In [1]:
```
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
```

In [2]:
```
img=image.load_img(r'D:\Data Science With AI Practise PDF\Training\happy\images
```

In [3]:
```
plt.imshow(img)
```

Out[3]:
```
<matplotlib.image.AxesImage at 0x1f427786870>
```

In [4]:
```
i1=cv2.imread(r'D:\Data Science With AI Practise PDF\Training\happy\download (1)
i1
```

```
Out[4]: array([[[255, 254, 244],
                [255, 254, 244],
                [253, 253, 247],
                ...,
                [209, 218, 228],
                [210, 219, 229],
                [210, 219, 229]],

               [[255, 254, 244],
                [253, 254, 245],
                [253, 253, 247],
                ...,
                [208, 217, 227],
                [209, 218, 228],
                [209, 218, 228]],

               [[253, 254, 245],
                [253, 254, 245],
                [253, 253, 247],
                ...,
                [206, 215, 225],
                [207, 216, 226],
                [208, 217, 227]],

               ...,

               [[  7,  18,  32],
                [  7,  18,  32],
                [  7,  18,  32],
                ...,
                [  7,  37,  48],
                [  6,  36,  47],
                [  6,  36,  47]],

               [[ 11,  22,  36],
                [ 11,  22,  36],
                [ 11,  22,  36],
                ...,
                [  5,  35,  46],
                [  4,  34,  45],
                [  3,  33,  44]],

               [[ 15,  26,  40],
                [ 15,  26,  40],
                [ 15,  26,  40],
                ...,
                [  2,  32,  43],
                [  1,  31,  42],
                [  0,  30,  41]]], dtype=uint8)
```

```
In [5]: i1.shape
```

```
Out[5]: (183, 276, 3)
```

```
In [6]: train=ImageDataGenerator(rescale=1/225)
        validation=ImageDataGenerator(rescale=1/225)
```

```
In [7]: train_dataset=train.flow_from_directory(r'D:\Data Science With AI Practise PDF\T
                                                target_size=(200,200),
                                                batch_size=3,
                                                class_mode='binary')

      validation_dataset=validation.flow_from_directory(r'D:\Data Science With AI Prac
                                                       target_size=(200,200),
                                                       batch_size=3,
                                                       class_mode='binary')
```

```
Found 11 images belonging to 2 classes.
Found 0 images belonging to 2 classes.
```

In [8]:
```
train_dataset.class_indices
```

Out[8]:
```
{'happy': 0, 'not happy': 1}
```

In [9]:
```
train_dataset.classes
```

Out[9]:
```
array([0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1])
```

In [10]:
```
model=tf.keras.models.Sequential([tf.keras.layers.Conv2D(16,(3,3),activation='re
                                  tf.keras.layers.MaxPool2D(2,2),

                                  tf.keras.layers.Conv2D(32,(3,3),activation='re
                                  tf.keras.layers.MaxPool2D(2,2),

                                  tf.keras.layers.Conv2D(64,(3,3),activation='re
                                  tf.keras.layers.MaxPool2D(2,2),

                                  tf.keras.layers.Flatten(),

                                  tf.keras.layers.Dense(512, activation='relu'),

                                  tf.keras.layers.Dense(1,activation='sigmoid')
                                  ])
```

```
c:\Users\DELL\AppData\Local\Programs\Python\Python312\Lib\site-packages\keras\src
\layers\convolutional\base_conv.py:99: UserWarning: Do not pass an `input_shape`/
`input_dim` argument to a layer. When using Sequential models, prefer using an `I
nput(shape)` object as the first layer in the model instead.
  super().__init__(
```

In [11]:
```
model.compile(loss='binary_crossentropy',
              optimizer=tf.keras.optimizers.RMSprop(learning_rate=0.001),
              metrics=['accuracy']
              )
```

In [12]:
```
model_fit=model.fit(train_dataset,epochs=15)
```

```
Epoch 1/15
```

```
c:\Users\DELL\AppData\Local\Programs\Python\Python312\Lib\site-packages\keras\src
\trainers\data_adapters\py_dataset_adapter.py:122: UserWarning: Your `PyDataset`
class should call `super().__init__(**kwargs)` in its constructor. `**kwargs` can
include `workers`, `use_multiprocessing`, `max_queue_size`. Do not pass these arg
uments to `fit()`, as they will be ignored.
  self._warn_if_super_not_called()
```

```
4/4 ━━━━━━━━━━━━━━━━━━━━ 4s 290ms/step - accuracy: 0.6545 - loss: 3.9715
Epoch 2/15
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 258ms/step - accuracy: 0.4040 - loss: 1.0005
Epoch 3/15
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 283ms/step - accuracy: 0.5990 - loss: 0.6059
Epoch 4/15
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 297ms/step - accuracy: 0.8326 - loss: 0.7088
Epoch 5/15
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 255ms/step - accuracy: 0.6232 - loss: 0.7604
Epoch 6/15
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 248ms/step - accuracy: 0.8409 - loss: 0.6658
Epoch 7/15
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 282ms/step - accuracy: 0.8320 - loss: 0.4619
Epoch 8/15
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 250ms/step - accuracy: 1.0000 - loss: 0.2376
Epoch 9/15
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 261ms/step - accuracy: 0.6973 - loss: 0.5700
Epoch 10/15
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 258ms/step - accuracy: 1.0000 - loss: 0.1273
Epoch 11/15
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 278ms/step - accuracy: 1.0000 - loss: 0.0503
Epoch 12/15
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 288ms/step - accuracy: 1.0000 - loss: 0.0139
Epoch 13/15
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 261ms/step - accuracy: 1.0000 - loss: 0.0130
Epoch 14/15
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 255ms/step - accuracy: 1.0000 - loss: 0.0032
Epoch 15/15
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 281ms/step - accuracy: 1.0000 - loss: 0.0018
```

