In [ ]:
```
import requests
from urllib.parse import urlparse
from io import BytesIO
from PIL import Image, ImageDraw
import matplotlib.pyplot as plt
import matplotlib.image as mpimg

import os, time, uuid

from azure.cognitiveservices.vision.customvision.training import CustomVisionTrainingClient
from azure.cognitiveservices.vision.customvision.prediction import CustomVisionPredictionClient
from azure.cognitiveservices.vision.customvision.training.models import ImageFileCreateBatch, ImageFileCreateEntry, Region
from msrest.authentication import ApiKeyCredentials
```

In [ ]:
```
def display_image(path):
    img = mpimg.imread(path)
    imgplot = plt.imshow(img)
    plt.show()
```

In [ ]:
```
train_endpoint = "https://udacityligtherdetection.cognitiveservices.azure.com/"
train_key = "cc3e781efc734e18ba44adb585399cc3"
train_resource_id = "/subscriptions/a70a7281-34b4-43ff-932a-1f9171daff2c/resourceGroups/aind-246477/providers/Microsoft.CognitiveServices/accounts/udacityligtherdetection"

pred_endpoint = "https://udacityligtherdetection-prediction.cognitiveservices.azure.com/"
pred_key = "ba7ce73bb517493db18927c49e94d379"
pred_resource_id = "/subscriptions/a70a7281-34b4-43ff-932a-1f9171daff2c/resourceGroups/aind-246477/providers/Microsoft.CognitiveServices/accounts/udacityligtherdetection-Prediction"
```

In [ ]:
```
pred_cred = ApiKeyCredentials(in_headers={"Prediction-key": pred_key})
prediction_model = CustomVisionPredictionClient(endpoint=pred_endpoint, credentials=pred_cred)

train_cred = ApiKeyCredentials(in_headers={"Training-key": train_key})
train_model = CustomVisionTrainingClient(train_endpoint, train_cred)
```

In [ ]:
```
obj_dete_domain = next(domain for domain in train_model.get_domains() if domain.type == "ObjectDetection" and domain.name == "General")

project_name = "UdacityLighterDetectionSDK"
project = train_model.create_project(project_name, domain_id=obj_dete_domain.id)
print("The project {} has been created.".format(project.name))
```

```
The project UdacityLighterDetectionSDK has been created.
```

In [ ]:
```
project_status = project.status
print("The project status is {}.".format(project_status))
```

```
The project status is Succeeded.
```

In [ ]:
```
lighter_tag = train_model.create_tag(project.id, "lighter")
```

In [ ]:
```
train_iteration = train_model.train_project(project.id)
while (train_iteration.status != "Completed"):
    train_iteration = train_model.get_iteration(project.id, train_iteration.id)
    print ("Training status: " + train_iteration.status)
    print ("Waiting 2min..")
    time.sleep(120)
```

```
Training status: Training
Waiting 2min..
Training status: Training
Waiting 2min..
Training status: Training
Waiting 2min..
Training status: Completed
Waiting 2min..
```

In [ ]:
```
train_iteration.id
```

Out[ ]: 'eab353bc-fb3e-4f6c-a339-6b3bd1f914c6'

In [ ]:
```
iteration_list = train_model.get_iterations(project.id)
for iteration_item in iteration_list:
    iteration_item_perf = train_model.get_iteration_performance(project.id, iteration_item.id)
    iteration_item_perf_dict = iteration_item_perf.as_dict()
    print("ID: {} The model precision is: {} The model recall is: {} The model mAP is: {}".format(iteration_item.id, iteration_item_perf_dict["precision"], iteration_item_perf_dict["recall"],iteration_item_perf_dict["avera
    # Something is wrong here, the values do not match the Custom Vision Portal
```

```
ID: eab353bc-fb3e-4f6c-a339-6b3bd1f914c6 The model precision is: 1.0 The model recall is: 0.06666667 The model mAP is: 0.8381862
ID: 3f492802-6498-406f-aeb4-e1c8ed2bf4ee The model precision is: 1.0 The model recall is: 0.13333334 The model mAP is: 0.7264351
```

In [ ]:
```
train_perf = train_model.get_iteration_performance(project.id, "eab353bc-fb3e-4f6c-a339-6b3bd1f914c6")
train_perf_dict = train_perf.as_dict()
print("The model precision is: {} The model recall is: {} The model mAP is: {}".format(train_perf_dict["precision"],train_perf_dict["recall"],train_perf_dict["average_precision"]))

# Something is wrong here, the values do not match the Custom Vision Portal
```

```
The model precision is: 1.0 The model recall is: 0.06666667 The model mAP is: 0.8381862
```

In [ ]:
```
publish_name = "detect-lighter-model-V2"
train_model.publish_iteration(project.id, train_iteration.id, publish_name, pred_resource_id)
print ("the model {} has been published.".format(publish_name))
```

```
the model detect-lighter-model-V2 has been published.
```

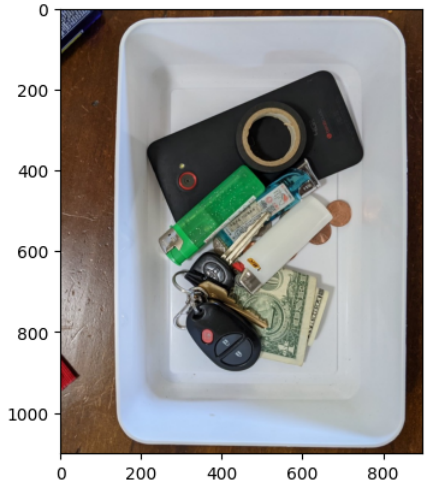
In [ ]:
```
image_path = "C:\\_Huissel\\PythonProjects\\workspace\\udacity\\cd0461-building-computer-vision-solutions-with-azure-project-starter\\starter\\sample_submission\\material_preparation_step\\lighter_test_images"

image_list = []
for root, dirs, files in os.walk(image_path):
    for file in files:
        if file.endswith(".jpg"):
            image_list.append(os.path.join(root, file))
```

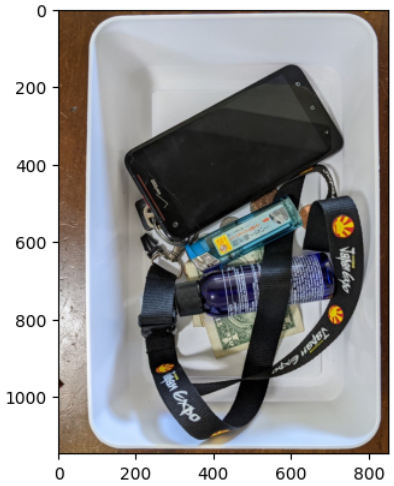
In [ ]:
```
def perform_prediction(test_image_path, model_id, model_iteration_name):
    with open(os.path.join (test_image_path), "rb") as test_image:
        results = prediction_model.detect_image(model_id, model_iteration_name, test_image.read())
        # Display the results.
        for n, prediction in enumerate(results.predictions):
            if prediction.probability > 0.1:
                print("\t" + prediction.tag_name +
                    ": {0:.2f}%".format(prediction.probability * 100))
            elif n == 0:
                 print("\t" + prediction.tag_name +
                    ": {0:.2f}%".format(prediction.probability * 100))
```

In [ ]:
```
for image in image_list:
    display_image(image)
    perform_prediction(image, project.id, publish_name)
```

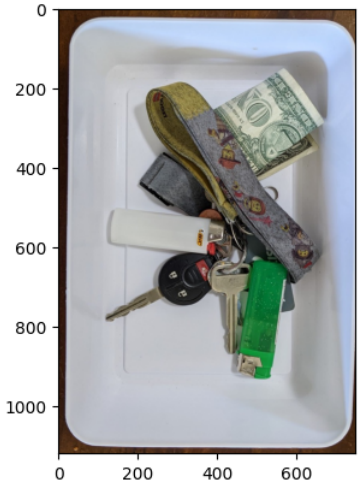

```
lighter: 65.05%
lighter: 26.22%
lighter: 18.40%
```



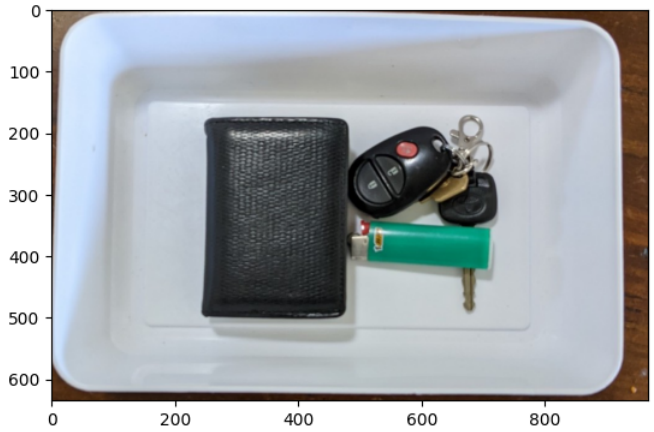

```
lighter: 31.19%
lighter: 16.46%
lighter: 12.89%
```



lighter: 6.59%



lighter: 31.70%
lighter: 23.01%




```
lighter: 14.10%
lighter: 13.18%
lighter: 12.88%
```

In [ ]: