BỘ THÔNG TIN VÀ TRUYỀN THÔNG

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Final Report

Foundation Internship

Project Title: Traffic License Recognition

Instructor:	Kim Ngoc Bach
Student Name :	Bui Xuan Hai
Student ID :	B22DCAT105
Class :	E22CQCN05-B

Hanoi - 2025

Table of contents

I. Project Introduction **2**
1. Key Features 3
2. Technologies Used 3
3.Reason for choosing this topic 3
4. Objectives and Scope of the Research 4
II. Foundation Knowledge **6**
1.Web Development Platforms 6
1.1 Bootstrap 6
1.2 FastAPI 7
2. AI Technologies 7
2.1 Image Processing 7
2.2 YOLOv5,v8 8
2.3 PaddleOCR 10
III. Project Management Report **11**
1. Requirement Analysis 11
a. Website Development Requirements 11
b. Recognition System Requirements 14
2.System Analysis and Design 17
b.Implementation 19
1. Backend API 19
2.Front-end 20
3.Project Schedule Management 24
4.Results of the project 25
IV. Conclusion **26**
1.Advantages: 26
2.Disadvantages: 27
3.Summary 27
V. References **27**

INTERNSHIP BASE REPORT

I. Project Introduction

This project aims to develop a web application that supports automatic recognition of vehicle license plates from uploaded images or videos. The system uses a deep learning model (trained by myself) to detect and recognize license plate numbers. This can be applied in real-life scenarios such as entry/exit management in parking lots or garages.

Users can interact with the system via a web interface, upload photos or videos, and the system will display the detected license plates. The backend is implemented using FastAPI, while the frontend is designed using HTML, CSS, and JavaScript (with optional integration of Bootstrap).

1. Key Features

- Allow users to upload vehicle images or videos.
- Automatically detect and recognize license plate numbers.
- Display recognition results in a user-friendly format.
- Possibility to store recognition history or export results.
- Designed for application in garage or parking management systems.

2. Technologies Used

- **Frontend**: HTML, CSS, JavaScript (Bootstrap).
- **Backend**: FastAPI (Python).
- **AI Model**: YOLO-based license plate detection + OCR model (e.g., PaddleOCR or custom model).

- **Storage**: Local server and database (MySQL)

3.Reason for choosing this topic

I chose this topic to deepen my expertise in artificial intelligence, particularly in computer vision, and to acquire new technical skills. By working on a license plate recognition system, I can gain hands-on experience with state-of-the-art models like YOLOv8 and PaddleOCR, enhancing my understanding of model training, inference optimization, and OCR integration. At the same time, I aim to develop a practical application that delivers real-world value—an automated solution for vehicle identification that can improve efficiency in parking and traffic management.

4. Objectives and Scope of the Research

The objective of this research is to design, implement, and evaluate a web-based license plate recognition (LPR) system that leverages state-of-the-art object detection and OCR technologies to automate vehicle identification in parking and traffic management scenarios. Specifically, the study aims to:

- Apply a YOLOv8 model on a Vietnamese license-plate dataset to achieve robust plate localization under varying lighting conditions and camera angles.
- Integrate PaddleOCR on the cropped plate images to reliably extract alphanumeric characters with high accuracy.

- Develop a FastAPI backend that orchestrates the end-to-end inference pipeline—receive uploaded images/video, run YOLOv8 detection, crop detected regions, perform OCR, and return results via a RESTful API.

- Build a responsive HTML/CSS/JavaScript (Bootstrap) frontend that allows users to upload photos or stream webcam input, displaying both the cropped plate image and recognized text with confidence scores in real time.

The scope of this research focuses on:

- Collecting and annotating a representative dataset of Vietnamese vehicle images and videos.
- Training and optimizing YOLOv8 for license-plate detection, experimenting with hyperparameters (learning rate, batch size, input size).
- Applying PaddleOCR preprocessing techniques (rotation correction, contrast enhancement) to improve character recognition on crops.
- Designing and implementing a modular FastAPI service to handle image uploads, inference requests, and database logging of recognition results.
- Creating a simple, user-friendly web interface for file upload, displaying detection overlays, and presenting recognition outputs.
- Evaluating system performance using detection metrics (mAP), OCR accuracy, and end-to-end inference latency under real-world conditions.

By addressing these areas, the project seeks to demonstrate how modern deep-learning models can be combined and deployed in a practical web application, offering a scalable solution for automated license-plate recognition in Vietnamese parking or toll-gate environments.

II. Foundation Knowledge

1.Web Development Platforms

This section explores the key technologies and programming languages that are instrumental in the development of websites, detailing their general benefits and applications.

1.1 Bootstrap

Bootstrap is a widely-adopted open-source CSS framework designed to streamline the development of responsive, mobile-first web interfaces. Originating from Twitter, Bootstrap offers a comprehensive set of predefined classes and components—such as its 12-column grid system, navigation bars, form controls, and utility helpers—that enable developers to construct consistent layouts and interactive elements without writing extensive custom CSS. By leveraging its grid and flexbox utilities, Bootstrap ensures that web pages automatically adjust to various screen sizes, improving user experience on desktops, tablets, and smartphones alike.

Beyond its core layout utilities, Bootstrap includes a rich library of reusable UI components—buttons, cards, dropdowns, modals, and more—each styled with a cohesive design language. This component-based approach simplifies code maintenance and accelerates development, as developers can compose complex interfaces by combining and customizing these building blocks. Additionally, Bootstrap’s JavaScript plugins (powered by Popper.js) provide interactive behaviors—such as dropdown toggles and responsive navbars—without requiring third-party scripts. As a result, Bootstrap serves as an ideal foundation for creating clean, user-friendly frontends, enabling rapid prototyping and consistent design across web applications.

1.2 FastAPI

FastAPI is a modern, fast (asynchronous) web framework for building APIs with Python, based on standard Python-type hints. It is designed for performance and ease of use, with automatic generation of OpenAPI documentation and support for asynchronous programming, making it ideal for handling real-time data or high-demand applications . One of the most significant advantages of FastAPI is its performance, rivaling frameworks like Node.js and Go. FastAPI’s use of Python-type annotations leads to automatic request validation and documentation, greatly reducing the amount of boilerplate code developers need to write. In web applications, particularly those that involve backend data handling (e.g., login systems, chatbots), FastAPI’s efficiency ensures faster response times and improved scalability, critical for delivering a responsive user experience.

