Faculty of Informatics Engineering
Department of Software Engineering

Traffic police assistance

-MARYAM ALBATOUL ALMASRI 4210120

-TASNIM DAKAK 4210254

Problem Statement:

Traffic violation management faces significant challenges in relying on manual documentation of information, which increases the likelihood of error and slows down police performance during field missions. In addition, current systems do not take advantage of modern technologies such as vehicle plate recognition or converting spoken speech into text, nor do they provide a unified platform that enables concerned authorities and citizens to effectively follow up on violations and appeals. This deficiency affects data quality, limits decision-making speed, and impairs transparency and governance.

Proposed Solution:

The project proposes developing an intelligent traffic violation management system based on artificial intelligence technologies to convert speech to text, recognize vehicle plates, and generate reports automatically. The system provides dedicated interfaces for various roles (policeman, police director, administrator, citizen), and provides a central platform for managing and following up on violations and appeals. It also enhances the accuracy of documentation, expands the possibility of oversight, and increases the efficiency of field and administrative work.

Project objective:

The project aims to create an intelligent and integrated system for managing traffic violations, based on artificial intelligence technologies to improve the speed and accuracy of violation documentation, through:

- Automate violation recording using voice-to-text services, vehicle plate recognition, and automatically generate reports.

- Providing customized interfaces for each role (policeman, manager, administrator, citizen).
- Enabling citizens to follow up on violations and submit appeals electronically
- Unifying data into a single platform that supports transparency and speed of decision-making

Scope of the Project:

The scope of the project includes the development and implementation of an integrated electronic platform for managing traffic violations, which includes the following elements:

1. **Managing users and roles** (citizen, policeman, police director, administrator).
2. Artificial Intelligence Services:
· Convert audio to text (Speech-to-Text).
· Vehicle plate recognition (OCR).
· Automatically generate violation report text (Report Generation).
3. Organizing violations by police officers via a smart field application.
4 .Traffic Appeals Department tracks its status from submission to decision.
5. Control panels are dedicated to each role to display violations, statistics, and permitted operations.
6. A central database for storing violations, their reports, and attachments.
7. Citizen portal for following up on violations, paying them, and submitting appeals

Project Implementation Phases:

1. scope Definition and Project Goals
2. Literature review and compare between similar systems
3. Requirements Elicitation and Stakeholder Identification
4. system Analysis
5. System Design and Architecture Selection
6. start the first increment (developing an account management subsystem)
7. testing the functionalities of the first increment
8. start the second increment (developing a violation management subsystem)
9. testing the functionalities of the second increment
10. start the third increment (developing an AI service(speech to text))
11. testing the functionalities of the third increment
12. start the fourth increment (developing an AI service(plate recognition))
13. testing the functionalities of the fourth increment
14. start the fifth increment (developing an AI service(violation generation))
15. testing the functionalities of the fifth increment
16. document every step of the work

Used Tools:

*Django:

Django is a Python web framework for rapid development, clean design, and hassle-free coding. It's free and open-source. DRF (Django REST Framework) simplifies building RESTful APIs by integrating with Django's core features.

*Flutter:

Flutter is an open-source UI toolkit by Google for building natively compiled mobile, web, and desktop apps from a single codebase. It offers pre-designed widgets and tools for fast, high-performance app development, minimizing platform-specific complexities. Known for its fast development cycle, expressive UI, and native performance, Flutter is free and widely popular.

*HTML, CSS, JavaScript

These are the core web technologies used to structure web pages (HTML), style the interface (CSS), and add interactivity (JavaScript) for an engaging user experience

*Laravel

Laravel is a PHP web framework that simplifies backend development with a clean structure, built-in security, and a powerful MVC architecture for efficient data handling

*visual studio code:
Visual Studio Code combines the simplicity of a source code editor with powerful developer tooling, like IntelliSense code completion and debugging. We use it to develop the whole project (frontend, backend).

*Postman:
Postman is a testing tool for APIs. It helps developers send requests, check responses, and debug backend endpoints efficiently before connecting them to the frontend

*Database:
MySQL is a widely-used, open-source relational database system. It organizes data into tables and supports SQL for managing and querying data. Known for its speed, reliability, and scalability, it is ideal for web applications and data-driven projects. MySQL integrates seamlessly with programming languages like Python, PHP, and Java.

NoSQL is a flexible type of database used to store large and varied kinds of data that don't fit well into traditional tables. It doesn't require a fixed structure, which makes it perfect for handling unstructured data like text, audio, or images.
In this project, NoSQL is used with AI services to store processed meeting data — such as transcriptions, summaries, and key insights — allowing fast access and analysis by the intelligent components of the system

The stakeholders of the system:

- Police officer:
  They can log in, record and transcribe voice notes, capture and extract car plate numbers,
  and generate traffic violations.
  They can also manage their own account and update their profile information.

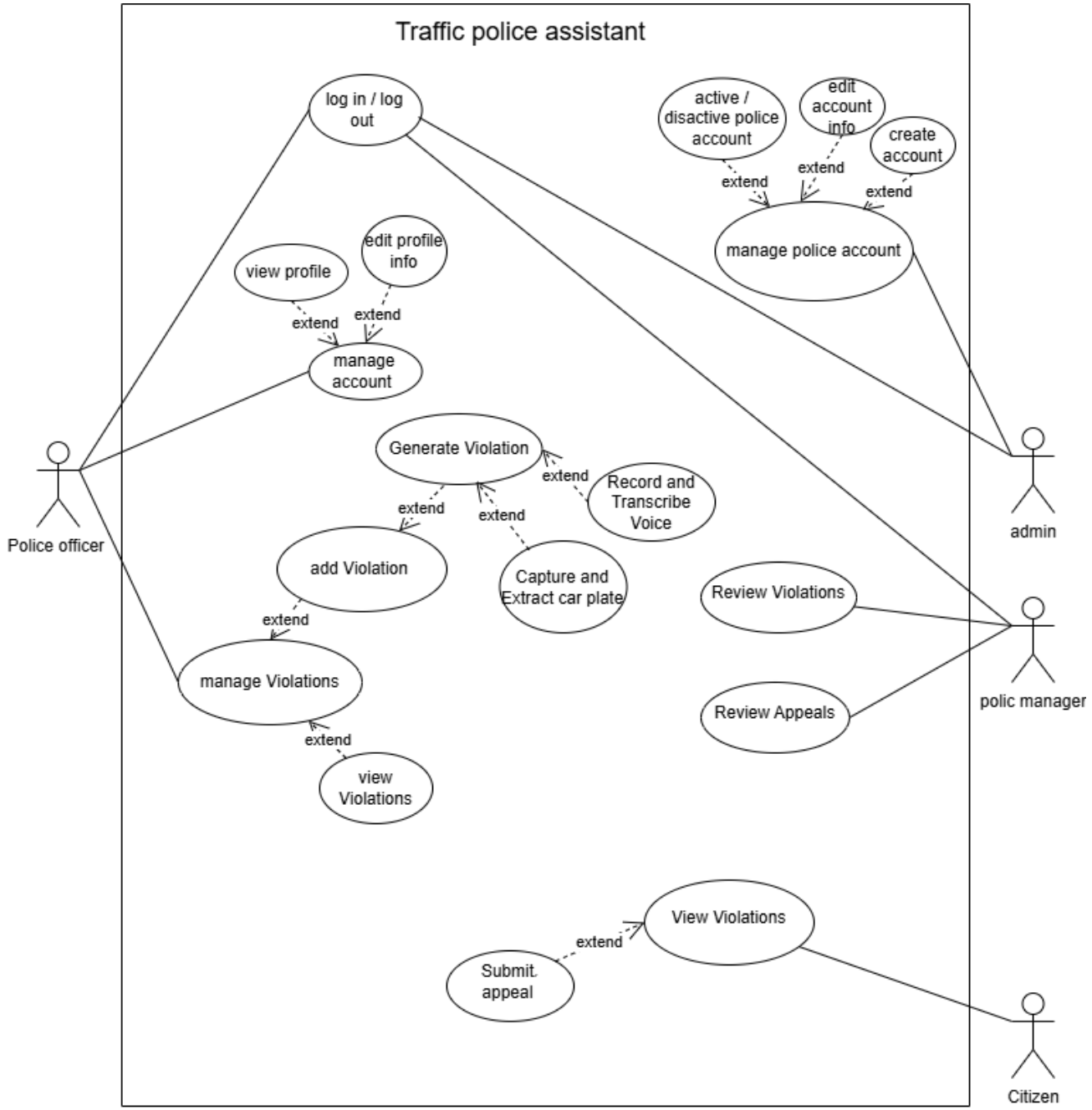
- The administrator:
  The Administrator manages the whole system.
  They are responsible for creating, editing, activating, and deactivating police officer accounts.
  The admin ensures data integrity, monitors system performance, and maintains access control for all users

- citizen:
  The Citizen interacts with the system to check their recorded traffic violations.
  They can view the details of each violation and submit an appeal if they believe a violation was issued by mistake.
- Police manager:
  They review the violations generated by officers, check submitted appeals from citizens, and make decisions to approve or reject them
  and enabling the police director to review the citizen's objection to the violation

Requirments database:

