WhatNext Vision Motors: Shaping the Future of Mobility with Innovation and Excellence

Project Overview

WhatNext Vision Motors, a rising innovator in the automotive industry sought to elevate its customer interactions and streamline internal operations through the implementation of customized Salesforce CRM solution. The primary focus of the project was to streamline vehicle order management, ensure accurate dealer assignment, and automation.

The manual ordering process brought difficulties, the solution was to create features that enhance vehicle ordering workflow. The system intelligently recommends the nearest dealer based on the customer's location, integrates real-time stock validation, minimizing errors and ensuring a smoother, more reliable experience. Automated status updates for bulk orders further strengthen operational accuracy that improves order tracking and customer communication.

Objectives

The main objective of this project is to implement and develop a customized salesforce CRM for WhatNext Vision Motors to enhance the customer experience and streamlining its operational processes.

By developing a centralized system to manage order creation, dealer assignment, and stock validation, the project aims to:

- **Automate Order and Dealer Assignment** by automatically assigning the nearest dealer based on customer's location.
- **Prevent Out-of-Stock Orders** by restricting the customer to place an order for the vehicle that is unavailable.
- **Send Test Drives Reminder** through automated emails that notify customers of a scheduled test drive.
- **Improve Customer Engagement and Experience** with the use of Lightning App and Dynamic Forms which makes clean and intuitive user interface.
- **Scalability of Backend** through Apex Classes for scheduled batch jobs to automate stock updates and order confirmations in bulk.

Phase 1: Requirement Analysis and Planning

The initial phase of the WhatNext Vision Motors project was to focus on planning the business needs and translates it into system requirements using the capabilities of Salesforce environment. The goal is to develop a CRM that supports vehicle ordering workflows.

Business Requirements

The following are the business requirements:

- Centralized data of vehicle, dealers, and orders.
- The vehicle stocks are automatically updated when an order is being placed.
- Dealers were automatically assigns based on customers location.
- Automatically notify customers about their test drives scheduled.
- Monitor vehicle service requests.
- Automation of key workflows to reduce human intervention.

Defining Project Scope

To meet business needs, the system was designed to have:

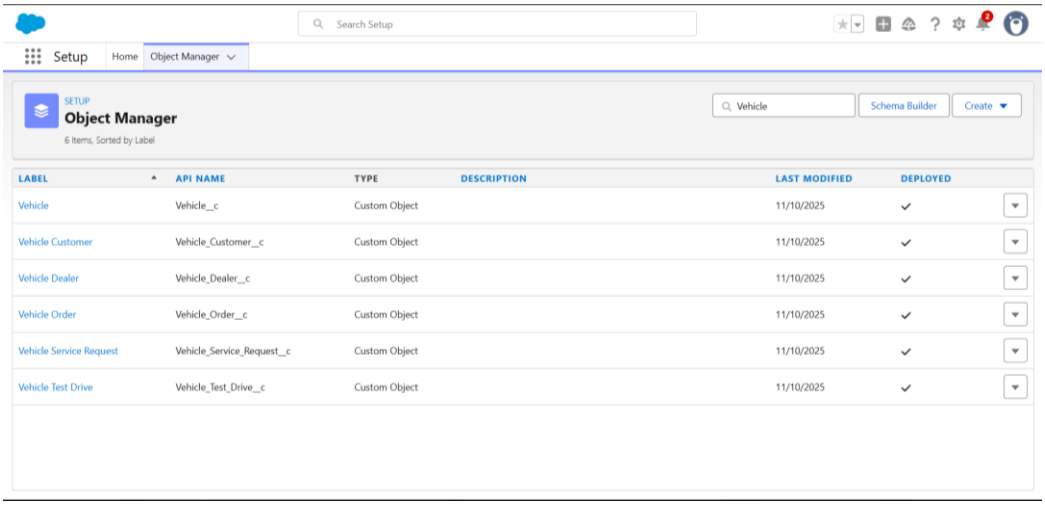
- Custom fields and objects for managing vehicles, orders, customers, dealers, test drives, and service requests.
- Record-triggered flows to automatically assigns dealer and send test drives notification.
- Apex triggers for stock availability and updates.
- Batch Apex to process pending orders based on stock availability.

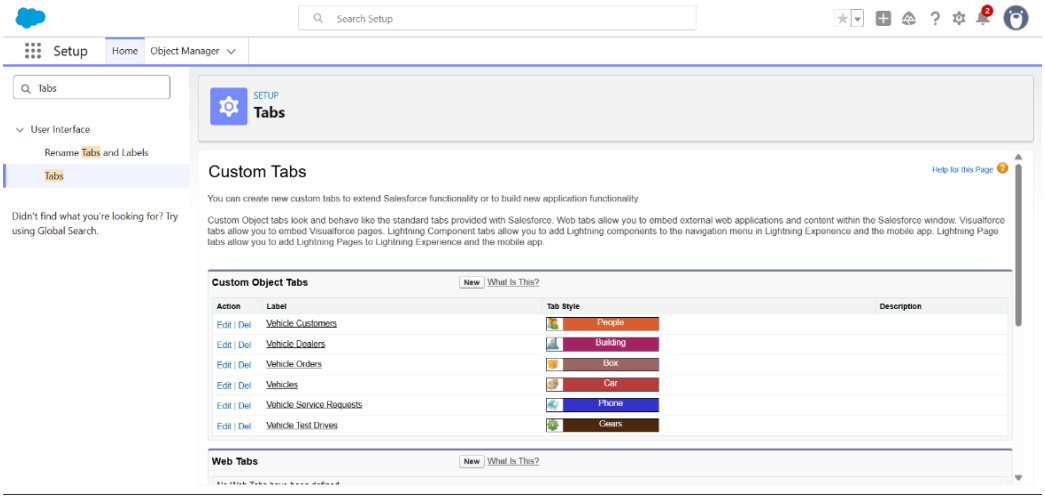
Data Model

Six custom objects were created to reflect business structure:

Object Name	Purpose
Vehicle__c	Stores vehicle details
Vehicle_Dealer__c	Stores authorized dealer info
Vehicle_Customer__c	Stores customer details
Vehicle_Order__c	Tracks vehicle orders
Vehicle_Test_Drive__c	Tracks test drives
Vehicle_Service_Request__c	Manages customer’s service requests

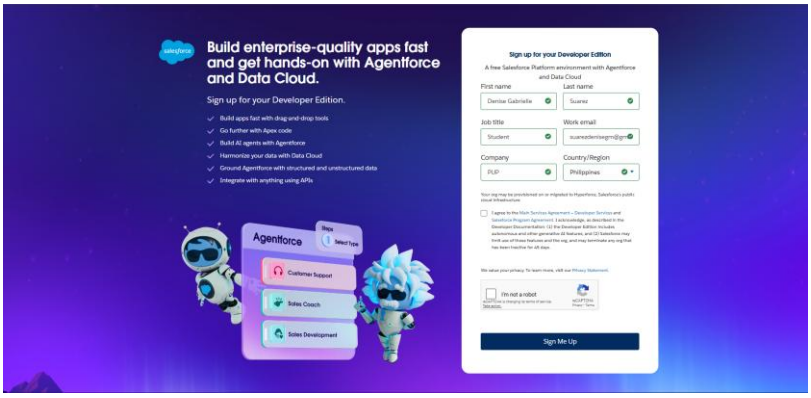
These objects are interconnected using lookup relationship to ensure data integrity.