2. AI Technologies

2.1 Image Processing

Image processing begins by acquiring input—either static images or video frames—and standardizing them into a consistent RGB format and resolution. Preprocessing techniques such as histogram equalization and Gaussian blur are applied to enhance contrast and reduce noise, ensuring license plates stand out under varied lighting conditions. For each input, YOLOv8 then predicts bounding boxes around candidate plate regions; these normalized coordinates are converted to pixel values and used to crop the exact region of interest (ROI). Optionally, a simple perspective correction can be applied when plates appear skewed, which aids in more accurate character recognition.

Once the ROI is extracted, it is resized to a fixed dimension (e.g., 240×80 pixels) and converted to grayscale for OCR readiness. Basic illumination adjustments—like gamma correction—and adaptive thresholding may be used to improve character visibility before passing the cropped image to PaddleOCR. In video scenarios, frame sampling (e.g., processing every 5th frame) and lightweight tracking reduce redundant detection calls, allowing the system to maintain near-real-time performance while minimizing computational load.

2.2 YOLOv5,v8

YOLO (You Only Look Once) is a family of real-time object detection models that process an image in a single forward pass, predicting bounding boxes and class probabilities directly from full images. YOLOv5 introduced significant improvements over its predecessors, leveraging a PyTorch implementation, a CSPDarknet-based backbone, and a PANet-like neck for multi-scale feature aggregation. Its architecture balances speed and accuracy by using efficient convolutional modules (Focus, BottleneckCSP) and optimized anchors, making it suitable for tasks that require rapid inference on limited hardware. YOLOv5 models come in different sizes (s, m, l, x) to accommodate various performance requirements, and include built-in auto-learning of augmentation strategies (mosaic, mixup) and hyperparameter evolution routines that simplify training on custom datasets.

YOLOv8 builds further on these foundations by adopting a more modular design, enhanced data augmentation, and improvements inspired by recent research in object detection. Its backbone uses a modified CSP approach with optimized convolution patterns to reduce latency, while the head leverages decoupled prediction (separate branches for classification and localization) to improve localization precision. YOLOv8 also integrates advanced training techniques—such as distribution-aware label assignment (SimOTA) and dynamic resizing during training—to increase robustness under varying object scales and aspect ratios. Additionally, YOLOv8 provides out-of-the-box support for both bounding-box detection and segmentation tasks within a unified framework. By streamlining the training pipeline, enhancing augmentation policies, and refining network modules, YOLOv8 achieves higher mean Average Precision (mAP) and faster inference speed compared to YOLOv5, making it an ideal choice for applications that demand state-of-the-art accuracy in real-time scenarios.

The Confidences scores = P(Object) x P(Class|Object) with:

P(Object)

- ○ The probability that the bounding box contains an object (license plate) instead of the background.
- ○ Computed from the output of the YOLOv8 network, based on the image features in the box region.

P(Class|Object):

- The conditional probability that an object in a box belongs to a particular class (license_plate).
- Computed from the class probabilities in the network output (usually via a softmax or sigmoid function).

2.3 PaddleOCR

PaddleOCR is an open-source optical character recognition toolkit developed by Baidu, designed for high accuracy and efficiency across multiple languages and scripts. It follows a two-stage pipeline: text detection and text recognition. In the detection stage, a lightweight text-detection model (such as DBNet) locates text regions by producing pixel-level probability maps, then extracts bounding boxes around each text instance. In the recognition stage, the cropped text regions are passed to a convolutional recurrent neural network (CRNN) or a transformer-based recognition model, which decodes the sequence of characters via a CTC (Connectionist Temporal Classification) loss or attention mechanism.

Key features of PaddleOCR include:

- Multilingual Support: Built-in models for English, Chinese, and several other languages, with the ability to fine-tune on custom scripts.
- Angle Classification: An optional module that predicts text orientation (0°, 90°, 180°, 270°), automatically rotating skewed crops to improve recognition accuracy.
- Lightweight Architecture: Detection and recognition networks are optimized for real-time inference on both GPU

and CPU, making them suitable for embedded or edge devices.
- Preprocessing Utilities: Integrated functions for image resizing, normalization, and adaptive thresholding to enhance low-contrast or low-resolution text.
- Easy Deployment: Models can be exported to ONNX or TensorRT, and integrated seamlessly into Python or C++ applications.

In a license-plate recognition pipeline, PaddleOCR excels at processing cropped plate images: after YOLOv8 identifies and extracts the plate ROI, PaddleOCR's recognition model interprets alphanumeric characters—often treating the plate as a single line of text—while the angle classifier corrects slight rotations. Its high-percentage accuracy (often above 95% in controlled conditions) and fast inference (tens of milliseconds per crop on GPU) make it ideal for real-time or near-real-time LPR systems.

III. Project Management Report

1. Requirement Analysis

a. Website Development Requirements

1. Login

- Objective: Allow users to securely access the application with a registered account.
- Requirements:

        - Users must enter both a valid username and password.
        - The system should verify credentials against stored user data.
        - On successful login, users are redirected to the main dashboard.
        - On failure, display an error message and allow the user to retry.

2. **Logout**

    - Objective: Enable authenticated users to sign out and return to the login page.
    - Requirements:
        - Provide a “Logout” button or link visible on all authenticated pages.
        - Terminate the user session securely upon logout.
        - After logout, redirect the user to the login page.
3. **Sign Up (Registration)**
    - Objective: Allow new users to create an account with required information.
    - Requirements:
        - The registration form must collect: Full Name, Email, Username, Password, and Role (e.g., admin or user).
        - Validate that all required fields are filled and that the username is unique.
        - Enforce password complexity: minimum length of 8 characters, containing uppercase, lowercase, and numbers.
        - On successful registration, display a success message and redirect the user to the login page.

        - On validation failure, highlight the erroneous fields and display appropriate error messages.

