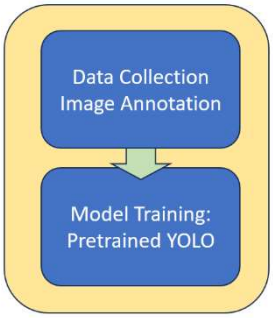
Machine Learning Model Development Process

First, we need our model to detect our objects. As we choose two similar bottles for this problem, we annotated some images as one class, “bottle”. Then we use the pretrained model YOLO8, and train it to detect this bottle.

Used bottles



The YOLO model can detect many classes, 80 class, including “bottle”. But it could not detect these bottles with the desired accuracy (98%), so we need to train our own model.

As we have relatively a small time for this challenge. We need to deal with it as fast as possible. Usually we use large dataset, 70% training, 15% validation, 15% testing, or something like this. But for this problem, we have used only 8 images with 13 object, containing our bottles. 11 object for training and 2 for validation. Mainly the videos were used for testing as we have no enough time for more labeling, and it works! At the end, Bottles could be detected with the desired accuracy. 150 epochs And we have a good trained model for this problem.

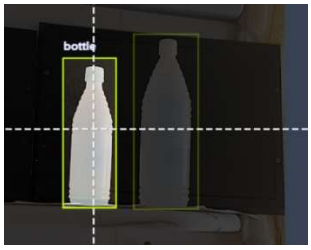


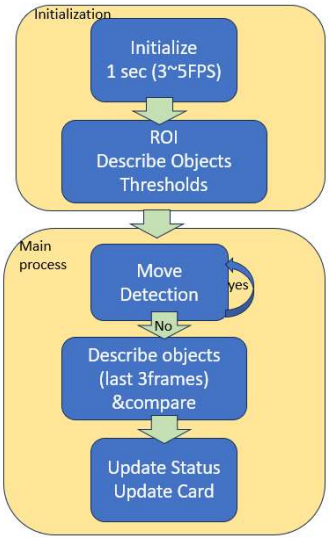
Image anntation

System design and architectural diagrams

The system is designed to deal with a video script for now, the camera should be fixed. Two bottles with different sizes should be used.

1 - Initialization:

At the start of video (1 second), the desired objects will be detected, and the region of interest ROI will be specified automatically. According to the desired objects those were founded, the thresholds will be stored. So if we have two bottles, we will detect the heights at the beginning, and the middle value for heights will be stored as a threshold.




```
shelf_bot  536 shelf_top  1193
Items in shelf: ['large bottle', 'small bottle']
initialization ended
```

2- Main Process:

So after initialization, we will have:

- List of objects, exa: [“large bootle”, “small bottle”].
- ROI, the shelf area.
- Thresholds of Height.

And the second part started.

Move detection: in ROI, move will be detected. If there is a move, “customer check”, we do nothing.

If there is no move, (move ended), we check the objects again and compare them to initial list of objects, and detect what the customer had bought or returned.

The trained model will detect bottles, and thresholds will recognize if it is large or small.

With every action, take or remove, the card state will be updated according to the data base.

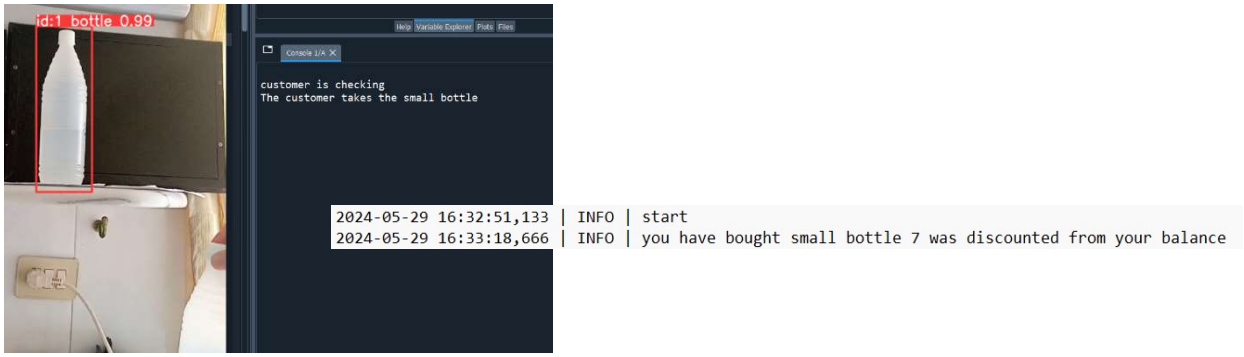
The database contains the bank account information, and the list of prices for the restaurant.

person_id	credit_num	bank	balance
115	11561761	boubyan	1500.86
120	51871561	boubyan	1455
125	71561861	boubyan	1200
130	54156411	boubyan	8000

product_id	product	price
1	bread	2
2	coffee_100g	8
3	coffee_200g	15
4	tea	10
5	small bottle	7
6	large bottle	10

The database will be updated immediately, according to action, and stored in the logs.

The logs will contains every detected action, and we can deal with in the desired way, like sending a notification to the user with every action.



Finally, a basket of the bought objects will appear to the user.

