WhatNext Vision Motors CRM
Project Documentation

Submitted by:
Ken Chico

Table of Contents

Project Overview

WhatNext Vision Motors CRM automates vehicle order processing in Salesforce, assigning orders to nearest dealers based on customer location, preventing orders for out-of-stock vehicles, and sending test drive email reminders one day prior to scheduled dates. Key features include custom objects for vehicles, customers, dealers, orders, test drives, and service requests with automation via Flows and Apex triggers.

Objectives

The primary goals are to streamline vehicle order assignment to geographically closest dealers, enforce stock availability validation to prevent overselling, and automate test drive reminders for improved customer experience. These objectives deliver business value through efficient order processing, reduced manual intervention, inventory accuracy, and enhanced customer satisfaction via timely communications.

Phase 1: Requirement Analysis & Planning

Understanding Business Requirements

- Automate vehicle order assignment to nearest dealer based on customer address matching dealer location.
- Prevent order creation for vehicles with zero stock quantity.
- Send automated email reminders to customers one day before scheduled test drives.
- Track vehicle orders, test drives, and service requests with status management.

Defining Project Scope and Objectives

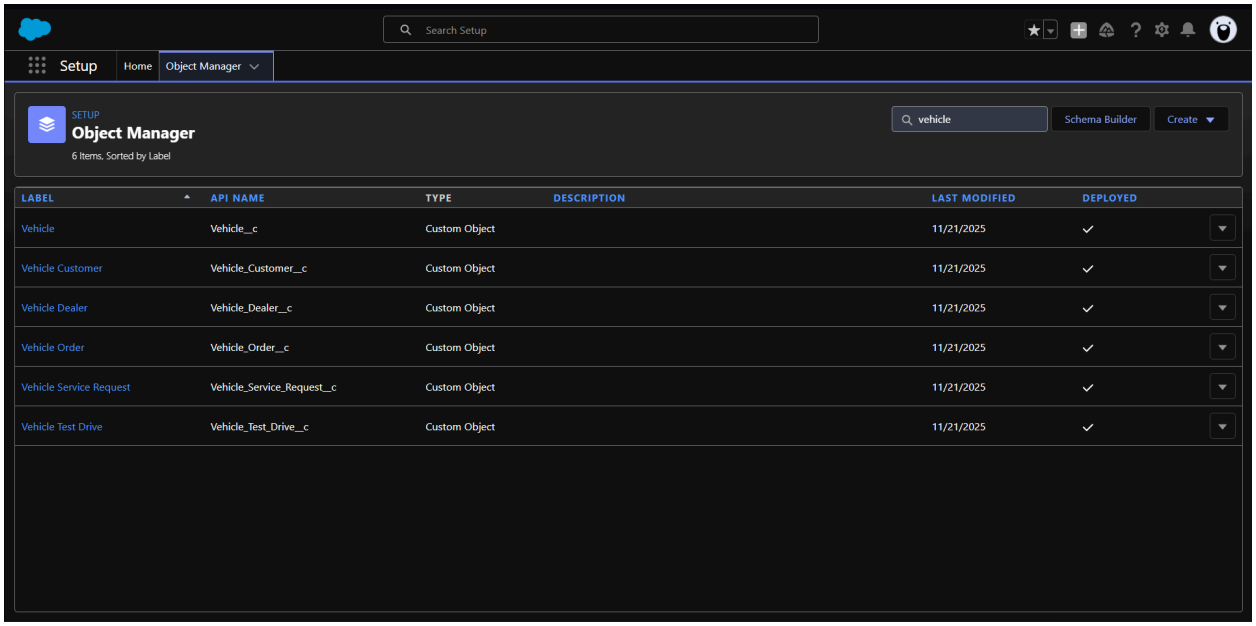
- In Scope: Custom objects (Vehicle, Vehicle Customer, Vehicle Dealer, Vehicle Order, Vehicle Test Drive, Vehicle Service Request), Flows for auto-assignment and email alerts, Apex triggers for stock management.
- Out of Scope: Payment processing, full inventory forecasting, mobile app integration.
- Objectives tied to operational efficiency and customer service improvement.

Design Data Model and Security Model

Figure 1 shows the full list of all the custom objects that were created for the development of WhatNext Vision Motors CRM.

