In [ ]:
```
!pip3 install torch torchvision torchaudio pandas pyYAML tqdm seaborn opencv-python
```

```
Requirement already satisfied: torch in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (1.12.0)
Requirement already satisfied: torchvision in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (0.13.0)
Requirement already satisfied: torchaudio in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (0.12.0)
Requirement already satisfied: pandas in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (1.4.3)
Requirement already satisfied: pyYAML in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (6.0)
Requirement already satisfied: tqdm in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (4.64.0)
Requirement already satisfied: seaborn in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (0.11.2)
Requirement already satisfied: opencv-python in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (4.6.0.66)
Requirement already satisfied: numpy in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (1.23.1)
Requirement already satisfied: typing-extensions in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from torch) (4.3.0)
Requirement already satisfied: pillow!=8.3.*,>=5.3.0 in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from torchvision) (9.2.0)
Requirement already satisfied: requests in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from torchvision) (2.28.1)
Requirement already satisfied: pytz>=2020.1 in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from pandas) (2022.1)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: colorama in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from tqdm) (0.4.5)
Requirement already satisfied: matplotlib>=2.2 in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from seaborn) (3.5.2)
Requirement already satisfied: scipy>=1.0 in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from seaborn) (1.8.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from matplotlib>=2.2->seaborn) (4.34.4)
Requirement already satisfied: cycler>=0.10 in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from matplotlib>=2.2->seaborn) (0.11.0)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from matplotlib>=2.2->seaborn) (3.0.9)
Requirement already satisfied: packaging>=20.0 in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from matplotlib>=2.2->seaborn) (21.3)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from matplotlib>=2.2->seaborn) (1.4.4)
Requirement already satisfied: six>=1.5 in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from python-dateutil>=2.8.1->pandas) (1.16.0)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from requests->torchvision) (2.1.0)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from requests->torchvision) (1.26.10)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from requests->torchvision) (2022.6.15)
Requirement already satisfied: idna<4,>=2.5 in c:\users\jmess\appdata\local\programs\python\python39\lib\site-packages (from requests->torchvision) (3.3)
```

In [ ]:
```
%matplotlib inline
```

In [ ]:
```
import torch
import cv2
from matplotlib import pyplot as plt
```

In [ ]:
```
model = torch.hub.load('ultralytics/yolov5', 'yolov5s')
```

```
Using cache found in C:\Users\jmess/.cache\torch\hub\ultralytics_yolov5_master
YOLOv5  2022-7-18 Python-3.9.13 torch-1.12.0+cpu CPU

Fusing layers...
YOLOv5s summary: 213 layers, 7225885 parameters, 0 gradients
Adding AutoShape...
```

In [ ]:
```
def load_image(path):
    return cv2.imread(path)[:, :, ::-1]
```

In [ ]:
```
def process_image(img):
    result = model([img], size=640)

    print(result.pandas().xyxy[0])

    return result
```

In [ ]:
```
IMGS_PATH = ['imgs/all.png', 'imgs/bus.png', 'imgs/people.png']
```

In [ ]:
```
IMGS = []

for i in IMGS_PATH:
    IMGS.append(load_image(i))
```

a)

In [ ]:
```
def getRGBfromI(RGBint):
    blue =  RGBint & 255
    green = (RGBint >> 8) & 255
    red =   (RGBint >> 16) & 255
    return red, green, blue
```

In [ ]:
```
def draw_result(frame, results):
    frame = frame.copy()
    for box in results.xyxy[0]:
            xB = int(box[2])
            xA = int(box[0])
            yB = int(box[3])
            yA = int(box[1])

            cv2.rectangle(frame, (xA, yA), (xB, yB), getRGBfromI(int(box[5]) * 100

    return frame
```

