untitled0

April 23, 2024

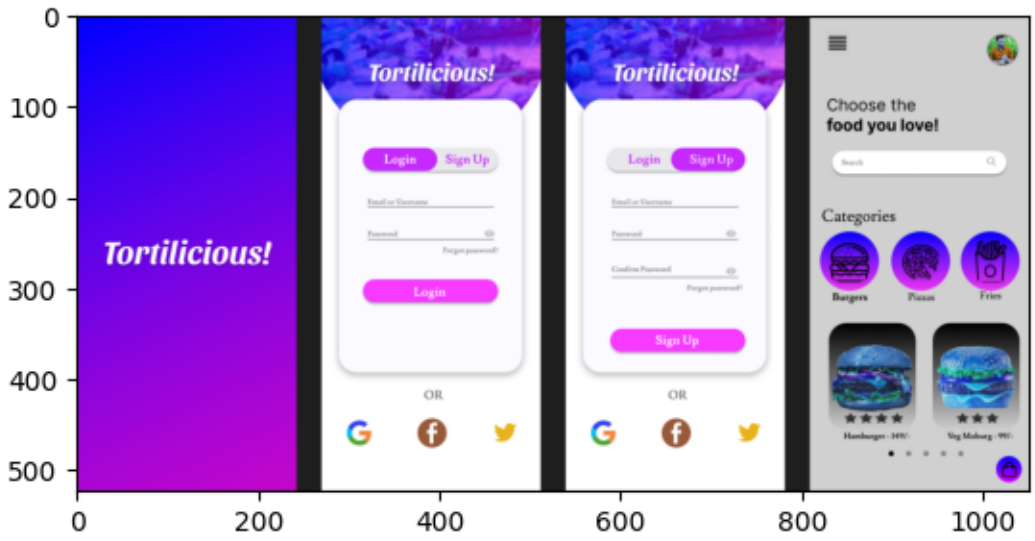
[1]:
```
!pip install opencv-python
```

```
Requirement already satisfied: opencv-python in /usr/local/lib/python3.10/dist-
packages (4.8.0.76)
Requirement already satisfied: numpy>=1.21.2 in /usr/local/lib/python3.10/dist-
packages (from opencv-python) (1.25.2)
```

[3]:
```
import cv2
import matplotlib.pyplot as plt
import numpy as np
```

[15]:
```
image1 =cv2.imread("/content/Screenshot 2024-04-23 154735.png")
plt.imshow(image1)
```

