Used Car Dealership:

Final Project Requirements Document

Overview

For this project, you will build a fully functional, **server-side rendered** web application for a **used car dealership**. The project will focus on backend development while ensuring **proper database design, code structure, and authentication**.

The application must be structured using **ESM (ECMAScript Modules)**, follow the **MVC pattern**, and be deployed on **Render** with a **PostgreSQL database**. Your site should include role-based authentication with different levels of access for **owners, employees, and users**.

You must aim to complete as much of this document as possible, with **70% being the absolute minimum to pass**.

Technology Stack

Your project must use the following technologies:

- **Node.js** with **Express.js** as the backend framework
- **EJS** for server-side rendering
- **ESM** (ECMAScript Modules) - No CommonJS (require is not allowed)
- **PostgreSQL** for the database
- **Deployed on Render** with a connected PostgreSQL database

---

Core Features

1. Home Page

- Displays featured vehicles.
- Displays links to vehicle categories (e.g., Trucks, Vans, Cars, SUVs).
- Users can click on vehicles to view their details.

2. Contact Page

- Includes a contact form where users can submit inquiries.

- Messages are stored in the database (not emailed).
- Employees can view messages in their dashboard.

3. Vehicle Browsing

- **View Vehicles by Category Page**
    - Displays all vehicles within a selected category (e.g., "All Trucks").
- **View Specific Vehicle Page**
    - Displays full details of a vehicle, including images, specifications, and price.
    - Logged-in users can leave **reviews**.
    - Users can edit and delete their own reviews.

4. User Authentication

- **Login Page**
- **Register Page**
- Uses **session-based authentication** (No JWT).
- Roles:
    - **Owner**: Full control over the system.
    - **Employee**: Limited permissions to manage inventory and user interactions.
    - **Standard User**: Can browse, submit inquiries, leave reviews, and track repair history.

5. User Dashboard

- Users can:
    - View their submitted reviews and edit/delete them.
    - See their **repair history** (if they have submitted vehicles for service).
    - Update their account information.

6. Admin Dashboard

- **Owner Dashboard**
    - Can add, update, and delete **vehicle categories**.
    - Can add, update, and delete **vehicles**.

    - Can manage **employee accounts**. Optional, can be hardcoded.
    - Can view all **customer inquiries (messages)**.
- **Employee Dashboard**
    - Can **edit some vehicle information** (e.g., pricing, availability, descriptions).
    - Can manage **vehicle reviews** (moderate/delete inappropriate ones).
    - Can view and manage **repair service requests**.

7. Vehicle Management

- **Only Owners** can:
    - Add, update, and remove **categories** (Truck, Van, Car, SUV, etc.).
    - Add, update, and remove **vehicles** from the system.
- **Employees** can:
    - Edit **vehicle details** (price, description, images, etc.).
    - Mark vehicles as **sold or available**.

8. Review System

- Logged-in users can:
    - **Leave reviews** on vehicle pages.
    - **Edit and delete** their own reviews.
- Reviews should be stored in the database and linked to users and vehicles.
- Employees can **moderate and remove inappropriate reviews**.

9. Repair Request & Tracking System

- Users can **submit a vehicle for repair** (similar to a ticket system).
- Repair requests go to an **employee dashboard**.
- Employees can update the **status of repairs** (Pending, In Progress, Completed).
- Users can see a **history of all their repair requests**.

10. Routing & Validation

- All routes requiring authentication must be protected.
- Implement proper **input validation**:

- Ensure user inputs are **sanitized** to prevent SQL injection.
- Ensure all required fields are properly validated before submitting data.

11. Deployment

- The project must be **deployed on Render**.
- The **PostgreSQL database** must be properly connected and configured.