch 10/10
3/3 [==============================] - 2s 675ms/step - loss: 0.0053 - accuracy: 1.0
000 - val_loss: 6.7771 - val_accuracy: 0.0000e+00
```

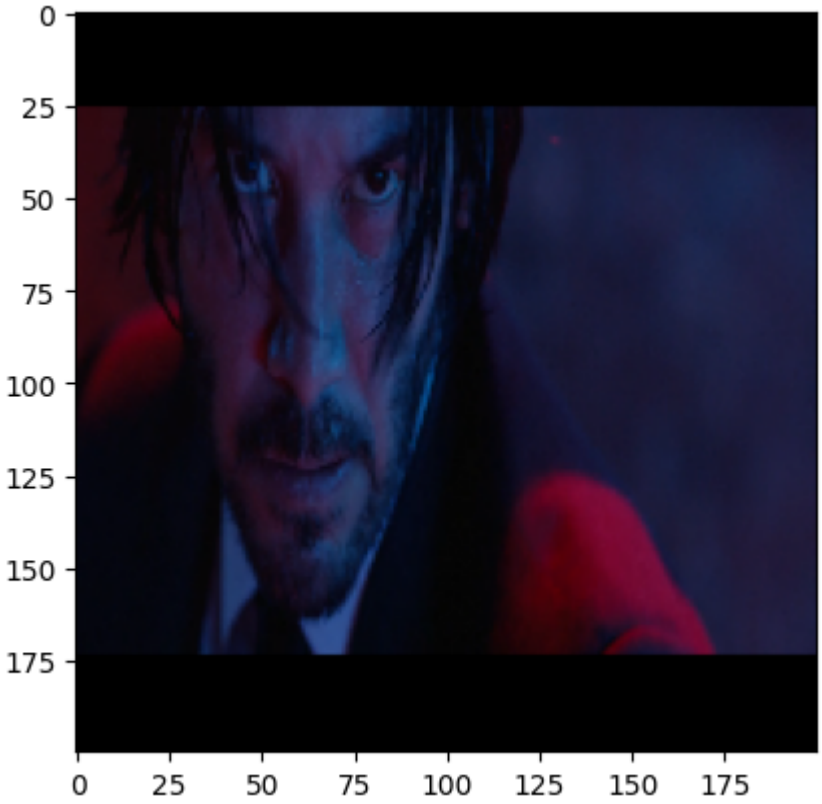
In [30]:
```
dir_path = r"C:\Users\gadel\VS Code projects\Emotion Detector\CNN\Testing"
for i in os.listdir(dir_path ):
    print(i)
    #img = image.load_img(dir_path+ '//'+i, target_size = (200,200))
   # plt.imshow(img)
   # plt.show()
```

