AI Lab-4

Data reading, analysis and visualization

Name: Ghulam Sarwar

CMS: 033-16-0017

Exercise

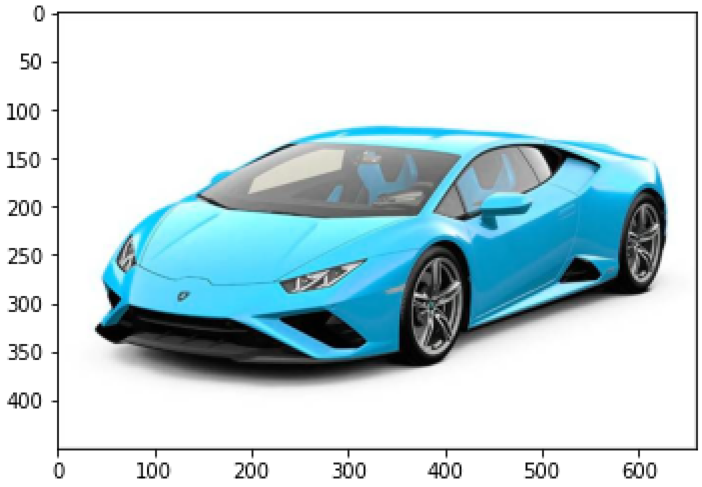
Take an image of a “Car” and a “Cup”. Perform following tasks:

- Read both images
- Resize image to 256 by 256
- Show both images
- Convert images to gray scale
- Normalize both images
- Show grayscale images
- Find contrast, energy and mean of both images
- Plot contrast, energy and mean of both images
- Discuss which feature among contrast, energy and is best for classification
- Perform the edge detection on both images and show its resultant image

```
In [38]:
```

```
#01.Read images
import cv2
import matplotlib.pyplot as plt
car = cv2.imread('../input/car-and-cup/car.jpg')
cup = cv2.imread('../input/car-and-cup/cup.jpg')
print(plt.imshow(car))
```

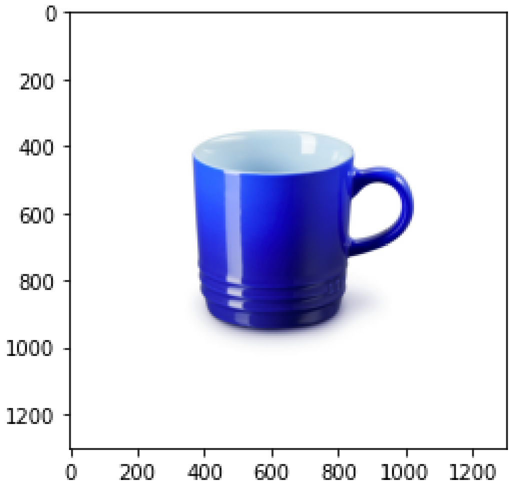
```
AxesImage(54,36;334.8x217.44)
```



In [39]:

```
print(plt.imshow(cup))
```

```
AxesImage(54,36;334.8x217.44)
```



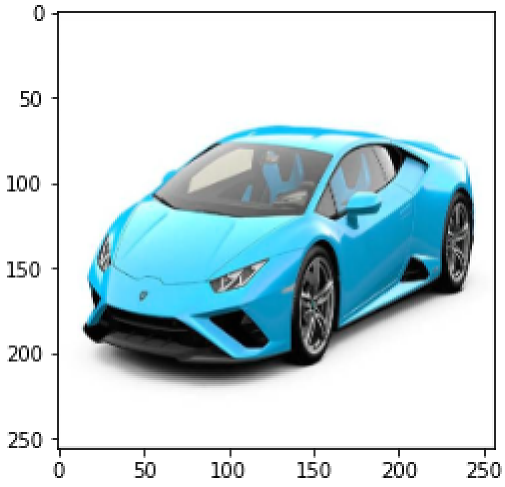
In [40]:

```
# Resizing car image
car = cv2.resize(car,(256,256))
print(car.shape)

print('Image 1')
plt.imshow(car)
```

```
(256, 256, 3)
Image 1
```

