■ Violation Detection System – Documentation

■ Purpose

This project is a Violation Detection System built using Flask, YOLOv8, and Supabase. It allows users to upload images or videos, runs real-time object detection on them, and highlights any detected objects as violations. The results are stored in Supabase for tracking and auditing.

■ Features

- Detection with YOLOv8 (images & videos supported) - Red bounding boxes with labels on detected objects - Stores processed files locally and in Supabase Storage - Logs violation details in Supabase database table - Provides REST API endpoints for health check and analysis

■ API Endpoints

1. GET /health → Returns system and model status 2. POST /analyze → Accepts an image or video, runs detection, returns annotated results 3. GET /results/ → Serves processed files from results directory

■■ How It Works

1. User uploads an image/video → Flask endpoint `/analyze` 2. YOLO model analyzes the file 3. Detections highlighted (if any) 4. Processed file saved locally and uploaded to Supabase Storage 5. Metadata inserted into Supabase database table `violations` 6. API responds with violation status, file URL, and timestamp

■ Requirements

- Python 3.9+ - Flask, Flask-CORS - Python-dotenv - Supabase-py - Ultralytics (YOLOv8) - OpenCV, NumPy

■ Environment Variables

SUPABASE_URL=your_supabase_url
SUPABASE_SERVICE_ROLE_KEY=your_supabase_service_key
SUPABASE_BUCKET=violations YOLO_WEIGHTS=yolov8n.pt PORT=5001

■ Example Workflow

1. Start server: python app.py 2. Check health: GET http://localhost:5001/health 3. Upload file: POST http://localhost:5001/analyze (form-data: file=) 4. Receive response with detection result and Supabase file URL