[15]:
```
<matplotlib.image.AxesImage at 0x78082424c130>
```



[16]:
```
image =cv2.imread("/content/re2.png")
plt.imshow(image)
```

[16]: <matplotlib.image.AxesImage at 0x780824356500>

[39]:
```
image2 =cv2.imread("/content/re1.png")
plt.imshow(image2)
```

[39]: <matplotlib.image.AxesImage at 0x78082418a9e0>

[7]:
```
type(image)
```

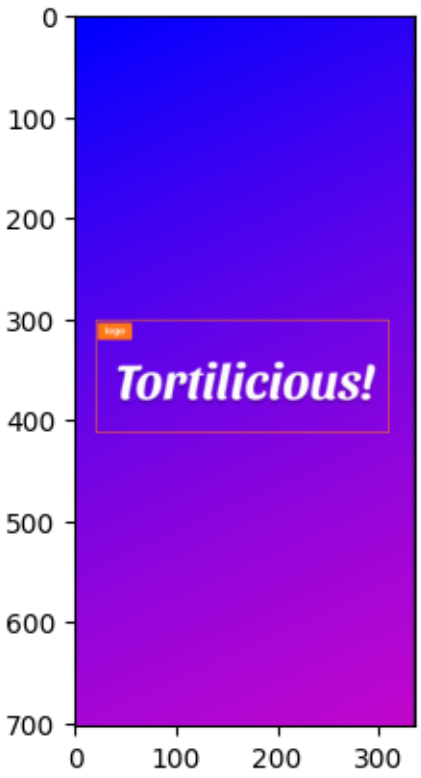
[7]:
```
numpy.ndarray
```

[8]:
```
image.shape
```

[8]:
```
(703, 336, 3)
```

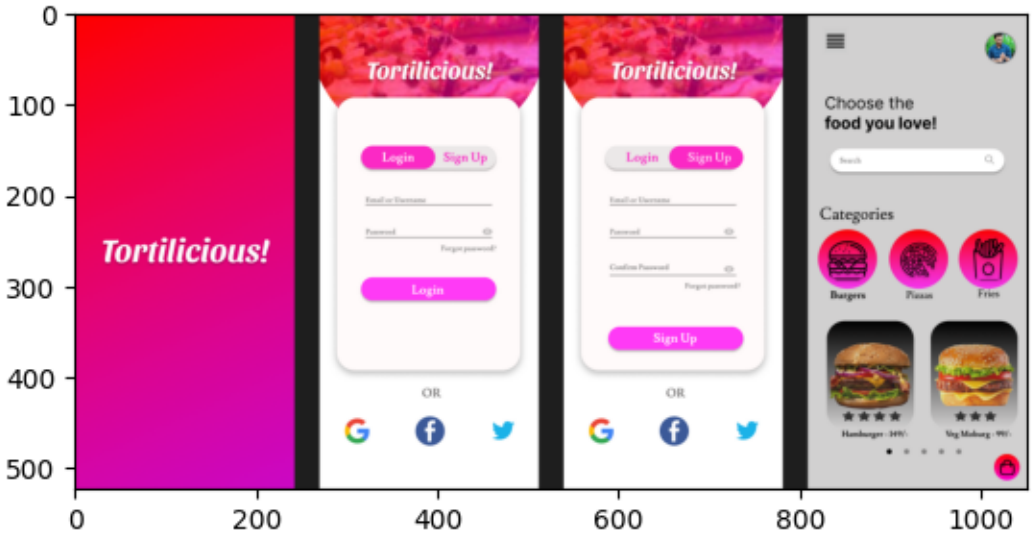
[40]:
```
plt.imshow(image)
```

[40]:
```
<matplotlib.image.AxesImage at 0x780823e0ad40>
```



[37]:
```
new_image = cv2.cvtColor(image1 , cv2.COLOR_BGR2RGB)
plt.imshow(new_image)
```

[37]: <matplotlib.image.AxesImage at 0x7808240ec940>



[11]:
```
"""
1. Splitting image channels

"""
r,g,b =cv2.split(new_image)
print('r', r.shape)
print('g', g.shape)
print('b', b.shape)
```

```
r (703, 336)
g (703, 336)
b (703, 336)
```

[12]:
```
"""
resize"""

s=10
w=int(new_image.shape[1]*s/100)
h=int(new_image.shape[0]*s/100)
dim= (w,h)
re_size = cv2.resize(new_image,dim, interpolation= cv2.INTER_AREA)
re_size.shape
```

[12]:
```
(70, 33, 3)
```

[13]:
```
import cv2
import matplotlib.pyplot as plt
import numpy as np
```

[18]:
```
yolo= cv2.dnn.readNet("/content/yolov3-tiny.weights", "/content/yolov3-tiny (1).
  ↪cfg")
```

[20]:
```
classes = []
with open("/content/coco.names",'r') as f:
  classes= f.read().splitlines()
```

[21]:
```
len(classes)
```

[21]:
```
80
```

[22]:
```
classes
```

[22]:
```
['person',
 'bicycle',
 'car',
 'motorbike',
```

```
'aeroplane',
'bus',
'train',
'truck',
'boat',
'traffic light',
'fire hydrant',
'stop sign',
'parking meter',
'bench',
'bird',
'cat',
'dog',
'horse',
'sheep',
'cow',
'elephant',
'bear',
'zebra',
'giraffe',
'backpack',
'umbrella',
'handbag',
'tie',
'suitcase',
'frisbee',
'skis',
'snowboard',
'sports ball',
'kite',
'baseball bat',
'baseball glove',
'skateboard',
'surfboard',
'tennis racket',
'bottle',
'wine glass',
'cup',
'fork',
'knife',
'spoon',
'bowl',
'banana',
'apple',
'sandwich',
'orange',
'broccoli',
```

```
 'carrot',
 'hot dog',
 'pizza',
 'donut',
 'cake',
 'chair',
 'sofa',
 'pottedplant',
 'bed',
 'diningtable',
 'toilet',
 'tvmonitor',
 'laptop',
 'mouse',
 'remote',
 'keyboard',
 'cell phone',
 'microwave',
 'oven',
 'toaster',
 'sink',
 'refrigerator',
 'book',
 'clock',
 'vase',
 'scissors',
 'teddy bear',
 'hair drier',
 'toothbrush']
```