Figure 1

Object Manager Overview

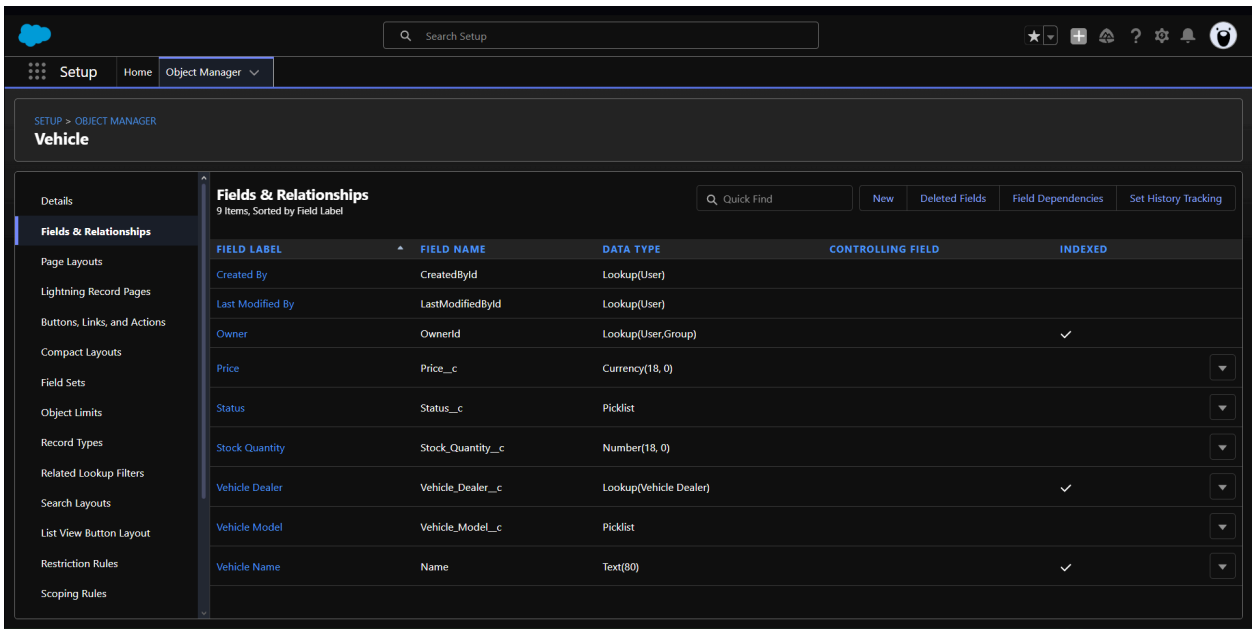


Data Model:

Figure 2 shows the screenshot of the fields and relationships of the vehicle object in the object manager.

Figure 2

Vehicle Object

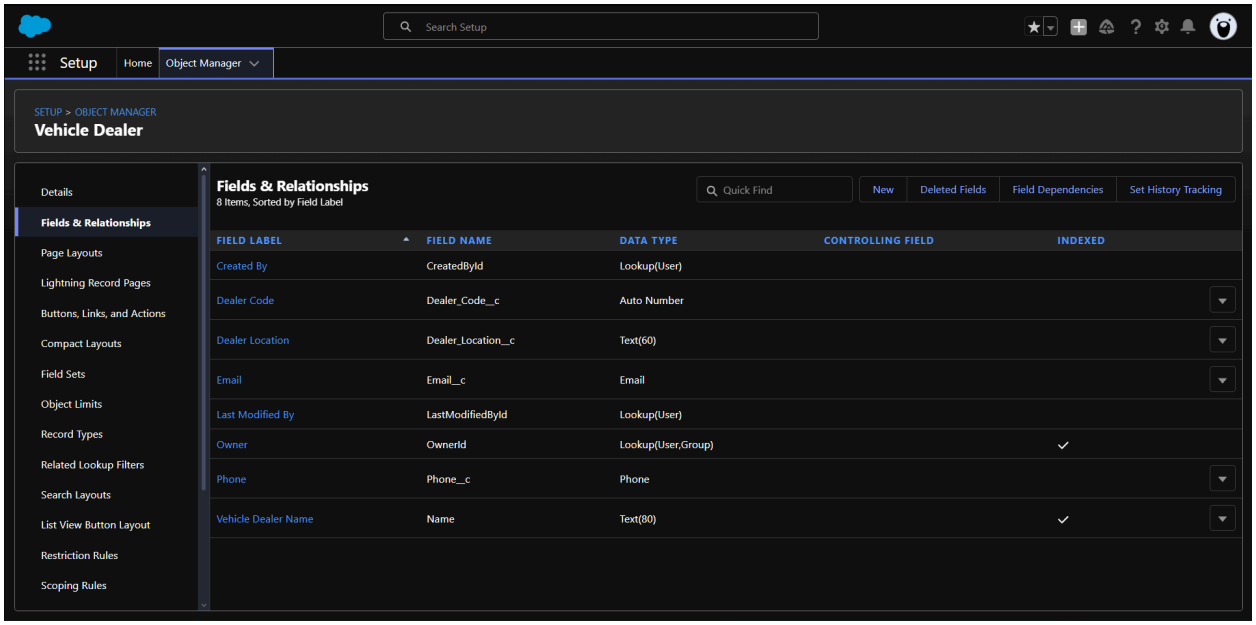


Vehicle: Name, Model (Picklist: Sedan, SUV, EV, etc.), Stock Quantity (Number), Price (Currency), Dealer (Lookup), Status (Picklist: Available, Out of Stock, Discontinued).

Figure 3 shows the screenshot of the fields and relationships of the vehicle dealer object in the object manager.

Figure 3

Vehicle Dealer

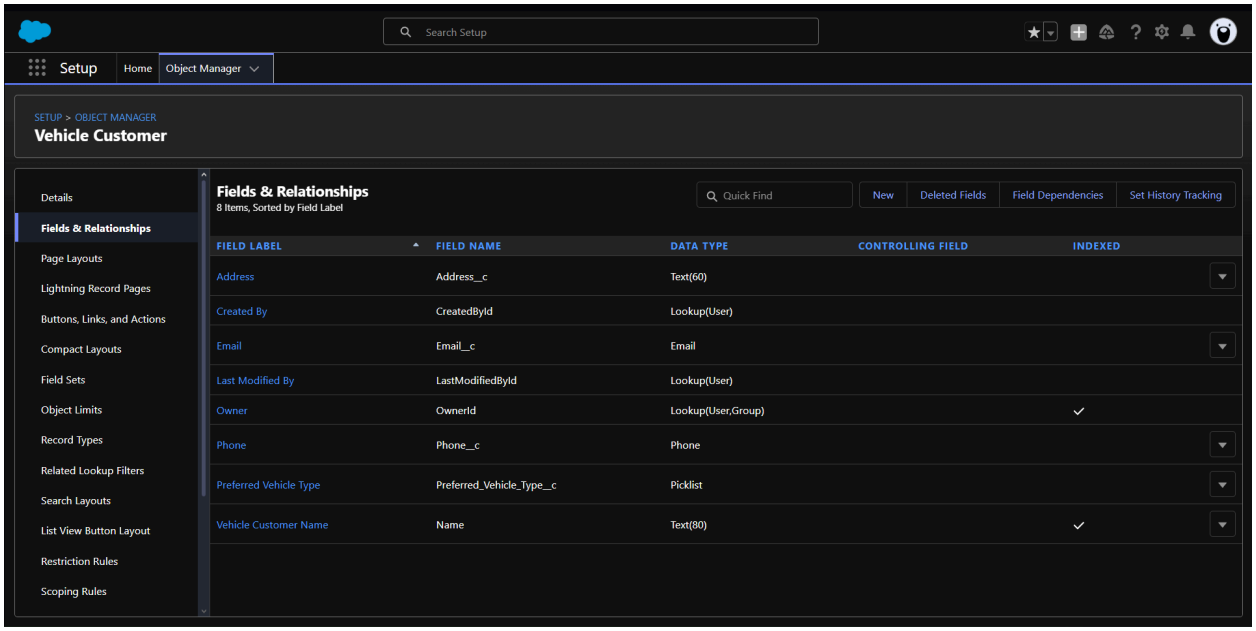


Vehicle Dealer: Name, Location (Text 60), Code (Auto Number: DC-0001), Phone, Email.

Figure 4 depicts the fields and relationships of the vehicle dealer object.

Figure 4

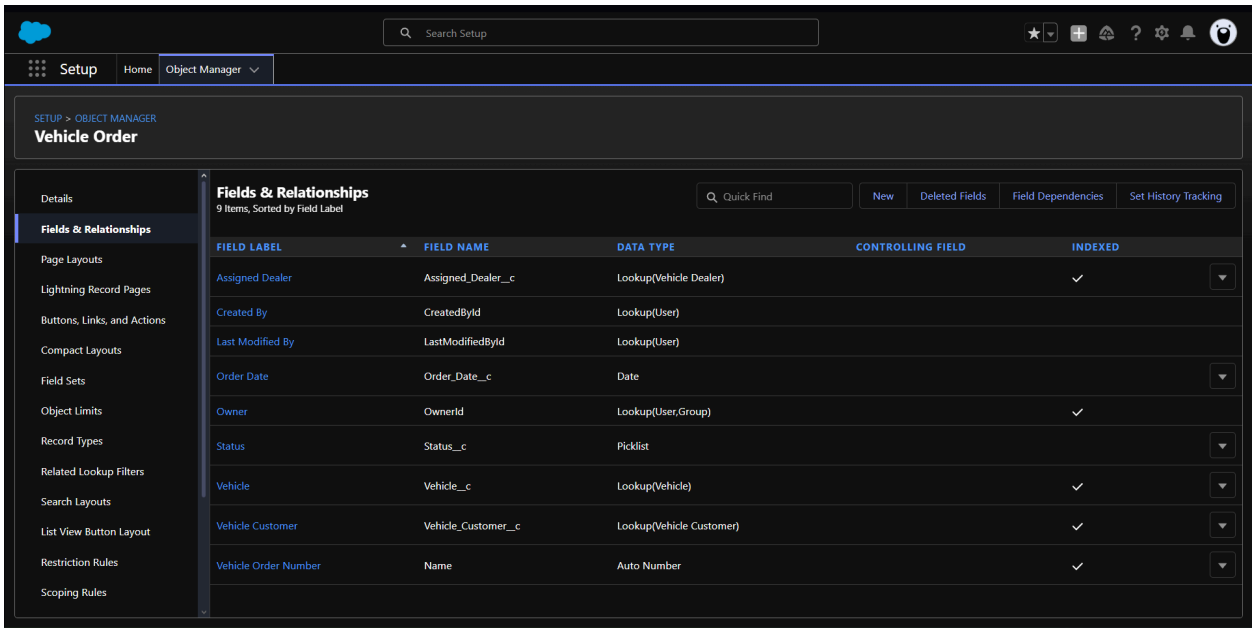
Vehicle Customer



Vehicle Customer: Name, Email, Phone, Address (Text 60), Preferred Vehicle Type (Picklist).

Figure 5 showcases the fields and relationships of the vehicle order in the object manager.