Out[40]:

```
<matplotlib.image.AxesImage at 0x7f01616adf10>
```



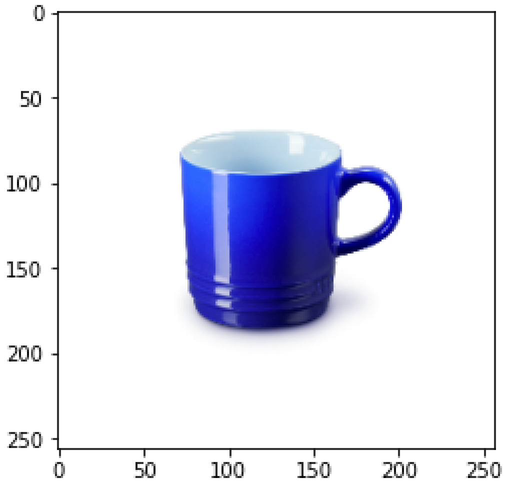
In [41]:

```
#resizing cup image
cup = cv2.resize(cup,(256,256))
print(cup.shape)

print('Image 1')
plt.imshow(cup)
```

```
(256, 256, 3)
Image 1
```

Out[41]:

```
<matplotlib.image.AxesImage at 0x7f016162a950>
```

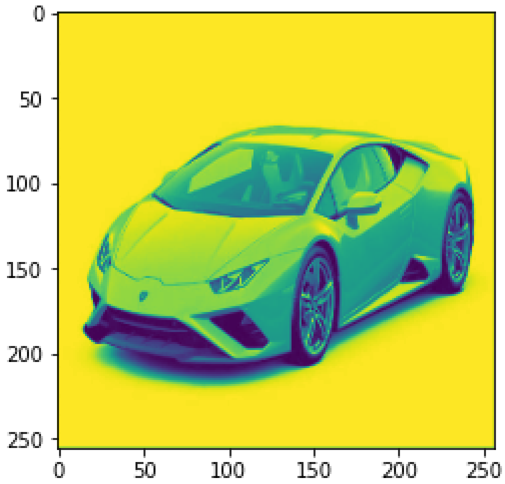


In [42]:

```
#Converting car image to gray scale
gray_car = cv2.cvtColor(car,cv2.COLOR_BGR2GRAY)
plt.imshow(gray_car)
```

Out[42]:

```
<matplotlib.image.AxesImage at 0x7f01615ab450>
```

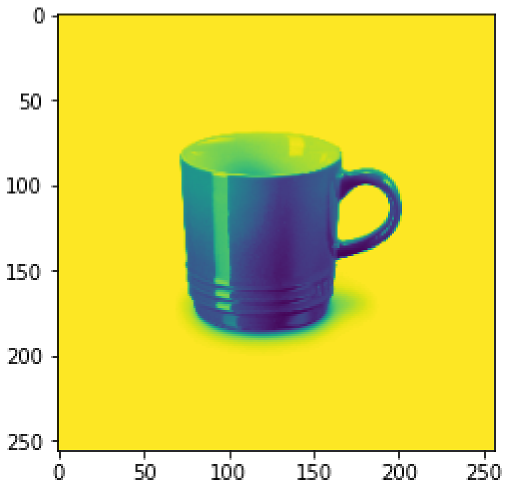


In [43]:

```
#Converting cup image to gray scale
gray_cup = cv2.cvtColor(cup,cv2.COLOR_BGR2GRAY)

plt.imshow(gray_cup)
```

Out[43]:

```
<matplotlib.image.AxesImage at 0x7f0161526b50>
```

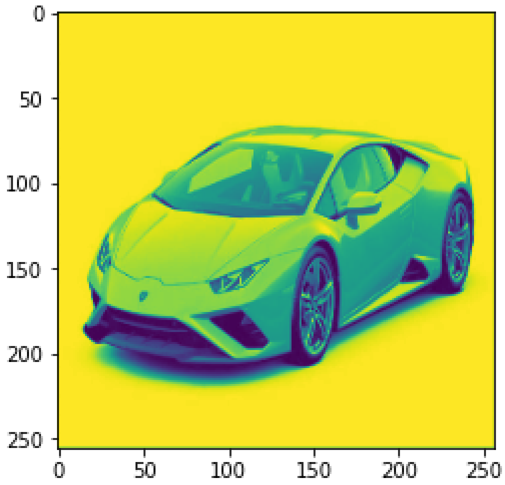


In [44]:

```
#Normalizing car image
normalize = np.zeros((250,250))
norm_car = cv2.normalize(gray_car,normalize,0,255,cv2.NORM_MINMAX)
plt.imshow(norm_car)
```

Out[44]:

```
<matplotlib.image.AxesImage at 0x7f01614a5e10>
```

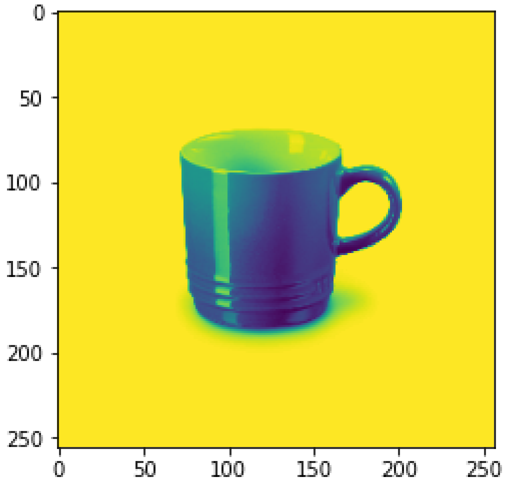


In [45]:

```
#Normalizing cup image
normalize = np.zeros((250,250))
norm_cup = cv2.normalize(gray_cup,normalize,0,255,cv2.NORM_MINMAX)
plt.imshow(norm_cup)
```

Out[45]:

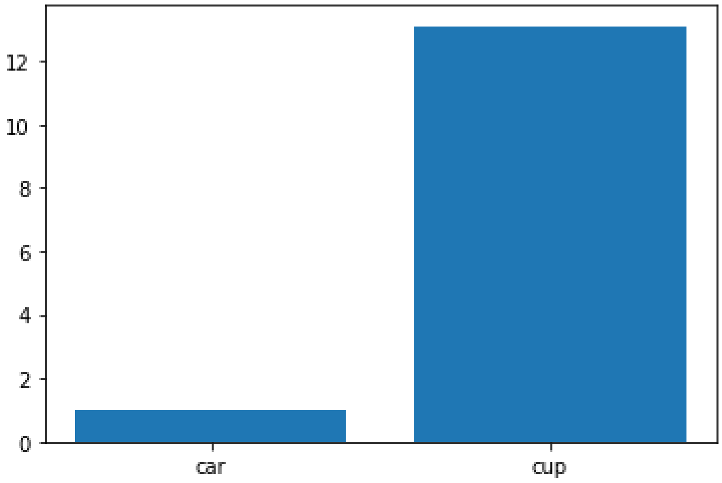
```
<matplotlib.image.AxesImage at 0x7f016141dbd0>
```



In [46]:

```
# Calculating contrast, energy and mean of both images
# for the applying some techniques to find

min_car = np.min(gray_car)
min_cup = np.min(gray_cup)

max_car = np.max(gray_car)
max_cup = np.max(gray_cup)

# formula for the contrast is as following

cont_car = (max_car-min_car)/(max_car+min_car)
cont_cup = (max_cup-min_cup)/(max_cup+min_cup)

print('Contrast of car is : ',cont_car)
print('Contrast of cup is : ',cont_cup)
print(plt.bar(['car','cup'],[cont_car,cont_cup]))
```

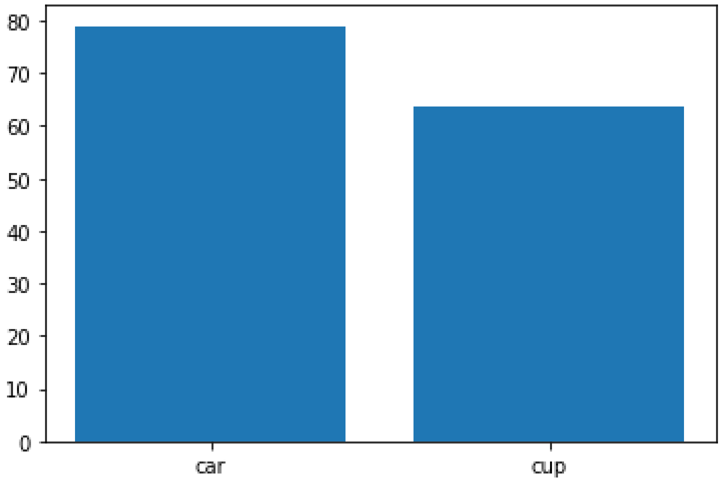
```
Contrast of car is :  1.0
Contrast of cup is :  13.111111
<BarContainer object of 2 artists>


/opt/conda/lib/python3.7/site-packages/ipykernel_launcher.py:13: Ru
ntimeWarning: overflow encountered in ubyte_scalars
  del sys.path[0]
```



