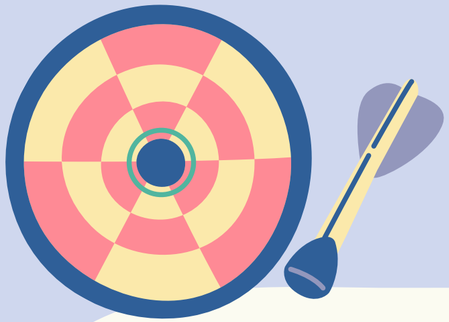
BORCELLE TOY

ระบบจัดการลานจอดรถ
โดยการอ่านป้ายทะเบียน

Member

นางสาวนริศรา	แป้งคุณญาติ	6320500581
นางสาวศุภลักษณ์	เหลือบุญชู	6320500671
นางสาวกุลวฏี	จารี	6320502355
นางสาวปภัสสินี	ราชประสิทธิ์	6320503050
นางสาวสุกัลยา	ขำเจริญ	6320503106



หลักการและเหตุผล

ในปัจจุบันมีผู้ใช้ยานพาหนะเพิ่มขึ้นเป็นจำนวนมาก และด้วยหอพักมีพื้นที่จำกัดในการจอดรถ ทำให้การหาที่จอดรถเป็นไปด้วยความยากลำบาก ต้องวนหาที่จอดรถเป็นเวลานาน ซึ่งทำให้เสียเวลา เปลืองพลังงาน หรือมีพื้นที่ไม่เพียงพอสำหรับบุคคลที่พักอาศัยภายในหอพัก เนื่องจากมีบุคคลภายนอกเข้าไปจอดรถ

โครงการนี้จึงพัฒนาระบบจัดการลานจอดรถโดยการอ่านป้ายทะเบียน เพื่อให้บุคคลภายในหอพักมีที่จอดรถเพียงพอ ป้องกันไม่ให้บุคคลภายนอกเข้ามาจอดรถได้ เพื่อความปลอดภัยของบุคคลภายในหอพักอีกด้วย

Responsible Individual

Deliverable	Responsible Individual
รวบรวมฐานข้อมูล	สุกัลยาและปภัสสินี
เขียนAI เทรน AI	กุลวฎี นริศราและศุภลักษณ์
ออบแบบเว็บไซต์	สุกัลยาและปภัสสินี
ทดสอบ AI	กุลวฎี นริศราและศุภลักษณ์
เชื่อมต่อโมเดลกับเว็บ	ทุกคน

ปัญหาที่พบ

1.ไม่สามารถเชื่อม Model กับเว็บได้

2.สมาชิกมีเวลาไม่ตรงกัน

3.Library ที่ใช้กับโปรแกรมนี้เก่าเกินไป และไม่รองรับกับ Python เวอร์ชันปัจจุบัน

เทคนิคที่ใช้ในการดำเนินงาน

1. 

HTML

2. 

my sql

3. 

python

โปรแกรมที่ใช้ในการดำเนินงาน

1. 

Visual Studio

2. 

pingendo

3. 

microsoft office

4. 

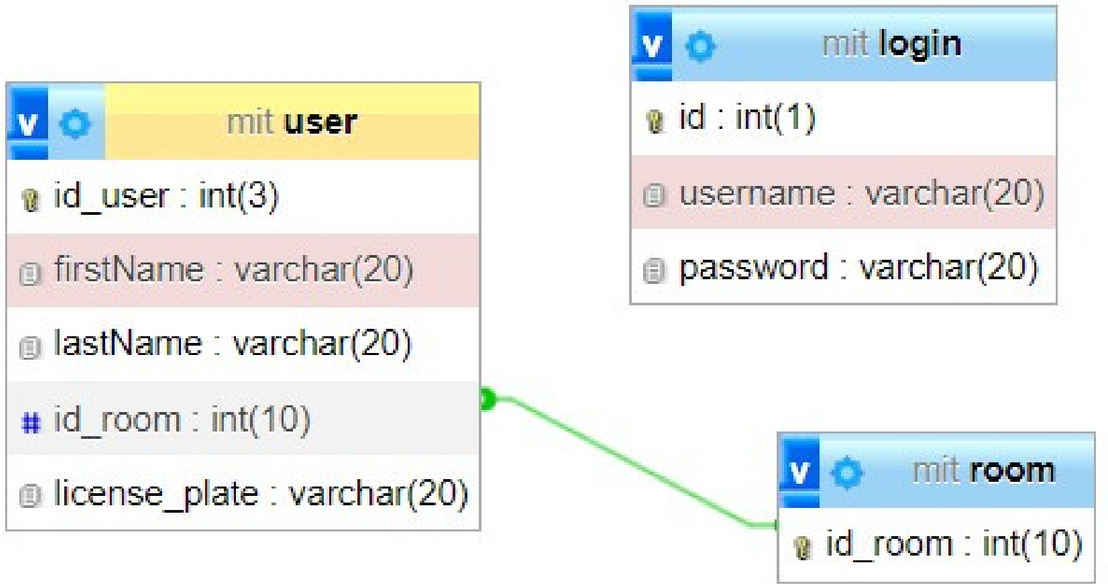
jupyter notebook

5. 

phpmyaddmin

ออกแบบฐานข้อมูล

ER diagram



Data Set

trian จำนวน 410 รูป

test จำนวน 22 รูป

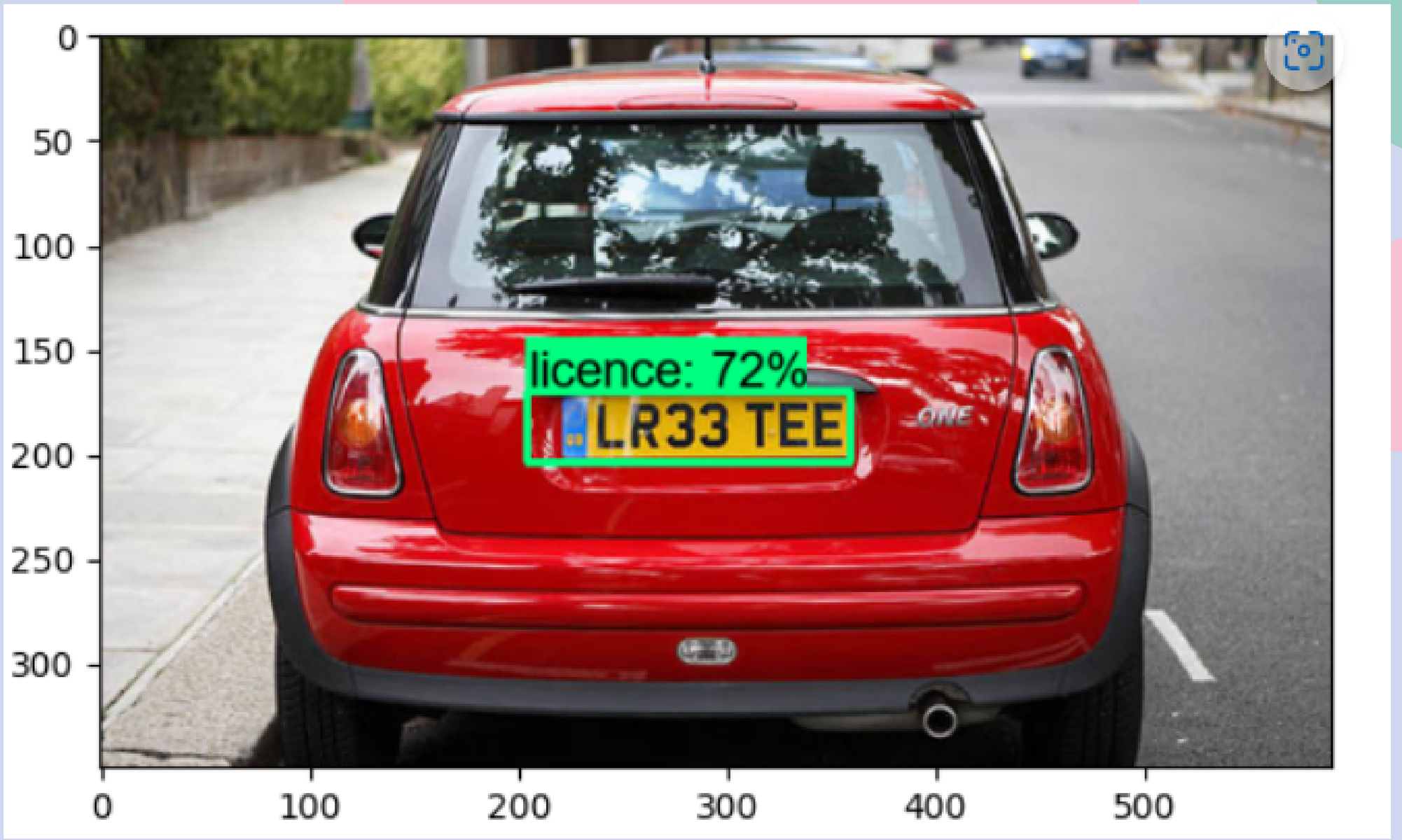
In [81]:
```
TRAINING_SCRIPT = os.path.join(paths['APIMODEL_PATH'], 'research', 'object_detection', 'model_main_tf2.py')
```

In [82]:
```
command = "python {} --model_dir={} --pipeline_config_path={} --num_train_steps=5000".format(TRAINING_SCRIPT, paths['CHECKPOINT_P
```

In [83]:
```
print(command)
```

```
python Tensorflow\models\research\object_detection\model_main_tf2.py --model_dir=Tensorflow\workspace\models\my_ssd_mobnet --pi
peline_config_path=Tensorflow\workspace\models\my_ssd_mobnet\pipeline.config --num_train_steps=5000
```

```
img = cv2.imread(IMAGE_PATH)
image_np = np.array(img)

input_tensor = tf.convert_to_tensor(np.expand_dims(image_np, 0), dtype=tf.float32)
detections = detect_fn(input_tensor)

num_detections = int(detections.pop('num_detections'))
detections = {key: value[0, :num_detections].numpy()
              for key, value in detections.items()}
detections['num_detections'] = num_detections

# detection_classes should be ints.
detections['detection_classes'] = detections['detection_classes'].astype(np.int64)

label_id_offset = 1
image_np_with_detections = image_np.copy()

viz_utils.visualize_boxes_and_labels_on_image_array(
            image_np_with_detections,
            detections['detection_boxes'],
            detections['detection_classes']+label_id_offset,
            detections['detection_scores'],
            category_index,
            use_normalized_coordinates=True,
            max_boxes_to_draw=5,
            min_score_thresh=.6,
            agnostic_mode=False)

plt.imshow(cv2.cvtColor(image_np_with_detections, cv2.COLOR_BGR2RGB))
plt.show()
```


licence: 72%
LR33 TEE
0
50
100
150
200
250
300
0
100
200
300
400
500

```python
def ocr_it(image, detections, detection_threshold, region_threshold):

    #Scores, boxes and classes above threshold
    scores = list(filter(lambda x: x> detection_threshold, detections['detection_scores']))
    boxes = detections['detection_boxes'][:len(scores)]
    classes = detections['detection_classes'][:len(scores)]

    #Full image dimension
    width = image.shape[1]
    height = image.shape[0]

    #Apply ROI filtering and OCR
    for idx, box in enumerate(boxes):
        roi = box*[height, width, height, width]
        region = image[int(roi[0]):int(roi[2]),int(roi[1]):int(roi[3])]
        reader = easyocr.Reader(['en'])
        ocr_result = reader.readtext(region)

        text = filter_text(region, ocr_result, region_threshold)

        plt.imshow(cv2.cvtColor(region, cv2.COLOR_BGR2RGB))
        plt.show()
        print(text)
        return text, region
```

```
text, region = ocr_it(image_np_with_detections, detections, detection_threshold, region_threshold)
```

```
WARNING:easyocr.easyocr:CUDA not available - defaulting to CPU. Note: This module is much faster with a GPU.
```


```
['LR33 TEE']
```