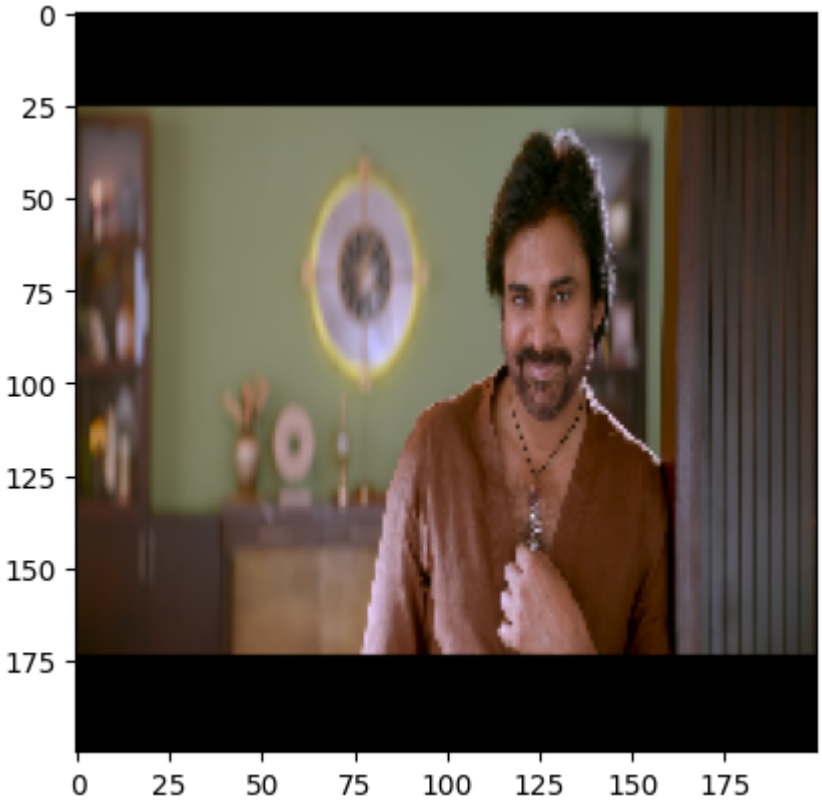
```
Screenshot (393).png
Screenshot (457).png
Screenshot (661).png
Screenshot (788).png
Screenshot (841).png
Screenshot (852).png
Screenshot (879).png
```

In [31]:
```
dir_path = r"C:\Users\gadel\VS Code projects\Emotion Detector\CNN\Testing"
for i in os.listdir(dir_path ):
    img = image.load_img(dir_path+ '//'+i, target_size = (200,200))
    plt.imshow(img)
    plt.show()