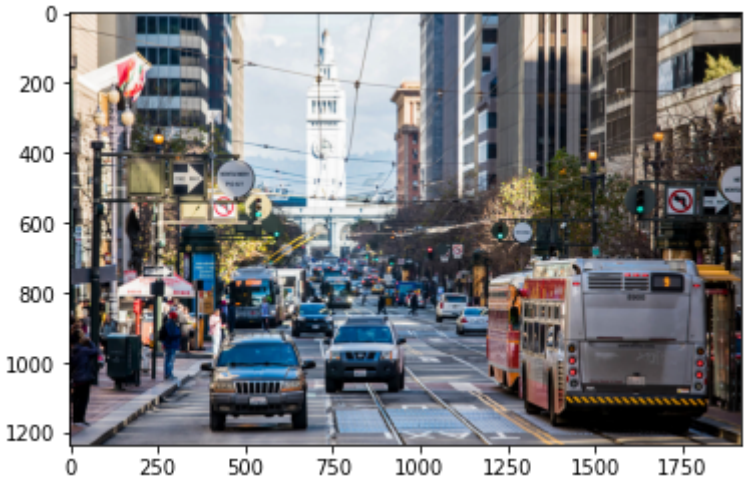
In [ ]:
```
def process(img):
    plt.imshow(img)
    plt.show()

    result = process_image(img)

    print(result)
```

```
    plt.imshow(draw_result(img, result))
    plt.show()
```

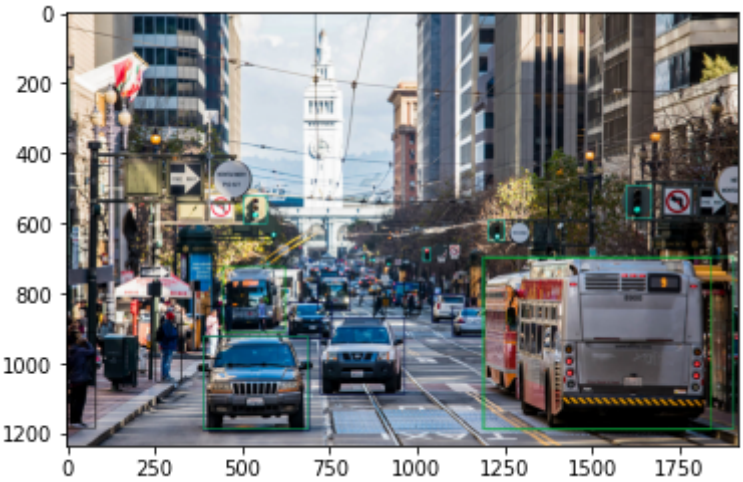
In [ ]:
```
process(IMGS[0])
```



	xmin	ymin	xmax	ymax	confidence	class
0	726.216492	870.891785	967.580078	1088.871460	0.846512	2
1	1104.301514	841.032959	1196.471558	923.366333	0.829974	2
2	393.870117	922.003052	694.894958	1192.261230	0.800172	2
3	1.707250	910.309021	84.191330	1184.690308	0.776526	0
4	636.185120	826.378418	760.100464	929.397766	0.770256	2
5	1045.868408	804.348999	1140.609253	883.489014	0.762862	2
6	1600.351807	493.414764	1672.191406	587.836914	0.680482	9
7	256.838928	852.810669	328.951660	1052.742432	0.631074	0
8	447.830383	725.431458	625.831116	910.149231	0.628749	5
9	1189.572998	697.095337	1907.240234	1186.575684	0.597191	7
10	1206.703735	593.303833	1252.806030	651.825256	0.585567	9
11	878.548706	793.447327	911.642090	870.598999	0.522277	0
12	971.822571	797.682190	1003.063477	869.149353	0.472607	0
13	718.716797	753.658752	812.098267	850.487915	0.451698	2
14	396.205566	838.769409	441.134613	986.398010	0.450832	0
15	507.221191	523.475037	573.016846	602.472778	0.437403	9
16	1020.958191	670.210266	1041.742432	708.367859	0.422663	9
17	392.386963	924.273560	691.774841	1186.112305	0.386347	7
18	1196.590454	700.699097	1843.417358	1188.745605	0.296920	5
19	964.837463	767.659180	1010.002258	861.787720	0.285839	0