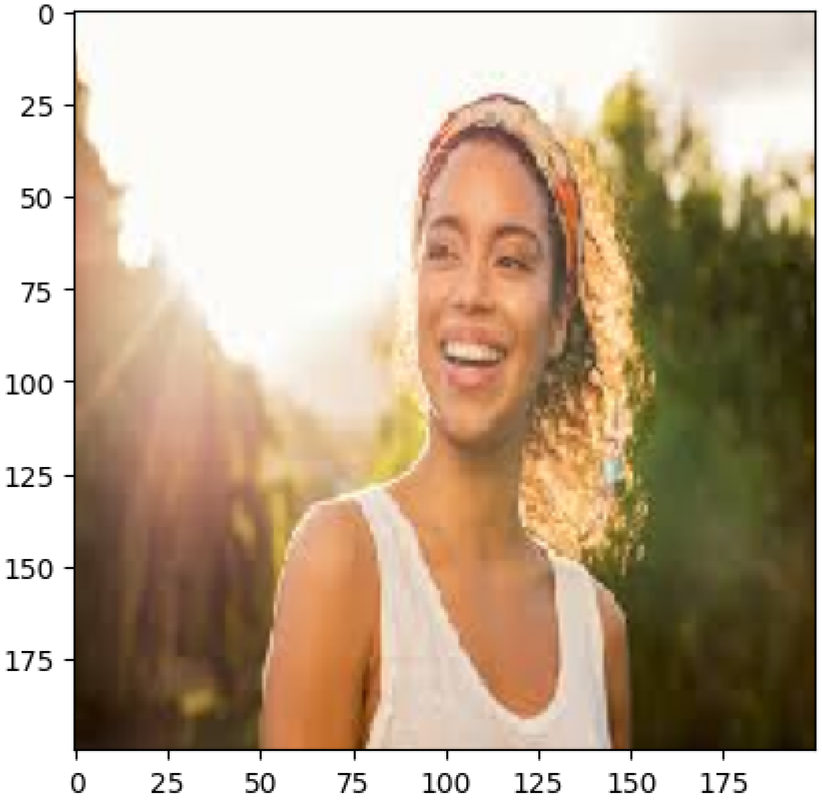
In [15]:

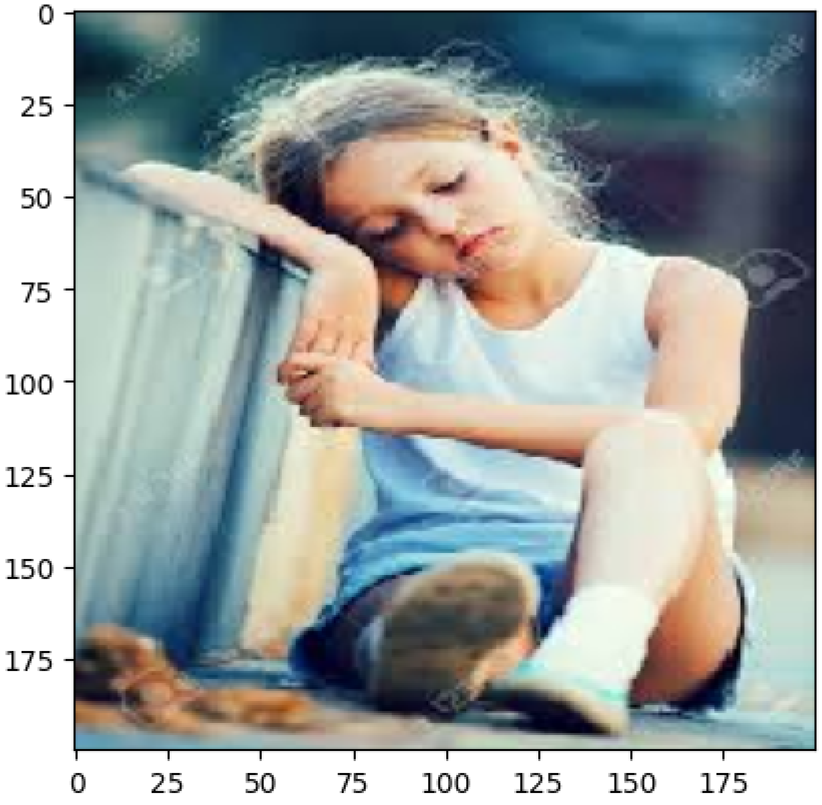
```python
dir_path=r'D:\Data Science With AI Practise PDF\Testing'
for i in os.listdir(dir_path):
    print(i)
```

