

# Accurate Detection and Recognition of Dirty Vehicle Plate Numbers for High-Speed Applications

Jitendra Singh

M. Tech PES University

# Agenda

- 1. Introduction & Problem Statement
- 2. Literature Review
- 3. Proposed Methodology
- 4. Novelty & Contribution
- 5. Gap Analysis
- 6. Timeline & Work Plan
- 7. Conclusion & Future Scope

# Introduction & Problem Statement

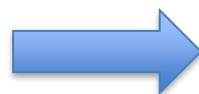
- - Importance of License Plate Recognition (LPR)
- - Challenges:
  - • Dirt/Mud/Obstructions
  - • Motion blur at high speeds
  - • Reduced recognition accuracy → fraud & enforcement gaps
- - Goal: Robust recognition of dirty/obscured plates at high speeds.

# Literature Review

- Existing ANPR Systems: Fail under dirt/motion blur
- OCR Optical Char Recognition (Tesseract, CNN): Reduced accuracy on noisy images
- GAN Generative Adversarial Networks-based Augmentation: Limited real-world use
- Conclusion: Need for robust dirty plate recognition.

# Proposed Methodology

- 1. Image Acquisition → Kaggle datasets
- 2. License Plate Detection → YOLO
- 3. Object Segmentation → Isolate plates
- 4. Character Recognition → Tesseract/CNN
- 5. Fraud Detection → Repeated failures flagged



YOLO



Gets Number  
Plat from Car



CNN Tesseract  
OCR



GY59KLL

Try to detect  
Actual Number



# Algorithms to Explore

- • YOLO → License Plate Detection
- • Tesseract OCR → Baseline Recognition
- • Custom CNN → Dirty plate recognition

# Novelty / Contribution

- - Dirty Plate Simulation → Image augmentation
- - Custom CNN for OCR → For noisy/dirty plates
- - Fraud Detection Layer → New addition
- - Real-time Suitability → Optimized for highways



# Gap Analysis

- Existing Systems:
  - - Good for clean plates
  - - Poor dirty plate accuracy
  - - Minimal fraud detection
- Proposed Work:
  - - Dirty plate accuracy improved
  - - Dedicated fraud detection
  - - Real-time, high-speed suitable

# Timeline of Proposed Work

- Phase 1 (Current) → Literature Review, Dataset Collection
- Phase 2 → Model Design & Training (YOLO & OCR)
- Phase 3 → Testing & Validation
- Phase 4 → Final Integration & Report Writing

# Conclusion & Future Scope

- - Robust LPR for dirty/high-speed vehicles
- - Future Scope:
  - Real-time system integration
  - Edge AI deployment
  - GAN-based preprocessing exploration

# Suggestions from Mentor REVIEW 2

- Grounding concept Visual LLM Visual Matric
- Which model does better on grounding
- Augmentation is needed classification for mud and dirty
- No further processing
- Visual models are doing really good text out of image
- Get data set and compare accuracy
- Image correction techniques
- Identify the data sets

# Data Sets

- <https://github.com/detectRecog/CCPD>
- <https://www.kaggle.com/datasets/saisirishan/indian-vehicle-dataset>
- We have added MP4 Vdos as inputs
- Usually criminals are captured in vdos

# Steps

- Step one : we will use Yolo / other models can be checked for accuracy
- Step 2 : we will use PaddleOCR or geminiv11 for Visual LLM capabilities

# Updates for review 2

- Develop portal using flask
- Database to be used Mysql
- Mp4 processing to be included
- Data to be kept in github repo
- <https://github.com/j33tu/mtechpes25.git>
- working model to be presented
- Compare multiple model for accuracy