```

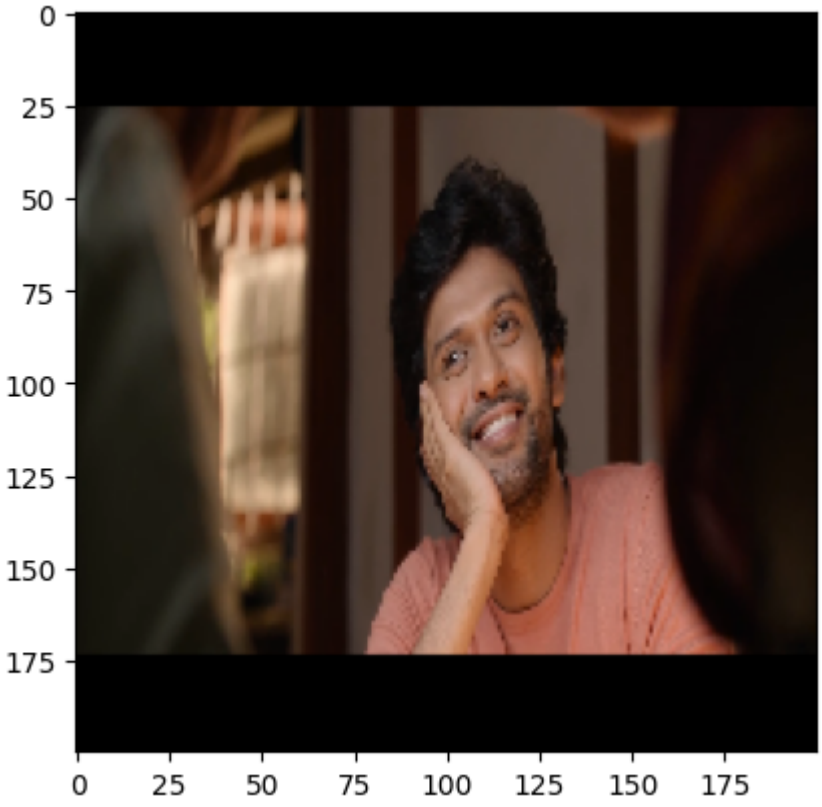

0
25
50
75
100
125
150
175
0
25
50
75
100
125
150
175

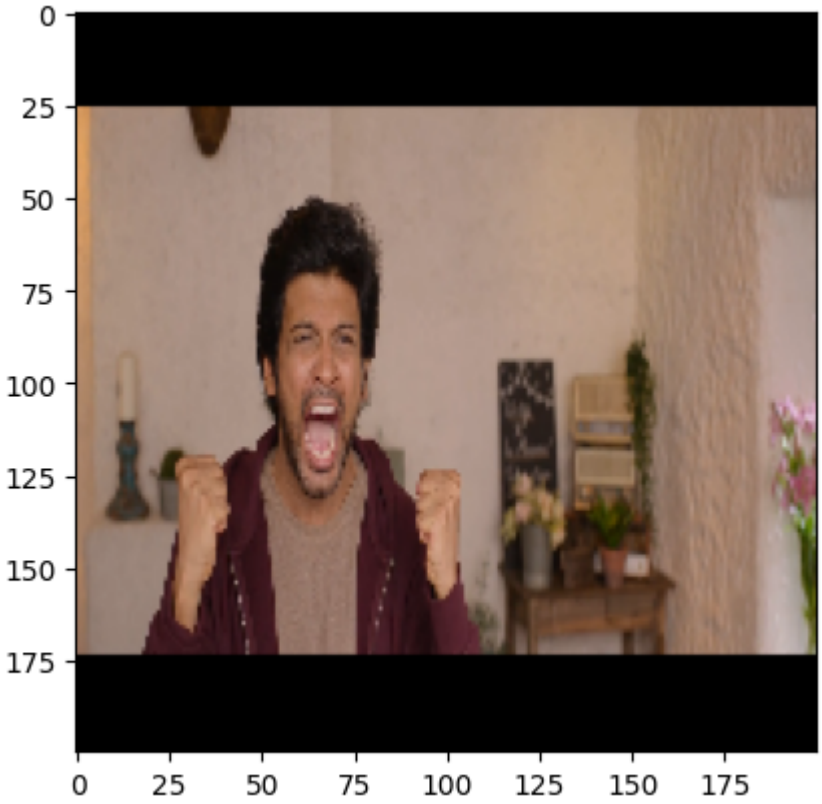

0
25
50
75
100
125
150
175
0
25
50
75
100
125
150
175


0
25
50
75
100
125
150
175
0
25
50
75
100
125
150
175

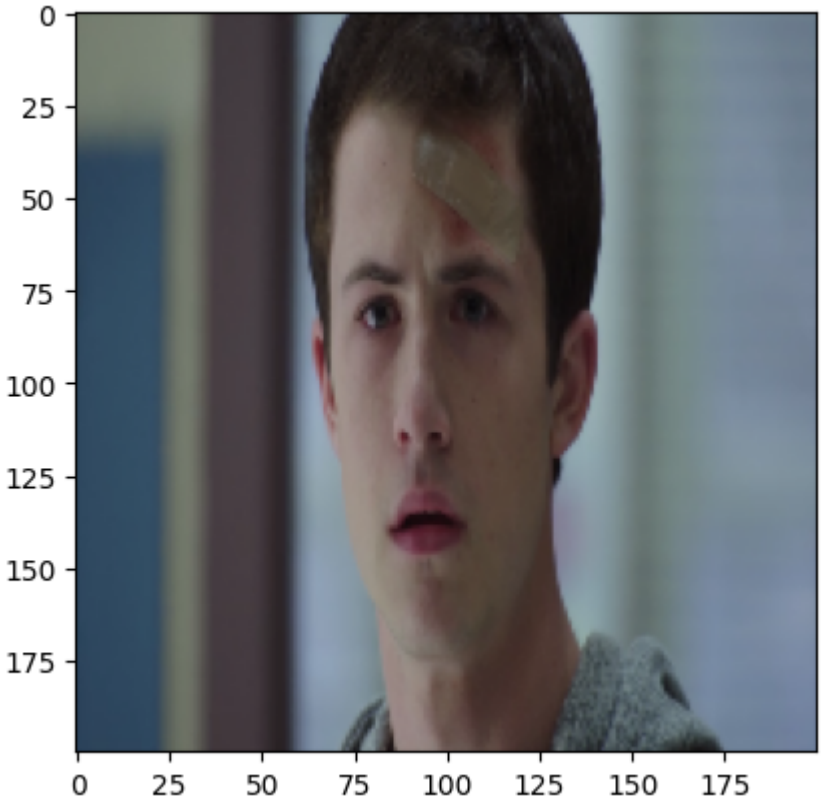

0
25
50
75
100
125
150
175
0
25
50
75
100
125
150
175


0
25
50
75
100
125
150
175
0
25
50
75
100
125
150
175

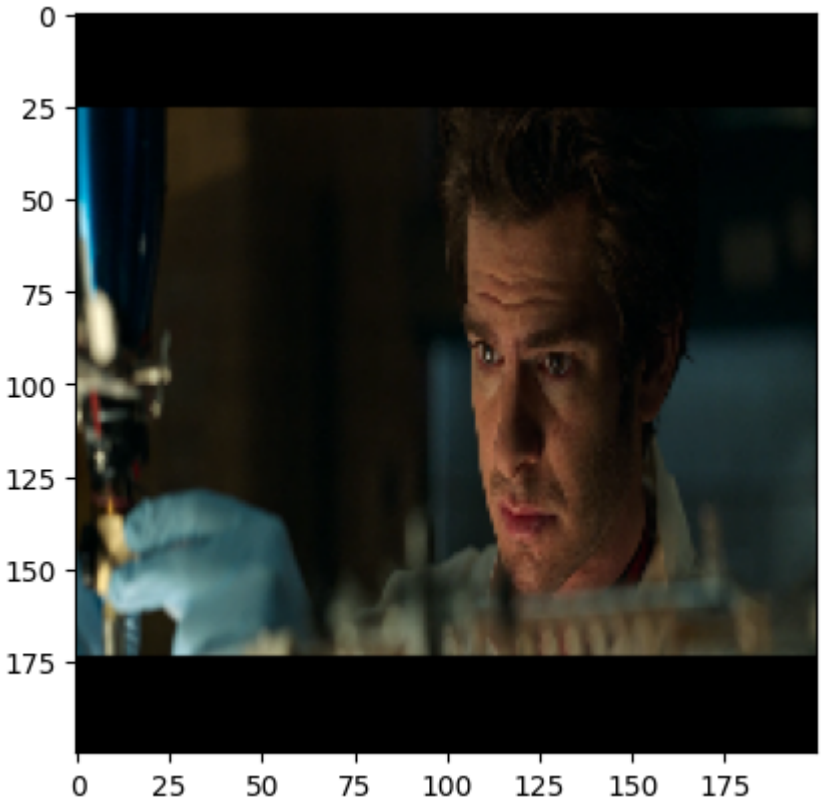

0
25
50
75
100
125
150
175
0
25
50
75
100
125
150
175



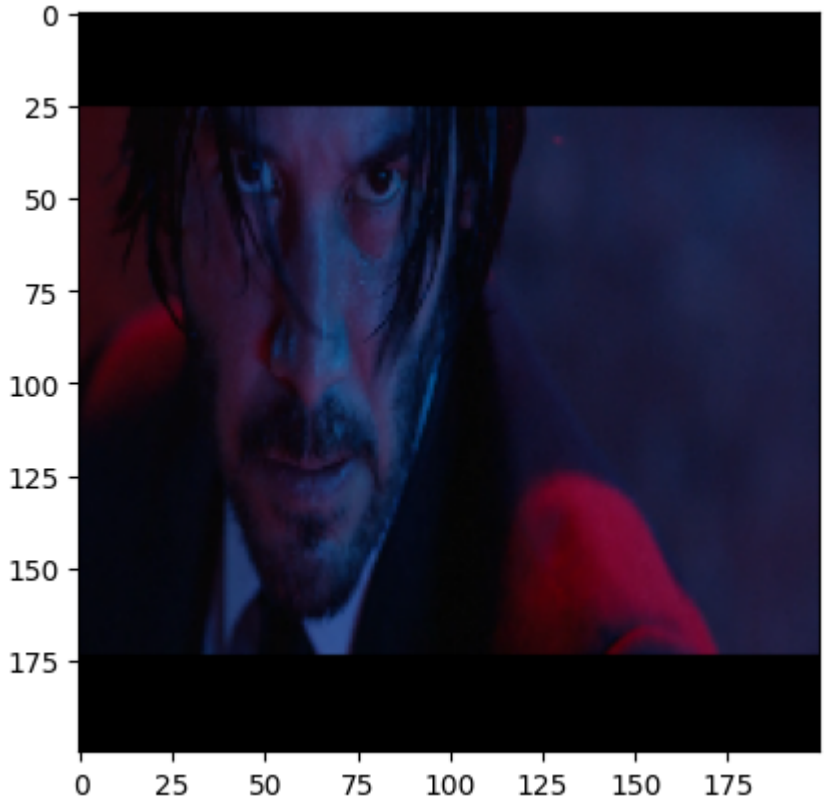
In [32]:
```
dir_path = r"C:\Users\gadel\VS Code projects\Emotion Detector\CNN\Testing"
for i in os.listdir(dir_path ):
    img = image.load_img(dir_path+ '//'+i, target_size = (200,200))
    plt.imshow(img)
    plt.show()

    x= image.img_to_array(img)
    x=np.expand_dims(x,axis = 0)
    images = np.vstack([x])