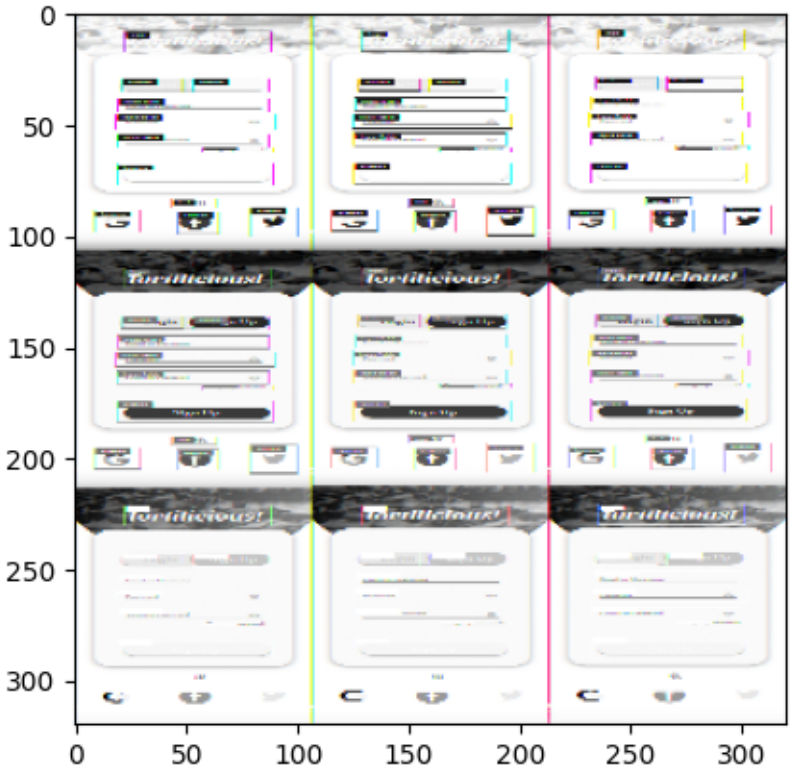
[42]:
```
img=cv2.imread("/content/re1.png")

blob = cv2.dnn.blobFromImage (img, 1/255, (320,320), (0,0,0), swapRB=True, crop␣
↪= False)
```

[24]:
```
blob.shape
```

[24]:
```
(1, 3, 320, 320)
```

[43]:
```
# print image

i = blob[0].reshape(320,320,3)
plt.imshow(i)
```

[43]:
```
<matplotlib.image.AxesImage at 0x78081af098d0>
```



[26]:
```
yolo.setInput(blob)
```

[27]:
```
output_layers_name = yolo.getUnconnectedOutLayersNames()
layeroutput = yolo.forward(output_layers_name)
```

[45]:
```
print(output_layers_name)
print(layeroutput)
```

```
('yolo_16', 'yolo_23')
(array([[nan, nan, nan, …,  0.,  0.,  0.],
       [nan, nan, nan, …,  0.,  0.,  0.],
       [nan, nan, nan, …,  0.,  0.,  0.],
       …,
       [nan, nan, nan, …,  0.,  0.,  0.],
       [nan, nan, nan, …,  0.,  0.,  0.],
       [nan, nan, nan, …,  0.,  0.,  0.]], dtype=float32), array([[nan, nan,
nan, …,  0.,  0.,  0.],
       [nan, nan, nan, …,  0.,  0.,  0.],
       [nan, nan, nan, …,  0.,  0.,  0.],
       …,
       [nan, nan, nan, …,  0.,  0.,  0.],
       [nan, nan, nan, …,  0.,  0.,  0.],
```

```
       [nan, nan, nan, …,  0.,  0.,  0.]], dtype=float32))
```

[53]:
```
import cv2

# Load YOLO model
yolo_weights_path = "/content/yolov3-tiny.weights"
yolo_config_path = "/content/yolov3-tiny(1).cfg"
yolo_net = cv2.dnn.readNet(yolo_weights_path, yolo_config_path)

# Get output layers' names
output_layers_name = yolo_net.getUnconnectedOutLayersNames()

# Assuming you have loaded your image as 'input_image'
# Perform forward pass
yolo_net.setInput(input_image)
layer_outputs = yolo_net.forward(output_layers_name)
```

```
---------------------------------------------------------------------------
error                                     Traceback (most recent call last)
<ipython-input-53-aeeff99038a1> in <cell line: 6>()
      4 yolo_weights_path = "/content/yolov3-tiny.weights"
      5 yolo_config_path = "/content/yolov3-tiny(1).cfg"
----> 6 yolo_net = cv2.dnn.readNet(yolo_weights_path, yolo_config_path)
      7
      8 # Get output layers' names

error: OpenCV(4.8.0) /io/opencv/modules/dnn/src/darknet/darknet_importer.cpp:21( :
 ↪ error: (-212:Parsing error) Failed to open NetParameter file: /content/
 ↪yolov3-tiny(1).cfg in function 'readNetFromDarknet'
```

[28]:
```
boxes = []
confidences = []
class_ids = []
for output in layeroutput:
    for detection in output:
        scores = detection[5:]  # Extract confidence scores
        class_id = np.argmax(scores)
        confidence = scores[class_id]

        if confidence > 0.7:
            center_x = int(detection[0] *width)
            center_y = int(detection[1] *height)
            w = int(detection[2] *width)
            h = int(detection[3] *height)

            x = int(center_x - w / 2)
```

```
            y = int(center_y - h / 2)

            boxes.append([x, y, w, h])
            confidences.append(float(confidence))
            class_ids.append(class_id)
```

[29]:
```
len(boxes)
```

[29]: 0

[46]:
```
indexes = cv2.dnn.NMSBoxes(boxes,confidences,.5,.4)
 print(indexes)
```

()

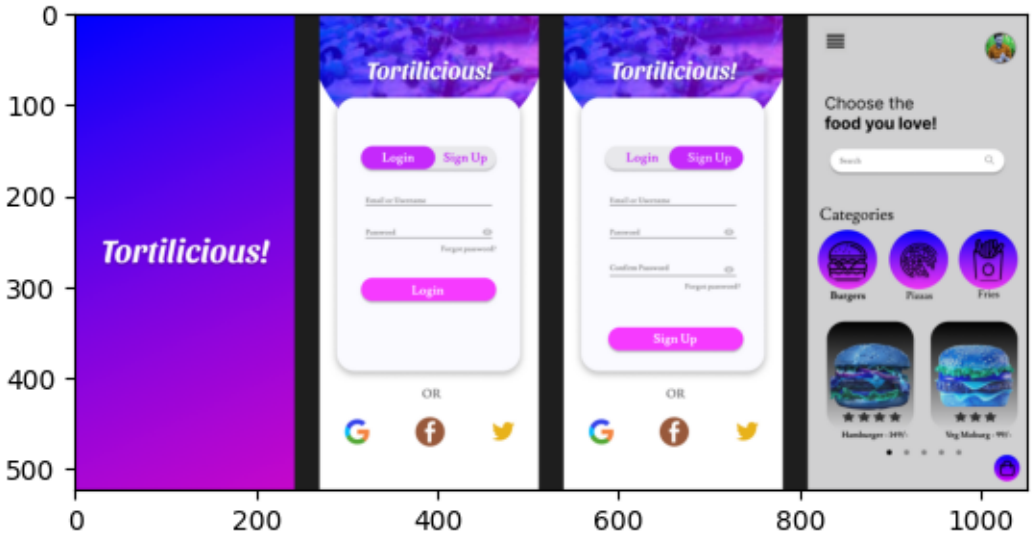
[48]:
```
font = cv2.FONT_HERSHEY_PLAIN
colors = np.random.uniform(0, 255, size = (len(boxes),3))

print(font)
print(colors)
```

1
[]

[51]:
```
for i in list(indexes):
            x,y,w,h = boxes[i]
            label = str(classes[class_ids[i]])
            confi = str(round(confidence[i], 2))
            color = colors[i]
            cv2.rectangle(img,(x,y),(x+w,y+h),color,2)
            cv2.putText(img,label + " " +confi, (x,y+20),font,2,(255,255,255),1)
```

[52]:
```
plt.imshow(image1)
```

[52]: <matplotlib.image.AxesImage at 0x78081af87d00>


0
100
200
300
400
500
0
200
400
600
800
1000
Tortilicious!
Tortilicious!
Login
Sign Up
Login
OR
Tortilicious!
Login
Sign Up
Sign Up
OR
Choose the
food you love!
Categories
Burgers
Pizzas
Fries