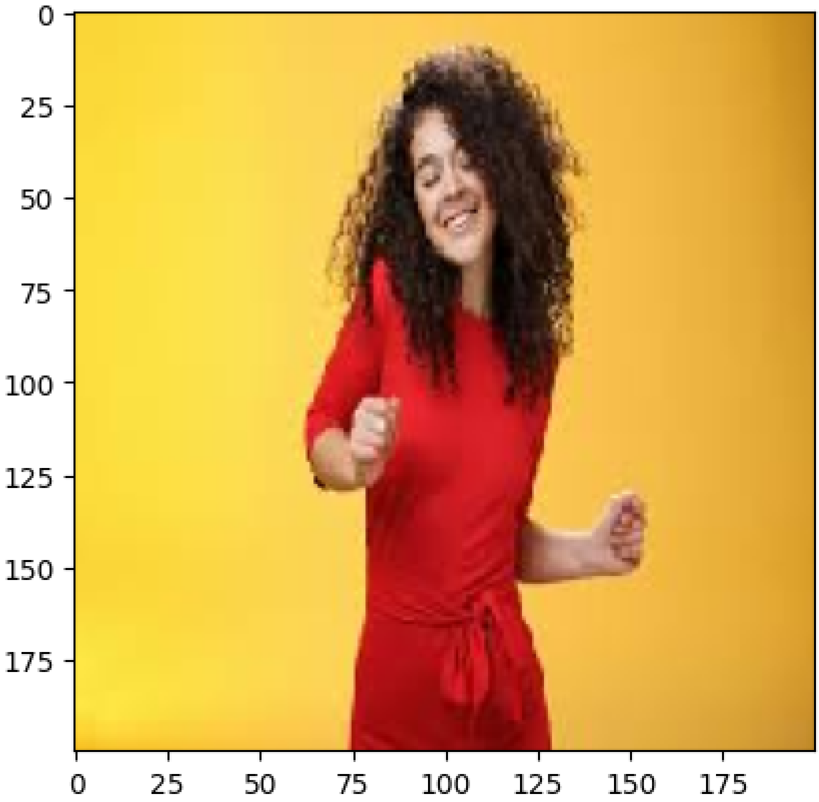
```
download (1).jpeg
download (2).jpeg
download.jpeg
images (1).jpeg
images (2).jpeg
images (3).jpeg
images.jpeg
istockphoto-1494508936-612x612.jpg
```

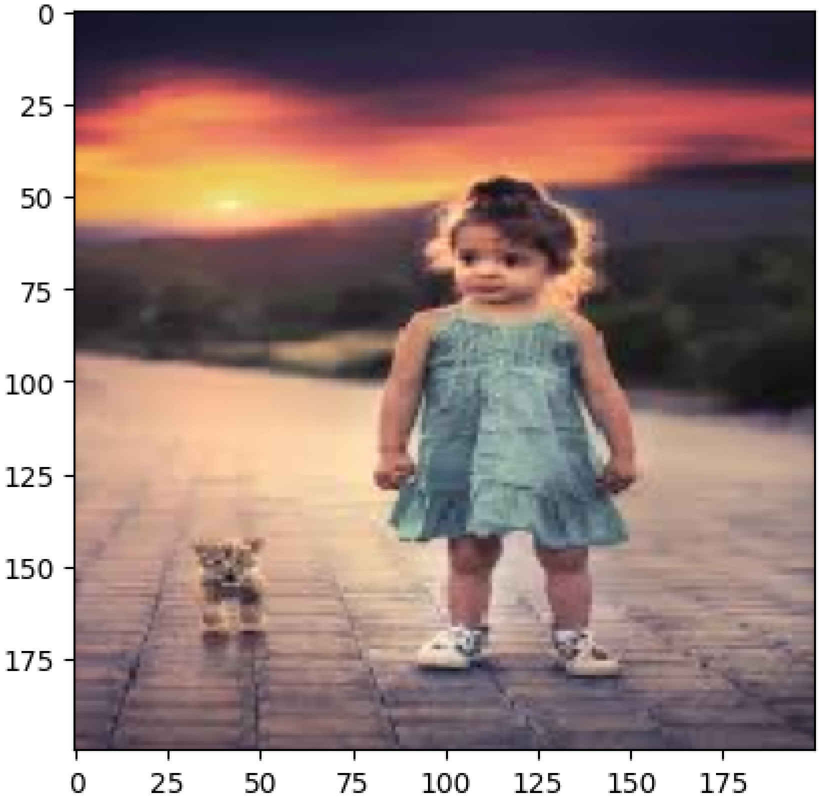
In [16]:

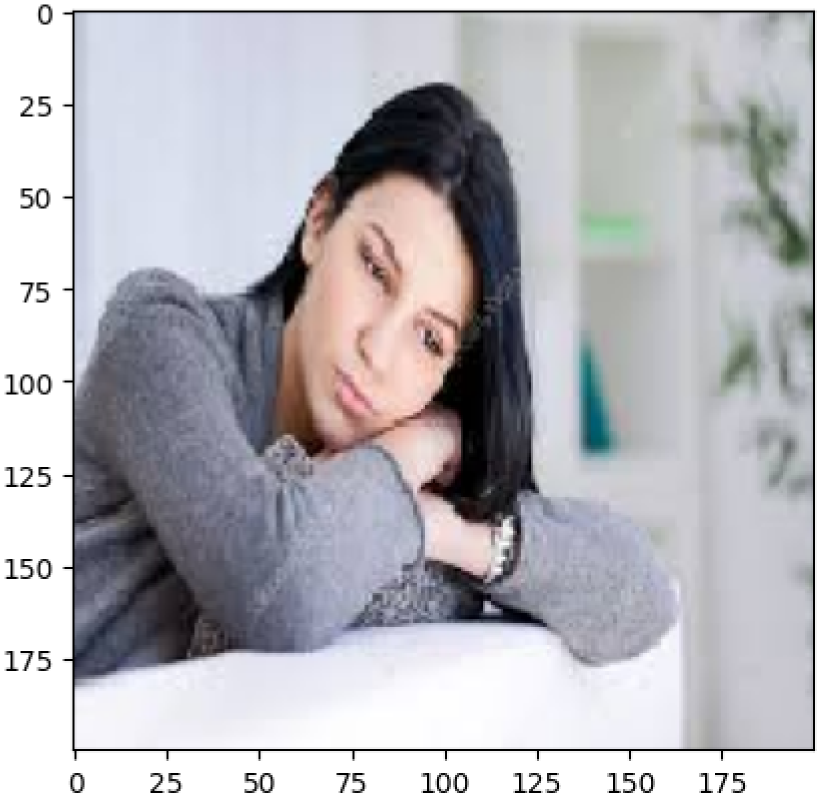
```python
dir_path=r'D:\Data Science With AI Practise PDF\Testing'
for i in os.listdir(dir_path):
    img=image.load_img(dir_path+'//'+i,target_size=(200,200))
    plt.imshow(img)
    plt.show()
```