    val = model.predict(images)
    if val == 0:
        print( ' i am not happy')
    else:
        print('i am happy')
```



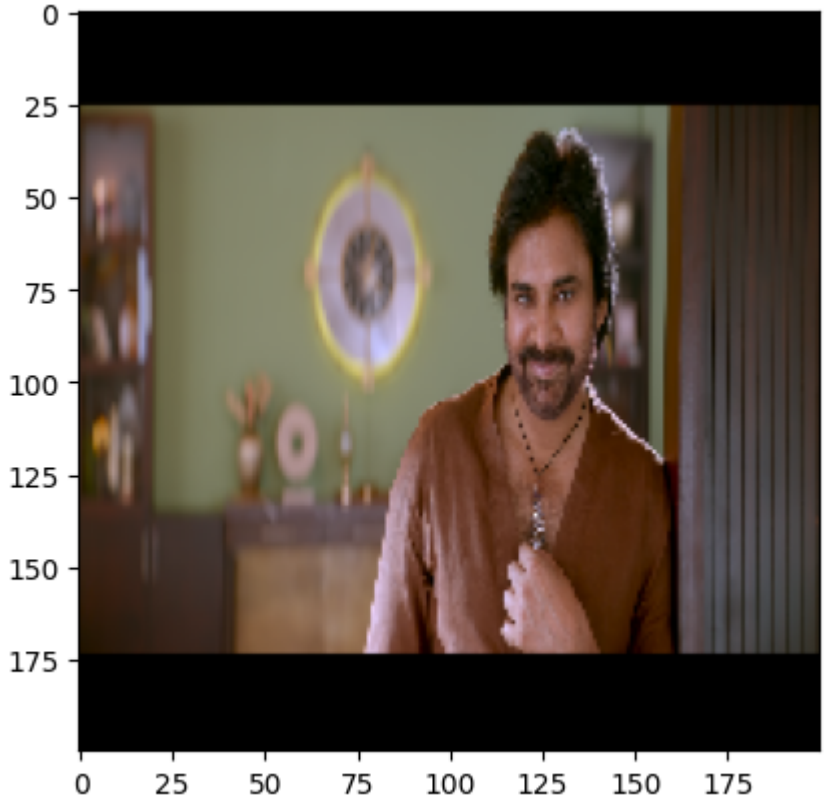

```
1/1 [==============================] - 0s 222ms/step
 i am not happy
```




```
1/1 [==============================] - 0s 61ms/step
 i am not happy
```

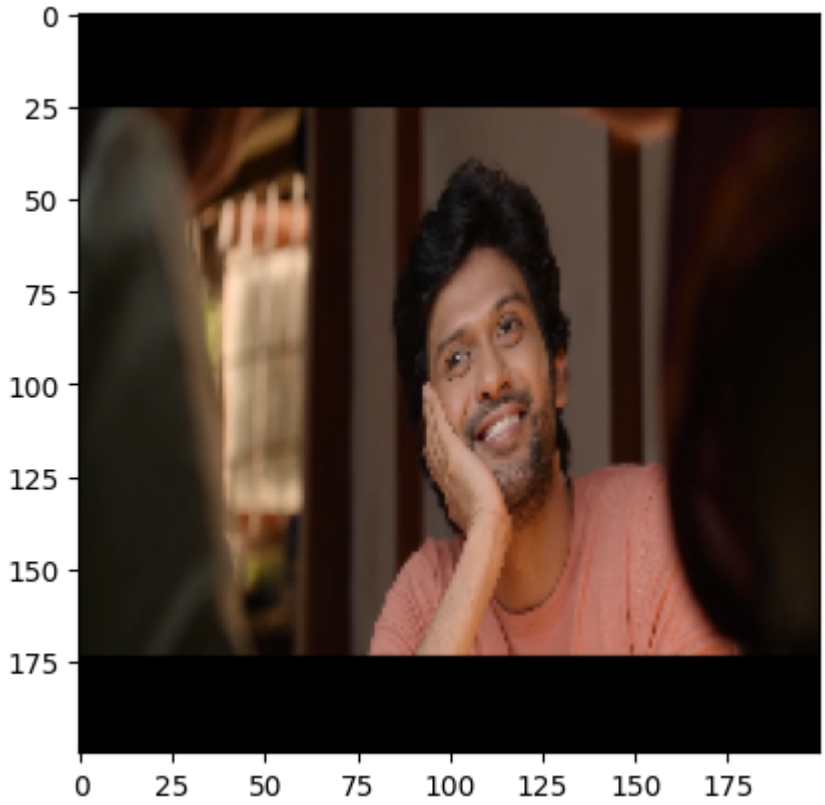



```
1/1 [==============================] - 0s 42ms/step