4. **View Homepage**
    - Objective: Display a landing page that differs for authenticated vs. unauthenticated users.
    - Requirements:
        - Unauthenticated users see a welcome screen with options to “Login” or “Sign Up.”
        - Authenticated users see a dashboard with links to “Upload Image/Video,” “View History,” and “Profile.”

        - The navigation bar should reflect the user’s authentication state (e.g., show “Logout” when logged in).
5. **Upload Image/Video**
    - Objective: Allow users to submit vehicle images or video clips for license plate recognition.
    - Requirements:
        - Provide an upload form accepting image files (JPEG, PNG) or video files (MP4).
        - Validate file type and ensure file size does not exceed predefined limits (e.g., 5 MB for images, 50 MB for videos).
        - On submission, display a loading indicator while the file is being processed.
        - If the upload fails (e.g., invalid format or size), show an error alert and prompt the user to retry.
6. **View Recognition Results**
    - **Objective**: Display detected license plate images and extracted text after processing.

- **Requirements:**
    - After processing, present a thumbnail of the cropped license plate alongside the recognized plate number and confidence scores.
    - Allow users to review multiple detected plates if more than one is found in the same upload.
    - Provide a “Download” link for the cropped plate image.
    - If no plate is detected, show a “Không thể nhận diện” message.

7. Profile Management

- **Objective:** Let users view and update their account information.
- **Requirements:**
    - Display current user details: Full Name, Email, Username, and Role.
    - Allow users to change their password (with current password verification).
    - On successful update, show a confirmation message.
    - Validate that the new password meets complexity requirements.

b. Recognition System Requirements

1.Image/Video Handling

- **Objective:** Ingest and preprocess uploaded media for detection.

- **Requirements:**
    - Accept image files (e.g., JPEG, PNG) and video files (e.g., MP4) via a POST endpoint.
    - Convert input to a standardized format (RGB, fixed resolution) before passing to the detection pipeline.
    - For videos, sample frames at a predefined rate (e.g., every 5th frame) to balance accuracy and performance.
    - If file conversion or sampling fails, return a descriptive error.

2.License Plate Detection (YOLOv8)

- **Objective:** Accurately locate license plate bounding boxes within the preprocessed image or video frame.
- **Requirements:**
    - Load a fine-tuned YOLOv8 model specialized on Vietnamese license plates.
    - For each input frame, output zero or more bounding boxes
    - (x1,y1,x2,y2) with associated confidence scores.
    - Discard detections below a confidence threshold (e.g., 0.5 by default).
    - If no plates are detected, record that “Không thể xác nhận” status and proceed to the response stage.

3.Region of Interest Extraction

- **Objective:** Crop and normalize each detected license plate region for OCR.
- **Requirements:**
    - Convert the normalized YOLO coordinates to pixel values based on the resized/preprocessed frame dimensions.

- Crop each license plate region and optionally apply perspective correction if the plate appears skewed.
- Resize each cropped ROI to a fixed size (e.g., 240×80 pixels) while preserving the aspect ratio.
- Save cropped images in a temporary directory, naming files uniquely (e.g., with a UUID or timestamp).

4.Character Recognition (PaddleOCR)

- **Objective:** Extract alphanumeric characters from each cropped license plate image.
- **Requirements:**
    - Load a PaddleOCR model configured for Latin characters (digits and uppercase letters).
    - For each ROI, apply OCR and obtain:
    - Extracted text string (e.g., "30A12345").
    - Confidence score for the recognized text (floating-point value between 0 and 1).
    - If OCR fails (e.g., low confidence or unreadable region), mark that ROI as "Không thể nhận diện" and include an appropriate message in the response.

c.Database Design

The system uses two main tables in MySQL to store:

1.users

- **Purpose:** Manage user account information.
- **Storage:**
    - Auto-incrementing ID (primary key)
    - Full name, email, login name (all unique), hashed password

- Role (admin or user)
- Account creation time

2.detection_logs

- **Purpose**: Record the license plate recognition results performed by the user.

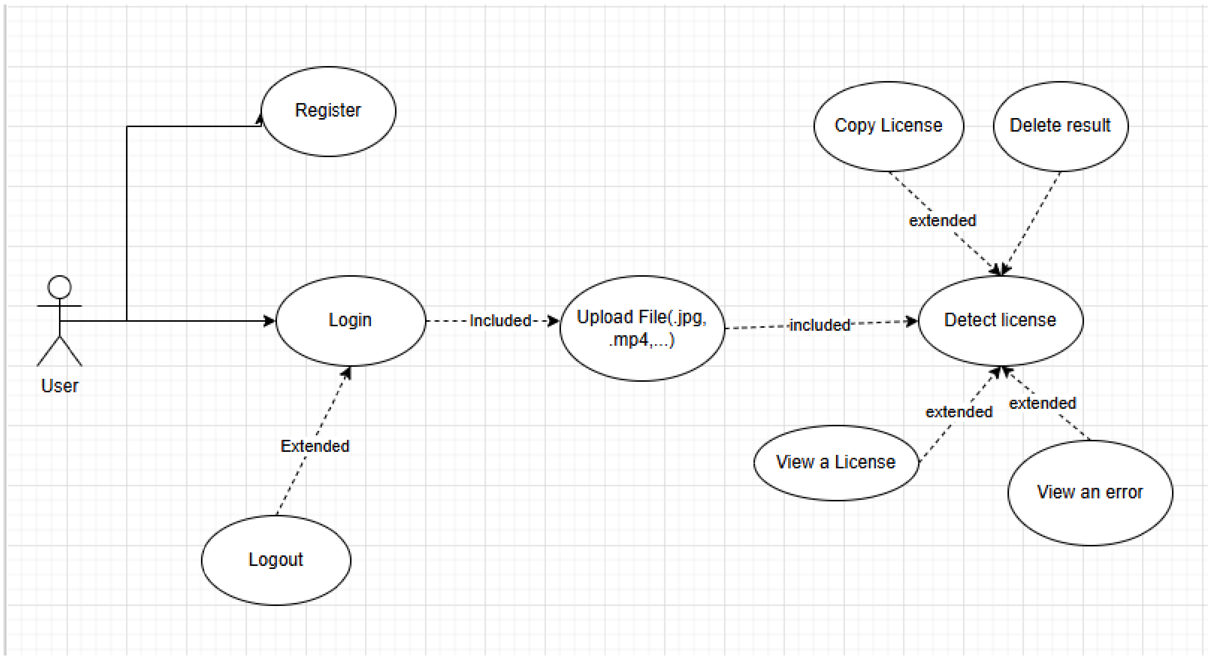
- **Storage:**
  - Auto-incrementing ID (primary key)
  - Foreign key user_id points to the users table to know which user performed
  - License plate string (license_plate) recognized by OCR
  - Confidence score of OCR result
  - Time of recognition
  - Path to cropped image of license plate (image_path)
  - (Optional) vehicle type (vehicle_type) and region/city (region) if available

These two tables ensure full storage of user information and recognition history, helping the system to decentralize, retrieve and make statistics easily.

2.System Analysis and Design

a.Overall Design

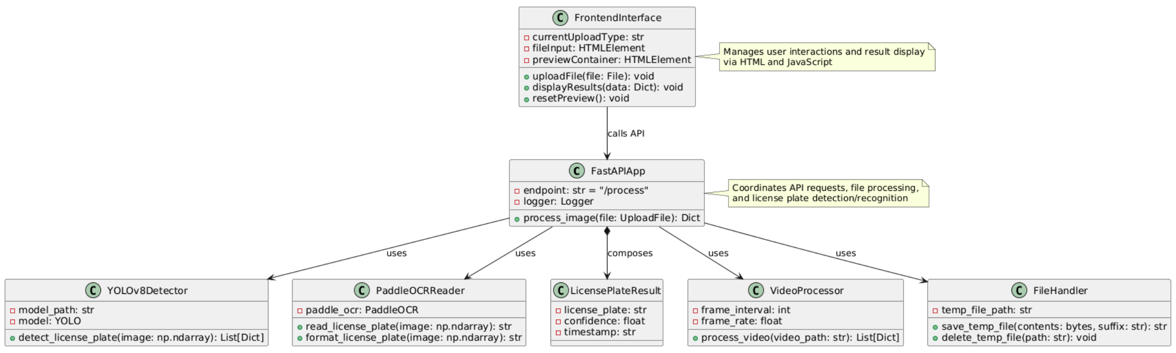
- **Use Cases:**



- Actor “User”: represents the end user of the system.
- Register Account / Login / Logout / Manage Profile: account management use cases, allowing users to register, log in, log out and edit their personal profiles.
- Upload Image/Video: the first step in the recognition process; the user selects a photo or video file containing the vehicle.
- Plate Recognition: a composite use case, consisting of a series of sub-steps to process the media file.
- Detect Plate (YOLOv8): uses the YOLOv8 model to detect the location of the license plate frame in an image or video frame.
- Crop & Preprocess ROI: crops the area containing the license plate, preprocesses (light correction, noise removal) to prepare for OCR.

- OCR Recognition (PaddleOCR): runs PaddleOCR on the cropped image to extract the license plate characters.
- Save Recognition Result: saves the result (plate string, confidence score, cropped image path) to a MySQL database.
- View Recognition Results: allows users to view the latest results or specific results after recognition.
- Download Cropped Image: provides the option to download the cropped license plate image.

-Class Diagrams


FrontendInterface
currentUploadType: str
fileInput: HTMLElement
previewContainer: HTMLElement
uploadFile(file: File): void
displayResults(data: Dict): void
resetPreview(): void
Manages user interactions and result display via HTML and JavaScript
calls API
FastAPIApp
endpoint: str = "/process"
logger: Logger
process_image(file: UploadFile): Dict
Coordinates API requests, file processing, and license plate detection/recognition
uses
uses
composes
uses
uses
YOLOv8Detector
model_path: str
model: YOLO
detect_license_plate(image: np.ndarray): List[Dict]
PaddleOCRReader
paddle_ocr: PaddleOCR
read_license_plate(image: np.ndarray): str
format_license_plate(image: np.ndarray): str
LicensePlateResult
license_plate: str
confidence: float
timestamp: str
VideoProcessor
frame_interval: int
frame_rate: float
process_video(video_path: str): List[Dict]
FileHandler
temp_file_path: str
save_temp_file(contents: bytes, suffix: str): str
delete_temp_file(path: str): void

b.Implementation

1. Backend API

- Authentication & Admin Management

Post	/register	Register a new account. User provides(fullname, email, …).
Post	/login	Authenticate user with username,password, return access token (JWT).