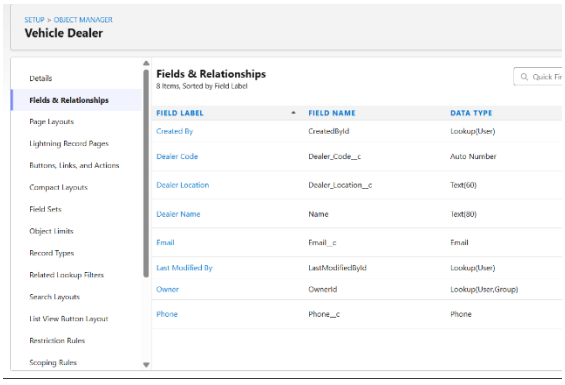
Phase 2: Salesforce Development - Backend & Configurations

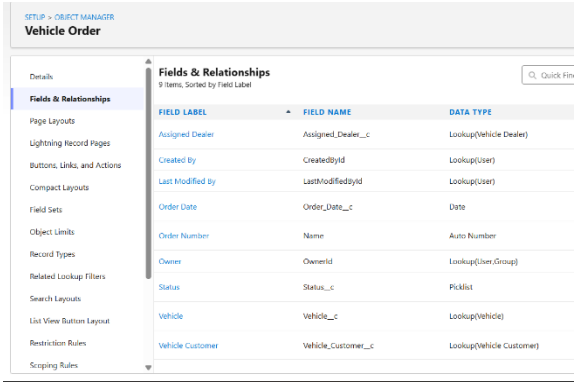
- **Setup Environment and DevOps Workflow**
  The development process starts with creating Salesforce Developer Org.

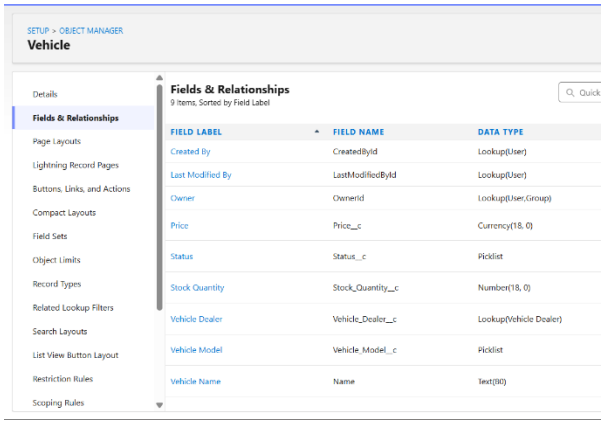


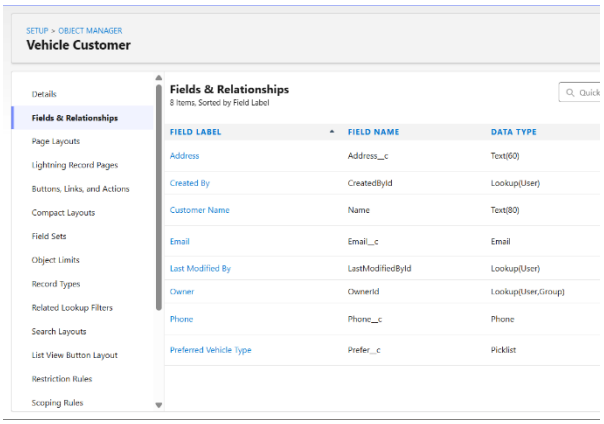
- **Environment:** Salesforce Lightning Experience (Developer Edition)
- **User Profiles/Roles:** Standard profiles were used for testing.

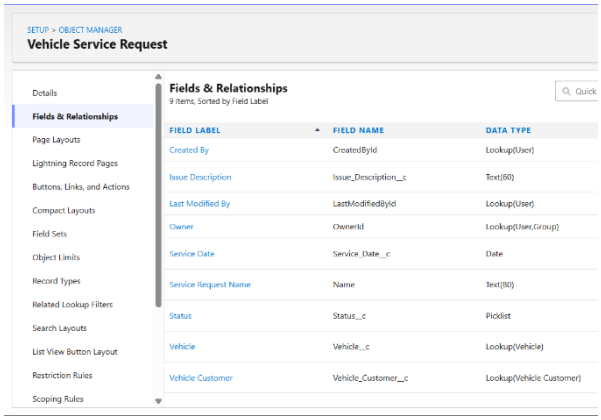
- **Customization of Objects, Fields, Validation Rules, Automation**
  - Six custom objects were created to store business-critical data. This includes:
    - **Vehicle** - Stores vehicle details.
    - **Vehicle Dealer** - Stores authorized dealer info.
    - **Vehicle Customer** - Stores customer details.
    - **Vehicle Order** - Tracks vehicle purchases.
    - **Vehicle Test Drive** - Tracks test drive bookings.
    - **Vehicle Service Request** - Tracks vehicle servicing requests.
  - The following custom objects were configured to support the business flow:
    - **Vehicle** – Store vehicle name, model, quantity, price, and status.
      - Dealer → Lookup Relationship
    - **Vehicle Dealer** – Store dealer name, location, code, phone, and email.
    - **Vehicle Order** – Store order date and status.
      - Customer → Lookup Relationship
      - Vehicle → Lookup Relationship
    - **Vehicle Customer** – Store customer name, email, phone, address, and preferred vehicle type.
    - **Vehicle Test Drive** – Store test drive date and status.
      - Customer → Lookup Relationship
      - Vehicle → Lookup Relationship
    - **Vehicle Service Request** – Store service date, issue description, and status.
      - Customer → Lookup Relationship
      - Vehicle → Lookup Relationship

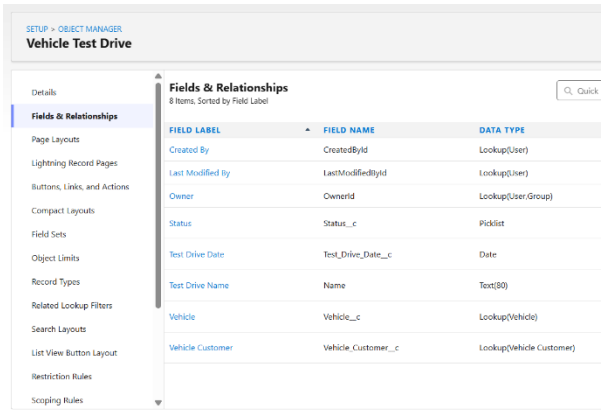






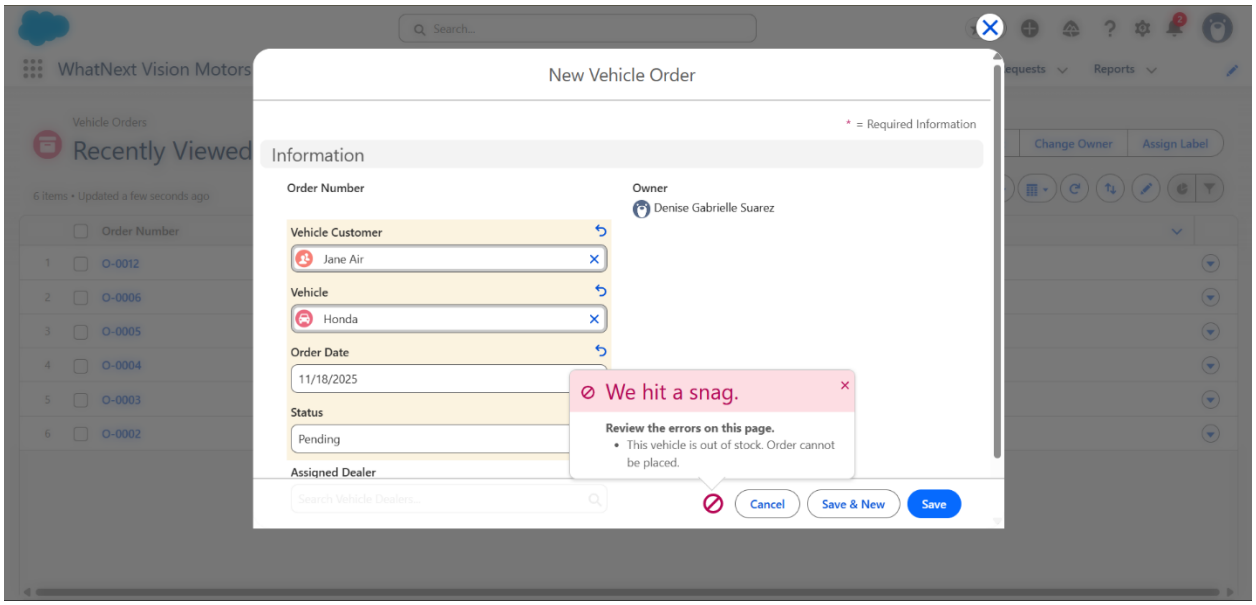






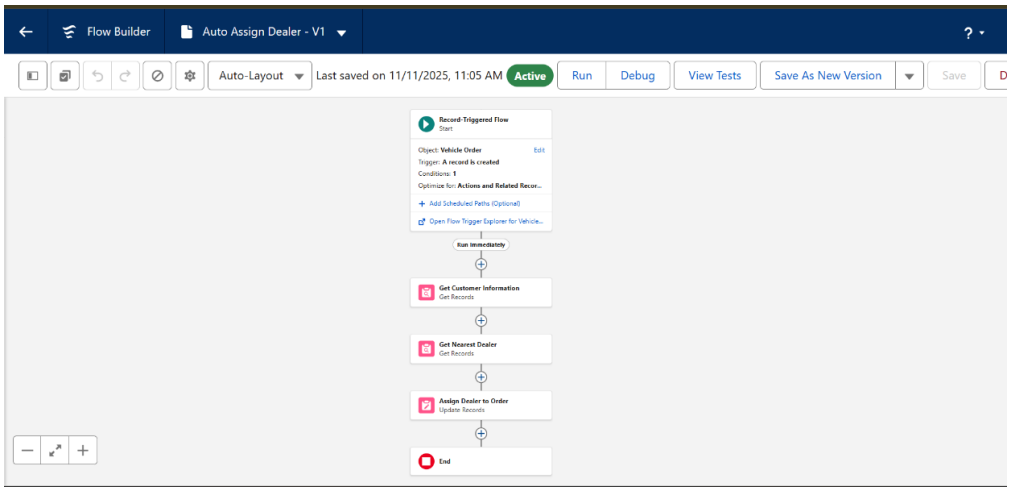
Validation Rules

- **Out-of-stock Blocker:** Prevent placing an order for out-of-stock vehicle.

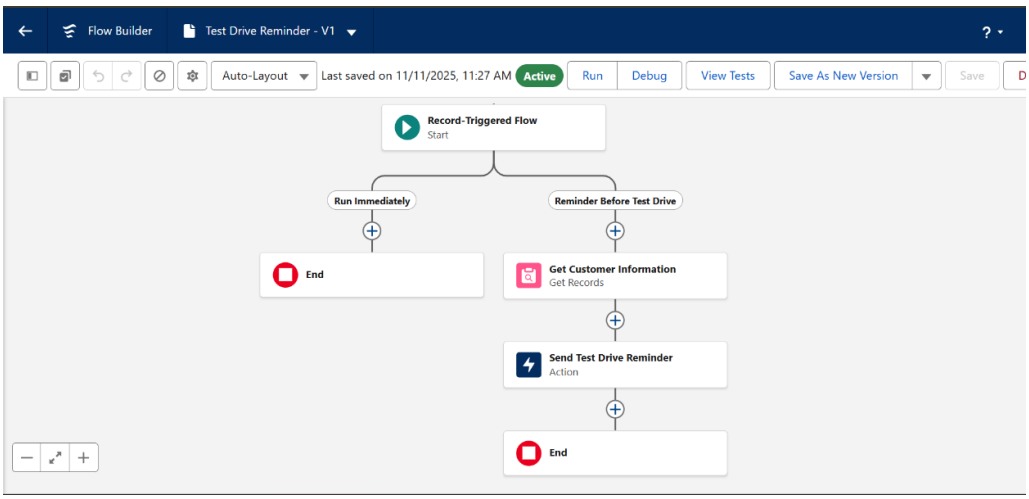


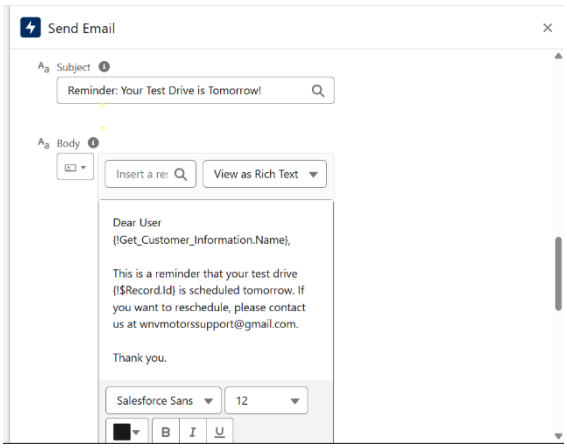
Automation

- **Flows (Record-Triggered)**
  - Auto Assign the nearest dealer based on customer's location using record-triggered flow.



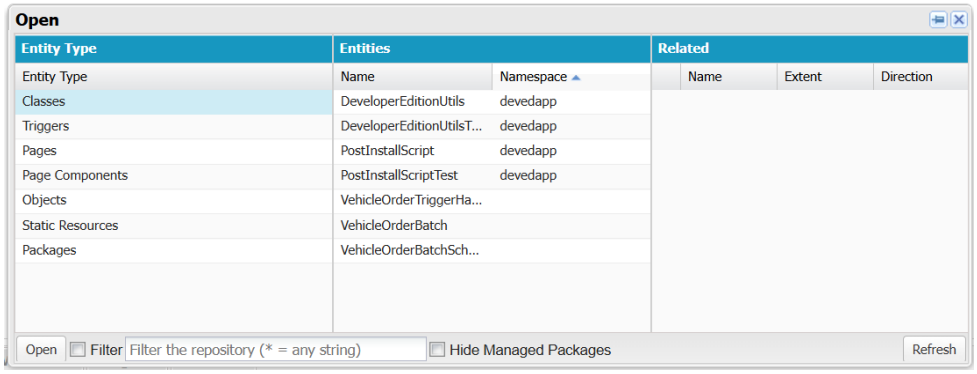
    - Test drive reminder flow to send an email using send emails action of record trigger flow.



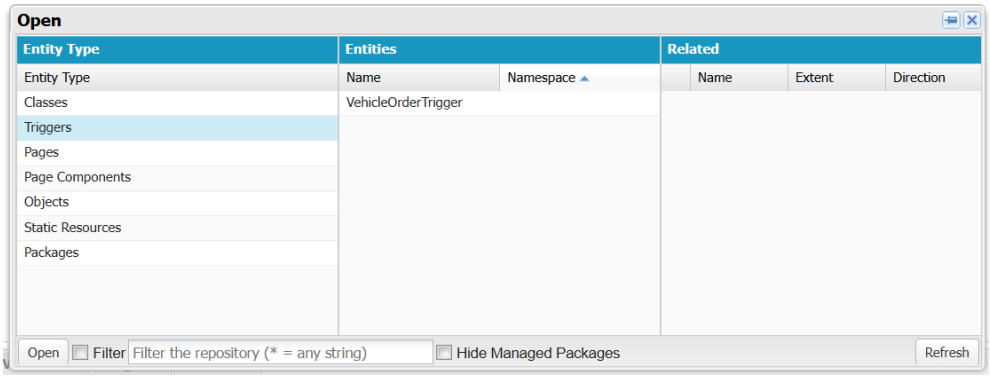


Apex Classes and Triggers

- **Apex Classes** were written to modularize the trigger logic and support backend automation:
    - VehicleOrderTriggerHandler handles stock checks and updates in the trigger.
    - VehicleOrderBatch checks for pending orders and confirms them if stock is available.
    - VehicleOrderBatchScheduler schedules the batch job to run daily at 12 PM.



- **Apex Trigger** was written on the Order object to perform stock availability validation and order status update logic (Pending or Confirmed).

