

# Real-Time Number Plate Detection System

A comprehensive overview of our real-time vehicle number plate detection system.



# Introduction

This document details the design, development, and implementation of a real-time vehicle number plate detection system using computer vision and deep learning. It identifies and reads license plates from live video, storing them in a local database. Suitable for traffic surveillance, parking management, and security checkpoints.

# Project Objectives



## Real-time Detection

Detect license plates from video streams.



## Text Extraction

Extract readable text using OCR.



## Data Storage

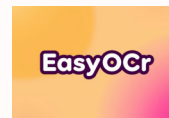
Store plate numbers and timestamps in a database.



## Efficiency

Develop an accurate, lightweight, and efficient system.

# Tools & Technologies



# System Architecture



# Implementation & Results

## Implementation

Step-by-step setup including package installation, model loading, database creation, and real-time detection loop using OpenCV and EasyOCR.

## Results

- Successfully detected and read plates in real-time.
- Satisfactory accuracy for frontal/near-frontal plates.
- Stored logs allowed easy review of captured data.



# Applications



## Smart Parking

Entry/exit tracking for parking systems.



## Traffic Enforcement

Law enforcement and surveillance.



## Toll Booths

Automation of toll collection.

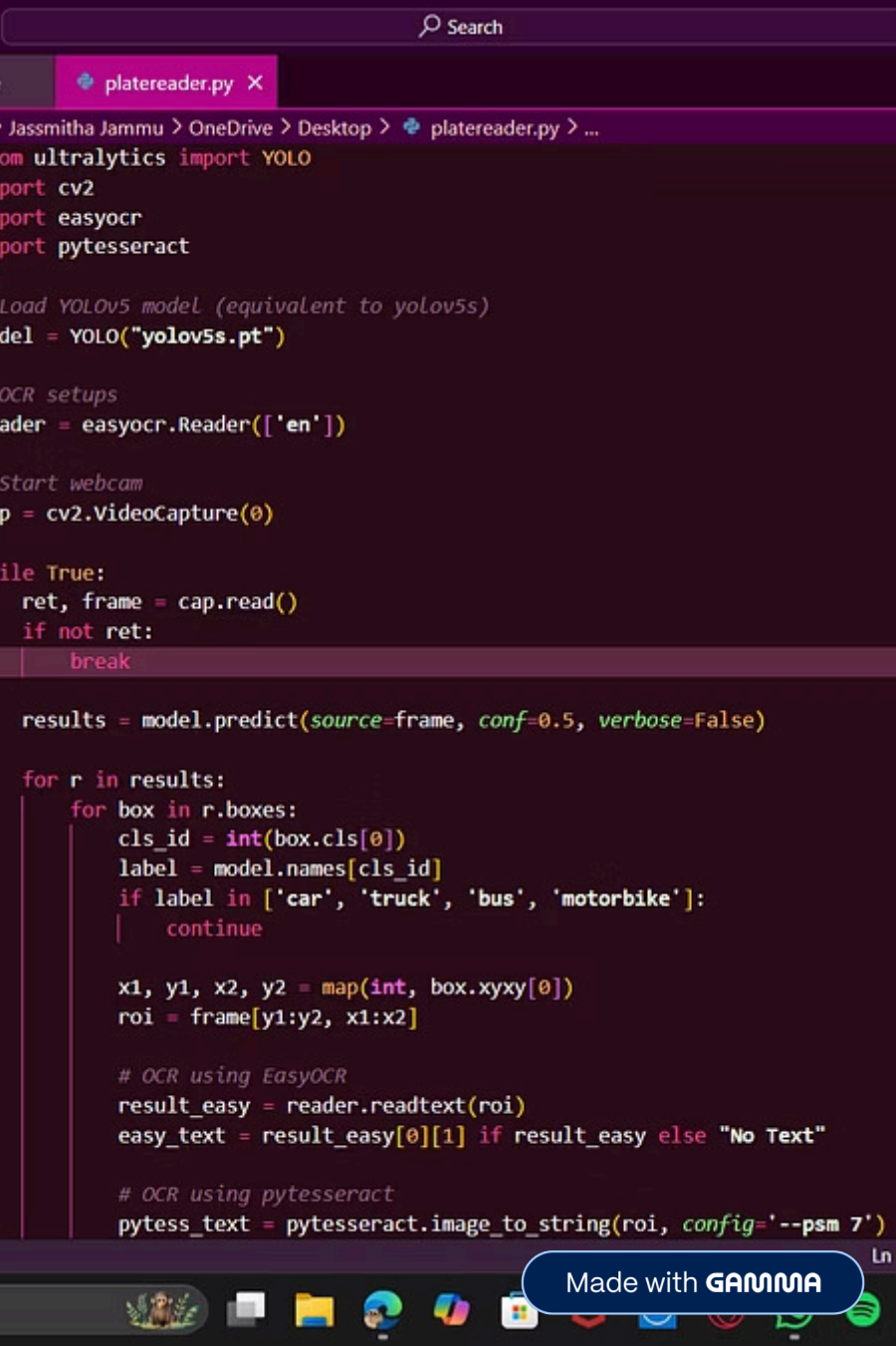


## Gate Security

Enhanced security at private premises.

# Future Work

- Integration with cloud database for centralized tracking.
- Night vision compatibility using IR imaging enhancements.
- GUI dashboard using Flask or Streamlit.
- Alert system for blacklisted vehicle detection.



The screenshot shows a code editor window titled 'platereader.py'. The code is written in Python and includes comments in a lighter font. The code imports 'YOLO' from 'ultralytics', 'cv2' from 'cv2', 'easyocr' from 'easyocr', and 'pytesseract' from 'pytesseract'. It loads a YOLOv5 model, sets up an EasyOCR reader for English, and starts a webcam. The main loop reads frames from the webcam, predicts objects using the YOLO model, and then uses EasyOCR and pytesseract to read text from the detected regions of interest (ROIs). The code is as follows:

```
platereader.py X
Jassmitha Jammu > OneDrive > Desktop > platereader.py > ...
from ultralytics import YOLO
import cv2
import easyocr
import pytesseract

Load YOLOv5 model (equivalent to yolov5s)
model = YOLO("yolov5s.pt")

OCR setups
reader = easyocr.Reader(['en'])

Start webcam
cap = cv2.VideoCapture(0)

while True:
    ret, frame = cap.read()
    if not ret:
        break

    results = model.predict(source=frame, conf=0.5, verbose=False)

    for r in results:
        for box in r.boxes:
            cls_id = int(box.cls[0])
            label = model.names[cls_id]
            if label in ['car', 'truck', 'bus', 'motorbike']:
                continue

            x1, y1, x2, y2 = map(int, box.xyxy[0])
            roi = frame[y1:y2, x1:x2]

            # OCR using EasyOCR
            result_easy = reader.readtext(roi)
            easy_text = result_easy[0][1] if result_easy else "No Text"

            # OCR using pytesseract
            pytess_text = pytesseract.image_to_string(roi, config='--psm 7')
```

Made with GAMMA



# Conclusion

This project demonstrates an effective, real-time solution for automatic number plate recognition using deep learning and computer vision. Its modular design and lightweight components make it a practical base for deployment in smart traffic and surveillance applications.

**Submitted by:** Jassmitha Jammu, CSE - AIML, Gitam University, Roll No: 2023000012