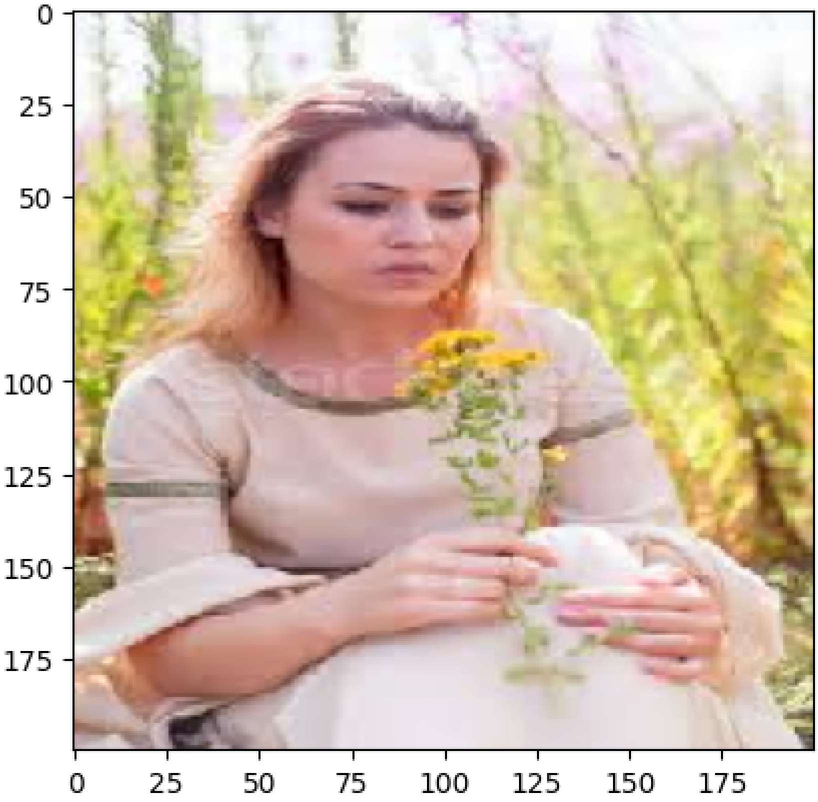

0
25
50
75
100
125
150
175
0
25
50
75
100
125
150
175

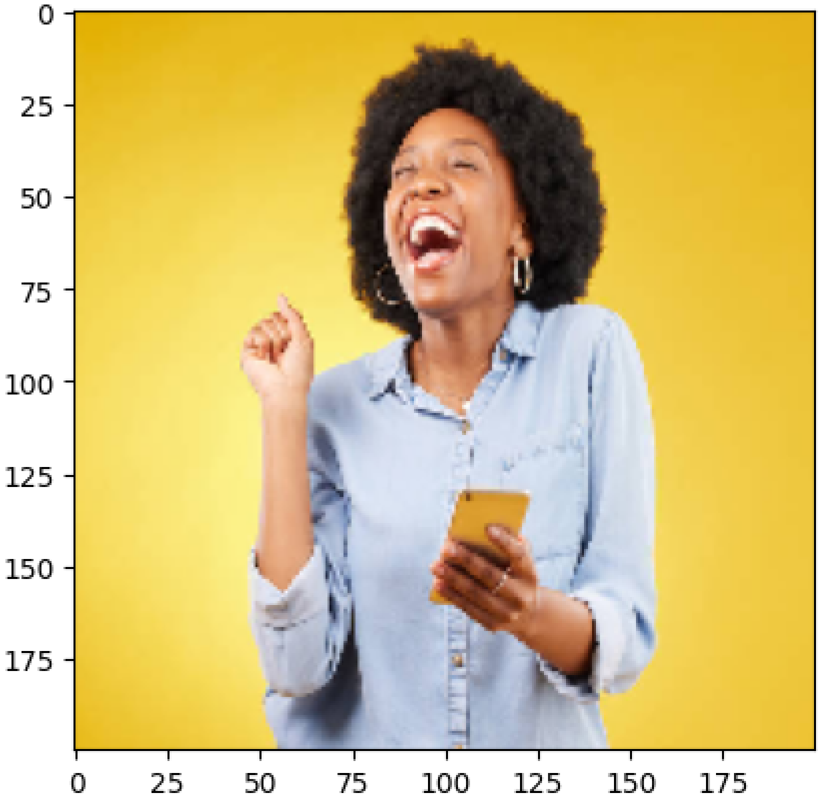

0
25
50
75
100
125
150
175
0
25
50
75
100
125
150
175


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

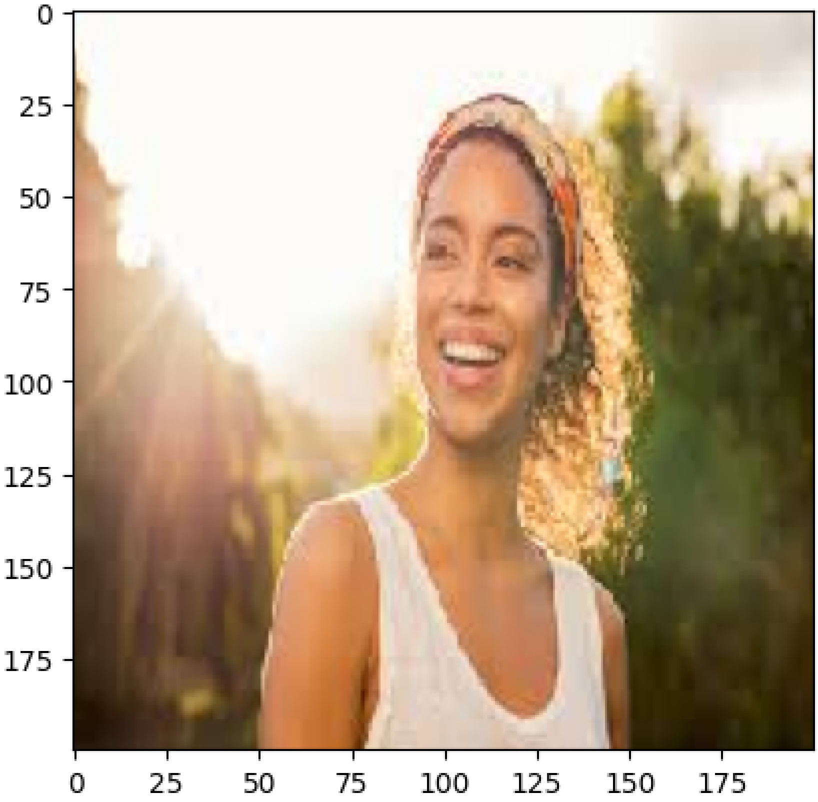

0
25
50
75
100
125
150
175
0
25
50
75
100
125
150
175


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

