This simple project uses a pre-trained object detection model to perform real-time inference on webcam input. It captures frames using OpenCV and runs detection with a PyTorch model.

⚙ Technologies

- Python 3.x
- PyTorch
- OpenCV
- [Model name: YOLOv5]

Make sure your webcam is accessible (e.g., /dev/video0 on Linux).

🦉 Output

The application opens a window showing the live webcam feed with detected objects highlighted.

🐏 2. Load the model (Code cell)

python
from yolov5 import YOLOv5
model = YOLOv5.load('yolov5s.pt') # Replace with your actual loading method

Explanation:
Here, a pre-trained YOLOv5 model is loaded to perform object detection. The model can be loaded locally or via a library like ultralytics.

✅ Clearly state how the model is loaded, and where to get the .pt file.

🐿 3. Define detect_from_webcam() (Code cell)

python
import cv2
import numpy as np

def detect_from_webcam():
    cap = cv2.VideoCapture(0, cv2.CAP_V4L2) # Use CAP_V4L2 for Linux

```
    if not cap.isOpened():
        print ("Error: Cannot access the webcam.")
        return

    while True:
        ret, frame = cap.read()
        if not ret:
            print ("Error: Failed to capture frame.")
            break

        results = model(frame) # Run inference

        rendered_img = np.array(results.render()[0]) # Convert to OpenCV format

        cv2.imshow("Live Detection", rendered_img)

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    cap.release()
    cv2.destroyAllWindows()
```

Explanation:
This function continuously reads frames from the cam, passes them to the model, and displays the output in a window with bounding boxes.