```
import cv2

import matplotlib.pyplot as plt

config_file='/content/ssd_mobilenet_v3_large_coco_2020_01_14 (1).pbtxt'
frozen_model='/content/frozen_inference_graph (1).pb'

print(config_file)

model=cv2.dnn_DetectionModel(frozen_model,config_file)

classLabels=[]
file_name='/content/coco.names'
with open(file_name,'rt') as fpt:
  classLabels=fpt.read().rstrip('\n').split('\n')

print(classLabels)

print(len(classLabels))

model.setInputSize(320,320)
model.setInputScale(1.0/127.5)
model.setInputMean((127.5,127.5,127.5))
model.setInputSwapRB(True)
```

```
#reading image
image=cv2.imread('/content/image.jpg')
plt.imshow(image) ##bgr image we nedd to convert into rgb color

plt.imshow(cv2.cvtColor(image,cv2.COLOR_BGR2RGB))

ClassIndex,confidece,bbox=model.detect(image,confThreshold=0.3)

print(ClassIndex)

print(ClassIndex)

font_scale = 3
font = cv2.FONT_HERSHEY_PLAIN
for ClassInd, conf, boxes in zip(ClassIndex.flatten(), confidece.flatten(), bbox):
   # Draw rectangle around the detected object
      cv2.rectangle(image, boxes, (255, 0, 0), 2)
      cv2.putText(image, classLabels[ClassInd - 1], (boxes[0] + 10, boxes[1] + 40),
                        font, font_scale, (0, 255, 0), thickness=3)

plt.imshow(cv2.cvtColor(image,cv2.COLOR_BGR2RGB))
```

input image:

output image

This Solution reads an image, uses a pre-trained SSD model with MobileNetV3 to detect objects in the image, draws bounding boxes around the detected objects, and labels them using class names. Finally, it displays the processed image with detections.

The process steps are:

1. Load model configuration and weights.
2. Set up image preprocessing.

3. Read and display the input image.
4. Run object detection on the image.
5. Draw bounding boxes and labels.
6. Display the resulting image with the detected objects.