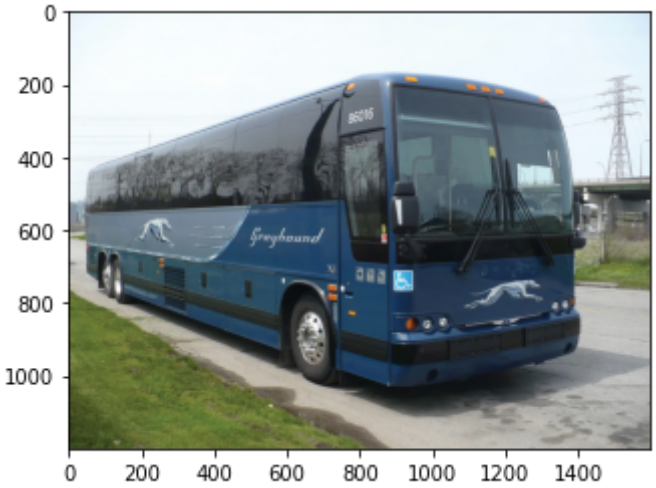
	name
0	car
1	car
2	car
3	person
4	car
5	car
6	traffic light
7	person
8	bus
9	truck
10	traffic light
11	person
12	person
13	car
14	person
15	traffic light
16	traffic light
17	truck
18	bus
19	person

```
image 1/1: 1235x1920 6 persons, 6 cars, 2 buss, 2 trucks, 4 traffic lights
Speed: 13.0ms pre-process, 106.7ms inference, 2.0ms NMS per image at shape (1, 3,
416, 640)
```

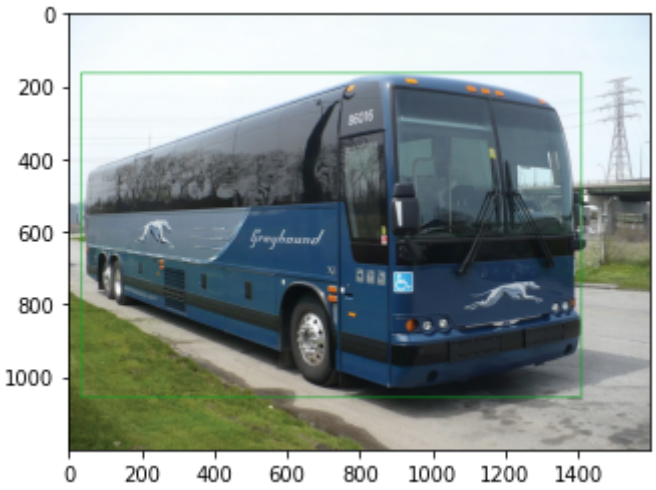


Como podemos ver, já que existem varios elementos na foto boa parte deles foi encontrado pelo modelo. Mas dá para ver que quanto mais profundo na foto o elemento está menor a confiança na classificação temos. Provavelmente devido menor informação de pixels para representá-lo dentro da foto.

In [ ]:
```
process(IMGS[1])
```

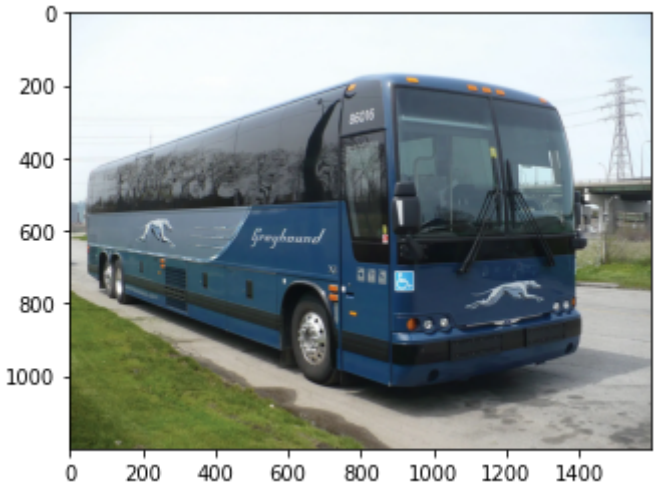



```
        xmin        ymin         xmax         ymax  confidence  class name
0  35.439453  164.285309  1411.653564  1055.357544    0.862536      5  bus
image 1/1: 1200x1600 1 bus
Speed: 13.0ms pre-process, 131.3ms inference, 2.0ms NMS per image at shape (1, 3,
480, 640)
```

