This task involves a simple image preprocessing step on an RGB image. First, the image is imported from a web address using the `!wget` command followed by the link to the image. The selected image captures a memorable moment: Ben Stokes celebrating his match-winning century in a test match, achieved with the last wicket remaining.

in this task i have tried implementing basic uses of image preprocessing on one single image using

Librariesfor Preprocessing:

1.matplotlib.image: For reading and displaying images.

2.Pillow: To perform basic image manipulations like resizing.

3.OpenCV: For advanced image processing tasks like converting to grayscale.

TO GET AND IMAGE USING '!wget' FROM GOOGLE ADDRESS

In [30]:

```
!wget 'https://c.ndtvimg.com/2020-08/0g5s6oug_ben-stokes-twitter_625x300_25_August_20.jpg'
```

```
--2024-12-21 18:41:15--  https://c.ndtvimg.com/2020-08/0g5s6oug_ben-stokes-twitter_625x300_25_August_20.jpg
Resolving c.ndtvimg.com (c.ndtvimg.com)... 23.197.17.134, 2600:1407:3c00:1587::24e8, 2600:1407:3c00:1582::24e8
Connecting to c.ndtvimg.com (c.ndtvimg.com)|23.197.17.134|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 70544 (69K) [image/jpeg]
Saving to: ‘0g5s6oug_ben-stokes-twitter_625x300_25_August_20.jpg’

0g5s6oug_ben-stokes 100%[===================>]  68.89K  --.-KB/s    in 0.02s

2024-12-21 18:41:16 (2.87 MB/s) - ‘0g5s6oug_ben-stokes-twitter_625x300_25_August_20.jpg’ saved [70544/70544]
```

IMPORTING NECESSARY MODULES

In [31]:

```
import matplotlib.image as mping
import matplotlib.pyplot as plt
```

LOAD THE IMAGE USING IMAGE MODULE

In [32]:

```
img = mping.imread('/content/benstokes.jpg')
```

In [33]:

```
type(img)
```

Out[33]:

```
numpy.ndarray
```

In [34]:

```
img.shape
```

Out[34]:

```
(605, 806, 3)
```

CONVERTING JPG IMAGE INTO NUMPY ARRAY

In [35]:

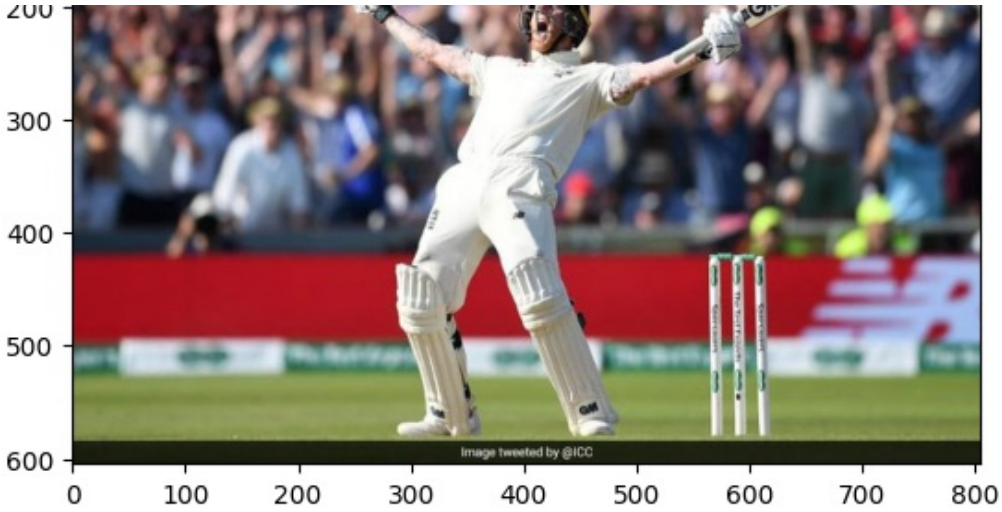
```
print(img)
```

```
[[[ 4  7 26]
  [ 4  7 26]
  [ 4  7 26]
  ...
  [ 1  2  6]
  [ 1  2  6]
  [ 1  2  6]]

 [[ 4  7 26]
  [ 4  7 26]
  [ 4  7 26]
  ...
  [ 1  2  6]
  [ 1  2  6]
  [ 1  2  6]]

 [[ 3  6 25]
  [ 3  6 25]
  [ 3  6 25]
  ...
  [ 1  2  6]
  [ 1  2  6]
  [ 1  2  6]]

 ...

 [[25 29 14]
  [25 29 14]
  [25 29 14]
  ...
  [29 33 19]
  [29 33 19]
  [29 33 19]]

 [[25 29 14]
  [25 29 14]
  [25 29 14]
  ...
  [29 33 19]
  [29 33 19]
  [29 33 19]]

 [[25 29 14]
  [25 29 14]
  [25 29 14]
  ...
  [29 32 21]
  [29 32 21]
  [29 32 21]]]
```

NOW DISPLAYING NUMPY ARRAY INTO JPG IMAGE

In [36]:

```
img_plot = plt.imshow(img)
```



USING 'pillow' LIBRARY : TO RESIZING THE IMAGE

In [37]:

```
from PIL import Image
```

In [38]:

```
img = Image.open('/content/benstokes.jpg')
```

In [39]:

```
img_resize = img.resize((400,400))
```

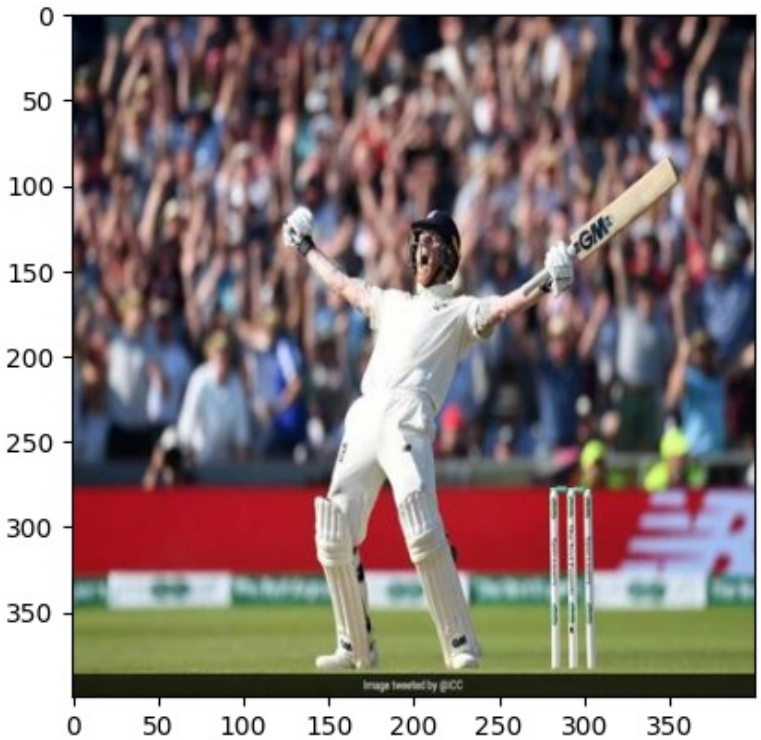
In [40]:

```
img_resize.save('benstokes_resized.jpg')
```

READING THE RESIZED IMAGE

In [41]:

```
img = mping.imread('/content/benstokes_resized.jpg')
resized_plot = plt.imshow(img)
plt.show()
```



In [42]:

```
img.shape
```

Out[42]:

```
(400, 400, 3)
```

USING 'opencv' : Now Converting RGB images to Grayscale image

IMPORTING THE LIBRARY

In [43]:

```
import cv2
```

In [44]:

```
img = cv2.imread('/content/benstokes.jpg')
```

In [45]:

```
grayscale_image = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
```

In [46]:

```
type(grayscale_image)
```

Out[46]:

```
numpy.ndarray
```

In [47]:

```
grayscale_image.shape
```

Out[47]:

```
(605, 806)
```

In [48]:

```
from google.colab.patches import cv2_imshow
```

cv2.imshow() will display the image. But this will not be allowed in Google Colab.

In [49]:

```
from google.colab.patches import cv2_imshow
```

SHOWING IMAGE

In [50]:

```
cv2_imshow(grayscale_image)
```

TO SAVE THE IMAGE

In [51]:

```
cv2.imwrite('dog_grayscale_image.jpg', grayscale_image)
```

Out[51]:

```
True
```

In this task, we looked at basic image preprocessing using three famous libraries: matplotlib.image, Pillow, and OpenCV. We started by importing an image from a web address and then showed how to load, display, resize, and convert an image into grayscale. Each of these steps was used to show the strengths and specific applications of the libraries used.

This not only demonstrates basic preprocessing steps but also gives an idea of how an image can be manipulated to get it ready for further analysis or application in computer vision. Such techniques are considered essential for preparing data on object detection, image classification, or feature extraction.