**Car Showroom Management System – Project Report**

**Project Title:** Car Showroom Management System  
**Submitted By:** Ahmed Raza (24k-1010), Simal Hassan (24k-0688), Laiba Jamil (24k-0812)  
**Course:** Object-Oriented Programming  
**Instructor:** Shafique - ur - Rehman  
**Submission Date:** 11th May 2025

**1. Executive Summary**

**Project Overview:**  
The Car Showroom Management System is a console-based C++ application developed using Object-Oriented Programming (OOP). The system aims to automate the operations of a car showroom by managing inventory, tracking sales, generating invoices, and handling customer information. The project addresses common issues in current showroom operations such as manual errors, poor inventory tracking, and inefficient data management.

**2. Introduction**

**Background:**  
Managing a car showroom manually or using spreadsheets is prone to errors and inefficiency. This project introduces an automated solution using C++ and OOP principles to manage vehicles, customers, and sales in an organized and secure manner.

**Objectives of the Project:**

* Implement an OOP-based console application in C++.
* Design modules for inventory, customers, invoices, and reports.
* Enable role-based access (Admin, Salesperson, Customer).
* Utilize file handling for persistent data storage.

**3. System Description**

**Current System Issues:**

* Manual errors in tracking data.
* No real-time inventory management.
* Lack of customer relationship tracking.
* Security and access control issues.

**Innovations and Features:**

* Admin/sales/customer login roles.
* Dynamic car catalog management.
* Customer search by name/history.
* Real-time stock updates.
* Invoice generation with tax/discounts.
* Daily/weekly sales reports.

**4. Programming Approach and Methodology**

**OOP Techniques Used:**

* **Encapsulation** for securing object data.
* **Inheritance** for role hierarchy (e.g., Admin, Salesperson, Customer).
* **Polymorphism** for method overloading/overriding in transactions.
* **File Handling** for saving and retrieving customer, car, and sales data.

**Algorithm and Design:**

* Classes for Car, User, Customer, Invoice, and Report.
* Role-specific interfaces using base class pointers.
* Unique customer ID generation.
* Searching/sorting algorithms for records.

**Data Management:**

* Binary and text file handling.
* Structured files for customer data, inventory, and transaction logs.

**5. System Rules and Functions**

**User Roles and Permissions:**

* **Admin:** Full access to inventory and reports.
* **Salesperson:** Manage customer orders and generate invoices.
* **Customer:** View available cars and make purchases.

**Core Functional Modules:**

* **Inventory Management:** Add/edit/delete car entries.
* **Customer Management:** Search/update customer records.
* **Invoice Generation:** Calculate price, tax, and discounts.

**6. Implementation and Development**

**Development Process:**  
The system was developed in C++ using modular programming. Each feature was developed as a class with clearly defined responsibilities and then integrated into the main console application.

**Programming Languages and Tools:**

* **Language:** C++
* **Environment:** Dev-C++ / Visual Studio
* **Files:** Binary & text-based storage

**Challenges Encountered:**

* Implementing role-based access with secure input.
* Managing binary file read/write operations.
* Generating dynamic reports based on sales logs.

**7. Team Contributions**

* **Ahmed Raza (24k-1010):** Designed admin module, class architecture, and user interface.
* **Laiba Jamil (24k-0812):** Implemented customer management, search, and invoice logic.
* **Simal Hassan (24k-0688):** Developed file handling and integration of all modules.

**8. Results and Discussion**

* All project objectives were achieved.
* Role-based system worked effectively.
* Inventory and customer data are persistently stored using file handling.
* Invoices are generated with tax and discount logic.
* Reports identify top-selling models and daily/weekly sales summaries.

**9. References**

* GitHub repository
* GeeksforGeeks – OOP and File Handling in C++
* Stack Overflow discussions on binary file I/O
* Object-Oriented Programming course lectures and notes