SYNOPSIS

Name of Project :- VEHICLE SHOWROOM MANAGEMENT SYSTEM

Team Member : Pruthviraj Karale
Pratiksha Kalambe
Kartik Kalbande
Sakshi Gabhane

Front-end Technologies: HTML
CSS
JAVASCRIPT

Back-end Technologies: JAVA
My-SQL

Objective : The main Objective of Project are

1. **Inventory Management:**
   - Implement a robust inventory tracking system to manage the showroom's vehicle stock efficiently.
   - Allow staff to add new vehicles, update details, and remove sold units from the inventory.
2. **Sales Management:**
   - Facilitate smooth sales transactions by providing a sales module to record customer details, vehicle information, and sales transactions.
   - Generate invoices and receipts automatically, ensuring accuracy and reducing manual errors.
3. **Customer Relationship Management (CRM):**
   - Develop a CRM component to store customer information, including purchase history and preferences.
   - Enable the showroom staff to provide personalized services and promotions based on customer data.

4. **Reporting and Analytics:**
    - Incorporate reporting features to generate insightful analytics on sales performance, inventory turnover, and other key metrics.
    - Provide customizable reports to assist in decision-making and strategic planning.
5. **User Authentication and Access Control:**
    - Implement secure user authentication to ensure that only authorized personnel can access and modify sensitive information.
    - Define roles and permissions to control access levels for different staff members.

6. **User-Friendly Interface:**
    - Design an intuitive and user-friendly interface for easy navigation and efficient use by showroom staff.
    - Ensure responsive design to accommodate various devices and screen sizes.
7. **Integration with External Systems:**
    - Explore integration possibilities with external systems, such as finance and accounting software, to streamline overall business operations.
8. **Scalability and Maintenance:**
    - Build the SVMS with scalability in mind to accommodate future expansion and changes in business requirements.
    - Develop a maintenance plan to address updates, bug fixes, and system enhancements.

Page Details :

1. **Home Page (index.jsp):**
    - Welcome message and brief introduction to the Vehicle Management System.
    - Quick links to essential features like Inventory, Sales, and Reports.
2. **Inventory Management Page (inventory.jsp):**
    - **Vehicle List:**
        - Render a dynamic table displaying vehicle details.
        - Fetch and display data from the backend using Java Servlets or Spring controllers.
    - **Add/Edit Vehicle:**
        - Form for adding or editing vehicle details.
        - Submit the form to a Java Servlet or Spring controller for processing and updating the database.
    - **Delete Vehicle:**
        - Button or link triggering a servlet to delete the selected vehicle.
3. **Sales Management Page (sales.jsp):**
    - **Sales Transaction Form:**
        - HTML form for collecting customer and sales information.
        - Use a Java Servlet or Spring controller to process the form data and update the database.
    - **Sales History:**
        - Display sales history using JSP, fetching and presenting data from the backend.

- **Sales Analytics:**
    - Use JavaScript libraries for interactive charts based on sales data.

4. **Customer Relationship Management (CRM) Page (crm.jsp):**
    - **Customer List:**
        - Dynamic table displaying customer details using JSP.
    - **Customer Details:**
        - Page showing detailed customer information fetched from the backend.
    - **Communication Log:**
        - Display a communication log using JSP, fetching data from the backend.
5. **Reporting and Analytics Page (reports.jsp):**
    - **Customizable Reports:**
        - Forms in JSP to set report parameters and submit them to a Java Servlet or Spring controller.
        - Generate dynamic HTML or PDF reports based on backend processing.
    - **Visual Analytics:**
        - Use JavaScript libraries (e.g., D3.js) to create interactive charts.
6. **User Management Page (user-management.jsp):**
    - **User Roles and Permissions:**
        - Forms and tables using JSP to manage user roles and permissions.
        - Java Servlets or Spring controllers to handle user-related operations.
7. **Settings Page (settings.jsp):**
    - **System Configuration:**
        - Form using JSP for configuring system settings.
        - Process form data using Java Servlets or Spring controllers and update the backend.
8. **Help and Support Page (help.jsp):**
    - **User Guides:**
        - Display user guides and documentation using JSP.
    - **Contact Support:**
        - Provide contact information and a form for user support.