 i am not happy
```

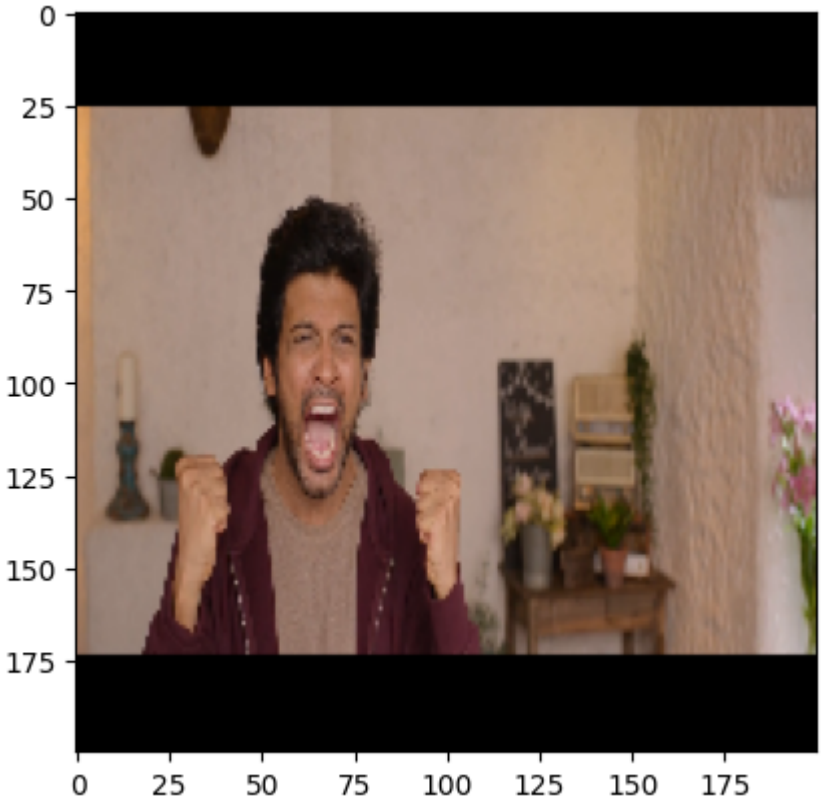



```
1/1 [==============================] - 0s 64ms/step
i am happy
```


```
1/1 [==============================] - 0s 75ms/step
i am happy
```




```
1/1 [==============================] - 0s 64ms/step
i am happy
```




```
1/1 [==============================] - 0s 152ms/step
 i am not happy
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: