EECS 581/582 Final Project

Team 18

Project Artifacts

The following diagram shows what the flow of our project would look like:

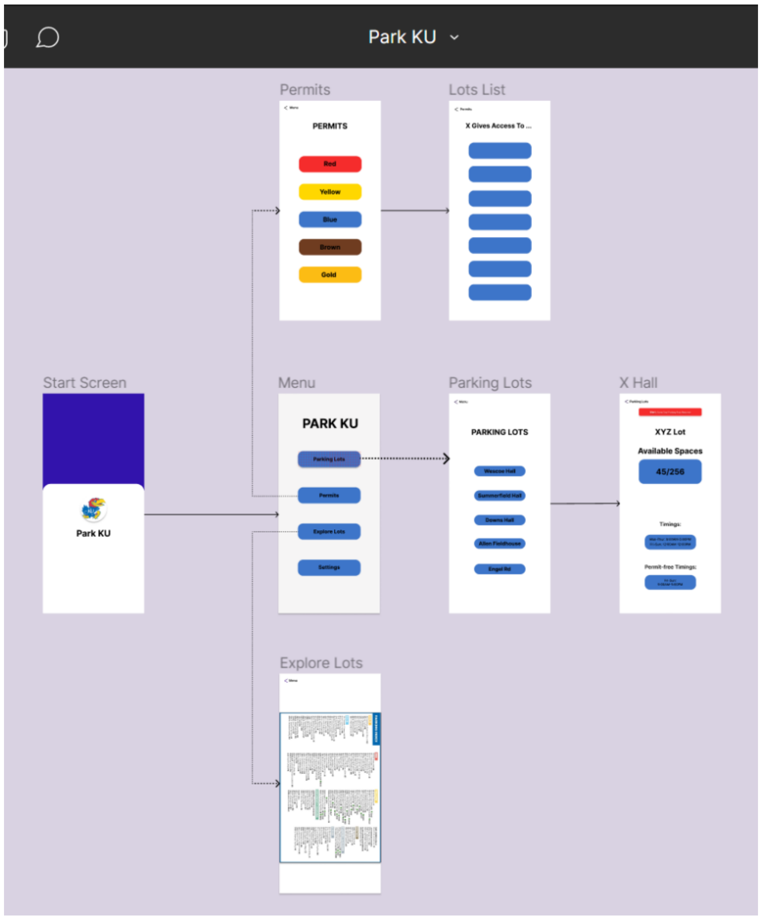


Fig5: The flow of our project; ParkKU

The tentative flow of our app can be viewed on the following link:

https://github.com/hzahid99/EECS-Final-Project/blob/main/Demo%20of%20the%20app.mp4

The following GitHub repository holds our progress so far:

https://github.com/hzahid99/EECS-Final-Project

Following are the screenshots of the screens we have made so far for the app:

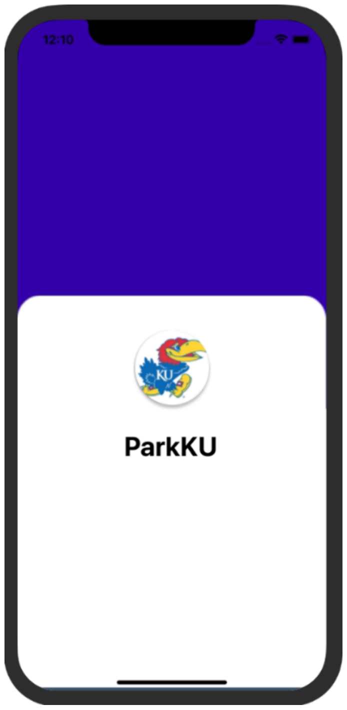
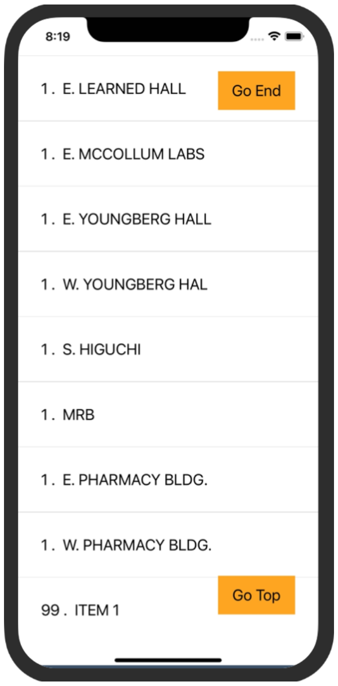


Fig5: Start Screen of the app



1 . E. LEARNED HALL
1 . E. MCCOLLUM LABS
1 . E. YOUNGBERG HALL
1 . W. YOUNGBERG HAL
1 . S. HIGUCHI
1 . MRB
1 . E. PHARMACY BLDG.
1 . W. PHARMACY BLDG.
99 . ITEM 1

Fig6: Screen that will list down all the lots and permits they require

This screen has the names of the lots but needs to be updated to include the color-coded permits and have the lots divided permit-wise.

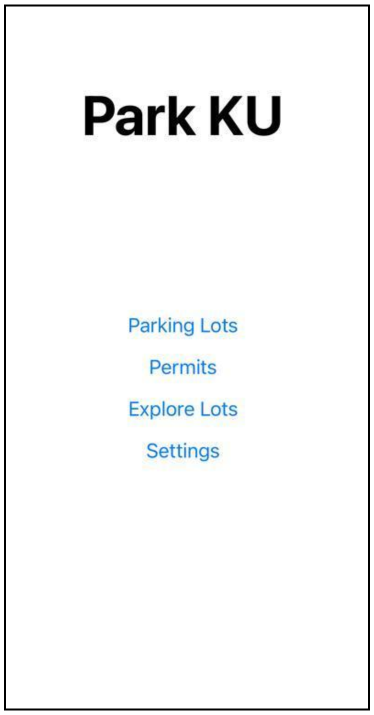


Fig7: Menu Screen

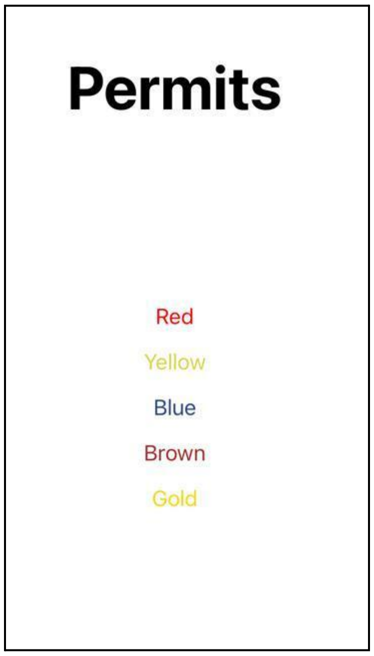


Fig8: Permits Screen

Worked on the recognition model and increased the epochs to 170. Ran the model over the video collected from our camera. The video is inside an engineering building. The model detects some cars for some reason inside, especially when the camera is being moved. But, we will have to debug the situation more. The camera still doesn't connect to the WIFI and we may need to take it to the IT so that they can help us connect it.

The validation error is going down which is a good indicator for the model accuracy. We will still keep working on it as we collect video from our camera.The test video has been uploaded on github.

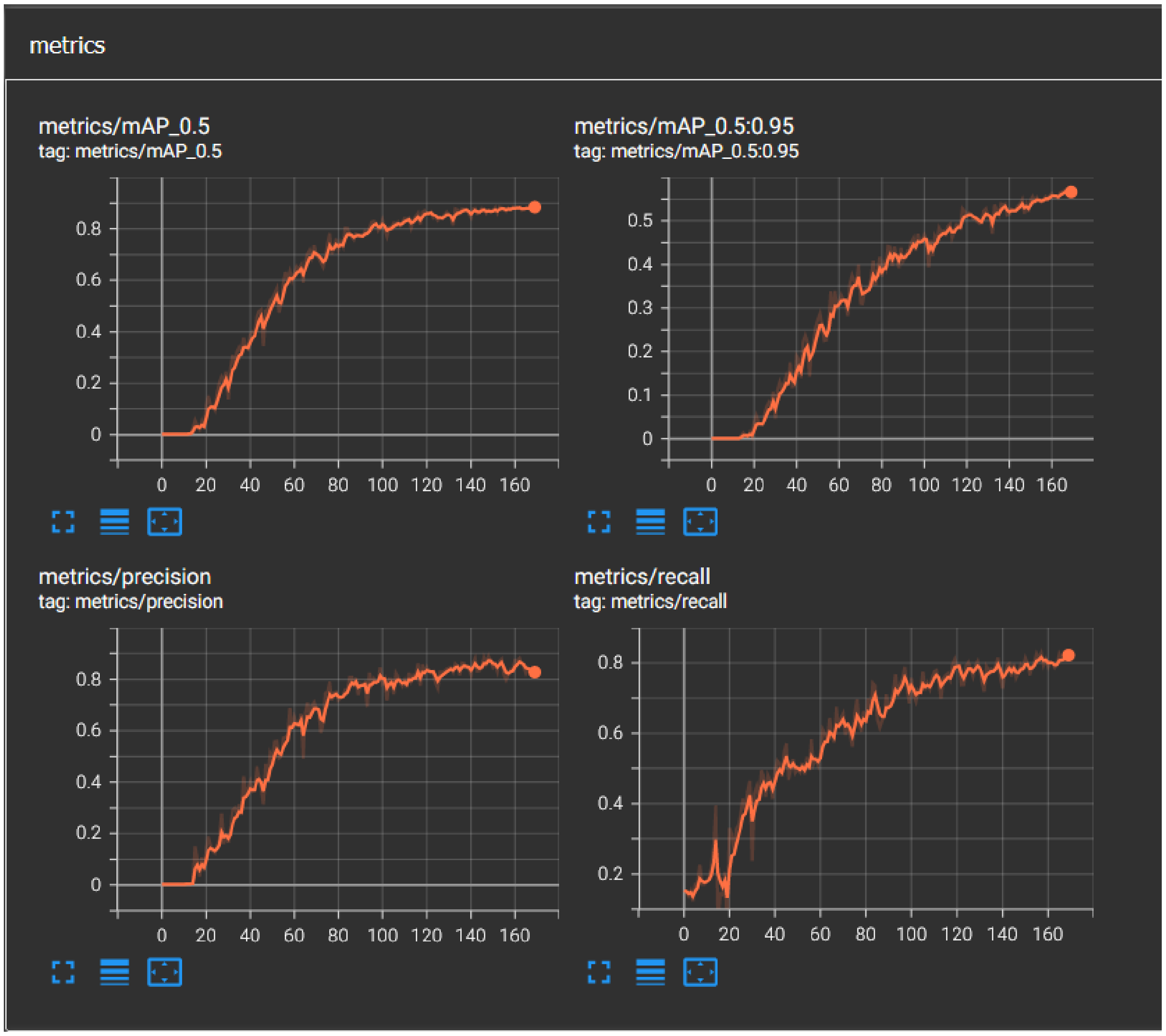


Fig9: Metrics on the model

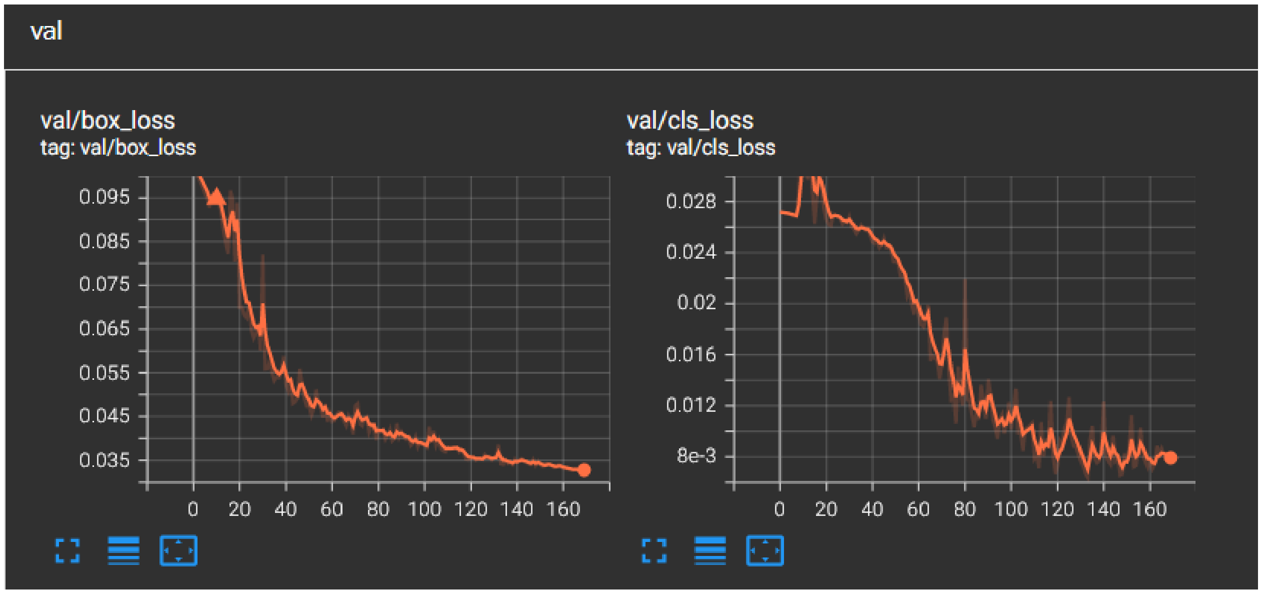
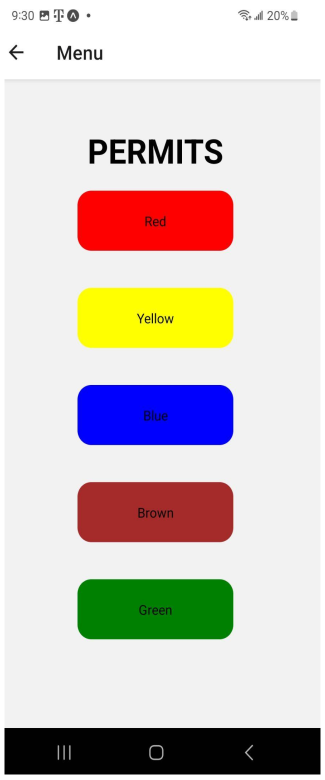
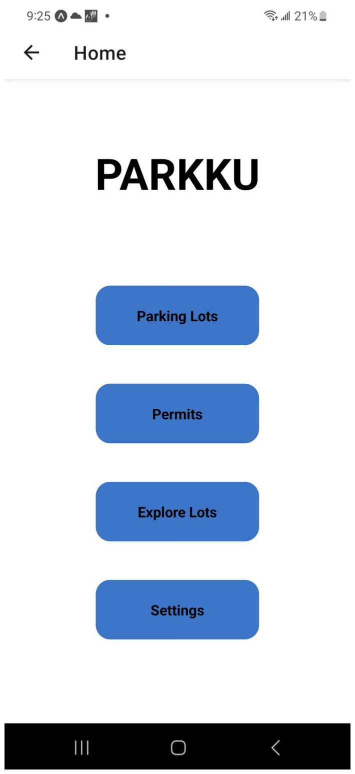


Fig9: Validation metric of the model

Requirements 34 and 43:





We have added the buttons to both the menu and permits screen. Navigation has been added to them too and from Menu, we can easi;y go to the permits screen by just hitting the permits button. For the Permits screen, we needed a little more work to do with the buttons since they required different colors. Each color represents the text inside it.

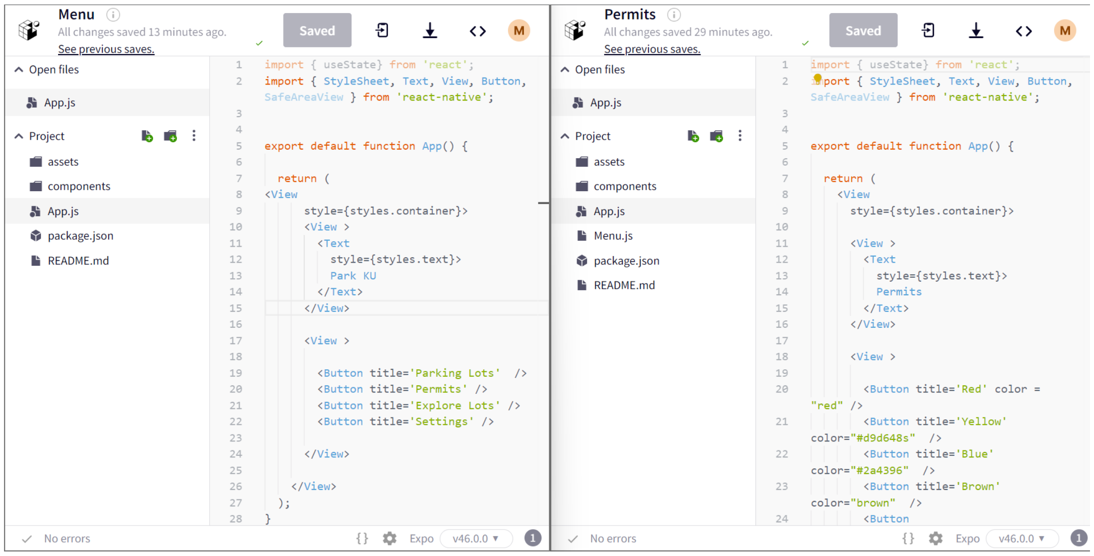
snippet of Permits screen code:

```
<View style = {styles.button}>

        <Pressable

        style = {[styles.pressable, {backgroundColor: "red"}]}

          onPress = {() => navigation.navigate('LotsList')}

          //onPressIn = {fontSize: 25 }

          android_ripple = {{color: 'LotsList'}}

          activeOpacity = {0.5}

        >

          <Text style = {styles.text}>

            Red

          </Text>


        </Pressable>


        <Pressable

      style = {[styles.pressable, {backgroundColor: "yellow"}]}

          onPress = {() => navigation.navigate('LotsList')}

          //onPressIn = {fontSize: 25 }

          android_ripple = {{color: 'green'}}

          activeOpacity = {0.5}

        >

          <Text style = {styles.text}>

            Yellow
          </Text>


        </Pressable>
```

snippet of menu screen code:

```
<View style = {{flex: 1, alignItems: 'center', justifyContent:
'space-evenly', backgroundColor: 'white'}}>

    <Text style = {{fontWeight: 'bold', fontSize: 45, textAlign: 'center'
}}> PARKKU </Text>


    <View>

      <Pressable

      style = {styles.pressable}

          onPress = {() => navigation.navigate('ParkingLots')}

          //onPressIn = {fontSize: 25 }

          android_ripple = {{color: 'green'}}

          activeOpacity = {0.5}

        >

          <Text style = {styles.text}>

            Parking Lots

          </Text>


        </Pressable>


        <Pressable

          style = {styles.pressable}

          onPress = {() => navigation.navigate('Permits')}

          //onPressIn = {fontSize: 25 }
          android_ripple = {{color: 'green'}}

          activeOpacity = {0.5}

        >

          <Text style = {styles.text}>

            Permits

          </Text>


        </Pressable>
```

Fig 10.0: Menu and Permit on expo snack emulator



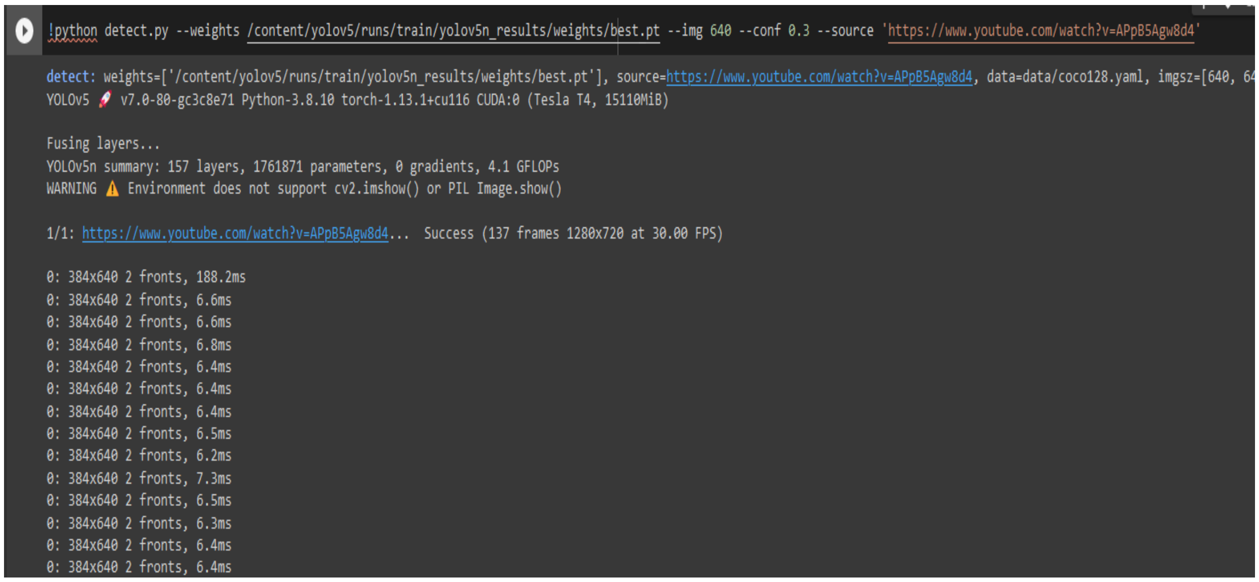
```
import { useState} from 'react';
import { StyleSheet, Text, View, Button,
SafeAreaView } from 'react-native';


export default function App() {

  return (
<View
      style={styles.container}>
      <View >
        <Text
          style={styles.text}>
          Park KU
        </Text>
      </View>

      <View >

        <Button title='Parking Lots'  />
        <Button title='Permits' />
        <Button title='Explore Lots' />
        <Button title='Settings' />

      </View>

    </View>
  );
}
```

```
import { useState} from 'react';
import { StyleSheet, Text, View, Button,
SafeAreaView } from 'react-native';


export default function App() {

  return (
    <View
      style={styles.container}>

      <View >
        <Text
          style={styles.text}>
          Permits
        </Text>
      </View>

      <View >

        <Button title='Red' color =
"red" />
        <Button title='Yellow'
color="#d9d648s"  />
        <Button title='Blue'
color="#2a4396"  />
        <Button title='Brown'
color="brown"  />
        <Button
```

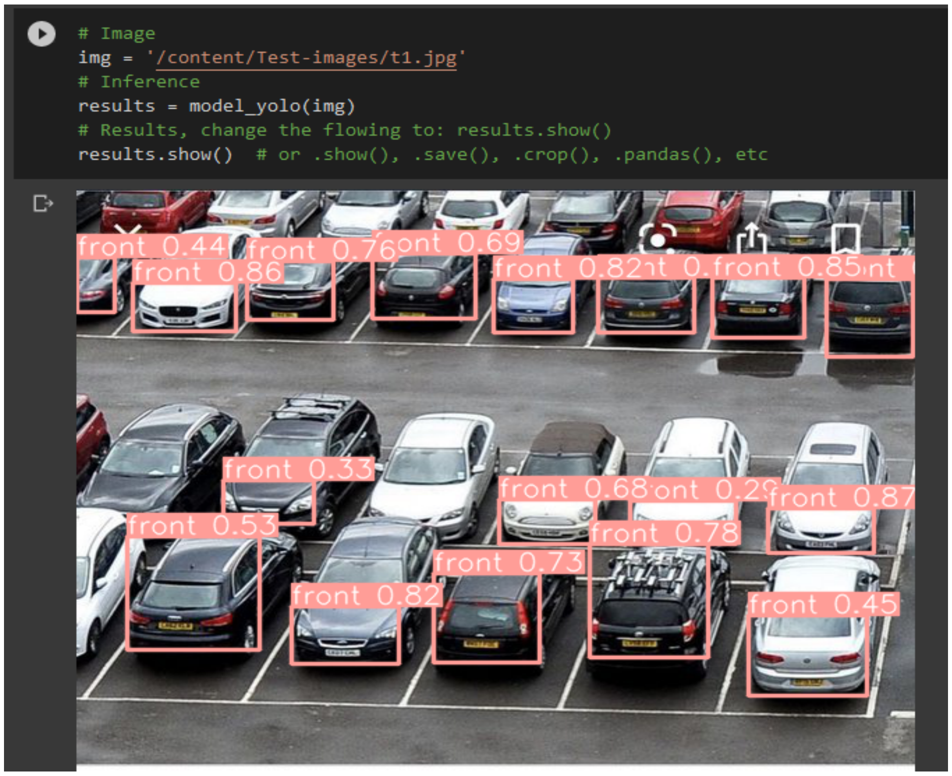
FIg 10.1: Menu and permit expo code snippet

Requirements 38 & 39

Our camera isn’t connected to the campus WIFI, and KU IT couldn’t help us further. We are moving forward with youtube videos for testing, we might use our camera to take videos outside engineering further.

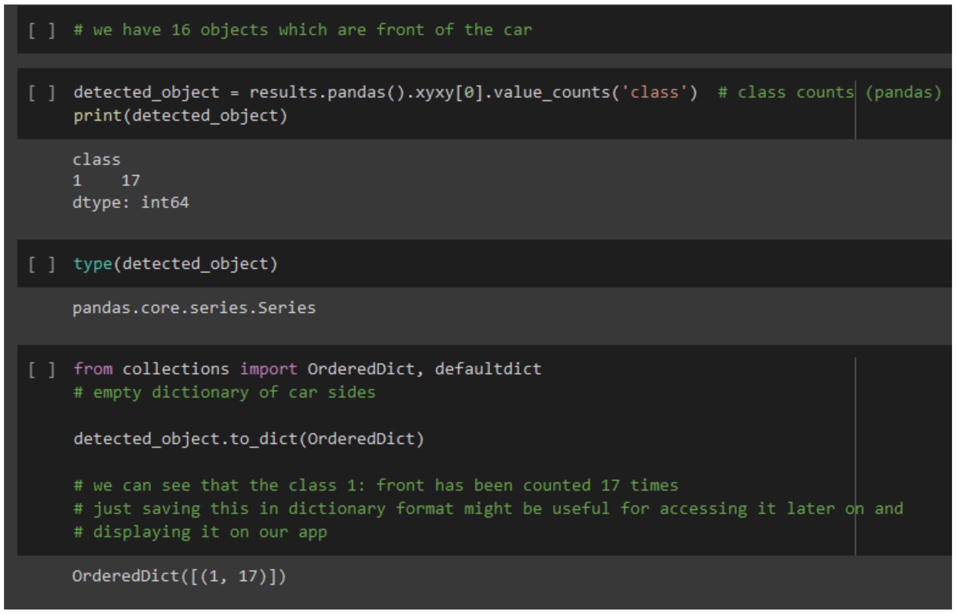


```
!python detect.py --weights /content/yolov5/runs/train/yolov5n_results/weights/best.pt --img 640 --conf 0.3 --source 'https://www.youtube.com/watch?v=APpB5Agw8d4'

detect: weights=['/content/yolov5/runs/train/yolov5n_results/weights/best.pt'], source=https://www.youtube.com/watch?v=APpB5Agw8d4, data=data/coco128.yaml, imgsz=[640, 64
YOLOv5 🚀 v7.0-80-gc3c8e71 Python-3.8.10 torch-1.13.1+cu116 CUDA:0 (Tesla T4, 15110MiB)

Fusing layers...
YOLOv5n summary: 157 layers, 1761871 parameters, 0 gradients, 4.1 GFLOPs
WARNING ⚠ Environment does not support cv2.imshow() or PIL Image.show()

1/1: https://www.youtube.com/watch?v=APpB5Agw8d4...  Success (137 frames 1280x720 at 30.00 FPS)

0: 384x640 2 fronts, 188.2ms
0: 384x640 2 fronts, 6.6ms
0: 384x640 2 fronts, 6.6ms
0: 384x640 2 fronts, 6.8ms
0: 384x640 2 fronts, 6.4ms
0: 384x640 2 fronts, 6.4ms
0: 384x640 2 fronts, 6.4ms
0: 384x640 2 fronts, 6.5ms
0: 384x640 2 fronts, 6.2ms
0: 384x640 2 fronts, 7.3ms
0: 384x640 2 fronts, 6.5ms
0: 384x640 2 fronts, 6.3ms
0: 384x640 2 fronts, 6.4ms
0: 384x640 2 fronts, 6.4ms
```



```
# Image
img = '/content/Test-images/t1.jpg'
# Inference
results = model_yolo(img)
# Results, change the flowing to: results.show()
results.show()  # or .show(), .save(), .crop(), .pandas(), etc
```

The team's app developers needed the number of sides being detected:



```
[ ] # we have 16 objects which are front of the car

[ ] detected_object = results.pandas().xyxy[0].value_counts('class')  # class counts (pandas)
    print(detected_object)

    class
    1    17
    dtype: int64

[ ] type(detected_object)

    pandas.core.series.Series

[ ] from collections import OrderedDict, defaultdict
    # empty dictionary of car sides

    detected_object.to_dict(OrderedDict)

    # we can see that the class 1: front has been counted 17 times
    # just saving this in dictionary format might be useful for accessing it later on and
    # displaying it on our app

    OrderedDict([(1, 17)])
```

Some of the demo videos are on Github:

https://github.com/hzahid99/EECS-Final-Project/tree/main/Demo

Note: I'm still re-running the model just so that I can come with an optimum number of epochs for training. However, our model is in good shape. I will start looking into the model deployment alongside collecting the parking video with the rest of the team in future.