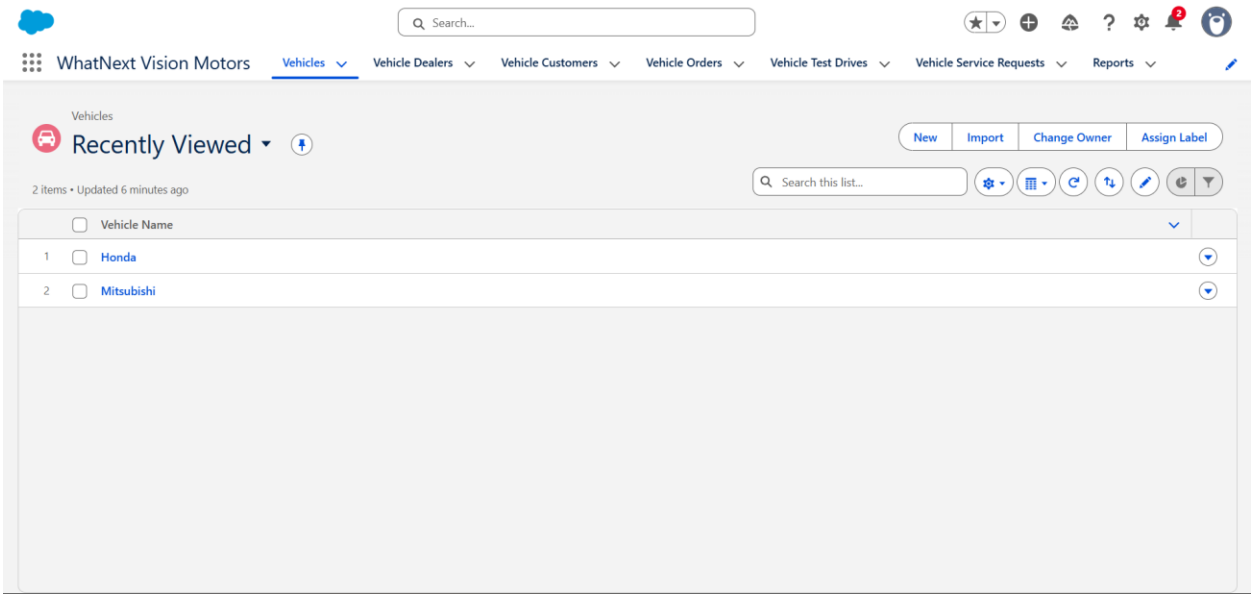


Phase 3: UI/UX Development & Customization

Lightning App Setup through App Manager

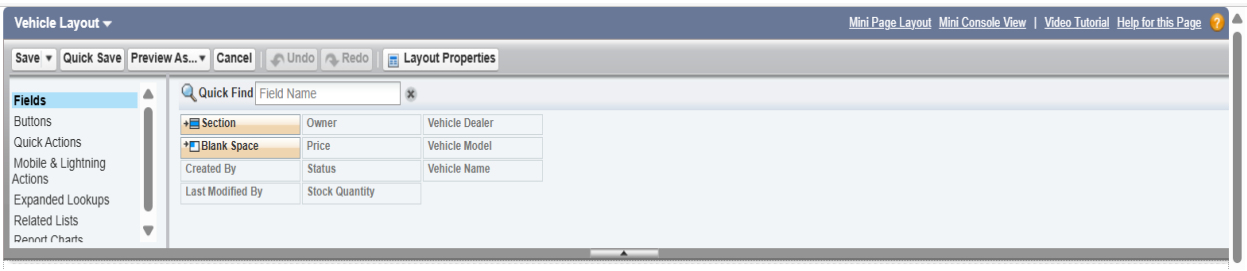
A custom Lightning App titled “WhatNext Vision Motors” was created using App Manager. This app includes relevant custom tabs like Vehicles, Dealers, Orders, Customers, Test Drives, and Service Requests for easy navigation.

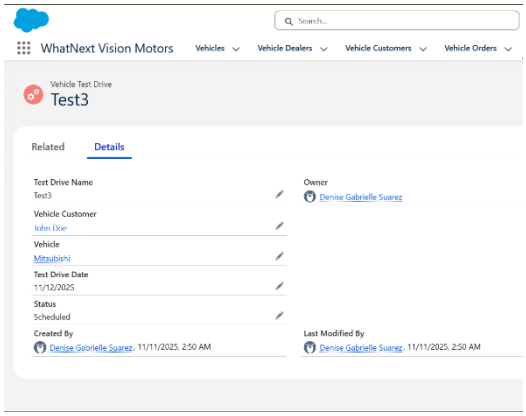
- Lightning App: *WhatNext Vision Motors*
- Tabs: Vehicles, Dealers, Customers, Orders, Test Drives, Services
- Used **Dynamic Forms** for fields based on status & availability
- Highlight panels, related lists added to Lightning Pages

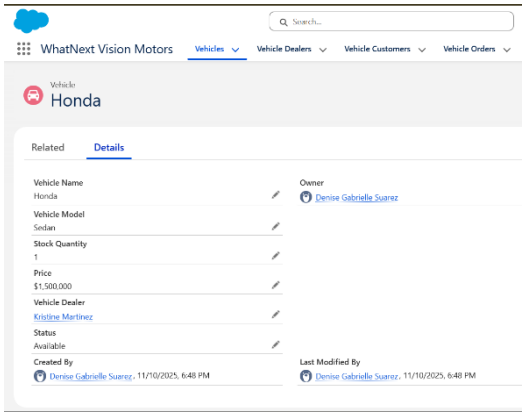


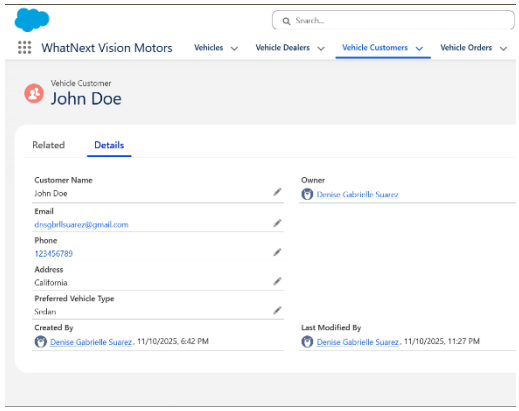
Page Layouts and Dynamic Forms

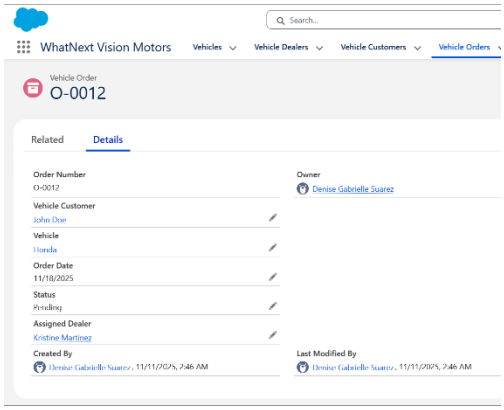
Page Layouts were customized to ensure clean and intuitive UI. Dynamic forms were used to place fields directly on the Lightning Record Page.

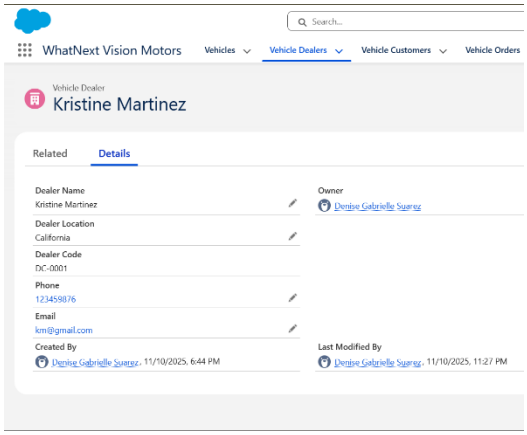



WhatNext Vision Motors
Vehicles
Vehicle Dealers
Vehicle Customers
Vehicle Orders
Vehicle Test Drive
Test3
Related
Details
Test Drive Name
Test3
Owner
Denise Gabrielle Suarez
Vehicle Customer
John Doe
Vehicle
Mitsubishi
Test Drive Date
11/12/2025
Status
Scheduled
Created By
Denise Gabrielle Suarez, 11/11/2025, 2:50 AM
Last Modified By
Denise Gabrielle Suarez, 11/11/2025, 2:50 AM


WhatNext Vision Motors
Vehicles
Vehicle Dealers
Vehicle Customers
Vehicle Orders
Vehicle
Honda
Related
Details
Vehicle Name
Honda
Owner
Denise Gabrielle Suarez
Vehicle Model
Sedan
Stock Quantity
1
Price
$1,500,000
Vehicle Dealer
Kristine Martinez
Status
Available
Created By
Denise Gabrielle Suarez, 11/10/2025, 6:48 PM
Last Modified By
Denise Gabrielle Suarez, 11/10/2025, 6:48 PM

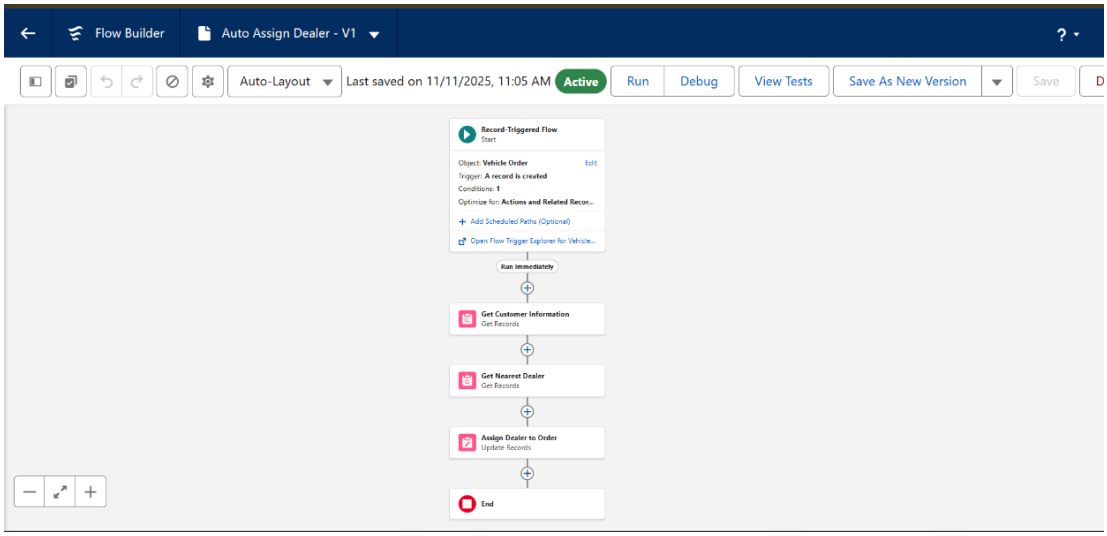

WhatNext Vision Motors
Vehicles
Vehicle Dealers
Vehicle Customers
Vehicle Orders
Vehicle Customer
John Doe
Related
Details
Customer Name
John Doe
Owner
Denise Gabrielle Suarez
Email
dnsgbrllsuarez@gmail.com
Phone
12345789
Address
California
Preferred Vehicle Type
Sedan
Created By
Denise Gabrielle Suarez, 11/10/2025, 6:42 PM
Last Modified By
Denise Gabrielle Suarez, 11/10/2025, 11:27 PM


WhatNext Vision Motors
Vehicles
Vehicle Dealers
Vehicle Customers
Vehicle Orders
Vehicle Order
O-0012
Related
Details
Order Number
O-0012
Owner
Denise Gabrielle Suarez
Vehicle Customer
John Doe
Vehicle
Honda
Order Date
11/18/2025
Status
Pending
Assigned Dealer
Kristine Martinez
Created By
Denise Gabrielle Suarez, 11/11/2025, 2:46 AM
Last Modified By
Denise Gabrielle Suarez, 11/11/2025, 2:46 AM


WhatNext Vision Motors
Vehicles
Vehicle Dealers
Vehicle Customers
Vehicle Orders
Vehicle Dealer
Kristine Martinez
Related
Details
Dealer Name
Kristine Martinez
Owner
Denise Gabrielle Suarez
Dealer Location
California
Dealer Code
DC-0001
Phone
123459876
Email
km@gmail.com
Created By
Denise Gabrielle Suarez, 11/10/2025, 6:44 PM
Last Modified By
Denise Gabrielle Suarez, 11/10/2025, 11:27 PM

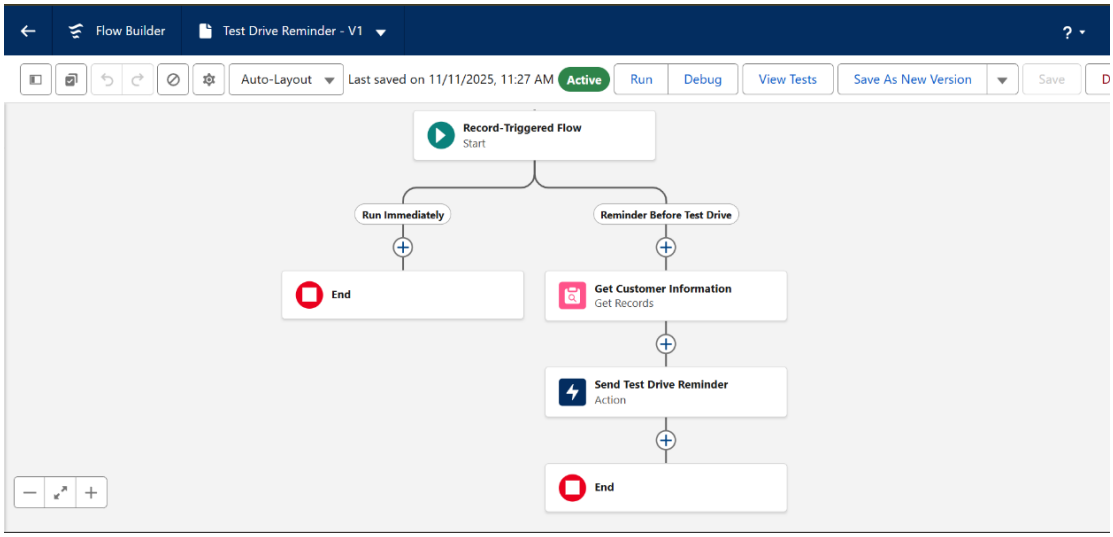
Flow 1: Auto Assign Dealer

This flow runs on Vehicle_Order_c creation and fetches customer's address from Get Customer Information, Get Nearest Dealer and Assigns that Dealer to the Order.



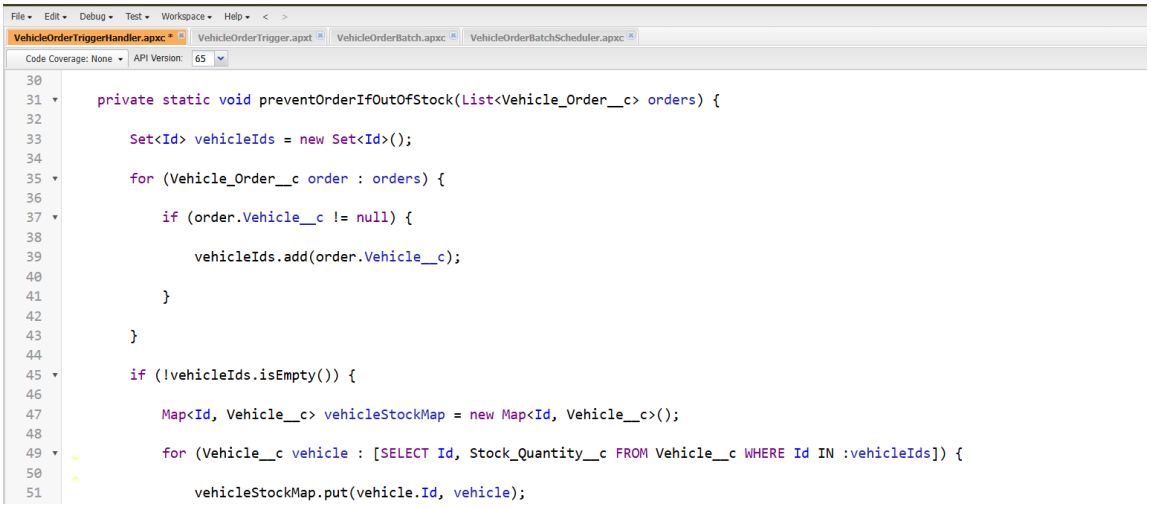
Flow 2: Test Drive Reminder

This flow runs on Vehicle_Test_Drive_c creation/update and sends email the day before the scheduled test drive.