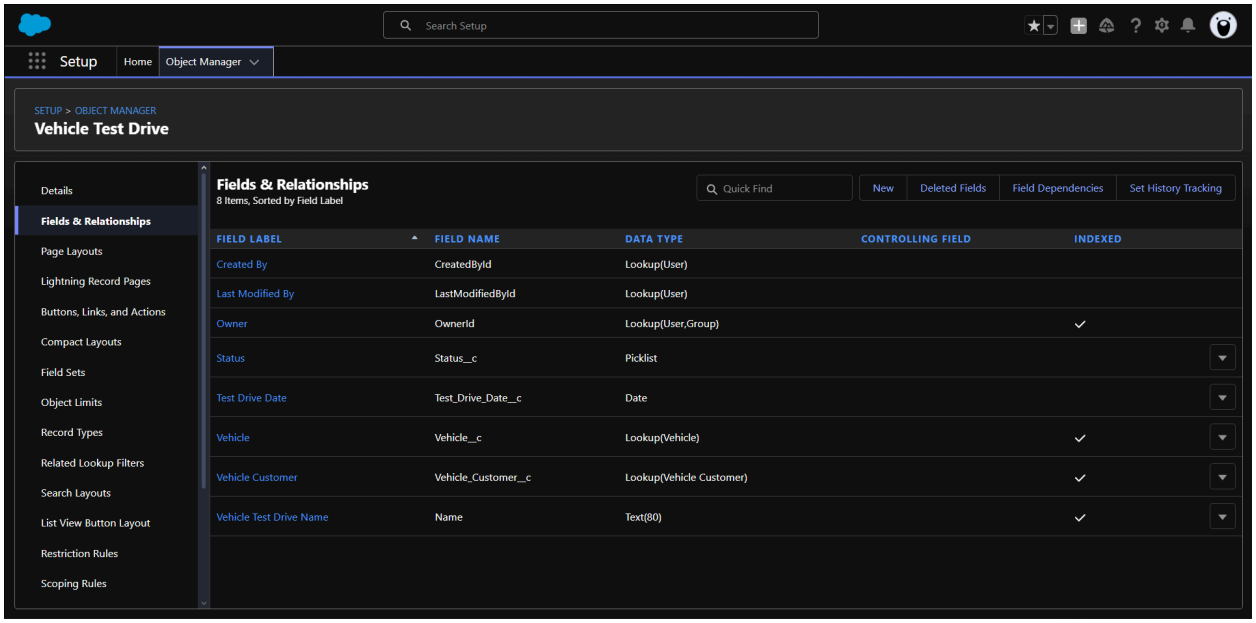
Figure 5
Vehicle Order



Vehicle Order: Order Number (Auto Number: O-0001), Customer (Lookup), Vehicle (Lookup), Order Date (Date), Status (Picklist: Pending, Confirmed, Delivered, Cancel), Assigned Dealer (Lookup).

Figure 6 presents the vehicle test drive custom object along with its fields and relationships in the object manager.

Figure 6
Vehicle Test Drive

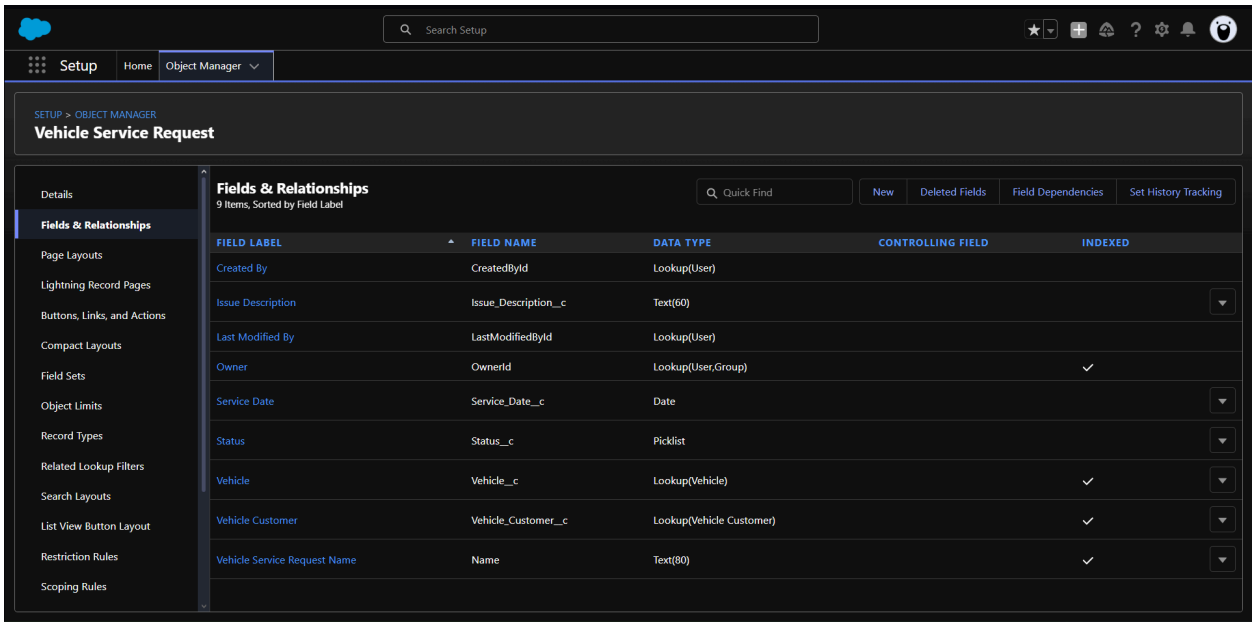


Vehicle Test Drive: Customer (Lookup), Vehicle (Lookup), Test Drive Date (Date), Status (Picklist: Scheduled, Completed, Cancel).

Figure 7 depicts the vehicle service request object with its fields and relationships.

Figure 7

Vehicle Service Request



Vehicle Service Request: Customer (Lookup), Vehicle (Lookup), Service Date (Date), Issue Description (Text 60), Status (Picklist: Requested, In Progress, Completed).
Security Model: System Administrator profile access; standard Salesforce sharing rules apply.

Stakeholders Mapping

- Primary: Dealers (order assignment), Customers (test drives/services), Admins (maintenance).
- Secondary: Salesforce Admins for monitoring Flows/Apex.

Execution Roadmap

1. Developer Org Setup → Object/Field Creation → App/Tabs → Flows → Apex → Testing.

Phase 2: Salesforce Development - Backend & Configurations

Setup Environment & DevOps Workflow

- Salesforce Developer Edition Org created via signup link.
- WhatNext Vision Motors Lightning App created via App Manager, including all custom objects + Reports/Dashboards.