ID	Requirement Title	Description	Category	Actor	Priority
RE-FR-1	The system must provide the policeman with the ability to log in	Enable the police officer to log in to his account using his credentials	account management	police officer	high
RE-FR-2	The system must provide the police officer with the ability to log out	Enable the police officer to log out of the system safely	account management	police officer	medium
RE-FR-3	The system must provide the display and modification of account information for the policeman	View and update police data (such as phone number or email)	account management	police officer	medium
RE-FR-4	The system must provide the policeman with the ability to record a violation by voice	Enable a police officer to create a new violation using voice commands	Registering violations	police officer	high
RE-FR-5	The system must provide the police officer with the ability to manually record a violation	Enable the police officer to manually enter violation details into the system	Registering violations	police officer	high
RE-FR-6	The system must provide the ability to take a photo of the violating vehicle	Enabling the policeman to take a photo of the violating vehicle as evidence of the violation	Registering violations	police officer	عالية
RE-FR-7	The system must provide review of the report generated via artificial intelligence	View the automated violation report before it is approved by the police officer	Report Management	police officer	medium
RE-FR-8	The system should provide the ability to modify reports generated by artificial intelligence	Allow the police officer to modify the data of the machine-generated report before it is approved	Report Management	police officer	medium
RE-FR-9	The system must provide a display of the violation record created by the police officer	View all violations created by the police officer along with their status and history	Violations management	police officer	high
RE-FR-10	The system must provide the ability to search for a violation within the register	Enable a police officer to search for a specific violation using different search criteria	Violations management	police officer	medium
RE-FR-11	The system must provide a personal violation record view for the citizen	View all violations recorded against citizens, along with their details and status	Violations management	citizen	high
RE-FR-12	The system must provide for an electronic appeal to a violation	Enabling citizens to file an appeal to a registered violation and follow up on it electronically	Appeals management	citizen	high
RE-FR-13	The system must provide tracking of the appeal status	Presenting the appeal status to the citizen (under review - accepted - rejected)	Appeals management	citizen	high
RE-FR-14	The system must provide the police director with the ability to log in	Enable the police director to log in to the system to access the administrative control panel	account management	police manager	high
RE-FR-15	The system must provide the police chief with the ability to log out	Enable the police director to safely log out of the system	account management	police manager	medium
RE-FR-22	The system must provide the police director with a review of the appeal sent by the citizen	Enabling the police director to review the citizen's appeal to the violation	Appeals management	police manager	medium
RE-FR-24	The system must provide the police director with a review of the appeal sent by the citizen	Enabling the police director to review the citizen's appeal to the violation	Appeals management	police manager	medium
RE-FR-16	The system must provide the display and modification of account information to the police manager	View police director data and update it when needed	account management	police manager	medium
RE-FR-17	The system must provide the addict with the ability to create a new police account	Enable the admin to create a new account for a police officer within the system	Users management	administrator	high
RE-FR-18	The system must provide the administrator with the ability to modify conditional account data	Enabling the admin to modify police personnel data	Users management	administrator	medium
RE-FR-19	The system must provide the administrator with the ability to delete a police account	Enable the admin to delete a police account from the system permanently	Users management	administrator	medium
RE-FR-20	The system must provide the admin with the ability to disable or activate a conditional account	Enable the admin to disable or reactivate a conditional account when needed	Users management	administrator	medium
RE-FR-21	The system must provide the administrator with an effective account display	Enable the admin to view a list of all active police account	Users management	administrator	medium
RE-FR-25	The system must provide the conversion of voice commands into text	Analyze voice commands issued by the policeman and convert them into processable texts	artificial intelligence	AI	high
RE-FR-26	The system must provide analysis of the captured images	Extract vehicle number and type from images using computer vision techniques	artificial intelligence	AI	high
RE-FR-27	The system must provide for the generation of automated violation reports	Create detailed violation reports that include location, time, type, and plate number	artificial intelligence	AI	high
RE-FR-28	The system must provide a classification of violations (the system must support several types)	Automatically classify violations according to their type (overspeeding, wrong alignment...)	artificial intelligence	AI	high
RE-NFR-01	The system must provide constant availability	The system ensures business continuity and service availability at all times with minimal downtime	availability	system	high
RE-NFR-02	The system must provide support for a large number of concurrent users	The system allows multiple requests to be processed simultaneously without delay or collapse	performance	system	high
RE-NFR-03	The system must provide scalability		scalability	system	high
RE-NFR-04	The system must provide encryption of all data and communications	Protecting data transferred between the user and servers using HTTPS protocols	security	system	high
RE-NFR-05	The system must provide password storage in a strong encrypted manner		security	system	high
RE-NFR-06	The system must provide daily data backup	Create automatic backups of the database to ensure they are restored when a malfunction occurs	reliability	system	high

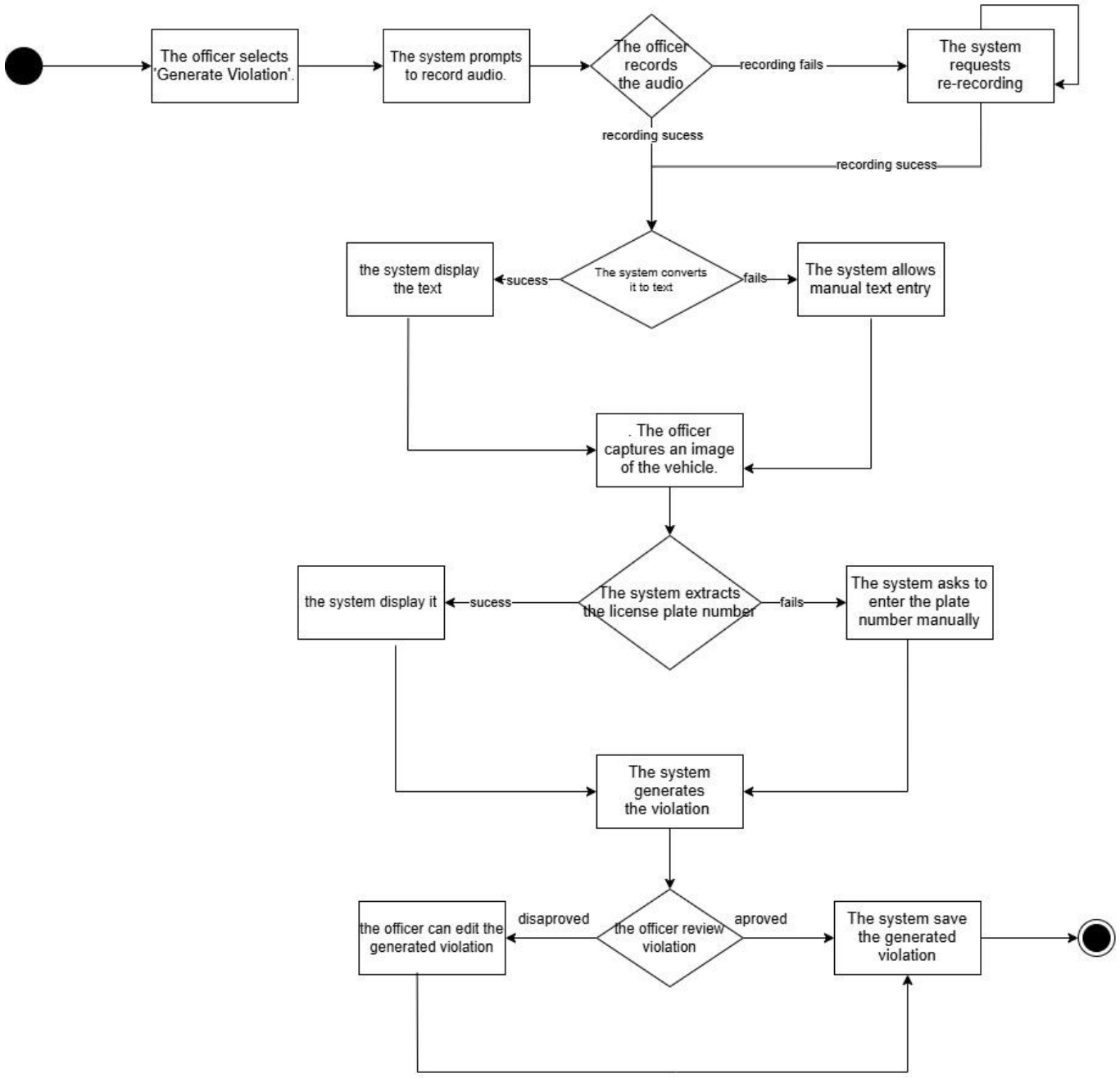
Use case diagram:



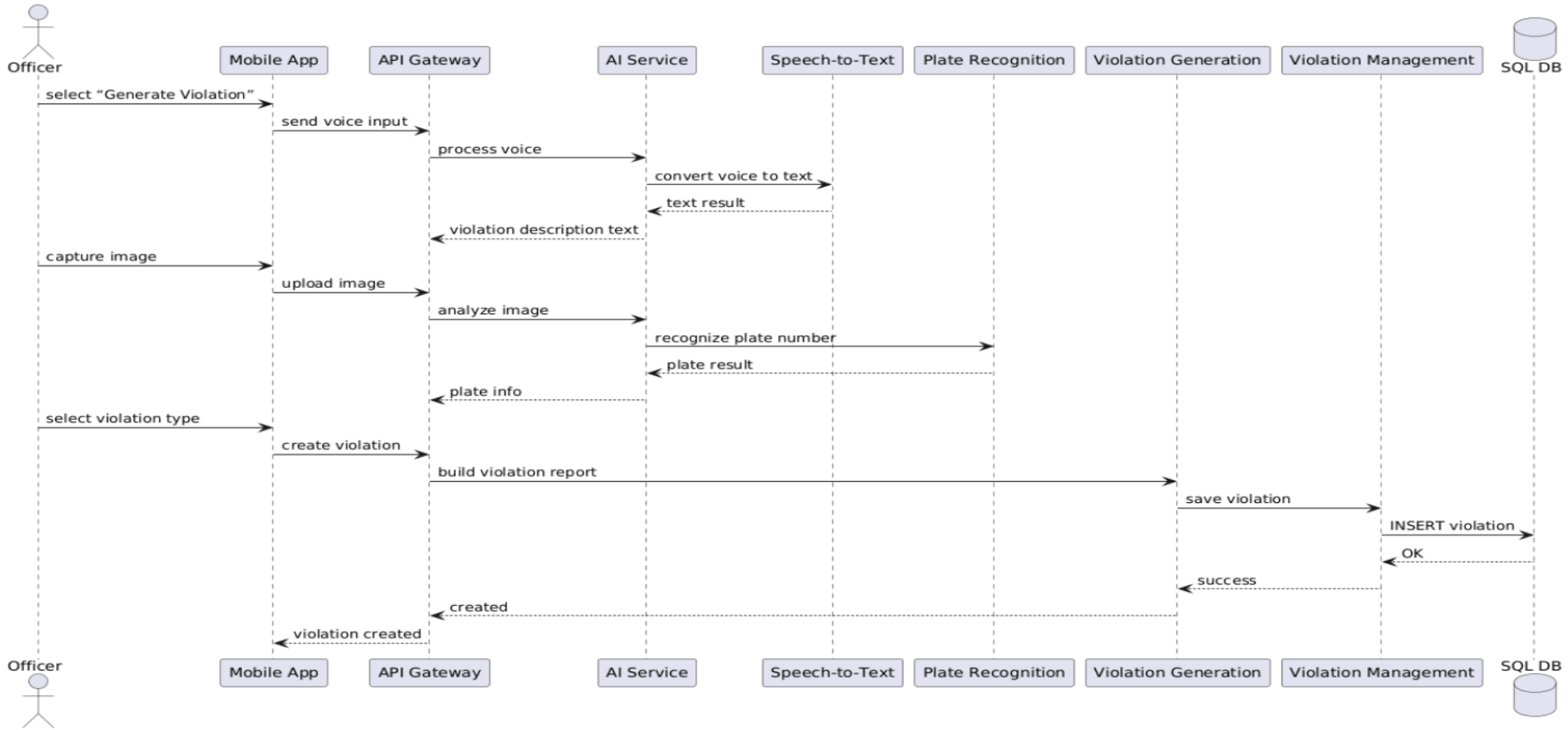
Use case specification:

9	UC_3
title	Generate Violation
description	This feature allows the police officer to create an AI-assisted traffic violation record.
actors	Police Officer
precondition	The officer must be logged in.
main flow	1. The officer selects 'Generate Violation'. 2. The system prompts to record audio. 3. The officer records the audio. 4. The system converts it to text and displays it. 5. The officer captures an image of the vehicle. 6. The system extracts the license plate number. 7. The officer selects the violation type. 8. The system generates the violation. 9.the officer review the violation 10. the officer can edit the generated violation 11. The officer confirms 12. the system save the generated violation.
postcondition	The violation is successfully created or temporarily saved for later submission.
alternative flow	First alternative flow A1: Start at step 3 where the recording fails. 4. The system requests re-recording. Second alternative flow A2: Start at step 4 where conversion fails. 5. The system allows manual text entry. Third alternative flow A3: Start at step 6 where extraction fails. 7. The system asks to enter the plate number manually.

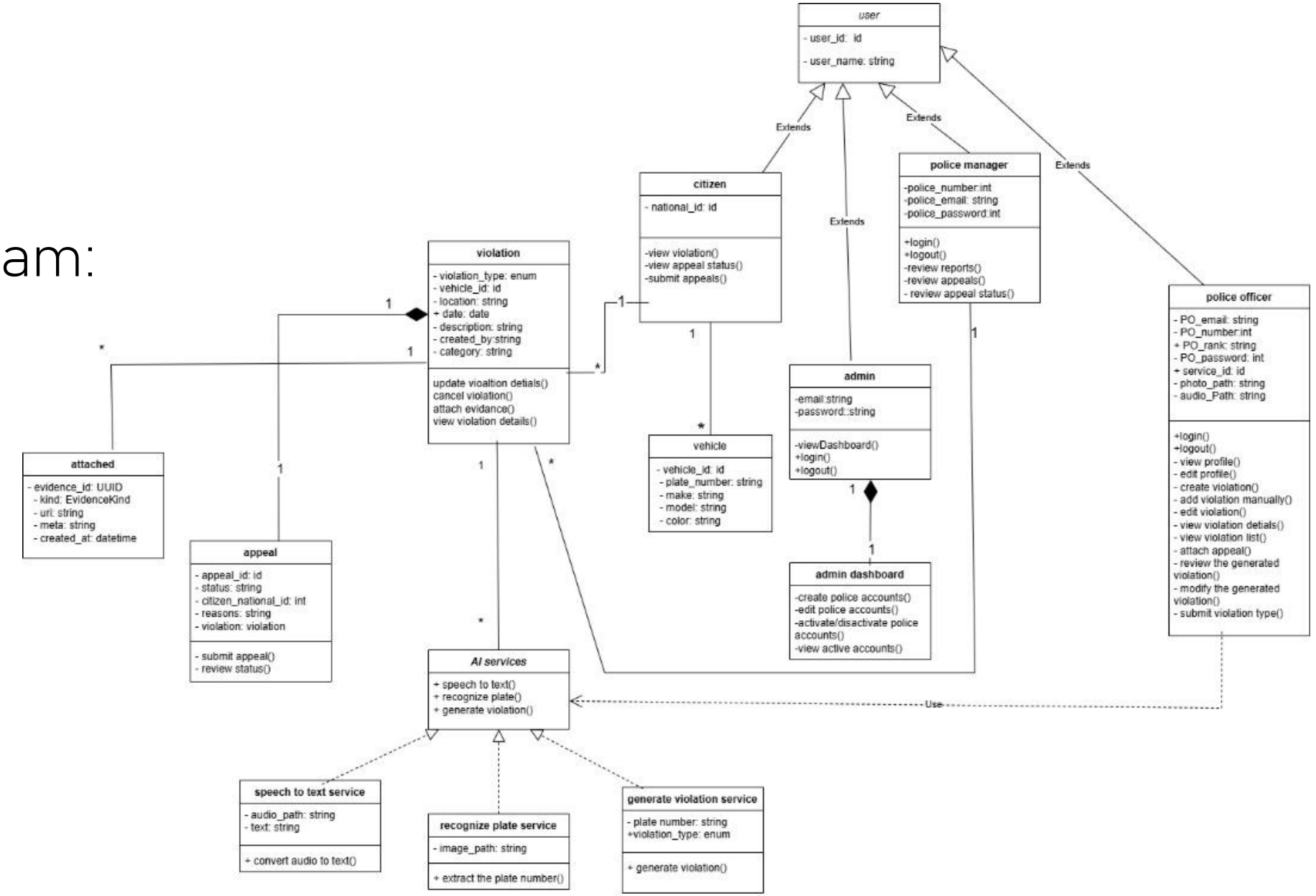
Activity diagram:



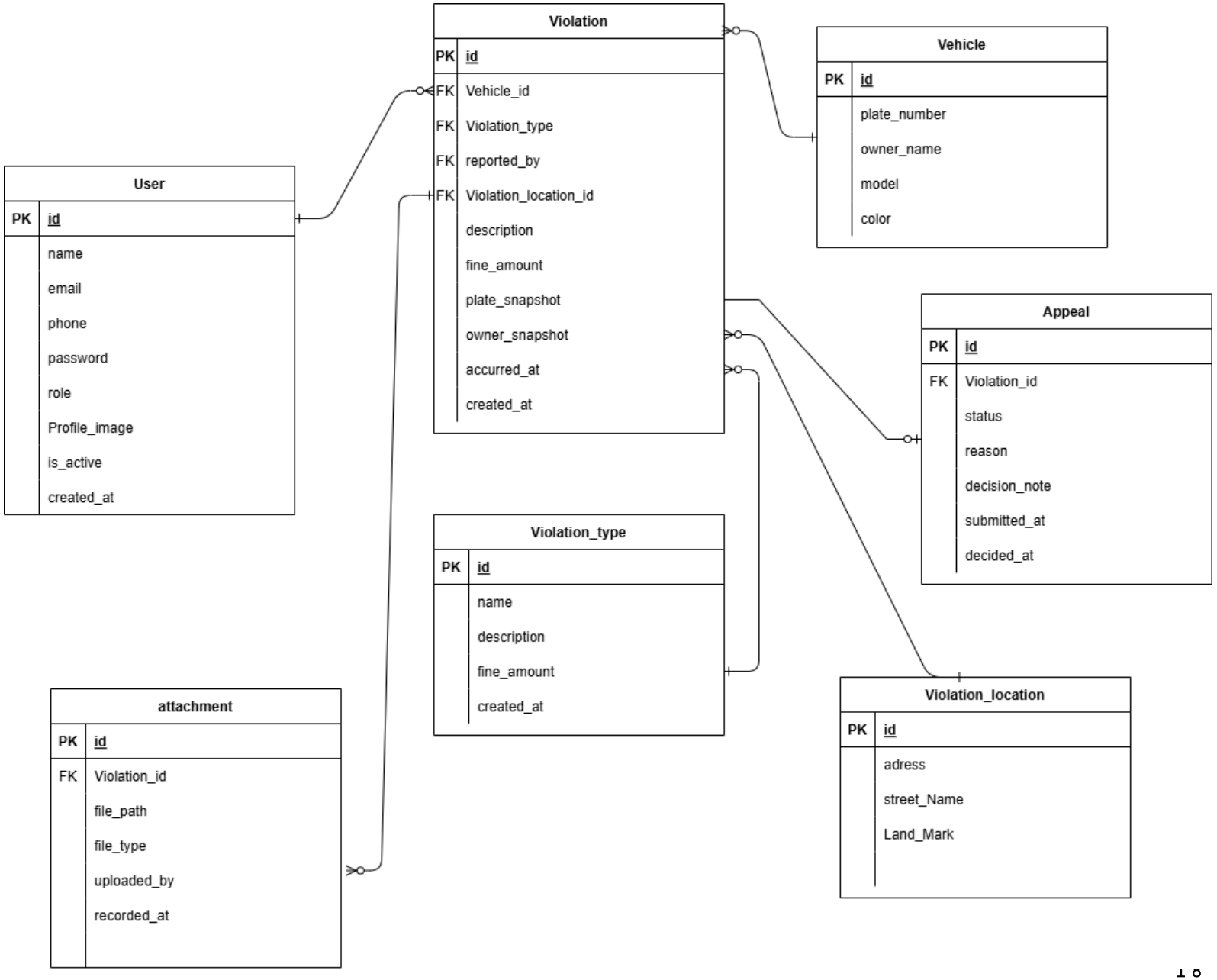
Sequence diagram :



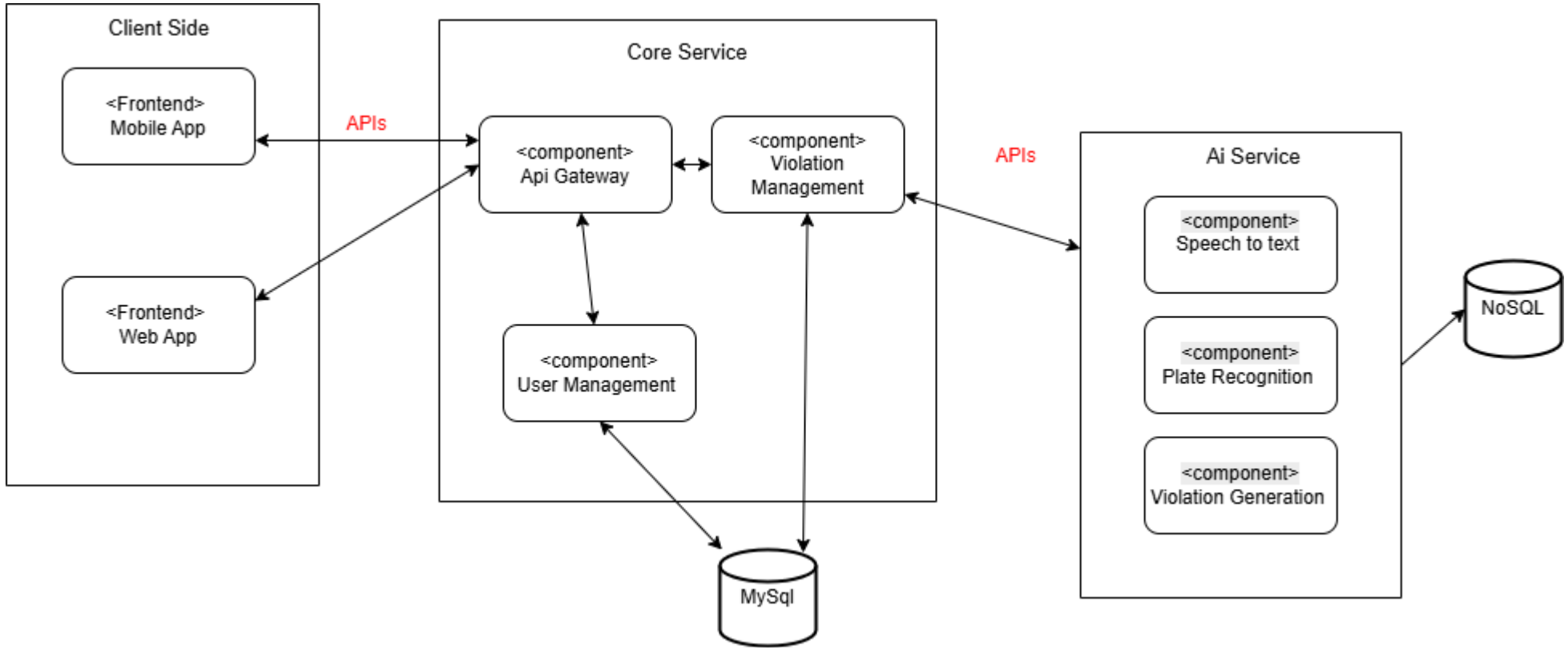
Class diagram:



ERD diagram:


Violation
PK id
FK Vehicle_id
FK Violation_type
FK reported_by
FK Violation_location_id
description
fine_amount
plate_snapshot
owner_snapshot
accurred_at
created_at
Vehicle
PK id
plate_number
owner_name
model
color
User
PK id
name
email
phone
password
role
Profile_image
is_active
created_at
Appeal
PK id
FK Violation_id
status
reason
decision_note
submitted_at
decided_at
Violation_type
PK id
name
description
fine_amount
created_at
attachment
PK id
FK Violation_id
file_path
file_type
uploaded_by
recorded_at
Violation_location
PK id
adress
street_Name
Land_Mark

Component diagram:



This High-Level System Decomposition diagram provides an overview of the system's architecture – **microservices architecture** – illustrating its main components and their interactions.

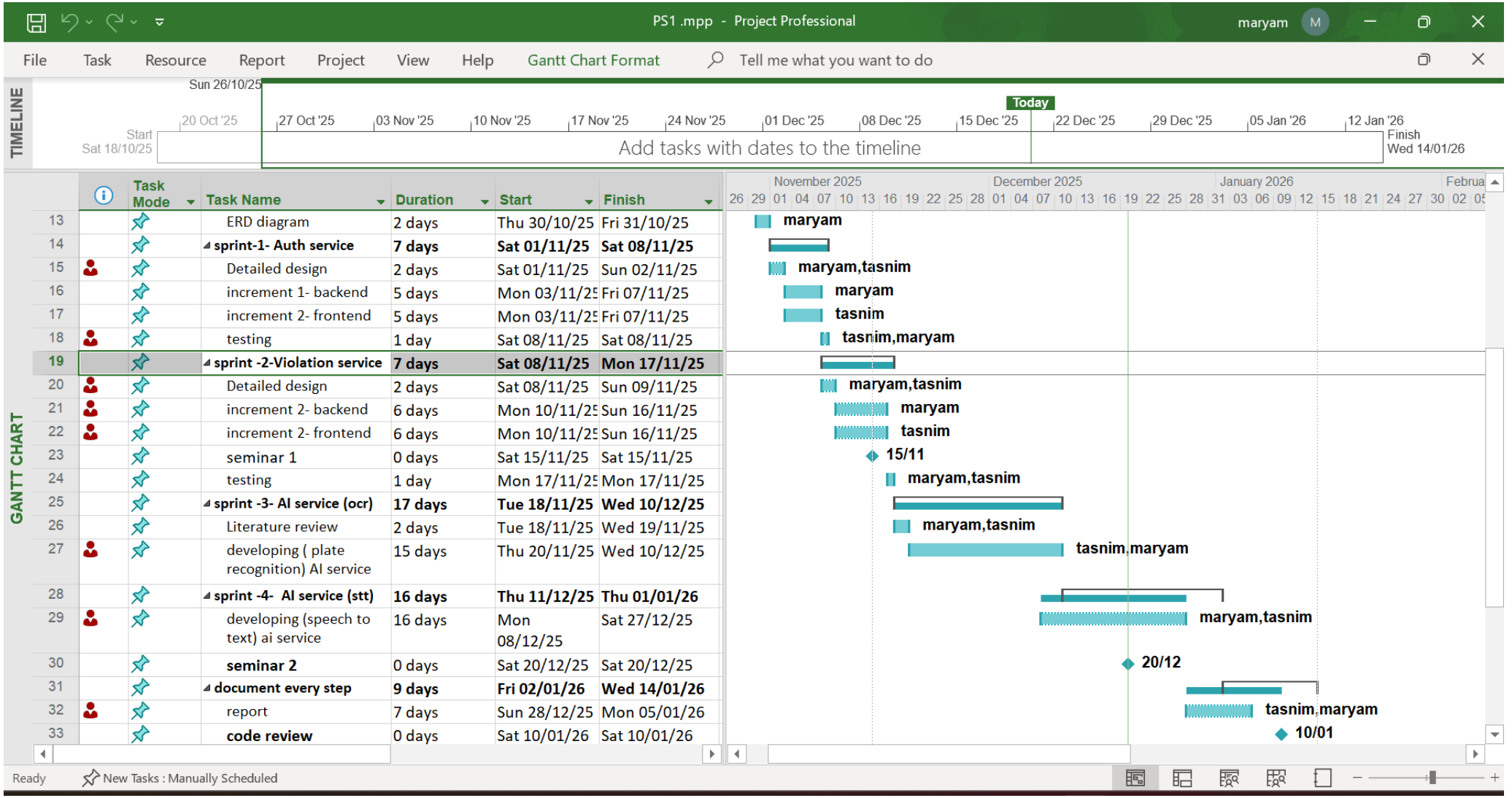
1.**Frontend:** The system consists of two frontend interfaces:
-Mobile App (for users accessing the system via mobile devices).
-Web App (for users accessing through web browsers).
2. **Backend**:
API Gateway Component:
Acts as the main entry point for all user requests coming from the web or mobile applications.
It routes the requests to the appropriate backend services such as user management or violation management.
This component also provides **authentication** and **authorization** mechanisms to secure the system.

User Management Component:
Handles all user-related operations for different roles (police officers, administrators, and citizens).
It manages **account creation**, **editing information**, and **activating or deactivating accounts**, and stores the related data in the **MySQL** database.
It communicates with the **API Gateway** via REST APIs.

Violation Management Component:
Responsible for managing all traffic violation operations, including **creation**, **modification**, **review**, and **storage**.
It interacts with police officers and managers through the **API Gateway**, and all violation records are stored in the **MySQL** database.
This component also integrates with the **AI Service** for image and text analysis.

AI Service Components:
A group of intelligent subsystems that automate the violation generation process. It includes:
Speech to Text Component:
Converts recorded audio from police officers into text using speech recognition techniques.
Plate Recognition Component:
Detects and extracts vehicle plate numbers from captured images using computer vision algorithms.
Violation Generation Component:
Automatically generates violation reports based on the extracted text and plate information, and stores the results in a **NoSQL** database.
Databases:
MySQL Database:
Stores core system data such as user accounts, violations, and configurations.
NoSQL Database:
Used for storing AI-related outputs like processed text, extracted images, and auto-generated reports.

Gantt char :


PS1 .mpp - Project Professional
maryam
File
Task
Resource
Report
Project
View
Help
Gantt Chart Format
Tell me what you want to do
TIMELINE
Sun 26/10/25
Today
20 Oct '25
27 Oct '25
03 Nov '25
10 Nov '25
17 Nov '25
24 Nov '25
01 Dec '25
08 Dec '25
15 Dec '25
22 Dec '25
29 Dec '25
05 Jan '26
12 Jan '26
Start
Sat 18/10/25
Add tasks with dates to the timeline
Finish
Wed 14/01/26
GANTT CHART
November 2025
December 2025
January 2026
Task Mode
Task Name
Duration
Start
Finish
13 ERD diagram 2 days Thu 30/10/25 Fri 31/10/25
14 sprint-1- Auth service 7 days Sat 01/11/25 Sat 08/11/25
15 Detailed design 2 days Sat 01/11/25 Sun 02/11/25
16 increment 1- backend 5 days Mon 03/11/25 Fri 07/11/25
17 increment 2- frontend 5 days Mon 03/11/25 Fri 07/11/25
18 testing 1 day Sat 08/11/25 Sat 08/11/25
19 sprint -2-Violation service 7 days Sat 08/11/25 Mon 17/11/25
20 Detailed design 2 days Sat 08/11/25 Sun 09/11/25
21 increment 2- backend 6 days Mon 10/11/25 Sun 16/11/25
22 increment 2- frontend 6 days Mon 10/11/25 Sun 16/11/25
23 seminar 1 0 days Sat 15/11/25 Sat 15/11/25
24 testing 1 day Mon 17/11/25 Mon 17/11/25
25 sprint -3- AI service (ocr) 17 days Tue 18/11/25 Wed 10/12/25
26 Literature review 2 days Tue 18/11/25 Wed 19/11/25
27 developing (plate recognition) AI service 15 days Thu 20/11/25 Wed 10/12/25
28 sprint -4- AI service (stt) 16 days Thu 11/12/25 Thu 01/01/26
29 developing (speech to text) ai service 16 days Mon 08/12/25 Sat 27/12/25
30 seminar 2 0 days Sat 20/12/25 Sat 20/12/25
31 document every step 9 days Fri 02/01/26 Wed 14/01/26
32 report 7 days Sun 28/12/25 Mon 05/01/26
33 code review 0 days Sat 10/01/26 Sat 10/01/26
maryam
maryam,tasnim
maryam
tasnim
tasnim,maryam
maryam,tasnim
maryam
tasnim
15/11
maryam,tasnim
maryam,tasnim
tasnim,maryam
maryam,tasnim
20/12
tasnim,maryam
10/01
Ready
New Tasks : Manually Scheduled

AI Literature review:

In the domain of vehicle license plate recognition, the process typically involves two key stages: 1) detecting the location of the license plate in an image and then 2) extracting the alphanumeric text from that detected region. Traditionally, these two tasks have been tackled by combining object detection and Optical Character Recognition (OCR) models. A common approach has been to use a detection model, such as YOLOv8, to locate the license plate, followed by the application of OCR models for text extraction. However, this workflow can be optimized by leveraging advanced vision-language models, such as Qwen3 VL:2b, which combine both detection and text extraction capabilities into a single unified model.

In this work, we compare the performance of various OCR models—including EasyOCR, Tesseract OCR, and Qwen3 VL:2b—in the context of vehicle license plate recognition. We also explore how the need for a separate object detection model like YOLOv8 can be eliminated by using Qwen3 VL:2b, which provides strong performance without the need for an additional detection stage.

1. YOLOv8 for License Plate Detection

YOLOv8 is a real-time object detection model that special in detecting and classifying objects in images quickly and accurately. YOLOv8 has been widely used for detecting various objects, including vehicle license plates. also YOLOv8 would be responsible for detecting the license plate's location in the image, which would then be followed by applying an OCR model to extract the alphanumeric information from the identified region

2. OCR Models for Text Extraction

Once the license plate is detected, the next step is to extract the alphanumeric text from the detected region using an OCR model. There are several OCR models available, each with unique features that cater to different use cases. In this review, we evaluate EasyOCR, Tesseract OCR, and Qwen3 VL:2b for this task

2.1 EasyOCR

EasyOCR is an open-source Optical Character Recognition (OCR) tool designed to extract text from images with high accuracy. It is built on deep learning models and supports over 80 languages, including both printed and handwritten text. EasyOCR is highly multipurpose and can handle a variety of OCR tasks, ranging from reading simple printed text to more complex scenarios like handwriting and distorted characters.

2.2 Tesseract OCR

Tesseract OCR is one of the most popular and widely used open-source OCR engines. Originally developed by HP and later maintained by Google, it is highly effective at extracting text from scanned documents or clear images.

2.3 Qwen3 VL:2b

Qwen3 VL:2b is a vision-language model designed to process both images and text. Qwen3 VL:2b is capable of directly handling image-to-text tasks without the need for pre-detection of objects. It excels at extracting detailed textual information from images, including complex scenarios like license plate recognition. The unique advantage of Qwen3 VL:2b lies in its ability to perform both detection and text extraction in one step, thereby eliminating the need for a separate object detection model such as YOLO. This integrated approach allows Qwen3 VL:2b to achieve strong performance in license plate recognition tasks while reducing the complexity of the workflow

GROUND TRUTH : 5108703

OCR OUTPUT : 5855103

ACCURACY : 0.429

CER : 0.571

EasyOCR With yolov8: Ex1:

```
📊 Evaluation Results
----------------------
Ground Truth : 5156173
OCR Output   : 6173
Accuracy     : 0.143
CER          : 0.429
```

Detected License Plate

EX2:

Evaluation Results

Ground Truth : 478817
OCR Output : 4788
Accuracy : 0.667
CER : 0.333

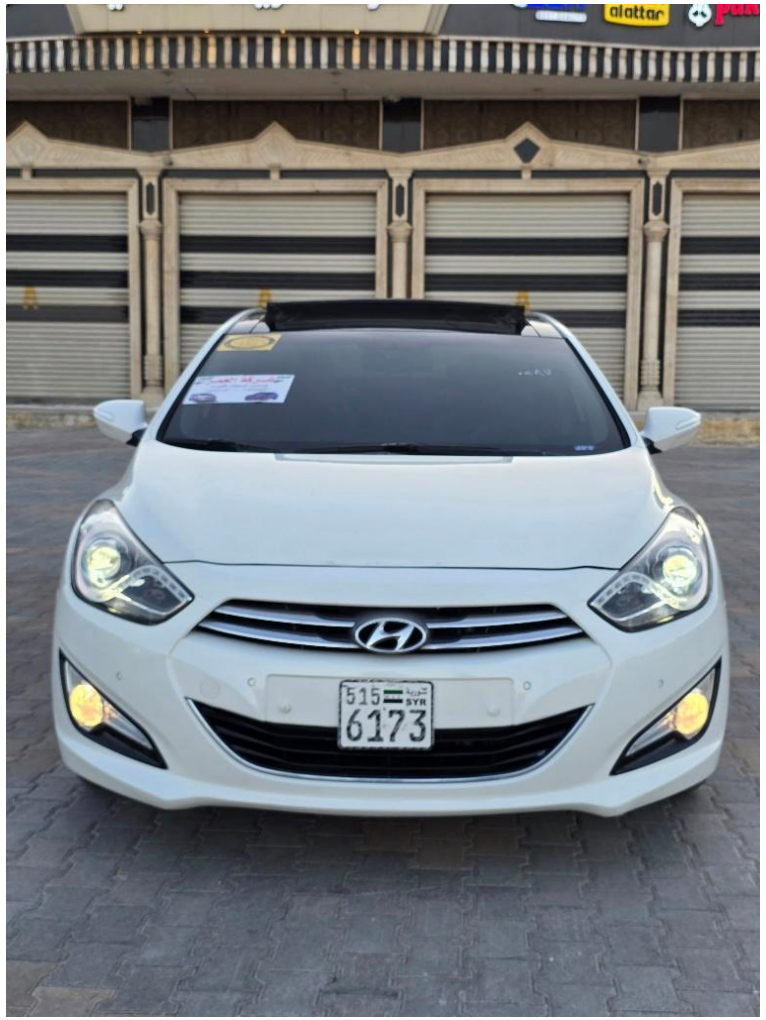
Detected License Plate

Ex3:

📊 Evaluation Results

Ground Truth : 5156173
OCR Output : 6173
Accuracy : 0.143
CER : 0.429

Detected License Plate



Using Qwen3 VL:2b.
Ex 1:

- ground truth: 5077237
- Qween model output: 50777237
- Accurecy: 1.00
- CER(character error rate): 0.0

extract this car plate number

Thought for 17.6 seconds

The car plate number is 5077237.

extract this car plate number

Thought for 9.5 seconds

The car plate number is 510 3682.

- ground truth: 5103682
- Qween model output: 510 3682
- Accurecy: 1.00
- CER(character error rate): 0.0

give me this car plate number

Thought for a moment

The car plate number is 233622.

- GROUND TRUTH: 233622
- QWEEN MODEL OUTPUT: 233622
- ACCURECY: 1.00
- CER(CHARACTER ERROR RATE):0.0

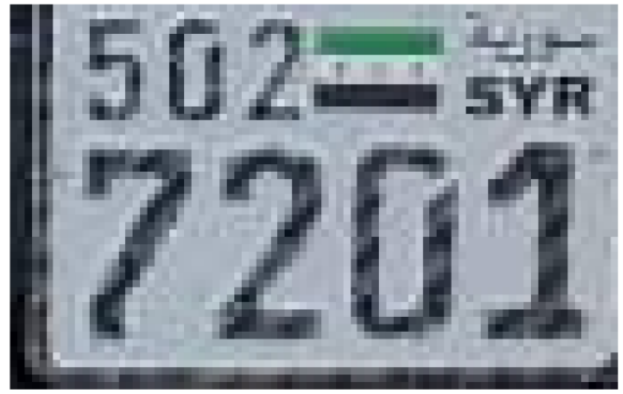
Using Tesseract OCR: Ex1:

📊 Evaluation Results

```
----------------------
Ground Truth : 5027201
OCR Output   : ml aed
Accuracy     : 0.0
CER          : 1.0
```

Using Tesseract OCR with YOLOV8:

Detected License Plate



📄 Extracted Text:

📊 Evaluation Results

```
----------------------
Ground Truth : 5027201
OCR Output   :
Accuracy     : 0.0
CER          : 1.0
```

Summary of Key Findings

Through the experiments conducted on various models for license plate recognition, we observed the following results:

EasyOCR, when used alone, did not provide satisfactory results. It only managed to extract part of the license plate number correctly, while the remaining digits were incorrectly recognized. This led to partial and inaccurate extraction of the full license plate number.

When EasyOCR was combined with YOLO for license plate detection, another issue arose: the model was able to read only the first part of the license plate or the second part, but not the entire number at once. This resulted in incomplete readings, reducing the overall accuracy in extracting the full license plate number.

On the other hand, Qwen3 VL:2b, which was the primary model in our study, proved excellent performance. It was able to accurately extract the full license plate number in every test, even for vehicles with license plates containing both Arabic and English numbers. We tested the model in various image conditions, and it consistently extracted the number correctly, highlighting its high accuracy and robustness for license plate recognition tasks.

Tesseract OCR also failed to produce satisfactory results, whether tested alone or combined with YOLO. It could not accurately extract the license plate numbers in either scenario.

In conclusion, the results indicate that Qwen3 VL:2b is the most reliable model for license plate recognition in our study. It provides high accuracy and eliminates the need for additional detection or OCR models, outperforming the other models tested in terms of both accuracy and CER(character error rate)

Summary of Key Findings

Through the experiments conducted on various models for license plate recognition, we observed the following results:

EasyOCR, when used alone, did not provide satisfactory results. It only managed to extract part of the license plate number correctly, while the remaining digits were incorrectly recognized. This led to partial and inaccurate extraction of the full license plate number.

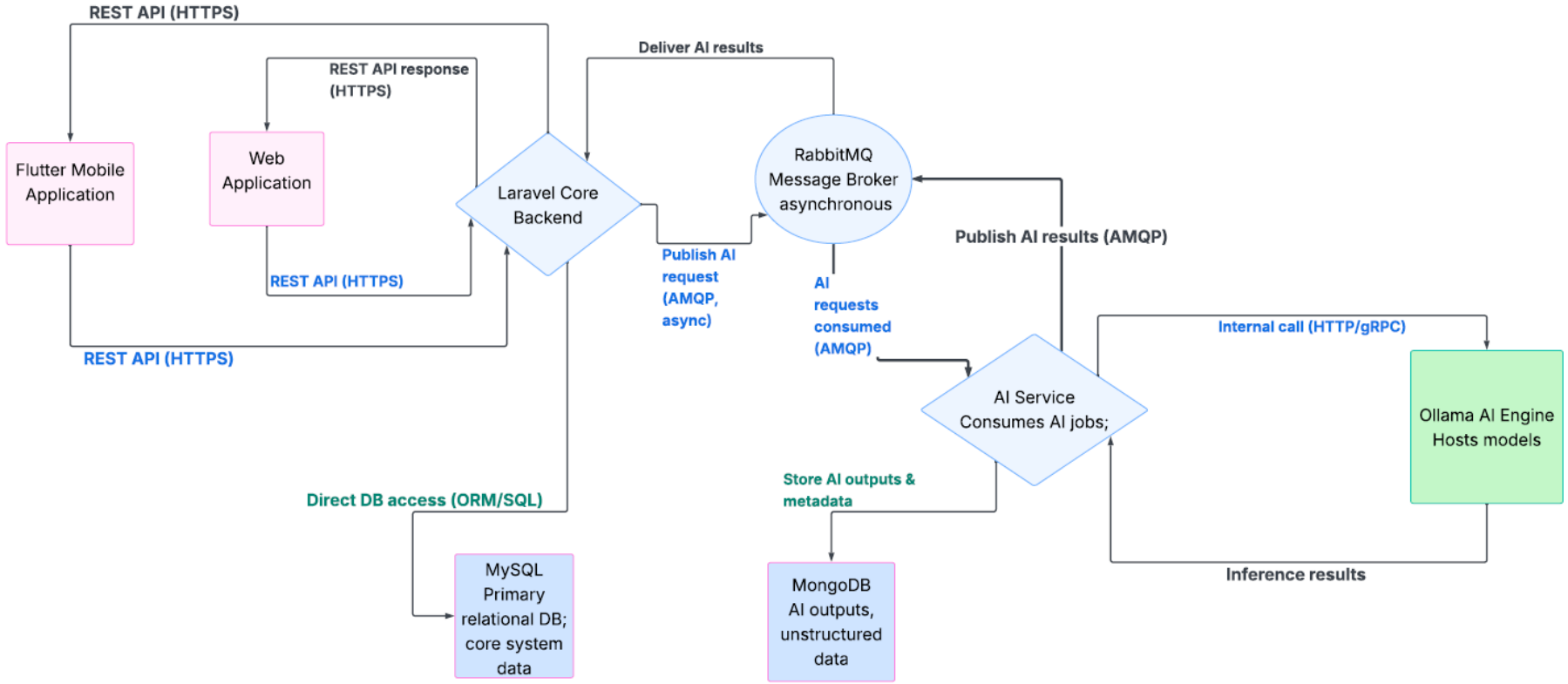
When **EasyOCR** was combined with **YOLO** for license plate detection, another issue arose: the model was able to read only the first part of the license plate or the second part, but not the entire number at once. This resulted in incomplete readings, reducing the overall accuracy in extracting the full license plate number.

On the other hand, **Qwen3 VL:2b**, which was the primary model in our study, proved excellent performance. It was able to accurately extract the full license plate number in every test, even for vehicles with license plates containing both Arabic and English numbers. We tested the model in various image conditions, and it consistently extracted the number correctly, highlighting its high accuracy and robustness for license plate recognition tasks.

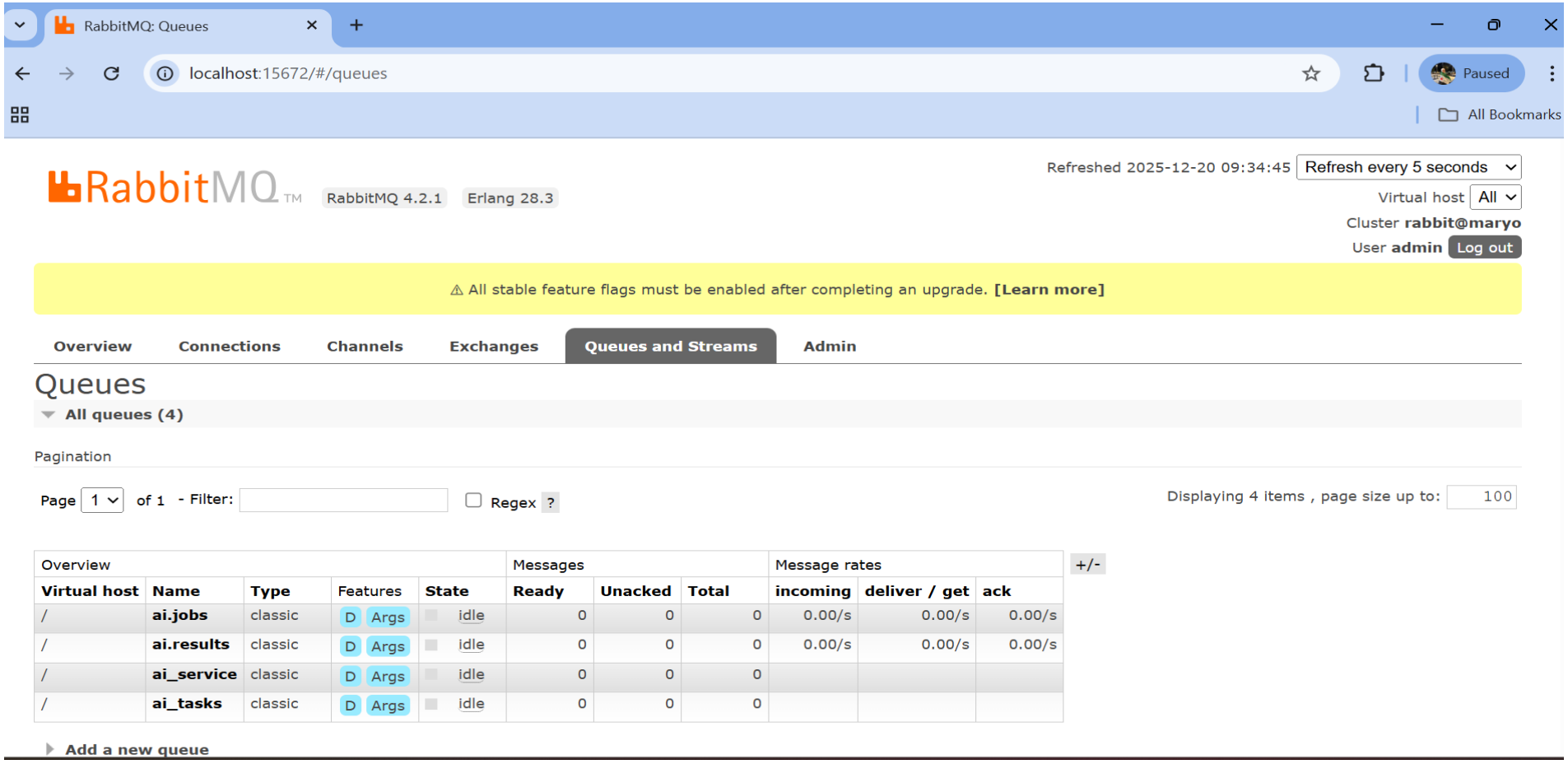
Tesseract OCR also failed to produce satisfactory results, whether tested alone or combined with **YOLO**. It could not accurately extract the license plate numbers in either scenario.

In conclusion, the results indicate that **Qwen3 VL:2b** is the most reliable model for license plate recognition in our study. It provides high accuracy and eliminates the need for additional detection or OCR models, outperforming the other models tested in terms of both accuracy and CER(character error rate)

How the component communicate with each other :

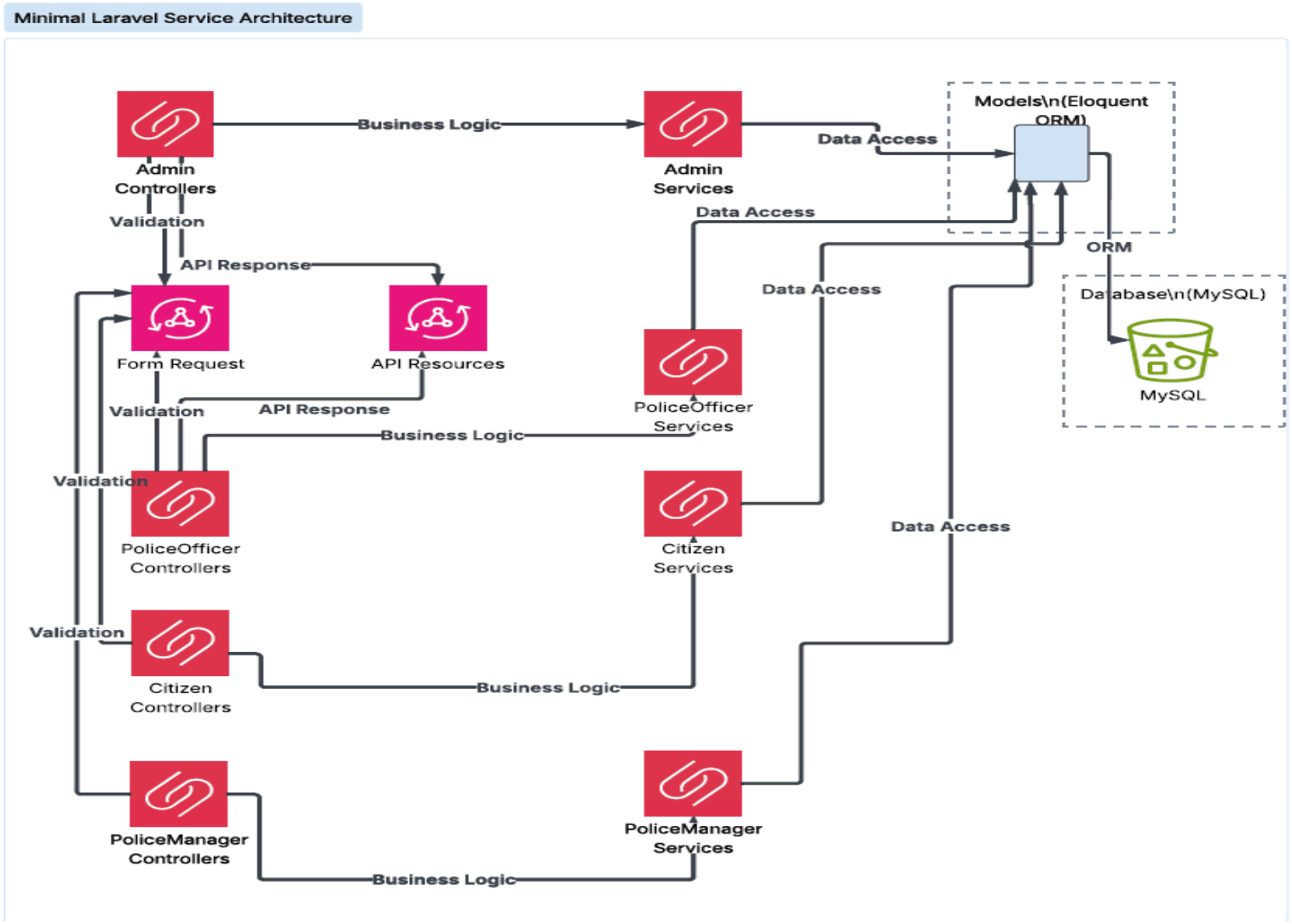


Rabbit MQ

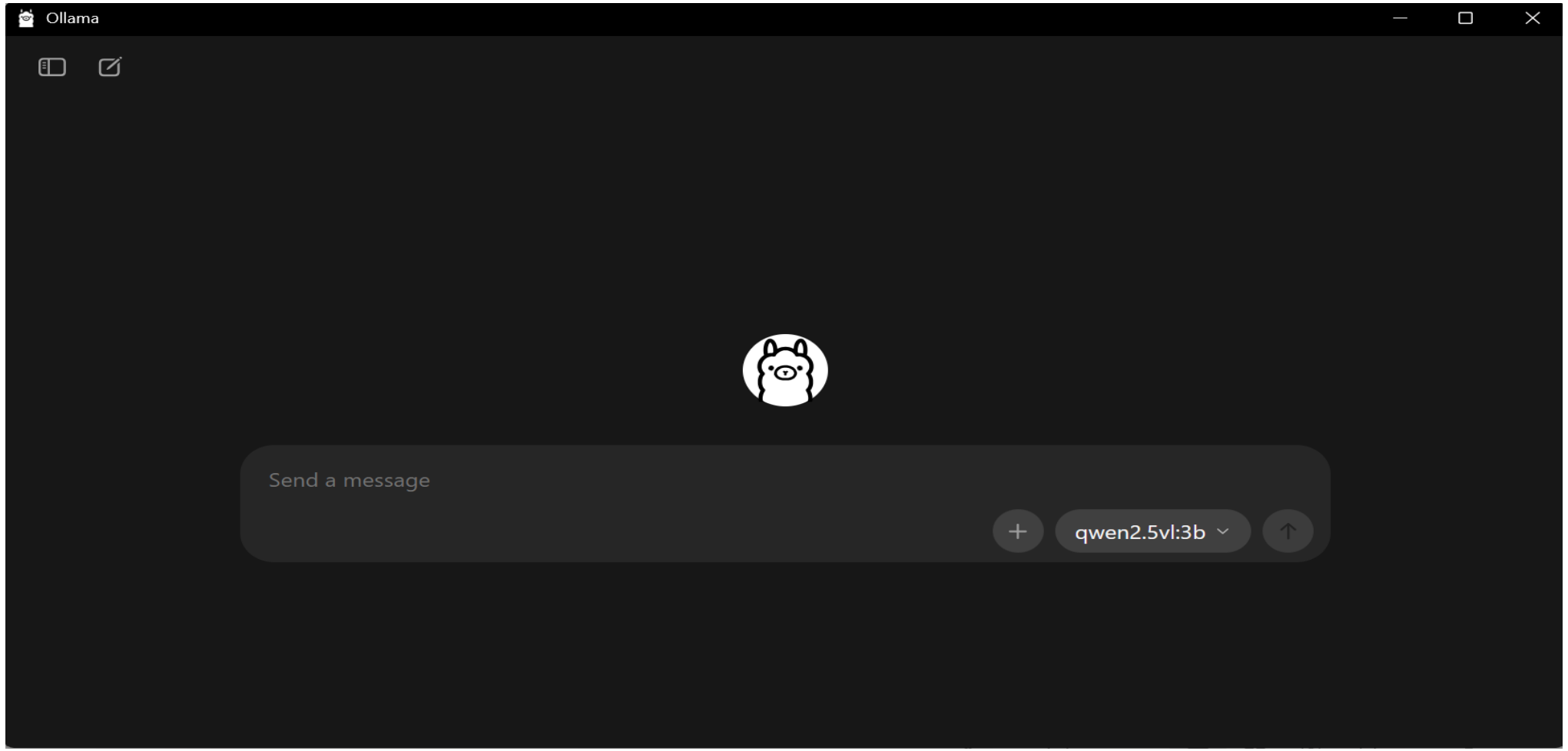


Overview					Messages			Message rates		
Virtual host	Name	Type	Features	State	Ready	Unacked	Total	incoming	deliver / get	ack
/	ai.jobs	classic	D Args	idle	0	0	0	0.00/s	0.00/s	0.00/s
/	ai.results	classic	D Args	idle	0	0	0	0.00/s	0.00/s	0.00/s
/	ai_service	classic	D Args	idle	0	0	0			
/	ai_tasks	classic	D Args	idle	0	0	0			

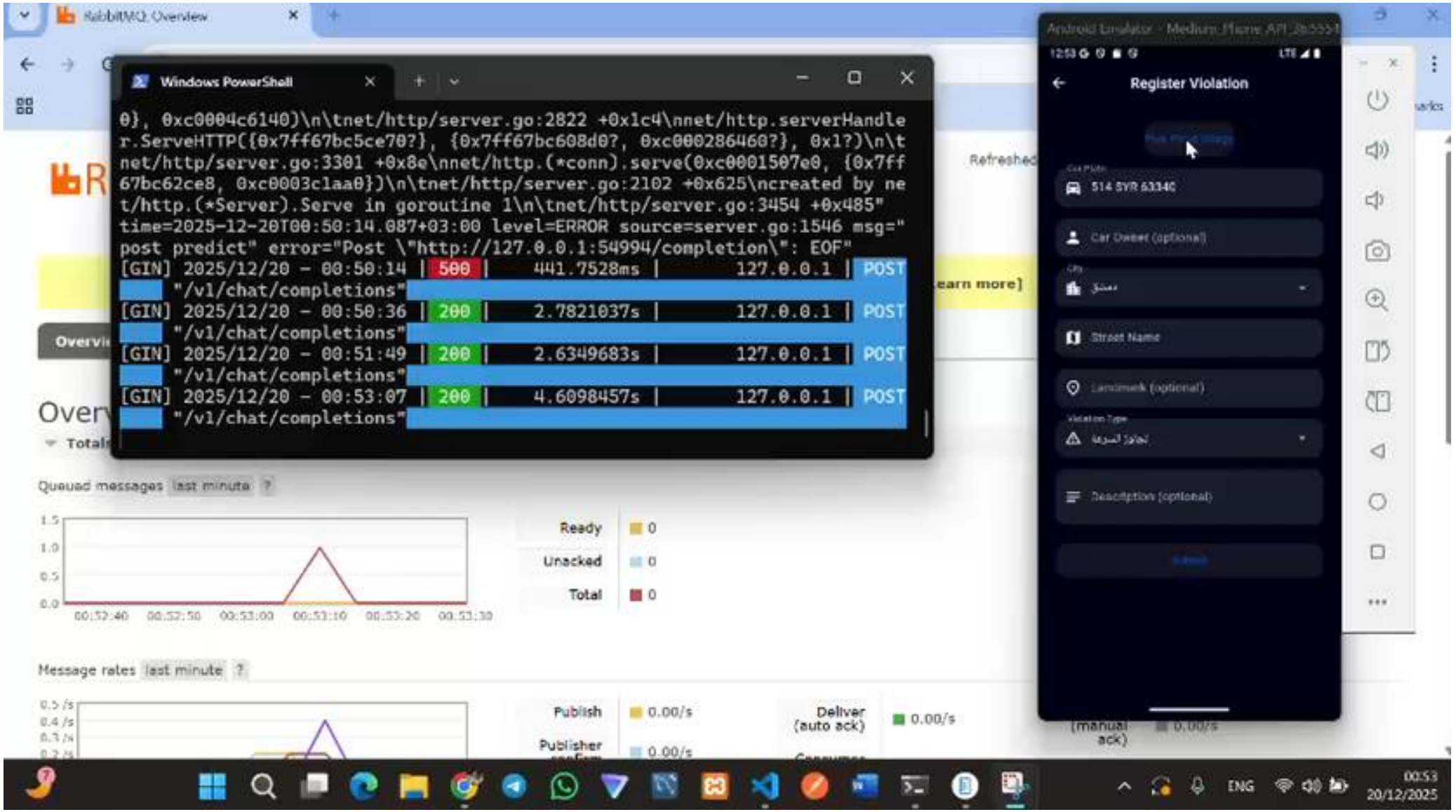
Core service : Laravel



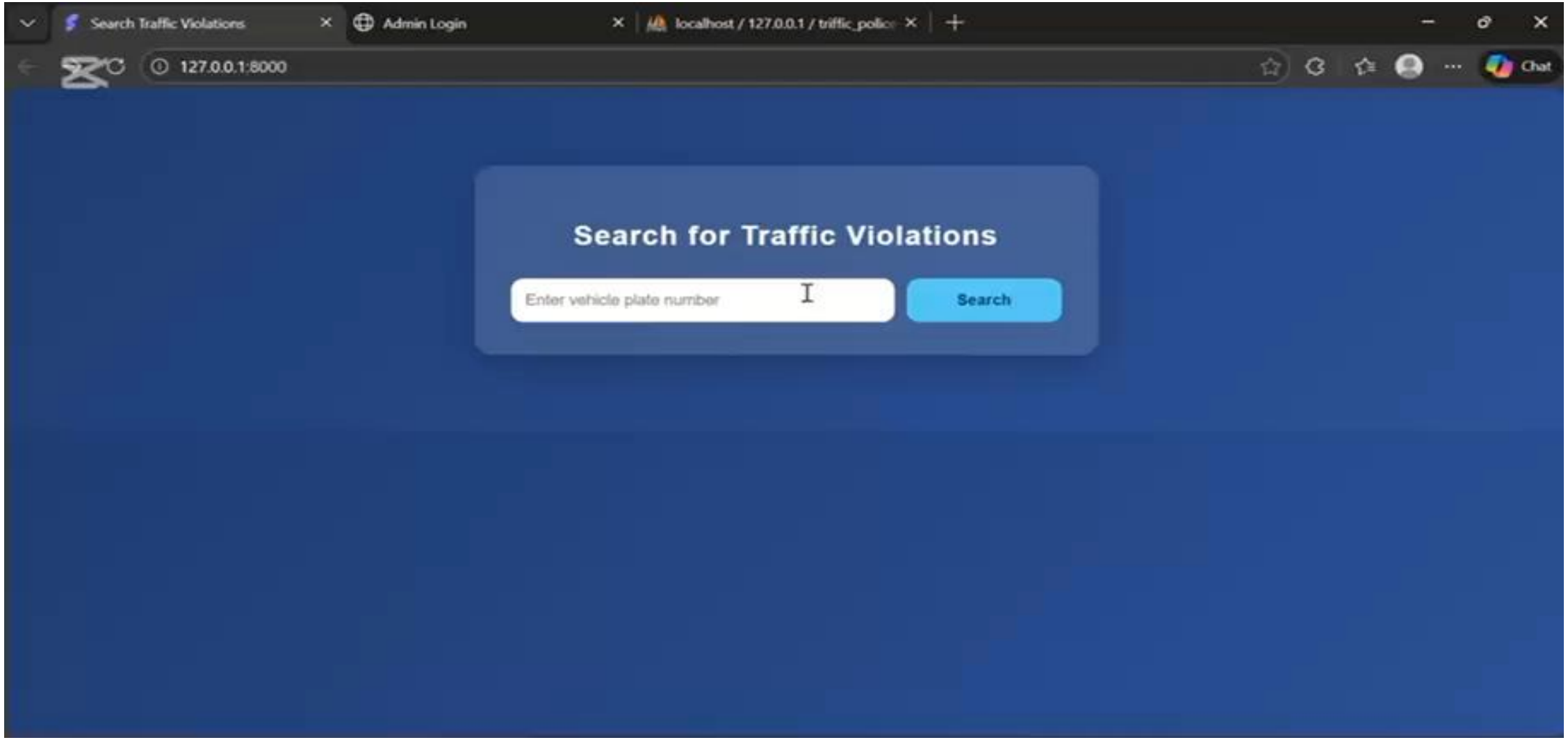
Ai initialize:ollama download , select the current qwen vision and serve



Ai process:


Windows PowerShell
0}, 0xc0004c6140)\n\tnet/http/server.go:2822 +0x1c4\nnet/http.serverHandle
r.ServeHTTP({0x7ff67bc5ce70?}, {0x7ff67bc608d0?, 0xc000286460?}, 0x1?)\n\t
net/http/server.go:3301 +0x8e\nnet/http.(*conn).serve(0xc0001507e0, {0x7ff
67bc62ce8, 0xc0003c1aa0})\n\tnet/http/server.go:2102 +0x625\ncreated by ne
t/http.(*Server).Serve in goroutine 1\n\tnet/http/server.go:3454 +0x485"
time=2025-12-20T00:50:14.087+03:00 level=ERROR source=server.go:1546 msg="
post predict" error="Post \"http://127.0.0.1:54994/completion\": EOF"
[GIN] 2025/12/20 - 00:50:14 | 500 | 441.7528ms | 127.0.0.1 | POST
"/v1/chat/completions"
[GIN] 2025/12/20 - 00:50:36 | 200 | 2.7821037s | 127.0.0.1 | POST
"/v1/chat/completions"
[GIN] 2025/12/20 - 00:51:49 | 200 | 2.6349683s | 127.0.0.1 | POST
"/v1/chat/completions"
[GIN] 2025/12/20 - 00:53:07 | 200 | 4.6098457s | 127.0.0.1 | POST
"/v1/chat/completions"
Register Violation
Queued messages last minute
Ready 0
Unacked 0
Total 0
Message rates last minute
Publish 0.00/s
Deliver (auto ack) 0.00/s
ENG
00:53
20/12/2025

Web demo:


Search Traffic Violations
Admin Login
127.0.0.1:8000
Chat
Search for Traffic Violations
Enter vehicle plate number
Search

App demo :