## 

**Online Automotive Dealership System**

Venkata Mahesh Babu Korrapati

A Project Submitted to

GRAND VALLEY STATE UNIVERSITY

In

Partial Fulfillment of the Requirements

For the Degree of

Master of Science in Applied Computer Science

School of Computing

April 2024



The signatures of the individuals below indicate that they have read and approved the project of Venkata Mahesh Babu Korrapati in partial fulfillment of the requirements for the degree of Master of Science in Applied Computer Science.

Dr. Robert Adams, Project Advisor Date 25/04/2024

Dr. Robert Adams, Graduate Program Director Date 25/04/2024

# Abstract

This online car dealership system aims to provide a platform for customers to buy, sell, and test drive cars. The system is designed to make the process of car management and order fulfillment more efficient for both customers and the dealership. The system's main features include car management, order management, and test drive management. The system also includes a login and dashboard module for admin use.

To develop this system, HTML and CSS were used to design the page layout, while JavaScript was used for validation tasks and animations. Python was used for implementing the business logic, and NoSQL database was used as the database for the project. The system was developed over the Django Framework.

The purpose of this project is to create a user-friendly and efficient online platform that streamlines the car buying and selling process. As of now, the development of the app is complete, and the app is ready for deployment.

# Project Management

**Introduction to the Project Management Approach**

**1. Project Initiation and Planning:**

- Project Scope : Having a platform where cars, trucks, and other vehicles may be sold online is essential in today's intensely competitive automotive dealer market. An automotive dealers management system was created to manage the store and sell automobiles online. An automobile store's staff can use this application to manage customers' orders and make it easier for customers to place their own. By categories, the vehicles in the shop are arranged. Each car is assigned a category, name, cost, and description. The staff can log in and manage the category details, product details, customer details, and order details along with the information about order date and time, bill amount, payment status, and delivery status. The following customer information will be maintained: Cust-Id, Name, Delivery Address, Phone number, and Email. The customer must register and log in to access the product page with the automobiles listed. The main point of developing this system is to help Dealers to manage their business and help customers with online ordering.

- The project has build in a scrum model which is an agile software development framework. And used the JIRA as a workflow management tool.

- Using JIRA Scrum, I have created a backlog of tasks, breaking down my project into manageable user stories.

**2. Sprint Planning:**

- With Scrum, I have planned my sprints, selecting user stories from the backlog to work on in each sprint.

- I have estimated the effort required for each user story using story points.

**3. Development:**

- As a solo developer, I have been responsible for designing and implementing the front-end and back-end components of the system.

- I used the technologies HTML,CSS,Javascript to build the UI(User Interface).all the backend business logic had written in python programming language and the project has been developed over the django framework. for Database I used the MySQL which is Relational Database Management System to solve complex queries.

**4. Deployment:**

- I took the AWS which is a cloud infrastructure service to host my application.

# Organization

The main architecture of application lies behind the features of the project. The system is a standalone web application. It follows a layered architecture with presentation, business logic, and data access layers. The presentation layer includes the user interface for interacting with the system. The business logic layer contains the modules for car management, order management, and test drive management. The data access layer handles database interactions for storing and retrieving data. The system can be deployed on a single server or in a cloud environment. Communication between modules is done via well-defined interfaces to ensure modularity and maintainability. The system consists of several modules:

**Login Module**: This module handles admin login functionality.

Interface: Login screen for entering username and password.

Functionality: Authenticates admin credentials.

**Logout Functionality**: Allows the admin to log out of the system.

**Dashboard:** Provides an overview of all Car and Car details.

Interface: Admin dashboard displaying relevant statistics and data.

Functionality: Displays summary information and links to other modules.

**Car Management Module:**

Adding New Car Details: Allows the admin to add details of a new car.

Edit the Existing Car Details: Allows the admin to modify existing car details.

View all the details of the Car: Displays all details of a specific car.

Listing of all Cars: Displays a list of all cars.

Interface: Forms for adding and editing car details, and a list view for displaying cars.

**Order Management Module:**

Adding New Order Details: Allows the admin to add details of a new order.

Edit the Existing Order Details: Allows the admin to modify existing order details.

View all the details of the Order: Displays all details of a specific order.

Interface: Forms for adding and editing order details, and a list view for displaying orders.

**Test Drive Management Module:**

Adding New Test Drive Details: Allows the admin to add details of a new test drive.

Edit the Existing Test Drive Details: Allows the admin to modify existing test drive details.

View all the details of the Test Drives: Displays all details of a specific test drive.

Interface: Forms for adding and editing test drive details, and a list view for displaying test drives.

# Reflection

The initial stage of the project is about the analysis of the requirements, understanding requirements in a natural language and transforming those requirements into diagrammatical representation of the whole system. The diagrammatical representations that I considered for this project are activity diagram and use-case diagram because of their unique advantages.

**Activity Diagram**

Activity diagrams can be useful in this online car dealership system to visually represent the flow of activities that occur within the system.

Activity diagrams can help to illustrate the steps involved in different processes within the system, such as adding a new car to the inventory, editing an existing car's details, or placing an order. By providing a visual representation of the steps involved in a process, activity diagrams can help to identify potential inefficiencies or bottlenecks in the system. In addition to helping with system design and optimization, activity diagrams can also be useful for communicating the system's functionality to stakeholders, such as developers, managers, or customers. Activity diagrams can provide a clear and concise overview of the system's processes, which can help stakeholders to understand how the system works and how it can be improved.

**A diagram of a company

Description automatically generated**

**Use Case Diagram**

A use case diagram can be useful in this online car dealership system to provide a high-level view of the system's functionality and how it interacts with its users. Use case diagrams can help to identify the different types of users who interact with the system, such as customers, dealers, or administrators, and the different use cases or scenarios in which they interact with the system.

For example, a use case diagram could include use cases such as "place an order," "view car details," or "schedule a test drive." These use cases can help to define the different functionalities of the system and how they relate to each other.

A diagram of a person's face

Description automatically generated

**Purpose of Database and Database model**

The purpose of the database in this automotive dealer management system is to store and manage essential data related to vehicles, customers, orders, and test drives. It serves as the backbone of the system, facilitating data retrieval, storage, and manipulation. The database is needed to:

● Store vehicle details: This includes information such as category, name, cost, and description of each vehicle, which is crucial for listing and managing the inventory of the automotive dealer.

● Store customer information: It should maintain customer data, including Cust-Id, Name, Delivery Address, Phone number, and Email, enabling the system to associate customers with their orders and contact them for updates or promotions.

● Store order details: The database should record order information, including order date and time, bill amount, payment status, and delivery status. This allows for order tracking and financial management.

● Store test drive details: It should also keep track of test drive information, such as the date, time, and details of the test drive. This is important for managing and scheduling test drives for potential customers.

**Input Data Available to the Database:**

● Vehicle details: Category, name, cost, description.

● Customer information: Cust-Id, Name, Delivery Address, Phone number, Email.

● Order details: Order date and time, bill amount, payment status, delivery status.

● Test drive details: Date, time, and related information for test drive scheduling.

A computer screen shot of a computer

Description automatically generated

**Technologies used**

● HTML: The page layout has been designed in HTML

● CSS: CSS has been used for all the designing part

● JavaScript: All the validation task has been developed by JavaScript

● Python: All the business logic has been implemented in Python

● RDBMS: MySQL has used as a database for this project

● Django: Project has been developed over the Django Framework

**Implementation**

**Home Screen**

**<add screen from your system>**

This is the landing page of the application where user can register an account or login here.

A red car parked on grass

Description automatically generated

**Login Screen**

**<add screen from your system>**

This is the login screen for the registered user and also users can navigate to the signup screen to register themselves as new user.

**Register Screen**

**<add screen from your system>**

This is a signup screen for new users.

A screenshot of a computer

Description automatically generated

**Cars Screen**

**<add screen from your system>**

This is the car's screen where the user can see all the available cars for sale and also they can filter the cars based on the category and fuel.

A screenshot of a computer

Description automatically generated

**<add screen from your system>**

This screen is showing the details of the car selected in the previous screen. On this screen, users can book a test drive and order the same car.