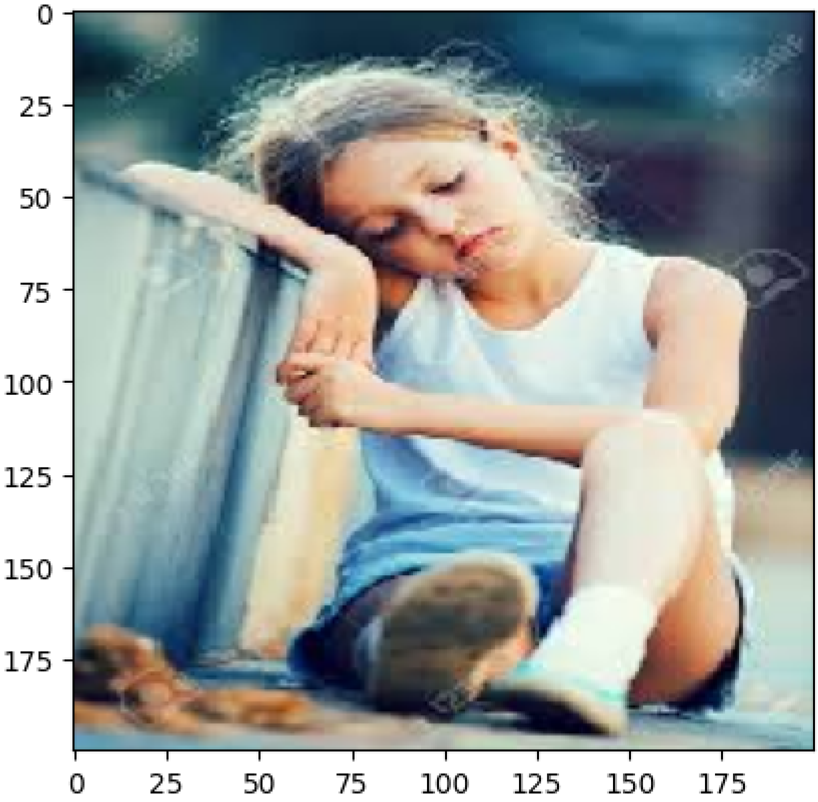


In [17]:

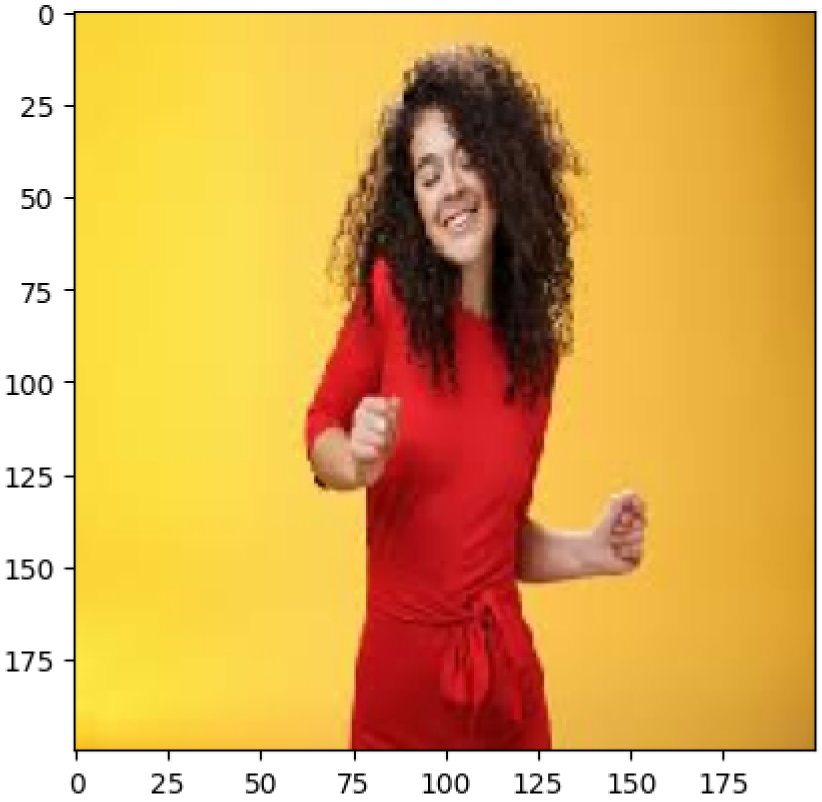
```
dir_path=r'D:\Data Science With AI Practise PDF\Testing'
for i in os.listdir(dir_path):
    img=image.load_img(dir_path+'//'+i,target_size=(200,200))
    plt.imshow(img)
    plt.show()


    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('iam not happy')
    else:
        print(' iam happy')
```



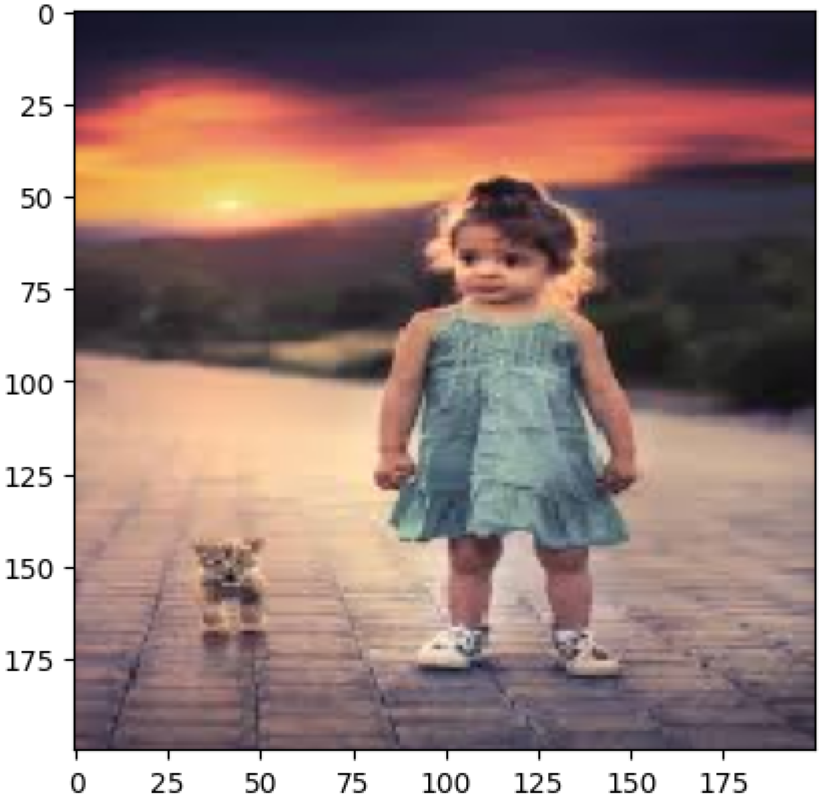
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 237ms/step
iam not happy
```



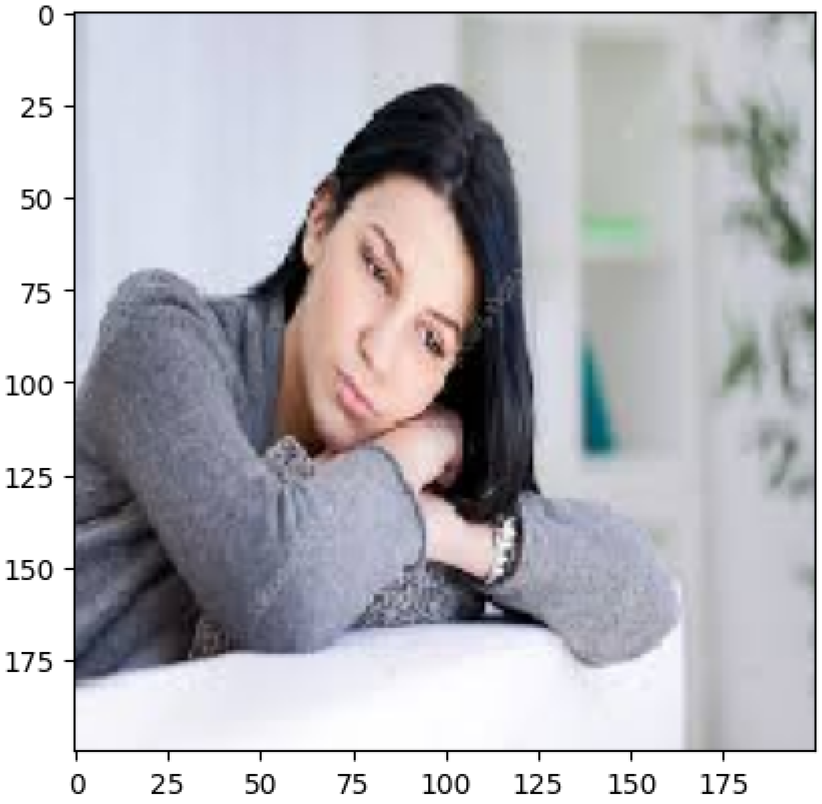
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 56ms/step
 iam happy
```