Customization of Objects, Fields, Validation Rules, Automation

- 6 Custom Objects created with "Allow Reports" and "Allow Search" enabled.
- Fields as detailed in Data Model above.
- Flows:

Flow Name	Trigger	Logic
Auto Assign Dealer	Vehicle Order (Status = Pending)	Get customer address → Match nearest Vehicle Dealer → Update Assigned Dealer field.
Test Drive Reminder	Vehicle Test Drive (Status = Scheduled, 1 day before Test Drive Date)	Get customer email → Send reminder email with test drive details.

Figures 8 and 9 show the actual flow via flow builder.

Figure 8
Auto Assign Dealer Flow

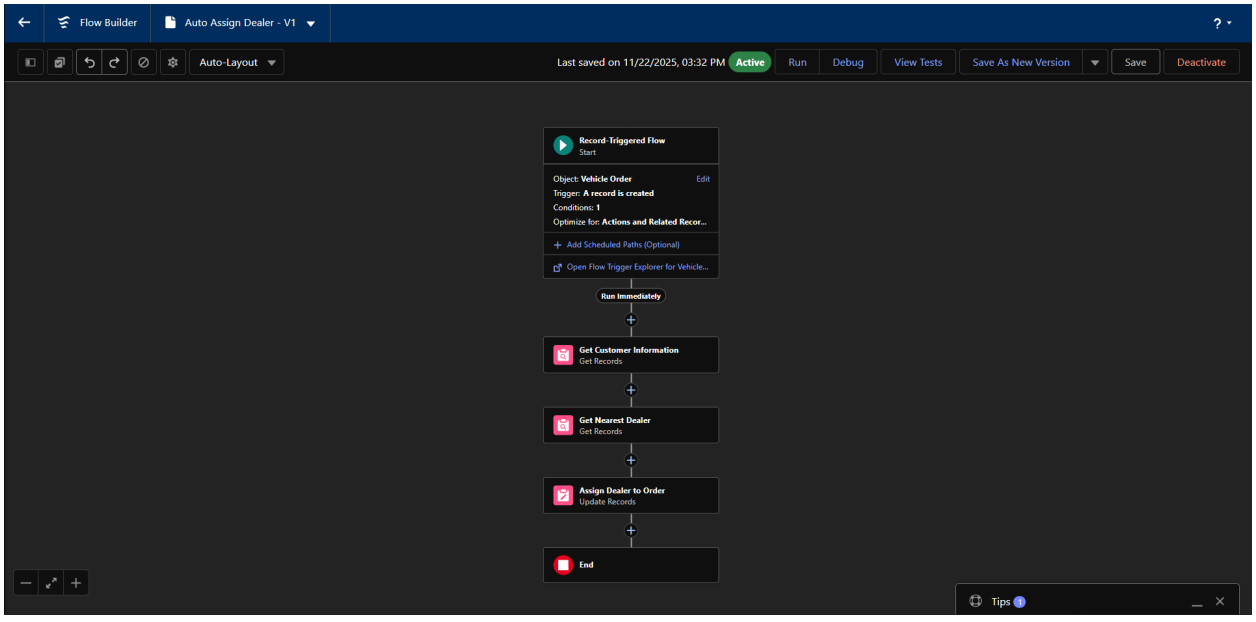
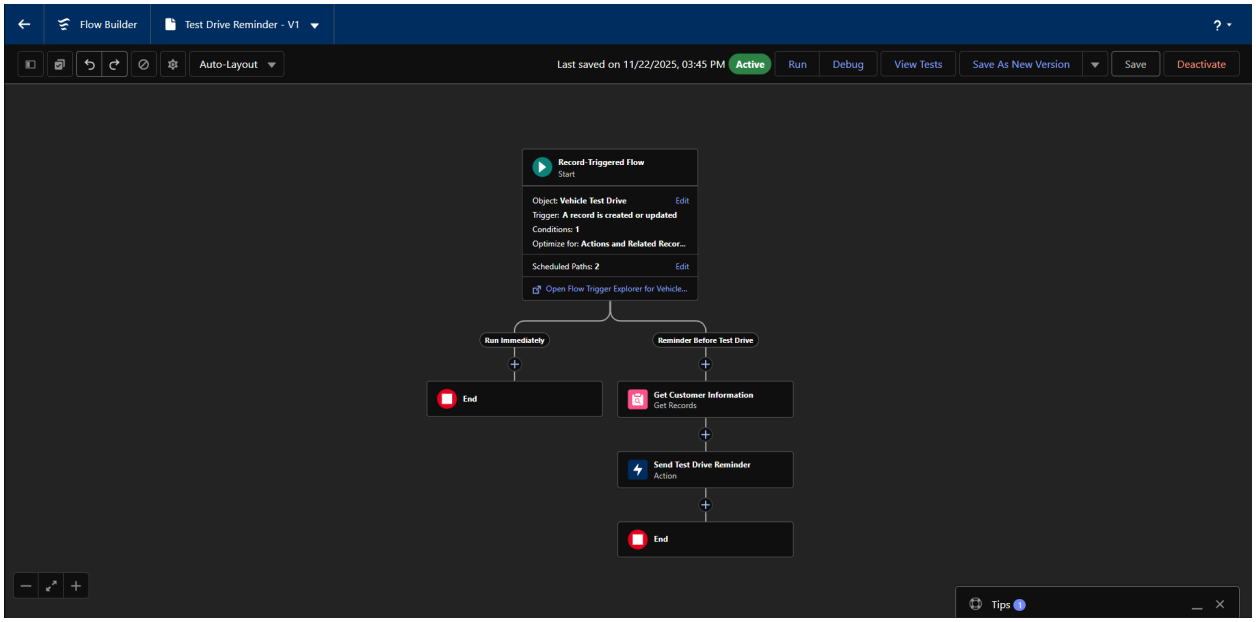


Figure 9
Test Drive Reminder Flow



Apex Classes, Triggers, Asynchronous Apex

Figures 10 to 13 shows the different apex classes and triggers that were made to handle the automation of the WhatNext Vision Motors CRM gracefully.

Figure 10

Vehicle Order Trigger Handler

Vehicle Order Trigger Handler (Apex Class): Handles stock reduction on Confirmed status; prevents orders if stock = 0.

Figure 11

Vehicle Order Trigger

```
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);

}
```

Vehicle Order Trigger: Fires on Vehicle Order insert/update.

Figure 12
Vehicle Order Batch

```
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>();
            List<Vehicle__c> vehiclesToUpdate = new List<Vehicle__c>();

            for (Vehicle_Order__c order : orderList) {
                Vehicle__c vehicle = vehicleStockMap.get(order.Vehicle__c);
                if (vehicle != null && vehicle.Stock_Quantity__c > 0) {
                    order.Status__c = 'Confirmed';
                    vehicle.Stock_Quantity__c -= 1;
                    ordersToUpdate.add(order);
                    vehiclesToUpdate.add(vehicle);
                }
            }

            if (!ordersToUpdate.isEmpty()) update ordersToUpdate;
            if (!vehiclesToUpdate.isEmpty()) update vehiclesToUpdate;
        }
    }

    global void finish(Database.BatchableContext bc) {
        System.debug('Vehicle order batch job completed.');
    }
}
```

Vehicle Order Batch (Batch Apex): Checks pending orders for out-of-stock vehicles and updates status.

Figure 13
Vehicle Order Batch Scheduler

```
global class VehicleOrderBatchScheduler implements Schedulable {

    global void execute(SchedulableContext sc) {

        VehicleOrderBatch batchJob = new VehicleOrderBatch();

        Database.executeBatch(batchJob, 50); // 50 is the batch size

    }

}
```

Vehicle Order Batch Scheduler (Schedulable Apex): Schedules batch execution.

Phase 3: UI/UX Development & Customization

Lightning App Setup

- WhatNext Vision Motors App: Contains tabs for all 6 custom objects + Reports/Dashboards; assigned to System Administrator.

Page Layouts, Dynamic Forms

- Standard page layouts used with all custom fields.

User Management

- System Administrator profile only.

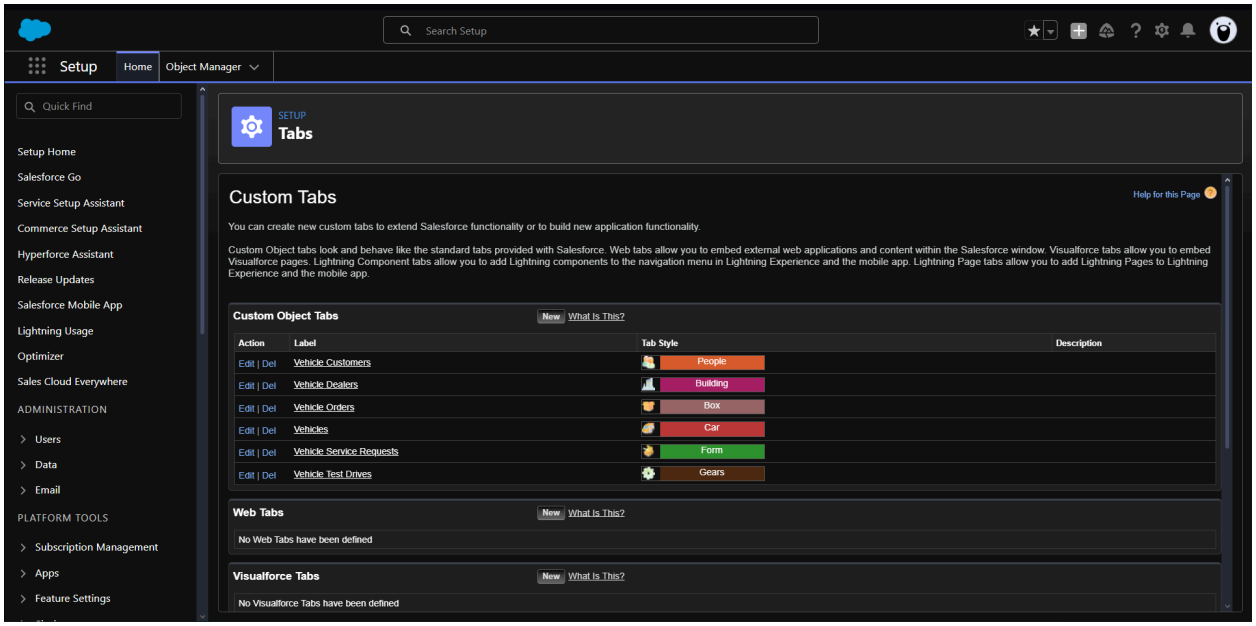
Reports and Dashboards

- Added to App; no custom reports developed in this phase.

Lightning Pages

Figure 14 shows the custom tabs that were made for the WhatNext Vision Motors CRM Project to display custom object data in a user-friendly format.

Figure 14
Custom Tabs



Custom tabs with icons: Vehicle (Car), Customer (People), Dealer (Building), Order (Box), Service (Form), Test Drive (Gear).

Phase 4: Data Migration, Testing & Security

Data Loading Process

- Manual record creation via New buttons for testing (Customers, Dealers, Vehicles).

Field History Tracking, Duplicate Rules, Matching Rules

- Not implemented in this project.

Profiles, Roles, Permission Sets, Sharing Rules

- System Administrator profile with full access.
- Standard role hierarchy. No custom sharing rules.

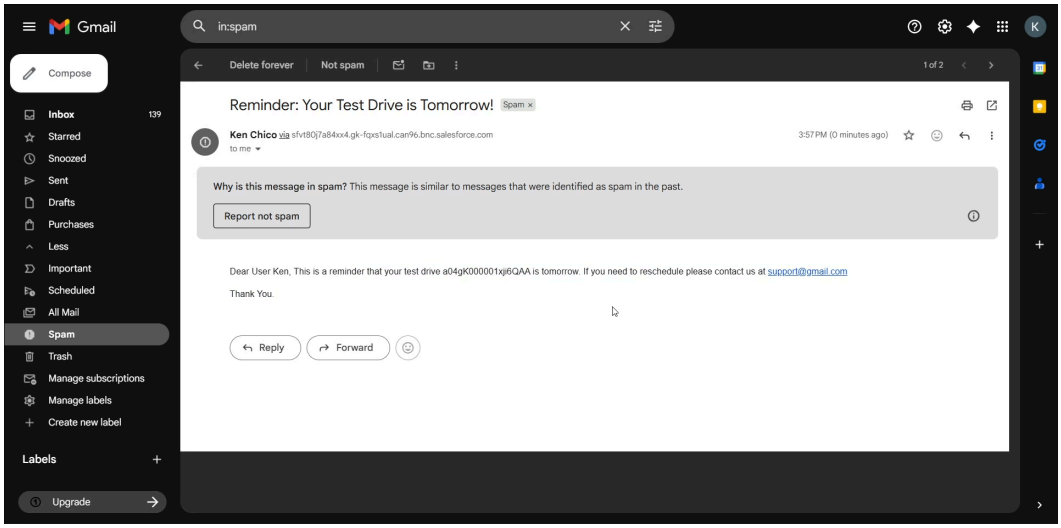
Creation of Test Classes

- Not shown; Apex classes include necessary logic for unit tests.

Test Cases with Screenshots

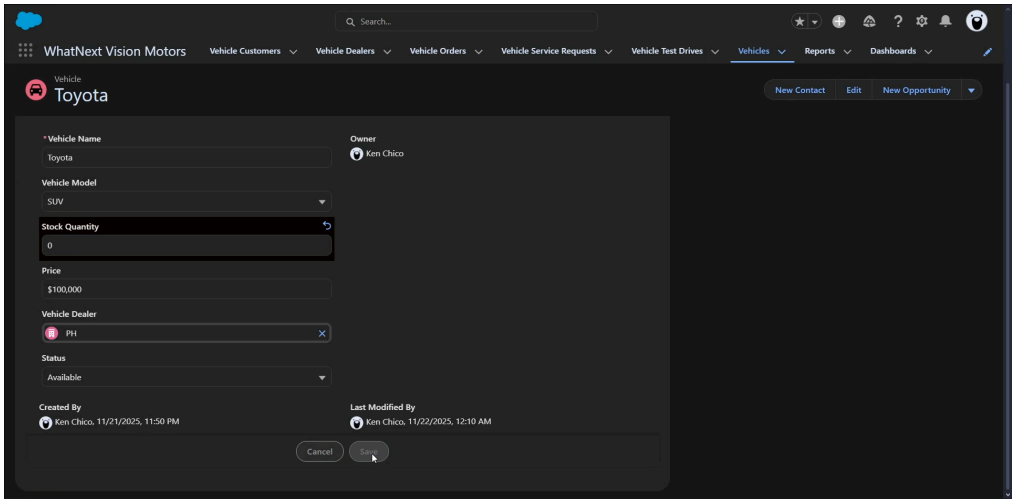
Test Case 1: Test Drive Reminder Flow

- Input: Create Vehicle Test Drive (Status=Scheduled, Date=Tomorrow, Customer=Ken).
- Expected Output: Email sent: "Reminder your test drive is tomorrow" to customer. Passed.

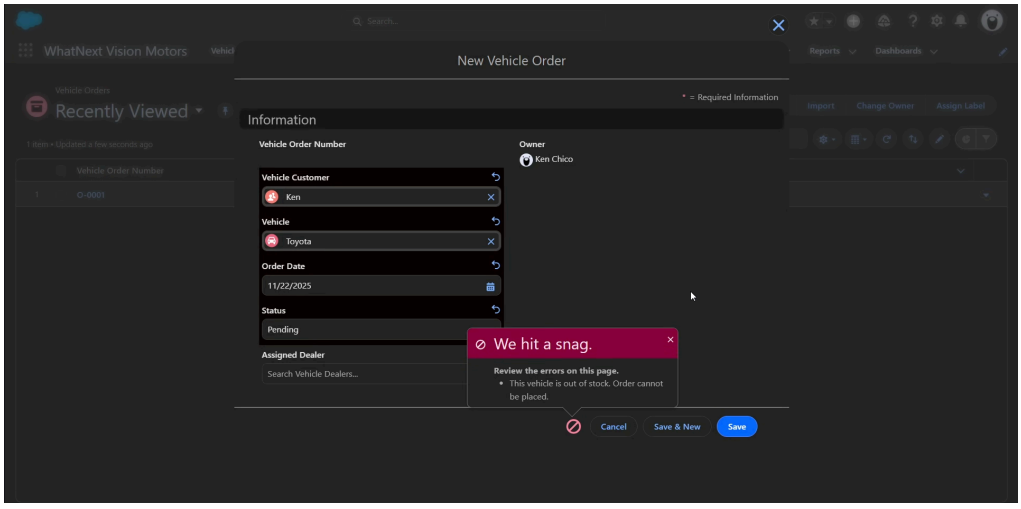


Test Case 2: Apex Stock Prevention

- Input: Stock Quantity = 0, Create New Vehicle Order.

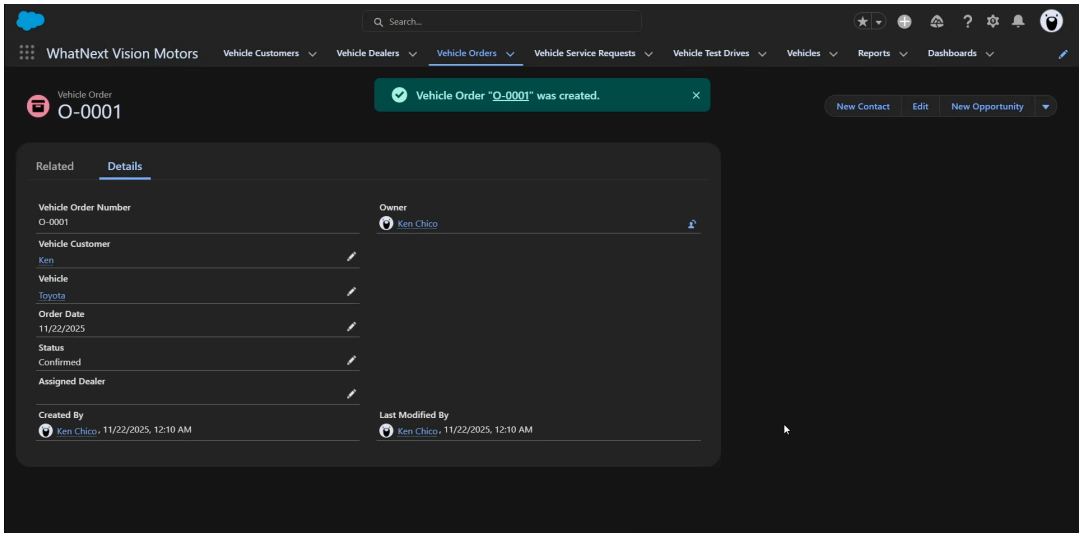


- Expected Output: Error "This vehicle is out of stock". Passed.

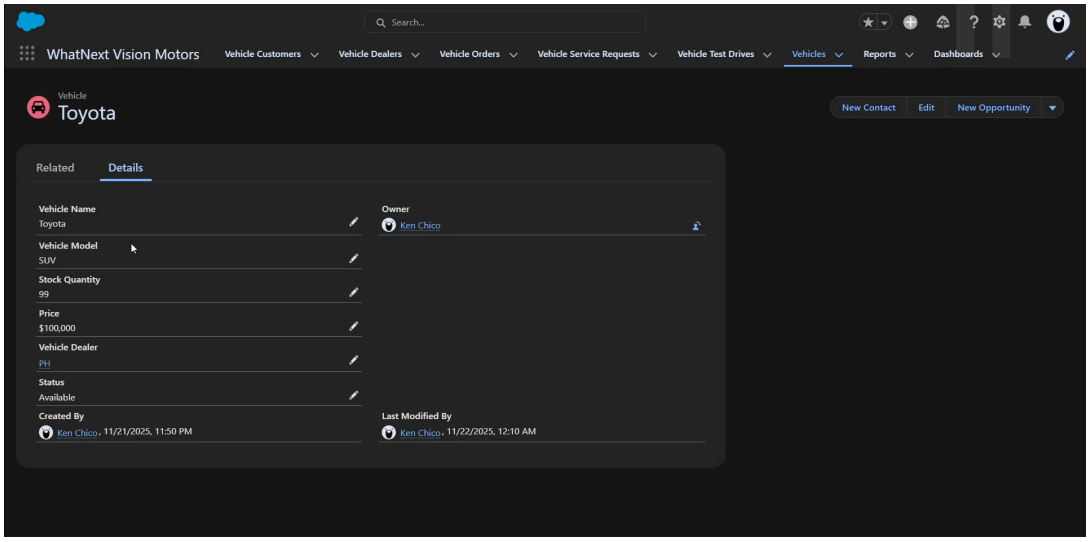


Test Case 3: Stock Reduction

- Input: Vehicle stock=100; Create Order (Status=Confirmed).



- Expected Output: Stock reduced to 99. Passed.



Testing Approach: Manual testing via record creation; verify Flows via debug logs; Apex via record updates.

Phase 5: Deployment, Documentation & Maintenance

Deployment Strategy

- Use Change Sets from Developer Org to Production; includes objects, fields, Flows, Apex classes/triggers.

System Maintenance and Monitoring

- Monitor Flows via Setup > Flows (activation status).
- Apex via Developer Console logs; Batch via scheduled jobs.

Troubleshooting Approach

1. Check Flow debug logs for assignment/email failures.
2. Review Apex triggers for stock errors.
3. Validate data (addresses matching) for geo-assignment.

Conclusion

The WhatNext Vision Motors CRM successfully automates core vehicle order processes, achieving efficient dealer assignment, stock management, and customer notifications through Salesforce native tools. All objectives met with Flows handling real-time automation and Apex ensuring data integrity.

Future Enhancements:

- Integrate Einstein AI for dealer recommendation scoring.
- Add chatbot for test drive scheduling.
- Implement full Data Loader for bulk migrations.
- Mobile Lightning Web Components for customers.