**Compare cars**

**<add screen from your system>**

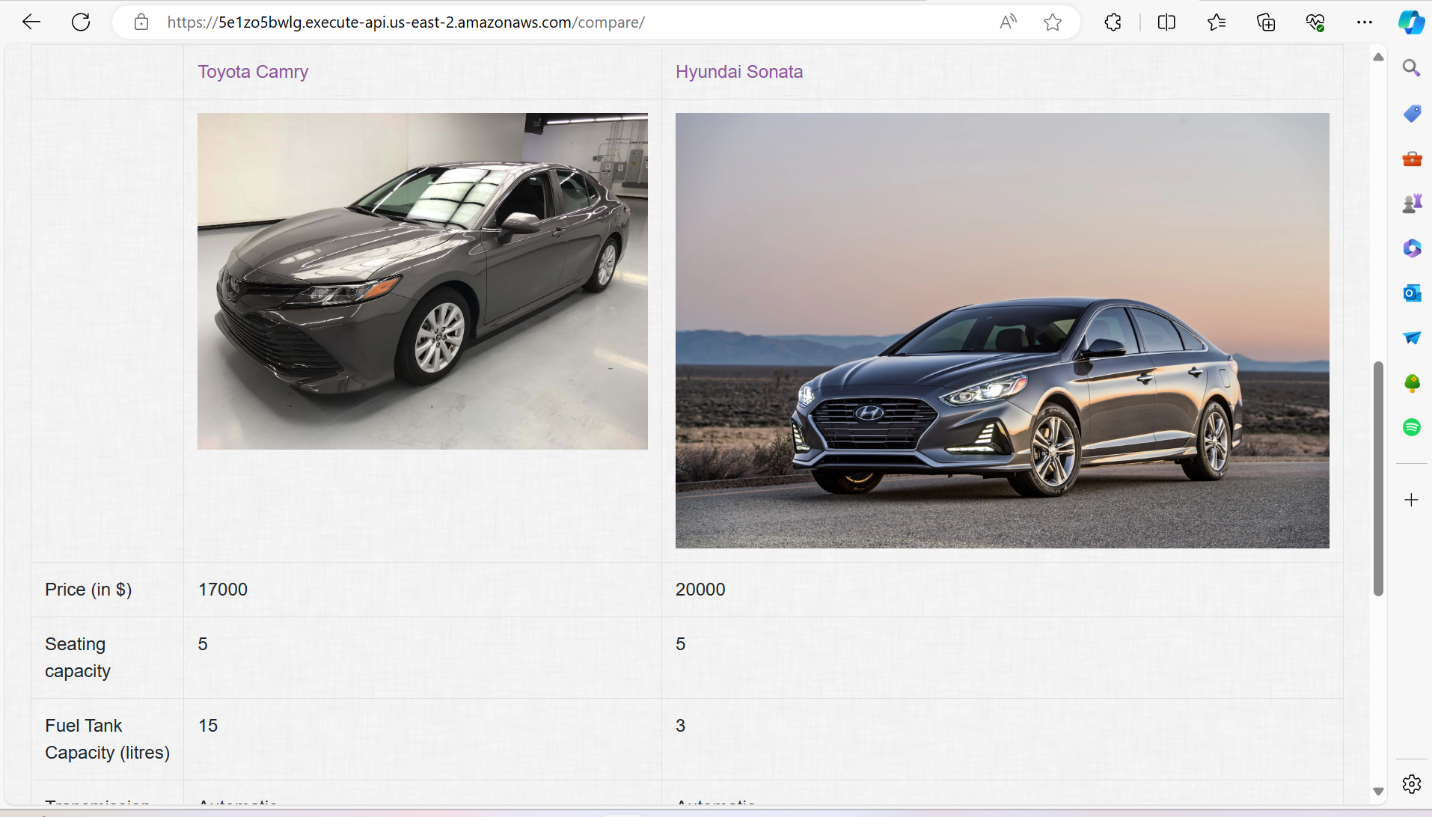
By clicking the compare nav item, the user can reach this page where the user can compare two different cars to understand the features.

A screenshot of a computer

Description automatically generated

**<add screen from your system>**

This is how the features are compared like this in a table where the user can also order or book a test drive from this screen itself. When the user submitted the form, the application says the success message on the screen to acknowledge the booking confirmation



**Screen**

**<add screen from your system>**

When a user clicks the book an order button, this dialog will pop up to get the data from the user to book an order for the car.

**<add screen from your system>**

When the user submitted the form, the application says the success message on the screen to acknowledge the booking confirmation.

**<add screen from your system>**

Once the user ordered a car, they can see the request and status of the request on the user profile page.

**Admin Login**

**<add screen from your system>**

Console to log in the Admin to manage the data in the application

A screenshot of a computer

Description automatically generated

**Adding Cars**

**<add screen from your system>**

This is the admin home screen where the admin can see the users registered, cars available, users booked for test drives and orders users made on the application with the information they entered

A screenshot of a computer

Description automatically generated

**<add screen from your system>**

The screen for adding and editing cars in this online car dealership system would typically include a form for the admin to input or edit the car details. The form may include fields for information such as the car's make, model, year, color, mileage, price, and any additional features or specifications.

To add a new car, the admin would fill out the form with the relevant details and submit it to the system. The system would then add the new car to the inventory.

A screenshot of a computer

Description automatically generated

**Editing Cars**

**<add screen from your system>**

To edit an existing car, the user would select the car from the list of existing cars, which may be displayed in a table or grid format, and then select the "edit" option. This would bring up the same form as used for adding a new car, but with the existing car's details pre-populated in the form fields. The admin could then make any necessary changes to the details and submit the updated form to the system, which would then update the car's details in the inventory.

**View/edit Orders**

**<add screen from your system>**

The screen where an admin can view and edit orders for a car ordered by a user on an application would typically include a variety of information and functionality to help the admin manage and track orders efficiently. The screen would likely display a list of all orders that have been placed, along with relevant details such as the order ID, customer name, car model, and order status.

A screenshot of a computer

Description automatically generated

**<add screen from your system>**

**View/edit Test drives**

**<add screen from your system>**

The screen where an admin can view and edit the test drive requests for a car requested from a user would typically include a table or list of all the test drive requests received by the dealership for a particular car. Each row in the table would represent a single test drive request and would display information such as the user's name, contact details, preferred test drive date and time

From this screen, the admin would be able to view and manage the test drive requests for the car. The admin could edit or cancel a test drive request if needed, by clicking on the appropriate action button next to each test drive request. The admin could also add new test drive requests for the car, by clicking on the "Add Test Drive Request" button.

**Infrastructure support(AWS)**

* AWS offers a wide range of services that can be used for deployment, including computing power (Amazon EC2), storage (Amazon S3), databases (Amazon RDS), and networking (Amazon VPC).
* I took AWS cloud services as the infrastructure support which means leveraging the various services offered by Amazon Web Services (AWS) to build and host my application and services.

**AWS Deployment**

1. Amazon EC2 (Elastic Compute Cloud):

a. EC2 instances will host our Django application.

2. Amazon RDS (Relational Database Service):

a. RDS is used to host our database.

3. Amazon VPC (Virtual Private Cloud):

a. Set up a Virtual Private Cloud to isolate our resources and provide network-level security.

4. Amazon Route 53:

a. Use Route 53 for domain registration and DNS management. It can route incoming traffic to your EC2 instances.

5. AWS Identity and Access Management (IAM):

a. Create IAM roles and policies to control access and permissions for AWS resources.

6. Elastic IP address

a. Amazon EC2 instance will have a public IP address

# Conclusions

In conclusion, this online car dealership system offers several advantages to both customers and the dealership. The system provides a user-friendly interface for customers to buy, sell, and test drive cars, which can increase sales and customer satisfaction. Additionally, the system's features, such as car management and order fulfillment, can help streamline dealership operations, leading to more efficient business processes.

However, there are also some potential disadvantages to consider. The use of a NoSQL database may not be suitable for large-scale operations, and additional testing may be required to ensure the system's scalability. Additionally, the system's reliance on JavaScript may lead to performance issues on older or less powerful devices.

For future work, the system could be expanded to include additional features, such as a recommendation engine to suggest cars based on customer preferences. The system could also be integrated with other third-party applications to enhance its functionality. Overall, this online car dealership system shows promise in improving the car buying and selling experience for both customers and dealerships and with further development and refinement, it could become an indispensable tool for the automotive industry.

# Appendices

View at URL : <https://5e1zo5bwlg.execute-api.us-east-2.amazonaws.com/>

Admin Dashboard : <https://5e1zo5bwlg.execute-api.us-east-2.amazonaws.com/admin>