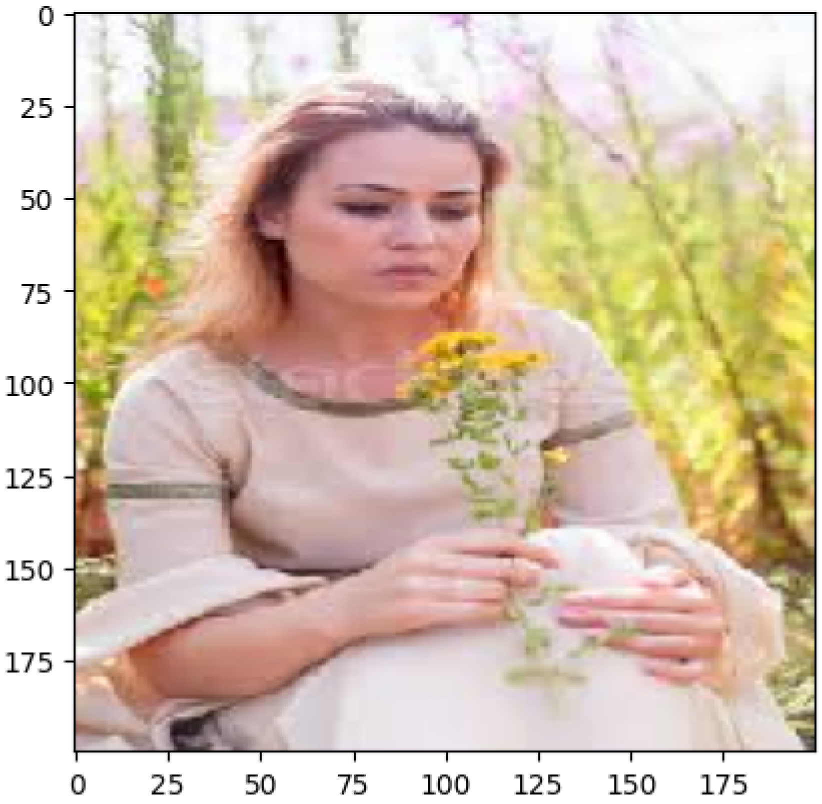
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 57ms/step
iam not happy
```



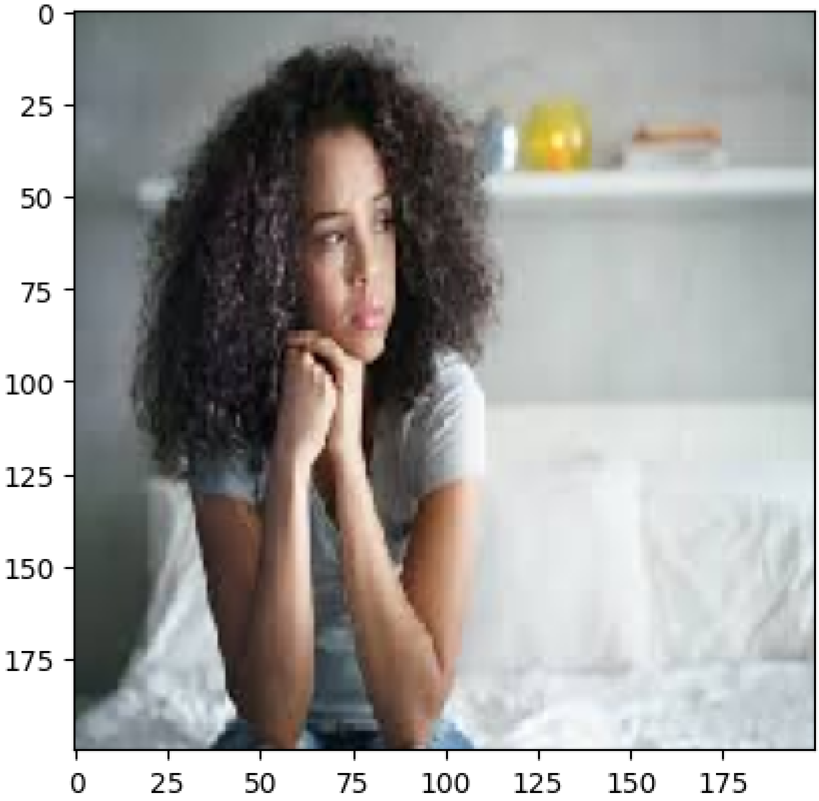
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 32ms/step
 iam happy
```



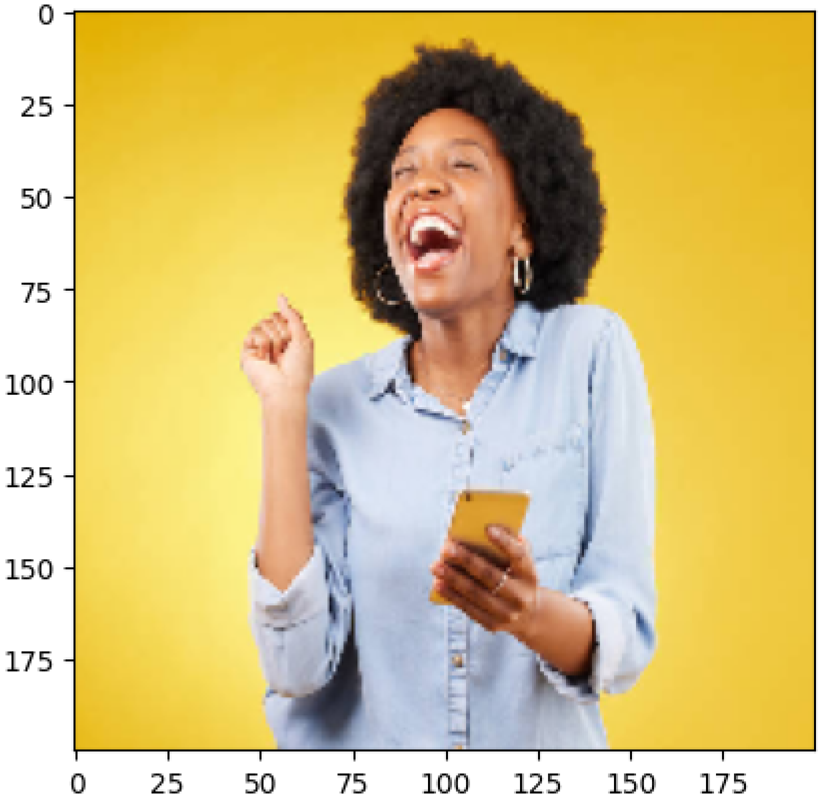
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 82ms/step
 iam happy
```



```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 111ms/step
 iam happy
```



```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 61ms/step
 iam happy
```



```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 71ms/step
iam not happy
```

In [ ]: