| **PROJECT NAME** | CCTV-AI |
| --- | --- |
| **REPOSITORY** | cctv-ai-cv |
| **PROJECT TYPE** | AI (Python) |
| **DATE** | 30-08-2023 |
| **VERSION** | 1.0 |

Containerization

Ensure that docker-desktop or docker instance is running. **And run the following command at the root of the solution**

Image can be built using the following command:

**docker build -t cctv-ai .**

Note: Docker file is available on the root of the project location.

**Please make the changes in Dockerfile to communicate with Dedicated GPUs for training purposes.**

As this is not a web application, there is no entry point for docker at the moment. You just need to build the image and execute individual scripts to run the use cases.

UseCase Scripts locations

You can run the individual use cases using the following command while in the docker container:

**python <scipt.py> <camera\_ip>**

The script locations are given as:

**PPE Kit**: <project\_root>/PPE\_Kit/src/ppe\_kit\_detection.py

**Zone Intrusion**: <project\_root>/zone\_intrusion/src/zone\_intrusion\_detection.py

**Water Edge:** <project\_root>/water\_edge/src/water\_edge\_detection.py

**Waiting Area**: <project\_root>/waiting\_area/src/waiting\_area\_detection.py

**4 Vehicles Per Crane**: <project\_root>/four\_vehicles\_per\_crane/src/vehicles\_in\_close\_proximity.py

**People close to the moving objects**: <project\_root>/people\_close\_to\_moving\_objects/src/people\_close\_to\_moving\_objects.py

**People under suspended load**: <project\_root>/people\_under\_suspended\_load/src/people\_under\_suspended\_load.py

**No-entry**: <project\_root>/traffic\_rules/src/no\_entry\_detection.py

**Over-speeding**: <project\_root>/traffic\_rules/src/over\_speeding\_detection.py

**Wrong turn**: <project\_root>/traffic\_rules/src/wrong\_u\_turn\_detection.py

**Illegal Parking**: <project\_root>/traffic\_rules/src/illegal\_parking\_violation\_detection.py

Adding New Camera

In the current scenario, whenever a camera is added on the UI layer, it needs to be triggered manually for the use cases to run on the AI server. But before triggering it, a testing preset needs to be added. Go to the admin panel of the camera, Select **PTZ** from the left sidebar and then **PTZ Setup**. There would be an intractable camera view as well as already existing presets for that camera. After setting up the zoom, pan and tilt adjustment, select the preset number that is to be added. Now, enter the name of the preset. You can now verify the preset using **Video & Audio** from the sidebar and then **Camera setup.** The created preset should be shown onto that camera.

Camera Calibration

After creating the preset on the newly added camera, you need to **calibrate** the camera for the specific preset that you just created. The calibration is required to project the camera view on the **bird-eye view** or **heatmap**. For that purpose, you need to activate the docker container again and while in the container, you need to run the following command:

python helpers/calibrate\_with\_mouse.py <camera\_ip>

This will run the calibration script on the **DEFAULT PRESET** of the specified camera\_ip.

**Note**: We are currently dealing with only one preset per camera and we call that DEFAULT PRESET of the specific camera.

The script displays a static frame from the live feed of the specified camera. We need to select four points on the canvas, displaying the road in front of the camera or any part of the camera which you want to be projected onto the heatmap.

**Note**: The points should be selected in clockwise or anticlockwise, and not in any other random sequence.

After the points are selected, they are stored in a yml file under **conf** directory with the name   
  
**config\_birdview\_<camera\_ip>.mp4**

Next time when you will run any use case on that camera with preset, it is gonna be loading that specific yml against that camera for projecting the bird eye view.

Dataset Locations

**PPE-Kit:**

/media/AISTORE/IQRAFOLDER/Code/Transformed PPE Dataset-v3i

/media/AISTORE/IQRAFOLDER/Code/PPE-Kit-demo/dataset

**People-under-suspended-load:**

/media/AISTORE/IQRAFOLDER/Code/people-under-suspended-load/datasets/custom\_dataset

Zone-specific Usecases

* Zone Intrusion (unauthorised zone)
* Water Edge (zone for edge of water)
* Waiting Area (zone for waiting area)
* 4 Vehicles per Crane (zone allowing x number of objects)
* No Entry Zone (No entry zone)
* Wrong Turn (Zone for unallowed turns)
* Illegal Parking (Illegal parking zone)

Whenever any zone-specific use case is executed, a canvas appears on which the zone needs to be drawn. **Zone points** need to be selected in a specific order as in the calibration process.

Code Cleaning**:**

We have removed redundant code chunks that were commented or useless altogether. Moreover, we have also started handling edge cases. We would continue these as part of our ongoing task.