Vehicle Registration System - Project Documentation

Tech Stack: -

Frontend: React.js, Tailwind CSS
Backend: Spring Boot, Microservices, Eureka Server, MySQL/PostgreSQL, Docker, Postman
Architecture: Microservice Architecture
Domain: Sri Lankan Vehicle Registration Service

1. Introduction

The Vehicle Registration System is a full-stack web application designed for Sri Lankan vehicle owners to register their vehicles online.
The system supports multiple Sri Lankan license plate formats “old”, “vintage”, and “modern” while ensuring accurate validation using backend algorithms.
The project consists of:

- ✓ React.js with Tailwind CSS frontend.
- ✓ Spring Boot microservice backend (with Eureka for service discovery).
- ✓ MySQL and PostgreSQL database.
- ✓ Docker containers to run databases.
- ✓ Postman for API testing.

2. System Architecture

2.1 Microservices

Gateway Service
Acts as the single-entry point for all frontend requests.

Registry Service
Handles CRUD operations for Owners registrations.

Vehicle Service
Handles CRUD operations for Vehicle registrations.

Eureka Server
For Service discovery for all microservices and interservice awareness.

Contains logic for:
Determining plate type (Old / Vintage / Modern)
Validating Sri Lankan vehicle license plate formats.

2.2 Database

PostgreSQL - Registry Service [firstName, middleName, lastName, address, mobileNumber, nic]
MySQL - Vehicle Service [numberPlate, plateType, vehicleColor, vehicleType, ownerNic]

This system uses Docker containers to run both the MySQL and PostgreSQL databases. Docker ensures that all microservices operate in a consistent, isolated, and unified environment without manual database setup. (No need to install MySQL or PostgreSQL locally.)

3. License Plate Formats (Sri Lanka)

Sri Lankan license plates have 3 categories:

3.1 **Old**

13 ශ්‍රී 9999

Format:

Two digits
Sinhala Letter "ශ්‍රී" symbol
Four digits
Example: 11 ශ්‍රී 2345

3.2 **Vintage**

250-9999
19-9999
250 9999

Format:

2-3 digits
Dash or space (optional extra spaces allowed)
Four digits
Examples: 250-1234, 19 2345, 250 2345

3.3 **Modern**

WP GA-9999
CAR-9999

Format:

Province code (1-3 letters) or vehicle class
Space or optional separator
1-3 letters
Dash or space
Four digits
Examples: WP GA 3456, CAR-1234, PA 9876

4. Backend Functional Requirements

4.1: Task 1 – Determine Plate Type

A backend function receives a String licensePlate and returns one of: "OLD", "VINTAGE", "MODERN"

4.2: Task 2 – Validate License Plate (Regex Based)

The backend must verify that a given string follows Sri Lankan license plate formats.

Regex Patterns:

OLD ➔ "^[0-9]{2}\\s*ශ්‍රී\\s*[0-9]{4}$"
VINTAGE ➔ "^([0-9]{2,3})\\s*-?\\s*([0-9]{4})$"
MODERN ➔ "^(?:[A-Z]{2}\\s)?([A-Z]{2,3})\\s*-?\\s*([0-9]{4})$"

4.3: Task 3 – Validate NIC

The backend must verify that a given string follows Sri Lankan NIC formats.

Regex Pattern:

OLD_NIC ➔ "^[0-9]{9}V$"
NEW_NIC ➔ "^[0-9]{12}$"

5. Frontend Functional Requirements

5.1 – Collect user input

- ✓ Owner details
- ✓ Vehicle details
- ✓ License plate number validation
- ✓ NIC number verification

5.2 – Communicate with backend microservices

Using Axios requests routed through the Gateway Service:

Base URL - http://localhost:8080/api

For registry-service:

POST Request - /registry
GET Request (collections) - /registry
GET Request (a specific resource) - /registry/${id}
PUT Request - /registry/${id}

DELETE Request - /registry/${id}
GET Request (NIC validation) - /registry/validate?${id}

For vehicle-service:

POST Request - /vehicles
GET Request (collections) - /vehicles
GET Request (a specific resource) - /vehicles/${id}
PUT Request - /vehicles/${id}
DELETE Request - /vehicles/${id}
GET Request (number plate validation) - /vehicles/validate?${id}
GET Request (receive plate type) - /vehicles/classify?${id}

5.3 – Display validation results

Valid / Invalid license plates
Valid / Invalid NIC
Plate type automatically detected (Old / Vintage / Modern)

5.4 – Perform CRUD operations

Create new owner
Create new vehicle
Edit/update records
Fetch all records
Delete records

5.5 – Provide responsive UI

Tailwind CSS for layouts
React components for modular structure