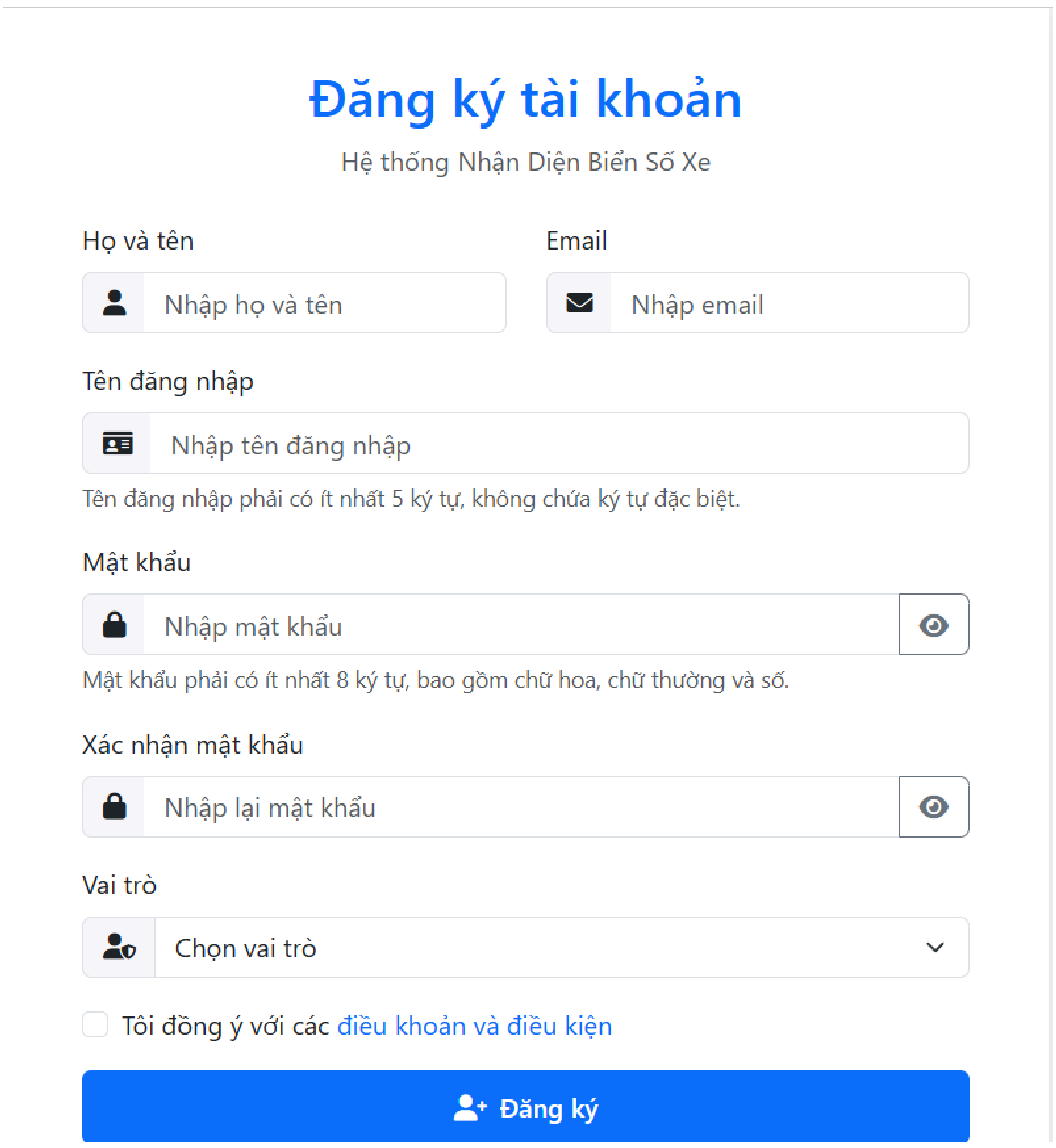
Post	/logout	Cancel login session.
Get	/me	Get details of logged in user
Get	/dashboard	API to get dashboard information

- Recognition

Post	/process	Receive image (.jpg, .png) or video (.mp4) file from client – Only authenticated users (with JWT) will be called. – Return JSON containing list of detected license plates,

2.Front-end

-When users create a new account:



Đăng ký tài khoản

Hệ thống Nhận Diện Biển Số Xe

Họ và tên: Nhập họ và tên

Email: Nhập email

Tên đăng nhập: Nhập tên đăng nhập

Tên đăng nhập phải có ít nhất 5 ký tự, không chứa ký tự đặc biệt.

Mật khẩu: Nhập mật khẩu

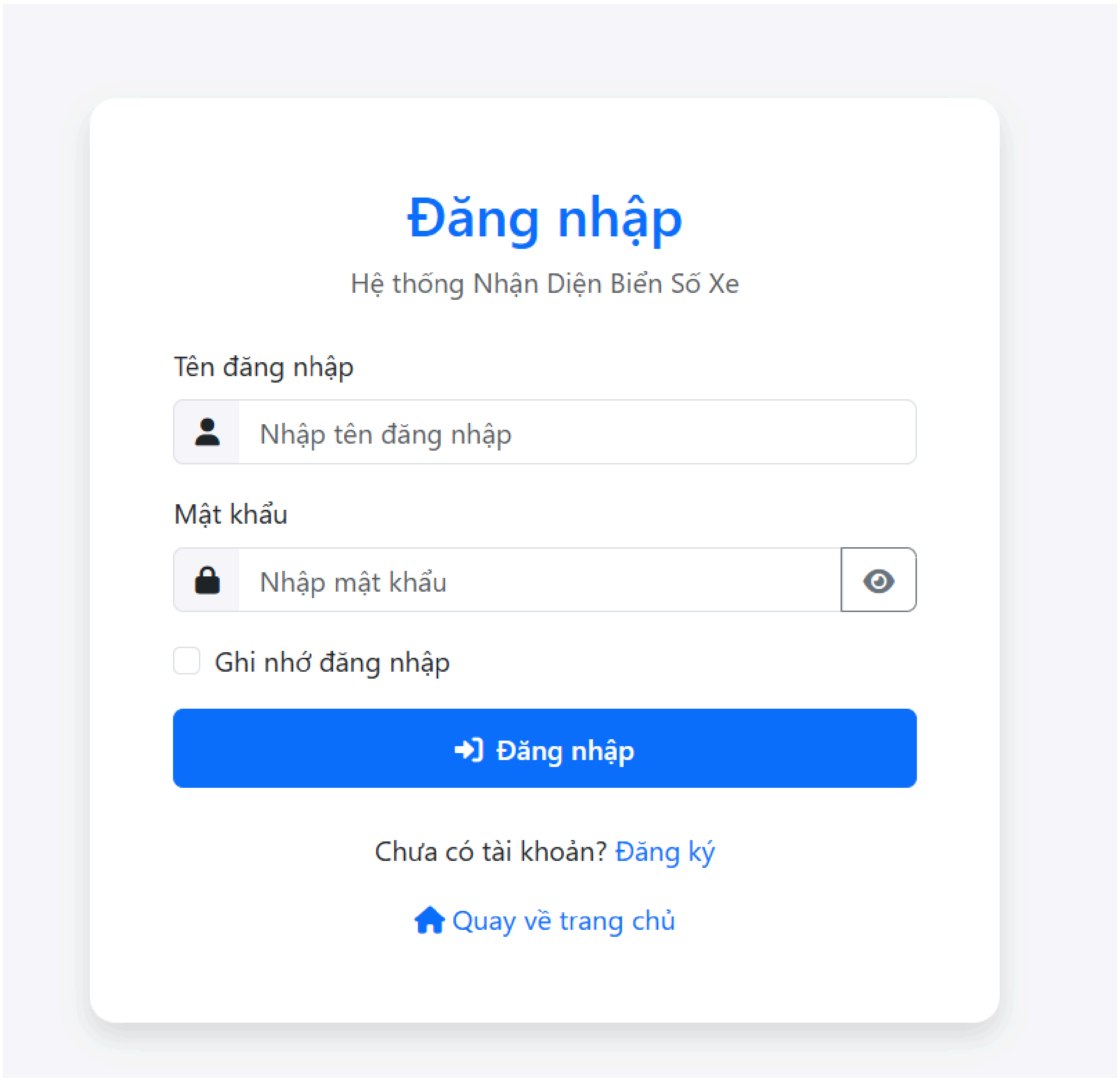
Mật khẩu phải có ít nhất 8 ký tự, bao gồm chữ hoa, chữ thường và số.

Xác nhận mật khẩu: Nhập lại mật khẩu

Vai trò: Chọn vai trò

☐ Tôi đồng ý với các điều khoản và điều kiện

-When users login



Đăng nhập

Hệ thống Nhận Diện Biển Số Xe

Tên đăng nhập

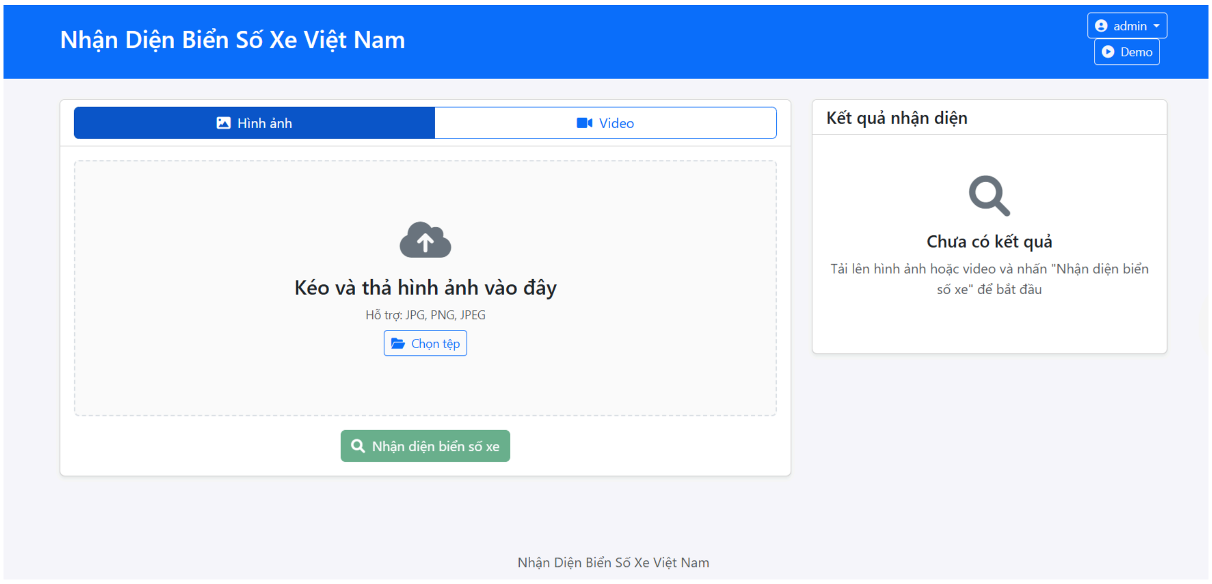
Nhập tên đăng nhập

Mật khẩu

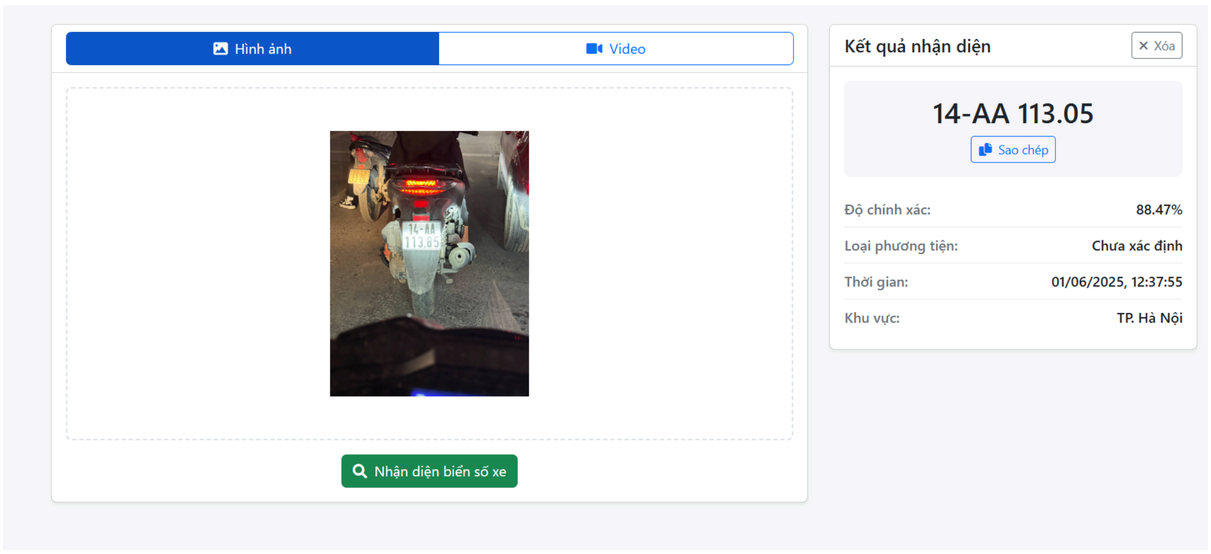
Nhập mật khẩu

Ghi nhớ đăng nhập

-The main interface

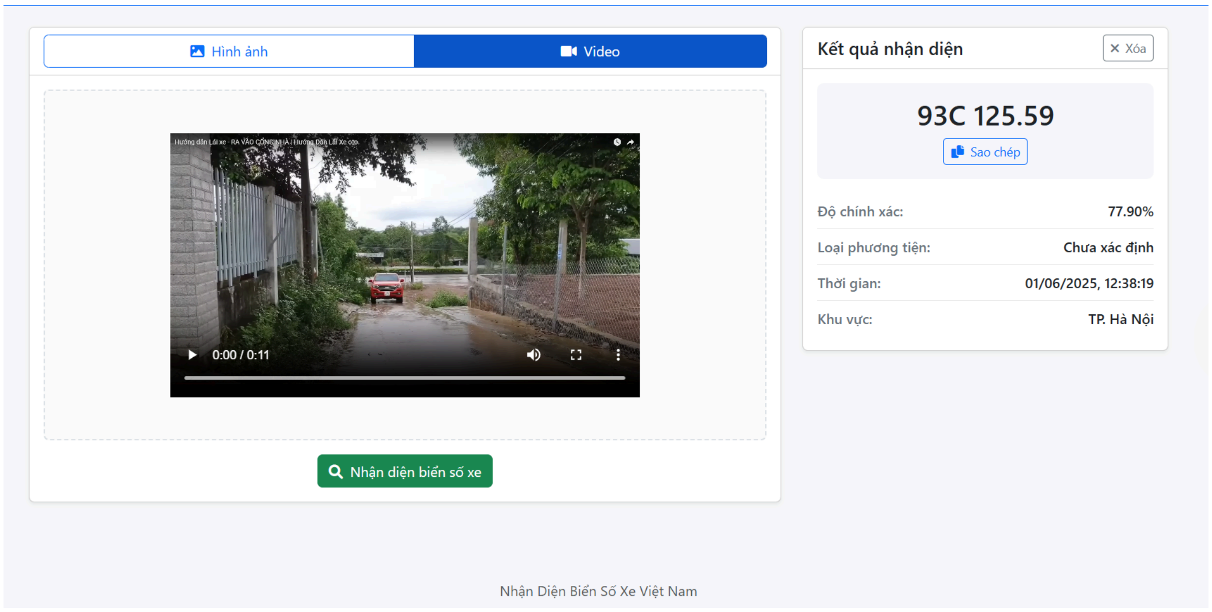

Nhận Diện Biển Số Xe Việt Nam
admin
Demo
Hình ảnh
Video
Kéo và thả hình ảnh vào đây
Hỗ trợ: JPG, PNG, JPEG
Chọn tệp
Nhận diện biển số xe
Kết quả nhận diện
Chưa có kết quả
Tải lên hình ảnh hoặc video và nhấn "Nhận diện biển số xe" để bắt đầu
Nhận Diện Biển Số Xe Việt Nam

- Result with image



Độ chính xác:	88.47%
Loại phương tiện:	Chưa xác định
Thời gian:	01/06/2025, 12:37:55
Khu vực:	TP. Hà Nội

-Result with video



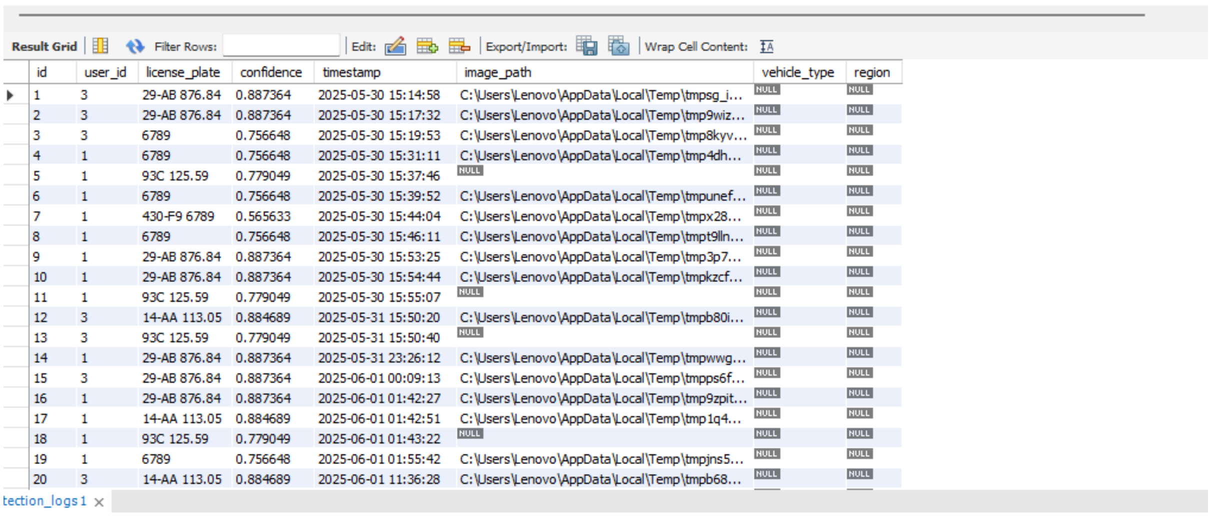
Kết quả nhận diện

93C 125.59

Độ chính xác:	77.90%
Loại phương tiện:	Chưa xác định
Thời gian:	01/06/2025, 12:38:19
Khu vực:	TP. Hà Nội

-Results saved on MySQL local

```
SELECT * FROM license_plate_db.detection_logs;
```



id	user_id	license_plate	confidence	timestamp	image_path	vehicle_type	region
1	3	29-AB 876.84	0.887364	2025-05-30 15:14:58	C:\Users\Lenovo\AppData\Local\Temp\tmpsg_i...	NULL	NULL
2	3	29-AB 876.84	0.887364	2025-05-30 15:17:32	C:\Users\Lenovo\AppData\Local\Temp\tmp9wiz...	NULL	NULL
3	3	6789	0.756648	2025-05-30 15:19:53	C:\Users\Lenovo\AppData\Local\Temp\tmp8kyv...	NULL	NULL
4	1	6789	0.756648	2025-05-30 15:31:11	C:\Users\Lenovo\AppData\Local\Temp\tmp4dh...	NULL	NULL
5	1	93C 125.59	0.779049	2025-05-30 15:37:46	NULL	NULL	NULL
6	1	6789	0.756648	2025-05-30 15:39:52	C:\Users\Lenovo\AppData\Local\Temp\tmpunef...	NULL	NULL
7	1	430-F9 6789	0.565633	2025-05-30 15:44:04	C:\Users\Lenovo\AppData\Local\Temp\tmpx28...	NULL	NULL
8	1	6789	0.756648	2025-05-30 15:46:11	C:\Users\Lenovo\AppData\Local\Temp\tmpt9lln...	NULL	NULL
9	1	29-AB 876.84	0.887364	2025-05-30 15:53:25	C:\Users\Lenovo\AppData\Local\Temp\tmp3p7...	NULL	NULL
10	1	29-AB 876.84	0.887364	2025-05-30 15:54:44	C:\Users\Lenovo\AppData\Local\Temp\tmpkzcf...	NULL	NULL
11	1	93C 125.59	0.779049	2025-05-30 15:55:07	NULL	NULL	NULL
12	3	14-AA 113.05	0.884689	2025-05-31 15:50:20	C:\Users\Lenovo\AppData\Local\Temp\tmpb80i...	NULL	NULL
13	3	93C 125.59	0.779049	2025-05-31 15:50:40	NULL	NULL	NULL
14	1	29-AB 876.84	0.887364	2025-05-31 23:26:12	C:\Users\Lenovo\AppData\Local\Temp\tmpwwg...	NULL	NULL
15	3	29-AB 876.84	0.887364	2025-06-01 00:09:13	C:\Users\Lenovo\AppData\Local\Temp\tmpps6f...	NULL	NULL
16	1	29-AB 876.84	0.887364	2025-06-01 01:42:27	C:\Users\Lenovo\AppData\Local\Temp\tmp9zpit...	NULL	NULL
17	1	14-AA 113.05	0.884689	2025-06-01 01:42:51	C:\Users\Lenovo\AppData\Local\Temp\tmp1q4...	NULL	NULL
18	1	93C 125.59	0.779049	2025-06-01 01:43:22	NULL	NULL	NULL
19	1	6789	0.756648	2025-06-01 01:55:42	C:\Users\Lenovo\AppData\Local\Temp\tmpjns5...	NULL	NULL
20	3	14-AA 113.05	0.884689	2025-06-01 11:36:28	C:\Users\Lenovo\AppData\Local\Temp\tmpb68...	NULL	NULL

3.Project Schedule Management

Task	Started on	Progress

Data collection of license plate images	Week 1	100%
Completed initial image preprocessing	Week 2	100%
Continued YOLO training—experimented with hyperparameters	Week 3	100%
Tested inference on real-world images on local(no interface)	Week 4	100%
Integrated PaddleOCR to recognize characters	Week 5	100%
Developed the FastAPI backend endpoint	Week 6	100%
Handled video input,create interface	Week 7	100%
Completed database integration	Week 8	100%
Add login register feature	Week 9	100%

4.Results of the project

- **Completed Features**:

1. User Registration (signup)
2. User Login and Logout
3. Upload Image for License Plate Recognition
4. Upload Video for License Plate Recognition (frame sampling every 10 seconds)
5. Plate Detection using YOLOv8
6. ROI Preprocessing and Cropping of Detected Plates
7. OCR Recognition of Cropped Plates via PaddleOCR
8. Storing Recognition Results in MySQL (detection_logs)

9. Downloading Cropped Plate Images
10. Viewing Full History of Past Recognition Events

- **Features Not Implemented**:

1. Vehicle-Type Classification (e.g., distinguishing car vs. motorbike)
2. Automatic Region Inference (mapping plate to province code, ex HaNoi:29, 30)
3. Real-Time Live Video Streaming Recognition (current implementation samples periodically)
4. Admin Dashboard for Managing All Users and Logs (only per-user history is available)
5. Advanced Analytics and Reporting (charts or trend graphs)

- Project Completion Level: 85%

IV. Conclusion

1.Advantages:

- The application implements all core features for license plate recognition, including user authentication, media upload (image and video), YOLOv8–based detection, and PaddleOCR recognition.
- ○The interface is clean, intuitive, and responsive, ensuring ease of use on both desktop and mobile devices.
- The end-to-end pipeline—from image preprocessing through database storage of results—operates reliably, providing accurate plate text and confidence scores.

2.Disadvantages:

- Certain advanced features remain basic or unimplemented, such as real-time continuous video streaming recognition and automatic mapping of detected plates to vehicle type or region.
- The admin dashboard and analytics modules are not yet complete, limiting oversight and reporting capabilities.
- Performance on low-light or heavily angled inputs can still be improved; occasional false negatives occur under challenging conditions.

3.Summary

During the process of learning and building the project, I have tried to learn and develop many features; however, there are still many shortcomings. In the future, I will research and learn more to complete the website system.

I sincerely thank my teacher for the guidance and support in completing this project.

V. References

[1] Ultralytics, "YOLOv8 Documentation," Ultralytics. Available at: https://docs.ultralytics.com/yolov8/ (Accessed: 01 June 2025).

[2] PaddlePaddle, "PaddleOCR: Open-Source OCR Toolkit," PaddleOCR Documentation. Available at: https://github.com/PaddlePaddle/PaddleOCR (Accessed: 01 June 2025).

[3] Sebastián Ramírez, “FastAPI: The high-performance web framework for building APIs with Python,” FastAPI Documentation. Available at: https://fastapi.tiangolo.com/ (Accessed: 01 June 2025).

[4] OpenCV Team, “Open Source Computer Vision Library,” OpenCV Documentation. Available at: https://opencv.org/ (Accessed: 01 June 2025).

[5] MySQL Documentation Team, “MySQL 8.0 Reference Manual,” MySQL Documentation. Available at: https://dev.mysql.com/doc/ (Accessed: 01 June 2025).

[6] Python Software Foundation, “Python 3.10 Documentation,” Python Docs. Available at: https://docs.python.org/3.10/ (Accessed: 01 June 2025).

[7] FastAPI Authors, “SQLAlchemy Integration with FastAPI,” FastAPI Tutorial. Available at: https://fastapi.tiangolo.com/tutorial/sql-databases/ (Accessed: 01 June 2025).