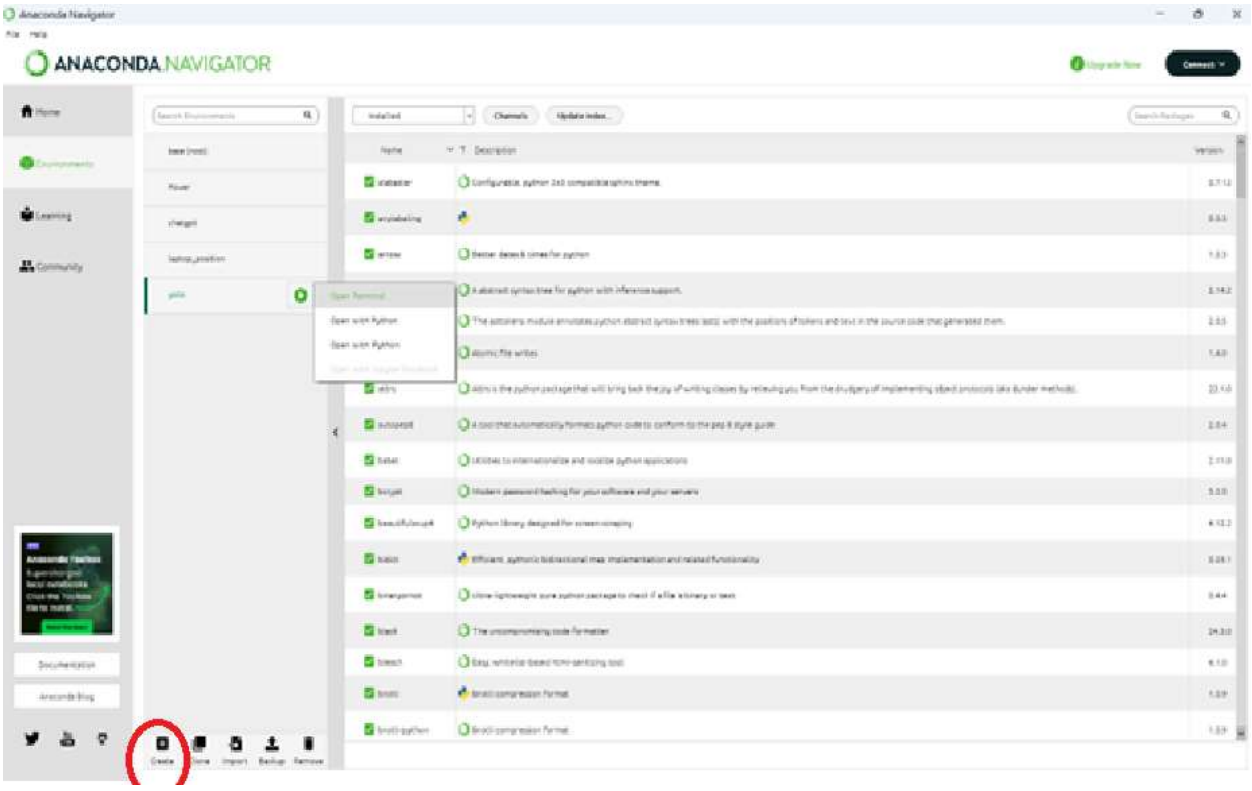
```
end of view

basket ['large bottle', 'small bottle']
```

How to run on Win11

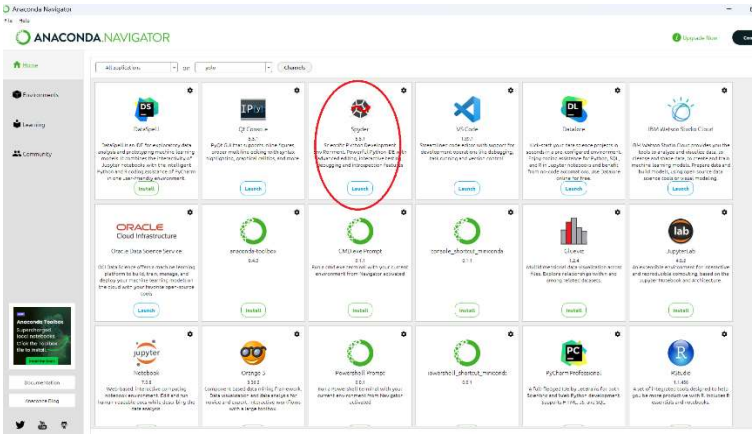
1) Download the project from github:
https://github.com/bashar-wannous/smart_shelf_challenge

2) you can use Anaconda software, or any python environment.
download the software, create new environment.
3) Basically you will need the following libraries:

ultralytics
cv2
numpy
pandas
flask
or you can go to environment terminal as in the photo, and use the following command:
pip install -r /path/to/requirements.txt
where requirements.txt is the file included in the project. Be careful about the right path.



4) Download Spyder



5) From spyder, open local_paths.py, set the paths for the files as it is in your working directory. Set the paths for the csv files, model, video , logfile.
6) Go to smart_shelf_v2.py and run!
7) If you want to check user frontend, “flask run”, from environment terminal, desired path.