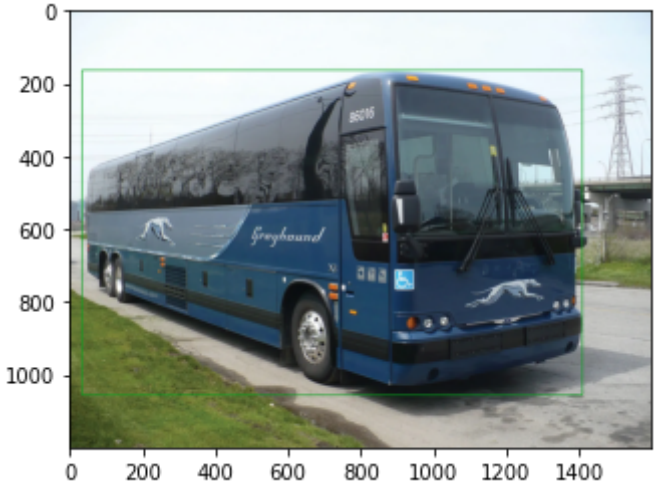


O unico elementos da foto propositalmente é um onibus e podemos ver que o modelo o identifica, com um nivel consideravel de confiança.

In [ ]:
```
process(IMGS[1])
```




```
        xmin        ymin         xmax         ymax  confidence  class name
0  35.439453  164.285309  1411.653564  1055.357544    0.862536      5  bus
image 1/1: 1200x1600 1 bus
Speed: 8.9ms pre-process, 109.6ms inference, 1.0ms NMS per image at shape (1, 3, 4
80, 640)
```



Os dois unico elementos da foto propositalmente são duas pessoas e podemos ver que o modelo o identifica, com um nivel consideravel de confiança as duas pessoas.

b)

In [ ]:
```python
# Transforma o video em imagens para ser usado no modelo

def get_frames(path):
    vidcap = cv2.VideoCapture(path)
    vidcap.set(cv2.CAP_PROP_FRAME_WIDTH, 1280)
    vidcap.set(cv2.CAP_PROP_FRAME_HEIGHT, 720)
    lista_results = []

    success,image = vidcap.read()
    while success:
        image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
        lista_results.append(image)
        success,image = vidcap.read()

    return lista_results
```

In [ ]:
```python
# Pega todos os frames
```

```
results = get_frames('videos/cross.mp4')
```

In [ ]:
```
# Executa os frames no modelo

t = model(results)
pd = t.pandas().xyxy
```

In [ ]:
```
# Verifica e calcula os centros dos quadrados demarcadores
def get_path(pd):
    path = []

    for j in pd:
        i = j[j['class'] == 0]

        if len(i) > 0:
            i = i.iloc[0]
            x = (i['xmax'] + i['xmin']) // 2
            y = (i['ymax'] + i['ymin']) // 2
            path.append((int(x), int(y)))

    return path
```

In [ ]:
```
# Pinta aréa a partir dos centros dos quadrados demarcadores calculado
def print_area(img, point, w):
    for i in range(point[0] - w, point[0] + w):
        for j in range(point[1] - w, point[1] + w):
            try:
                img[j][i] = (255, 0, 0)
            except IndexError:
                pass
```

In [ ]:
```
# Angraria os centros
a = get_path(pd)
```

In [ ]:
```
# Pinta o caminho na imagem

img = results[0].copy()
for i in a:
    print_area(img, i, 30)
```

In [ ]:
```
# Exibe a imagem

plt.imshow(img)
plt.show()
```