In [47]:

```
#calculating standrad deviation of the image for energy
car_std = np.std(car)
cup_std = np.std(cup)

print(car_std,cup_std)
print(print(plt.bar(['car','cup'],[car_std,cup_std])))
```

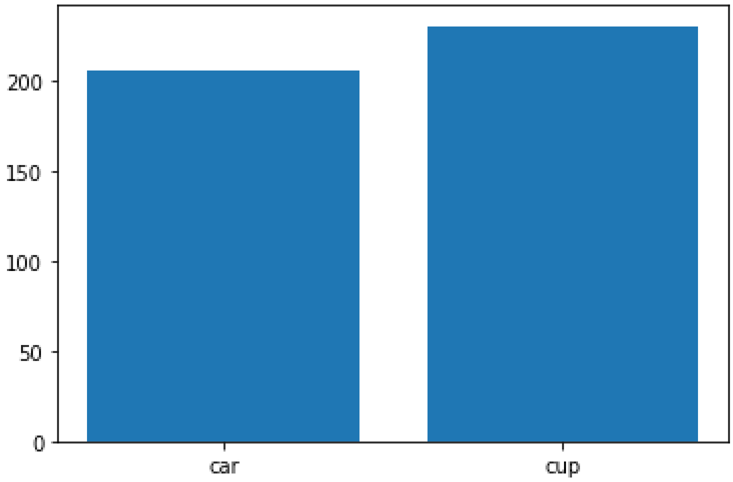
```
78.98270651860223 63.57742619056855
<BarContainer object of 2 artists>
None
```



In [48]:

```
#Calculating mean of both images

mean1= car.mean()
mean2 = cup.mean()

print(mean1,mean2)
print(print(plt.bar(['car','cup'],[mean1,mean2])))
```

```
205.40687561035156 230.45027669270834
<BarContainer object of 2 artists>
None
```



In [49]:

```
energy_dif = (car_std) - (cup_std)
cont_dif = (cont_car) - (cont_cup)
mean_dif = (mean1) - (mean2)

print(energy_dif)
print(cont_dif)
print(mean_dif)
```

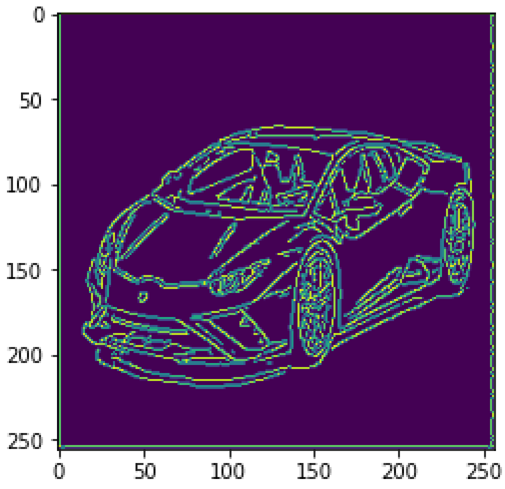
```
15.405280328033676
-12.111111
-25.04340108235678
```

In [50]:

```
#Edge detectionn in python
car_edg = cv2.Canny(gray_car, threshold1=30, threshold2=100)
plt.imshow(car_edg)
```

Out[50]:

```
<matplotlib.image.AxesImage at 0x7f0161271dd0>
```

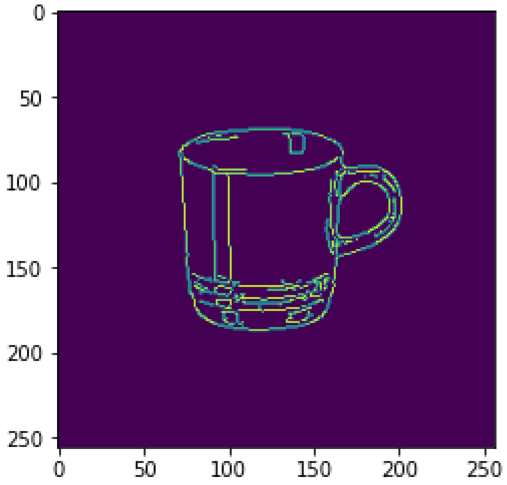


In [51]:

```
cup_edg = cv2.Canny(gray_cup, threshold1=30, threshold2=100)
plt.imshow(cup_edg)
```

Out[51]:

```
<matplotlib.image.AxesImage at 0x7f01611d3f50>
```


0
50
100
150
200
250
0
50
100
150
200
250