# Front end

## Upload Video for Plate Detection

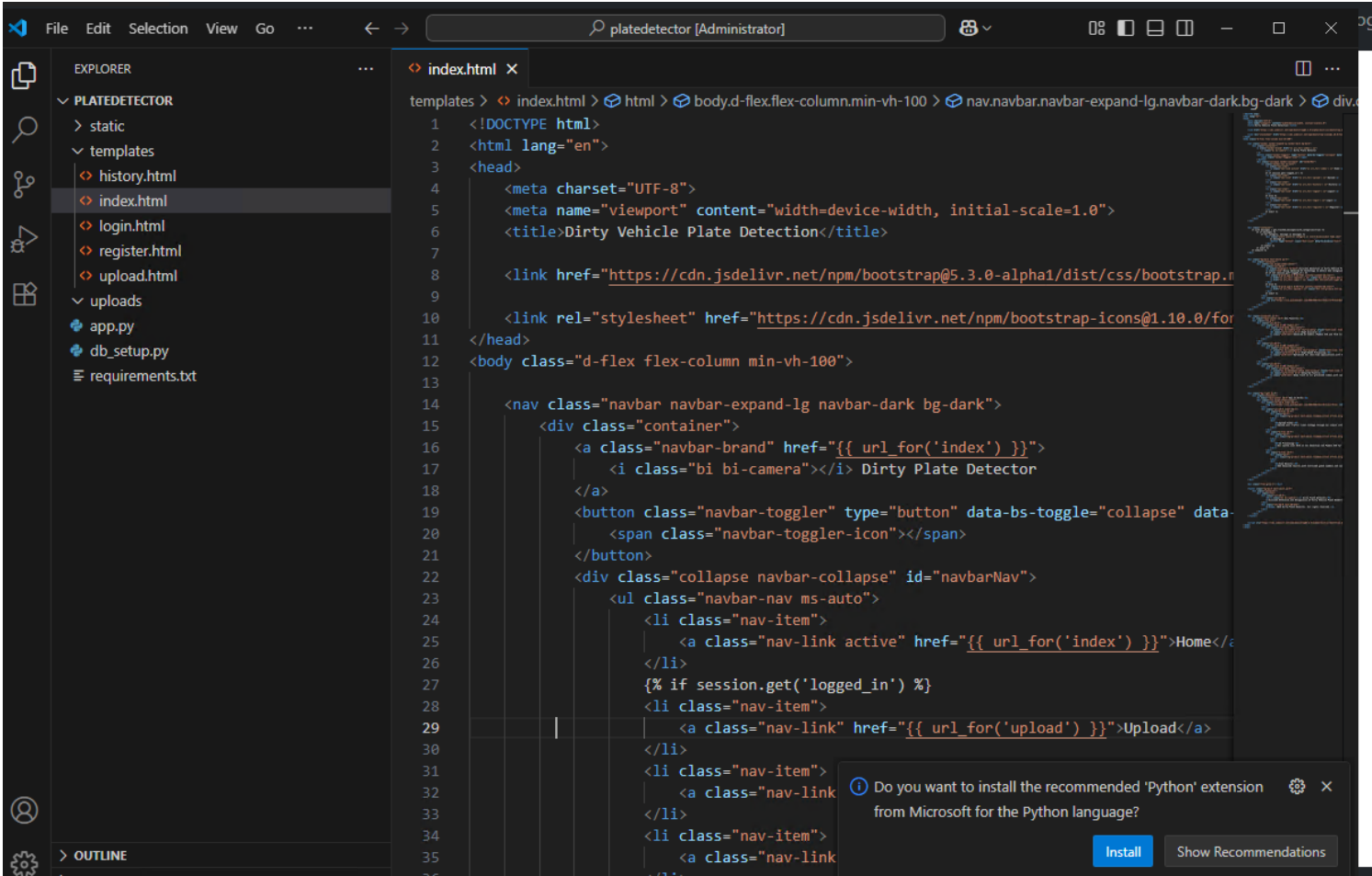
Select Video File

Choose file No file chosen

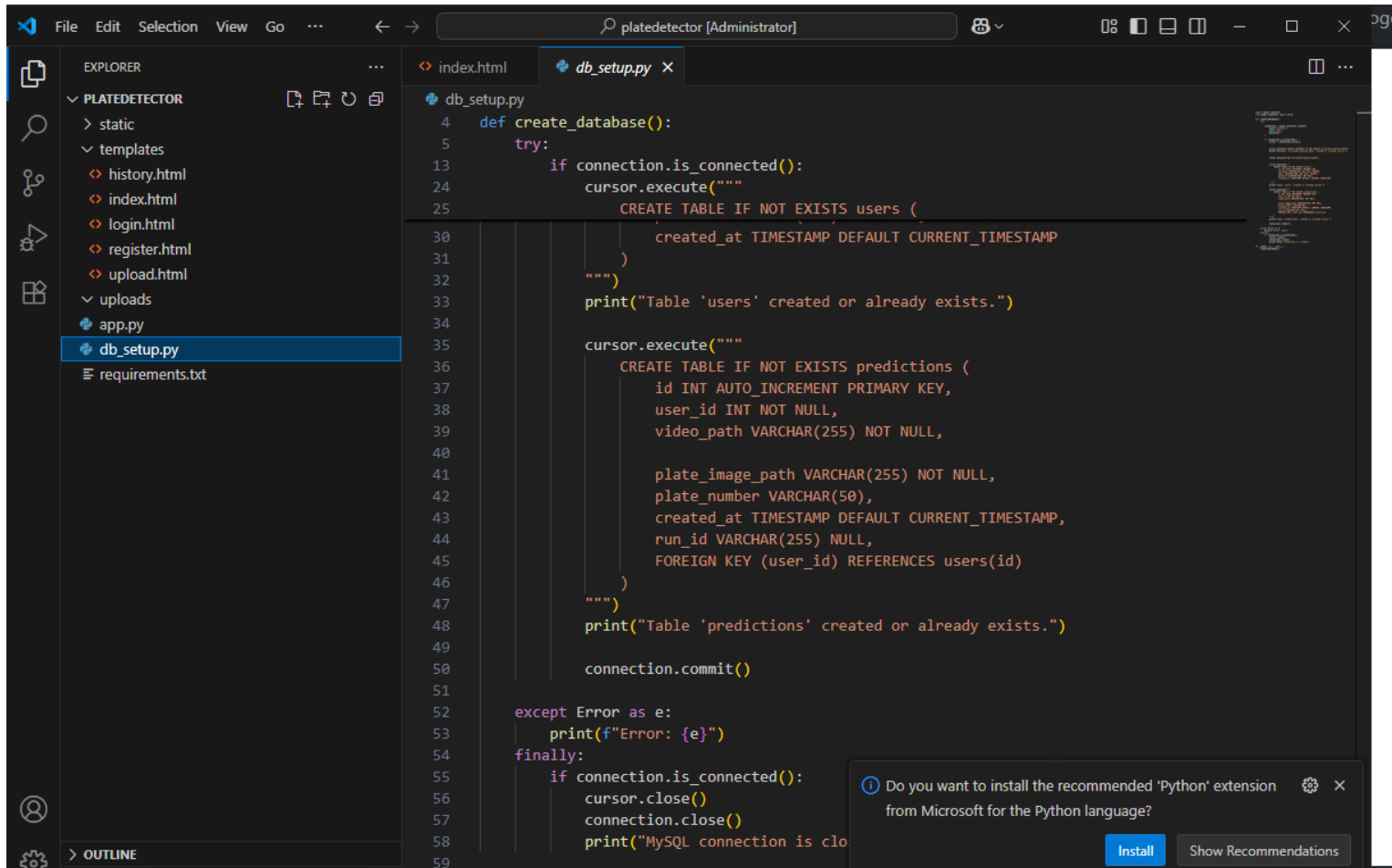
Upload and Process



# Front end code



# Database mysql



The screenshot shows a Visual Studio Code editor window with the file explorer on the left and a code editor in the center. The file explorer shows a project named 'PLATEDETECTOR' with subfolders 'static' and 'templates', and files 'history.html', 'index.html', 'login.html', 'register.html', 'upload.html', 'uploads', 'app.py', 'db\_setup.py', and 'requirements.txt'. The 'db\_setup.py' file is selected. The code editor shows the following Python code:

```
4 def create_database():
5     try:
13         if connection.is_connected():
24             cursor.execute("""
25                 CREATE TABLE IF NOT EXISTS users (
30                     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
31                 )
32             """)
33             print("Table 'users' created or already exists.")
34
35             cursor.execute("""
36                 CREATE TABLE IF NOT EXISTS predictions (
37                     id INT AUTO_INCREMENT PRIMARY KEY,
38                     user_id INT NOT NULL,
39                     video_path VARCHAR(255) NOT NULL,
40
41                     plate_image_path VARCHAR(255) NOT NULL,
42                     plate_number VARCHAR(50),
43                     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
44                     run_id VARCHAR(255) NULL,
45                     FOREIGN KEY (user_id) REFERENCES users(id)
46                 )
47             """)
48             print("Table 'predictions' created or already exists.")
49
50             connection.commit()
51
52     except Error as e:
53         print(f"Error: {e}")
54     finally:
55         if connection.is_connected():
56             cursor.close()
57             connection.close()
58             print("MySQL connection is clo
59
```

A notification dialog box is visible in the bottom right corner, asking: "Do you want to install the recommended 'Python' extension from Microsoft for the Python language?". It has "Install" and "Show Recommendations" buttons.