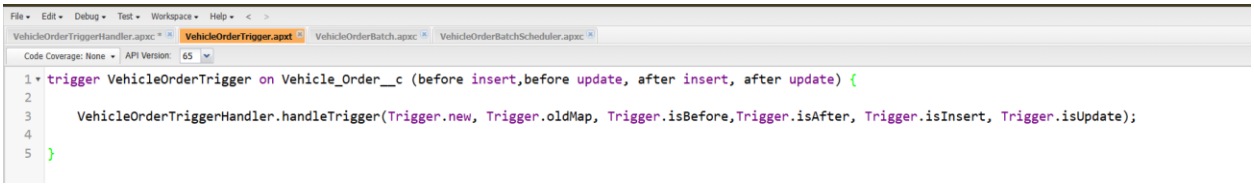


Apex Trigger and Handler

- Apex trigger named *VehicleOrderTrigger* while handler is *VehicleOrderTriggerHandler*.
    - The *VehicleOrderTriggerHandler* prevents out-of-stock orders and updates stock when the status is confirmed.



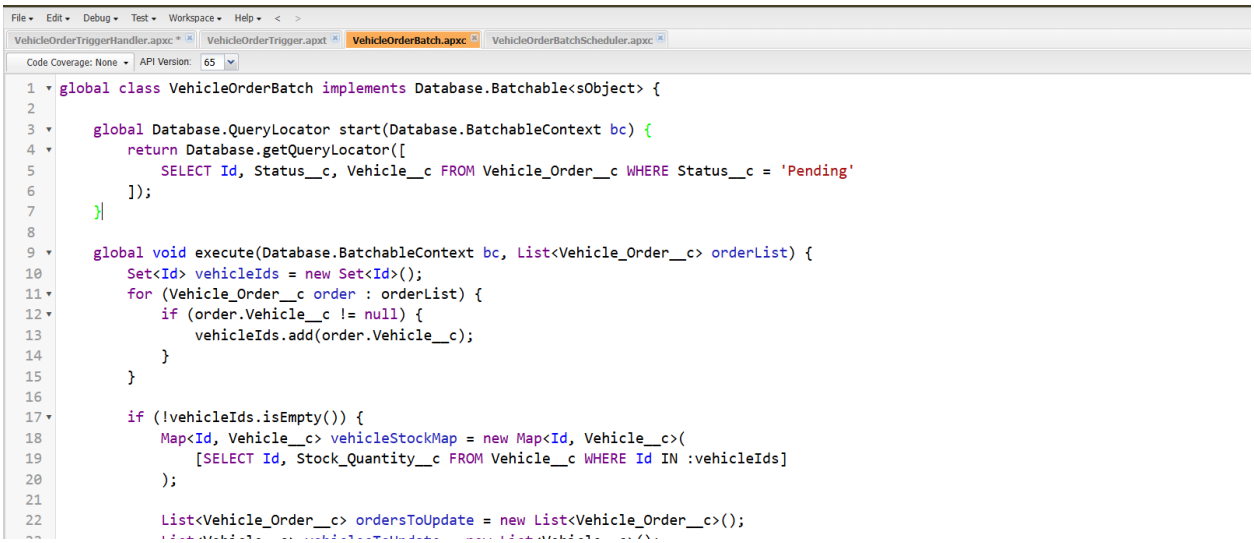
```
    private static void preventOrderIfOutOfStock(List<Vehicle_Order__c> orders) {

        Set<Id> vehicleIds = new Set<Id>();

        for (Vehicle_Order__c order : orders) {

            if (order.Vehicle__c != null) {

                vehicleIds.add(order.Vehicle__c);

            }

        }

        if (!vehicleIds.isEmpty()) {

            Map<Id, Vehicle__c> vehicleStockMap = new Map<Id, Vehicle__c>();

            for (Vehicle__c vehicle : [SELECT Id, Stock_Quantity__c FROM Vehicle__c WHERE Id IN :vehicleIds]) {

                vehicleStockMap.put(vehicle.Id, vehicle);
```



```
trigger VehicleOrderTrigger on Vehicle_Order__c (before insert,before update, after insert, after update) {

    VehicleOrderTriggerHandler.handleTrigger(Trigger.new, Trigger.oldMap, Trigger.isBefore,Trigger.isAfter, Trigger.isInsert, Trigger.isUpdate);

}
```

Apex Batch Class

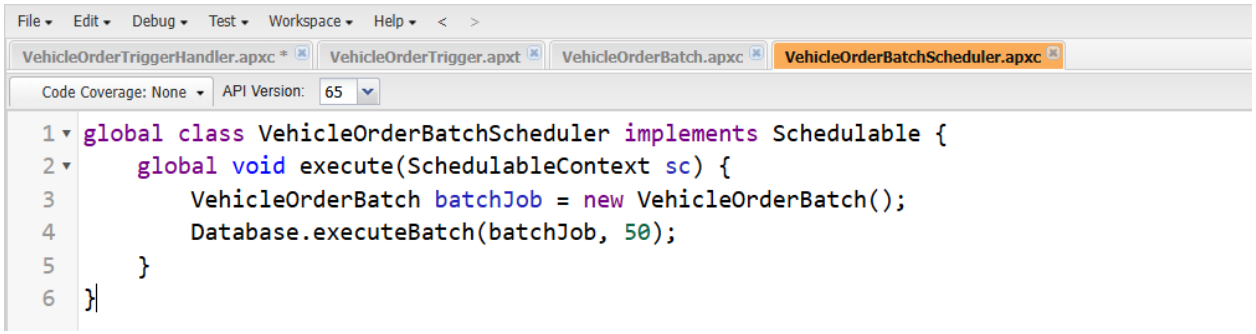
- The class *VehicleOrderBatch* runs daily as it checks for pending orders, available stocks, updates status when it is *Confirmed* and reduces the stock.



```
global class VehicleOrderBatch implements Database.Batchable<sObject> {

    global Database.QueryLocator start(Database.BatchableContext bc) {
        return Database.getQueryLocator([
            SELECT Id, Status__c, Vehicle__c FROM Vehicle_Order__c WHERE Status__c = 'Pending'
        ]);
    }

    global void execute(Database.BatchableContext bc, List<Vehicle_Order__c> orderList) {
        Set<Id> vehicleIds = new Set<Id>();
        for (Vehicle_Order__c order : orderList) {
            if (order.Vehicle__c != null) {
                vehicleIds.add(order.Vehicle__c);
            }
        }

        if (!vehicleIds.isEmpty()) {
            Map<Id, Vehicle__c> vehicleStockMap = new Map<Id, Vehicle__c>(
                [SELECT Id, Stock_Quantity__c FROM Vehicle__c WHERE Id IN :vehicleIds]
            );

            List<Vehicle_Order__c> ordersToUpdate = new List<Vehicle_Order__c>();
```

Apex Batch Scheduled

- The *VehicleOrderBatchScheduler* runs daily at 12pm as it executes batch class automatically.



Phase 4: Data Migration, Testing & Security

Profiles and Roles

- Standard profiles like **Standard User** and **System Administrator** were used.

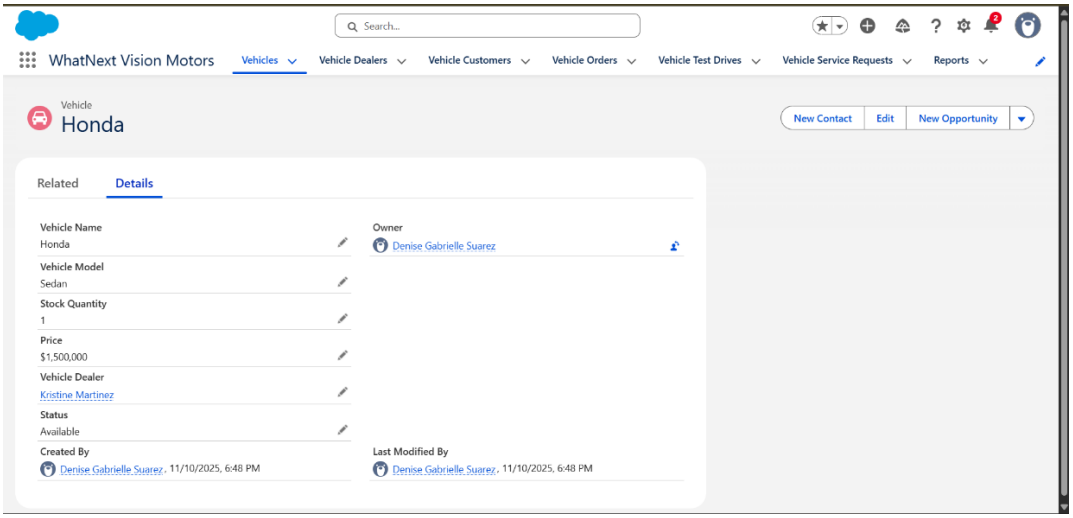
Sharing Rules

- **Public Read/Write** for most custom objects.

Preparation of test cases for each salesforce features like creation of Vehicle, Vehicle orders, Approval Process, Automatic Task creation, flows, triggers etc.

1. **Create a Vehicle:**
   INPUT:
   - **Vehicle Name**: Honda
   - **Vehicle Model**: Sedan
   - **Stock Quantity**: 1
   - **Price**: $1,500,000
   - **Vehicle Dealer:** Kristine Martinez
   - **Status:** Pending

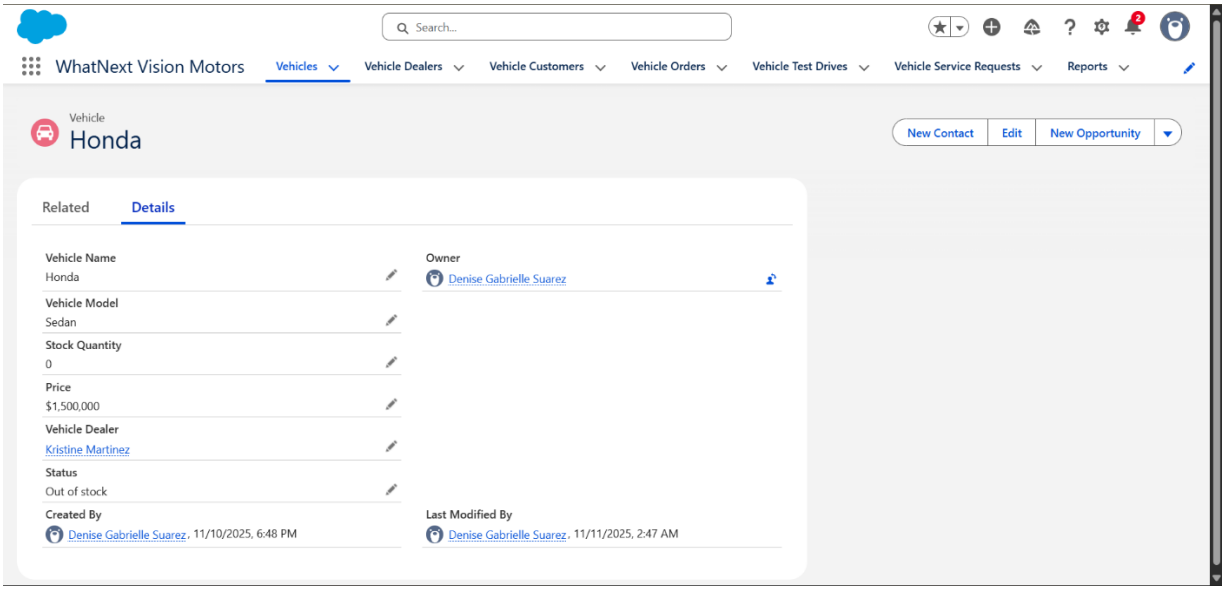
   OUTPUT:



2. **Confirmed Order**
   INPUT:
   - Go to Vehicle Orders tab → Click New.
   - **Vehicle:** Honda
   - **Status:** Confirmed
   - **Customer:** John Doe

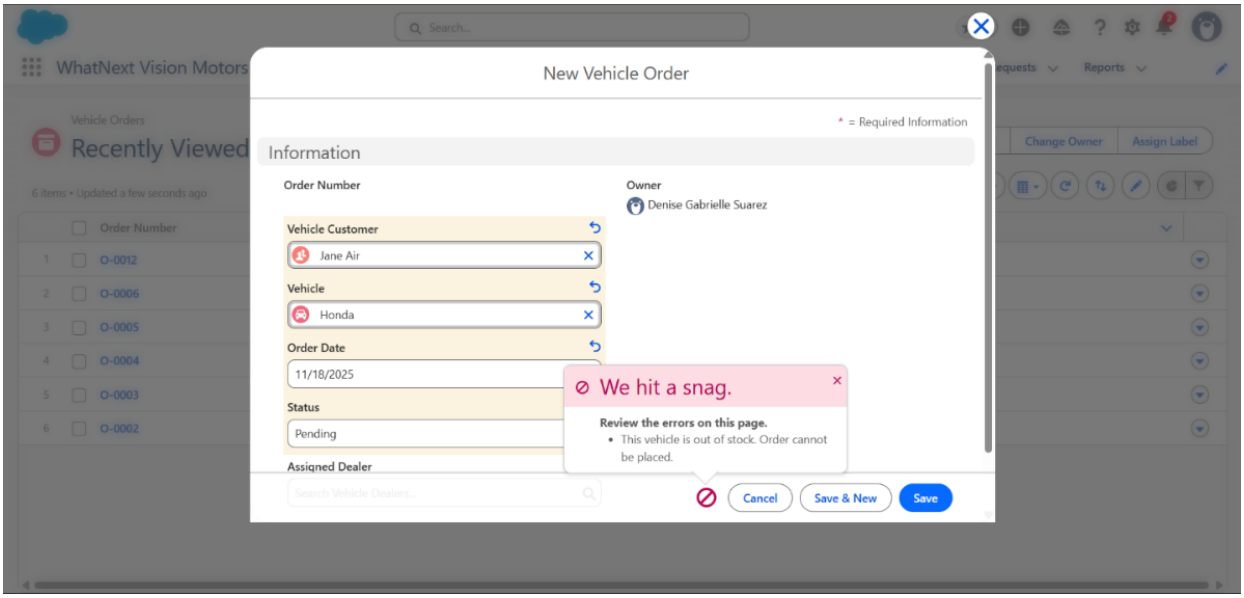
   OUTPUT:

   The stock became 0.



3. **Stock = 0 (Create new order)**
   INPUT:
   - **Vehicle:** Honda
   - **Status:** Pending
   - **Customer:** John Doe

   Prevent to place an order for 0 stock vehicle.

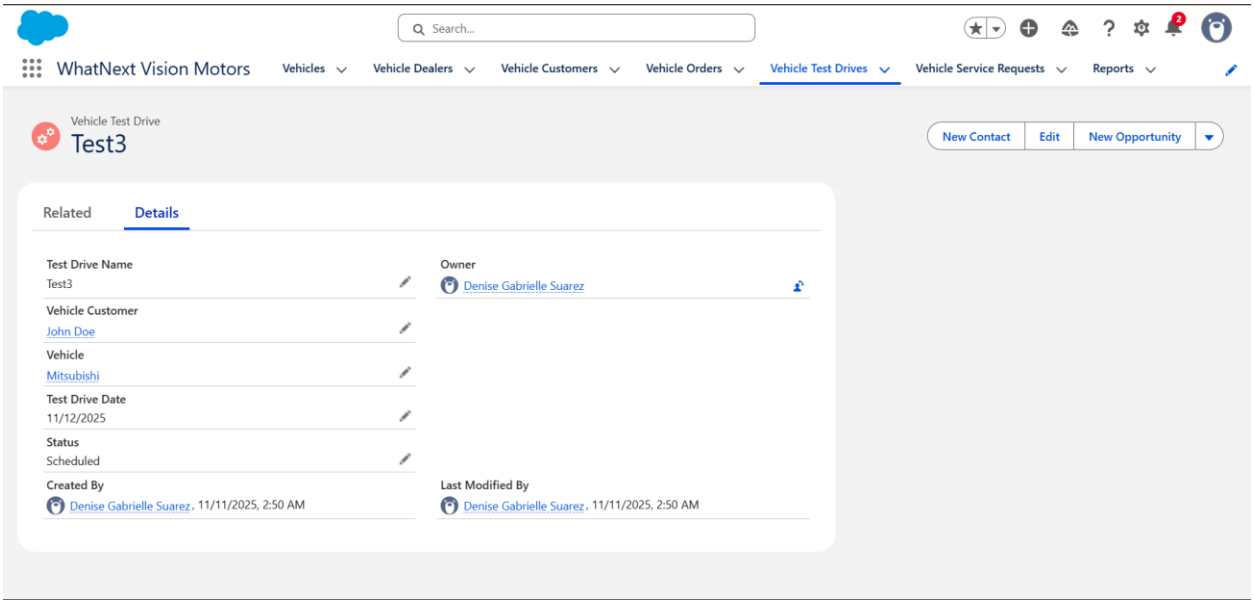


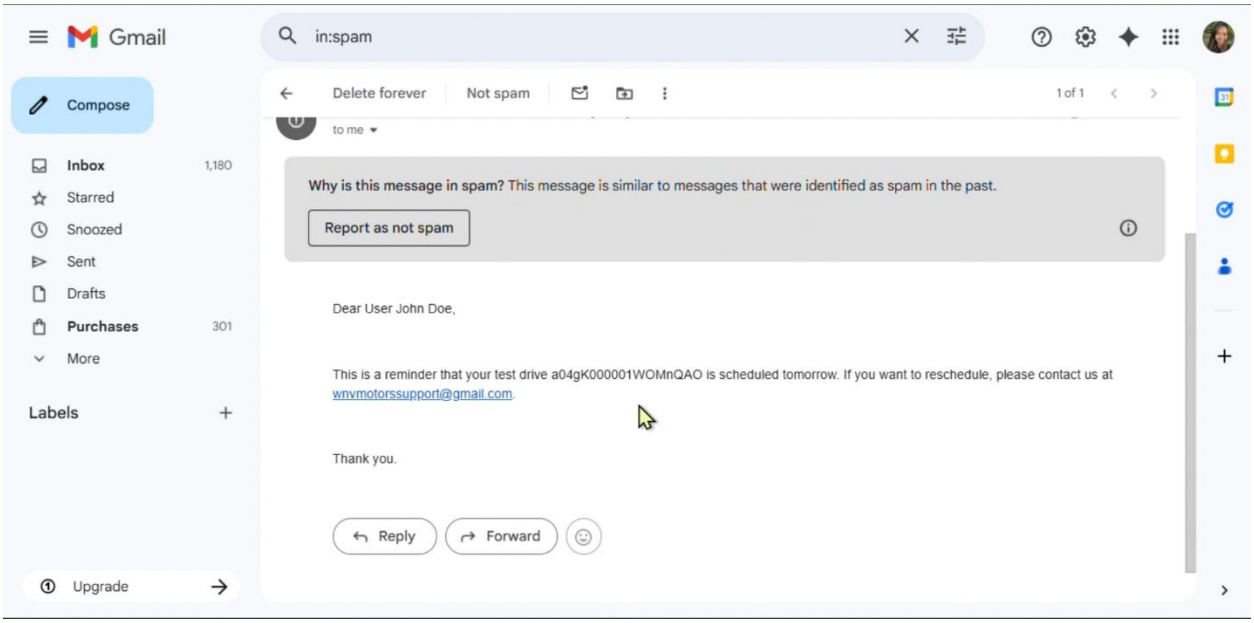
4. **Test Drive Reminder Email**

INPUT:

- **Customer:** Select any customer with email
- **Test Drive Date**: Tomorrow (pick tomorrow's date)
- **Status:** Scheduled

OUTPUT:





Phase 5: Deployment, Documentation & Maintenance

Deployment Strategy

The Change Set deployment method was used to deploy the developed features from the Developer Org to the live/production environment.

Deployment Steps:

1. Created an Outbound Change Set in the source org.
2. Added all custom components such as Custom objects, fields, flows, validation rules, triggers, and Apex classes.
3. Uploaded the Change Set to the Org.
4. Validated and deployed it from Inbound Change Sets in the org.
5. Post-deployment manual verification was done to ensure everything works as expected.

Testing & Sample Scenarios

Test Cases:

Several real-use scenarios were tested to confirm that the system behaves as expected:

- Attempting to create a vehicle order with zero stock should return an error.
- Setting stock to 2 and placing an order should correctly decrease the stock to 1.
- Creating a pending order and then updating the stock should allow the batch job to automatically confirm the order

System Maintenance and Monitoring

The following are the basic maintenance strategy to ensure smooth system performance:

1. **Monitoring**
    - Apex Jobs were used to monitor scheduled jobs and batch processes.
    - Debug Logs helped trace system errors or unusual behavior.
    - Email alerts were enabled for test drive reminders.

2. **Updates and Fixes**
    - Small updates such as adjusting field labels or adding help text were made in the sandbox and deployed through Change Sets.
    - Quarterly system reviews were scheduled to assess potential enhancements and UI improvements.

Troubleshooting Approach

1. **Reproduce the Issue**
   Attempt to replicate the problem in a developer org.

2. **Enable Debug Logs**
   Set up debug logs for the affected user to analyze the flow or Apex execution.

3. **Check Apex Jobs or Flows**
   For issues related to background processes, check Apex job failures or Flow error notifications.

4. **Fix and Retest**
   Update the logic in the Flow or Apex code.
   Retest the fix in the sandbox, then redeploy once confirmed working.

Conclusion

The WhatNext Vision Motors CRM system built on Salesforce successfully enhances both customer experience and operational efficiency. By automating key processes such as dealer assignment, stock validation, and test drive reminders, the system significantly reduces manual workload and minimizes the risk of errors. Features like Lightning Apps and Dynamic Forms further improve usability, ensuring that internal users can navigate and manage data with ease. Overall, the implemented solution provides a streamlined, reliable, and customer-centric workflow that strengthens the company's service quality and operational performance.

Several Salesforce configurations were created to support the system's functionality. *Validation rules* were implemented to prevent incorrect or incomplete data entry, such as restricting order creation when stock is zero. *Approval processes* were set up for requests that require managerial review, ensuring that important actions like order confirmation. *Automation flows* were developed to handle key processes, including updating stock, sending reminders, and confirming pending orders through scheduled or record-triggered flows. Additional components, such as custom objects and Apex classes, were also created to support system operations and data monitoring.

Testing was done in the sandbox environment before deployment. Each flow was tested by triggering the corresponding action for example, creating or updating a record to confirm field updates, email alerts, and automation behavior. Validation rules were tested by intentionally entering invalid data to ensure the rule triggers the correct error message. Approval processes were tested by submitting sample records for approval and verifying each approval step. After changes were deployed, final verification was done in production to ensure all components functioned as expected.

Future Enhancements

1. **Customer Portal Integration**
    - A portal or community where customer can login and view orders.

2. **Mobile Application Support**
    - Extend CRM features to a mobile app to provide customers and dealers easier access to order updates and service information.

3. **Reports and Dashboard**

- Develop enhanced reporting tools to give management deeper insights into sales performance, dealer efficiency, and customer behavior.

4. **Chatbot and Virtual Assistant Integration**
    - Add automated customer support to provide real-time answers, guide users through the ordering process, and improve response times.

5. **SMS Integration**
    - Notify customers via SMS about order updates.

6. **Financial System Integration**
    - Connect the system using API payment gateways to streamline billing, invoicing, and loan processing.