

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

ABSTRACT

The automotive industry is rapidly evolving, and customer expectations demand faster service, real-time communication, and personalized experiences. WhatNext Vision Motors, a forward-thinking mobility brand, recognized the need for digital transformation to scale operations and enhance customer satisfaction. This Salesforce CRM solution focuses on automating the vehicle ordering process, enabling real-time inventory tracking, and integrating intelligent logic for dealer assignment and order validation. Built on the robust Salesforce platform, the project combines low-code tools like Flows with Apex-driven automation to deliver a scalable, secure, and customer-first CRM infrastructure.

OBJECTIVE

The primary goal of this project is to build a scalable and intelligent Salesforce CRM tailored to the operations of WhatNext Vision Motors, with the following key objectives:

- **Smart Order Management:** Prevent out-of-stock orders with real-time stock validation.
- **Automated Dealer Assignment:** Suggest the nearest dealer based on the customer's location.
- **Order Status Automation:** Update order records using scheduled batch processes.
- **Test Drive Reminder System:** Send automated email reminders to customers before test drives.
- **Enhanced UX with Lightning:** Provide a clean, fast, and mobile-friendly CRM experience.

TECHNOLOGY DESCRIPTION

This solution was built using a combination of declarative and programmatic Salesforce tools:

- **Custom Objects:** Vehicle, Customer, Dealer, Orders, Test Drives, Services.
- **Flow Builder:** Record-triggered flows for dealer assignment and email reminders.
- **Apex Triggers:** Real-time validations for stock checks and auto stock reduction.
- **Batch Apex & Scheduler:** Scheduled job to confirm orders if inventory is available.
- **Validation Rules:** Enforces clean data (e.g., status rules, required fields).
- **Lightning App Builder:** Custom navigation with relevant tabs and role-specific layouts.
- **Permission Sets & Profiles:** Secure access based on roles (Admin, Dealer, Service Rep).

PROJECT EXECUTION PHASES

1. Developer Org Setup

- Created a Developer Edition org.
- Enabled Lightning Experience and APIs.
- Configured security roles and profiles for Admins and Dealers.
- Visualized schema using Schema Builder.

2. Lightning App & UI Setup

- Created a Lightning App: WhatNext Vision Motors *CRM*.
- Added custom tabs: Vehicle, Dealer, Customer, Orders, Test Drives, Service Requests.

- Customized layouts with dynamic forms, quick actions, and compact layouts.

3. Custom Object Configuration

Each object was defined with relationships and automation:

- **Vehicle__c**: Stores vehicle details, price, status, and stock quantity.
- **Vehicle_Customer__c**: Stores personal info, address, and preferred vehicle type.
- **Vehicle_Dealer__c**: Stores dealership data and location.
- **Vehicle_Order__c**: Linked to customer and vehicle, includes order date and status.
- **Vehicle_Test_Drive__c**: Tracks customer test drive bookings and status.
- **Vehicle_Service_Request__c**: Manages after-sales service appointments.

4. Business Logic Implementation

- **Validation Rule**: Prevents placing an order if stock is zero.
- **Trigger Handler**: Automatically reduces vehicle stock after order confirmation.
- **Batch Apex**: Updates pending orders to confirmed if stock becomes available.
- **Scheduled Job**: Runs batch job daily at midnight to update inventory/order status.

BUSINESS PROCESS AUTOMATION

1. Auto-Assign Nearest Dealer (Flow)

- **Trigger**: When a new Vehicle Order is created with status = Pending.
- **Logic**: Retrieves customer address, matches with dealer location, and assigns the nearest one.
- **Outcome**: Reduces manual effort and ensures faster service delivery.

2. Email Reminder for Test Drive (Scheduled Flow)

- **Trigger**: One day before the scheduled test drive.
- **Logic**: Fetches customer details and sends a personalized email.
- **Outcome**: Increases customer engagement and reduces test drive no-shows.

3. Order Validation (Apex Trigger)

- **Trigger**: Before Insert/Update on Vehicle_Order__c.
- **Logic**: Checks vehicle stock and prevents the order if quantity is zero.
- **Outcome**: Improves order accuracy and eliminates invalid bookings.

4. Batch Apex Order Processor

- **Trigger**: Scheduled via Apex Scheduler.
- **Logic**: Fetches all pending orders, confirms them if stock exists, and updates records.
- **Outcome**: Keeps order data fresh and reduces manual review.

REAL WORLD WORKFLOW EXAMPLE

Scenario: Kunal places an order for an EV sedan.

1. **Customer Record Creation:** Kunal's details (name, email, address) are entered and saved.

WhatNext Vision Motors

Vehicle Customer AYESHA

Related Details

Vehicle Customer Name
AYESHA

Email
ayeshacherkuri@gmail.com

Phone
(891) 994-5294

Address
sri sainath nagar

Preferred Vehicle Type
Sedan

Created By
AYESHA CHERUKURU, 12/5/2025, 7:50 AM

Owner
AYESHA CHERUKURU

Address
sri sainath nagar

Preferred Vehicle Type
Sedan

Created By
AYESHA CHERUKURU, 12/5/2025, 7:50 AM

Last Modified By
AYESHA CHERUKURU, 12/7/2025, 12:08 AM

Cancel Save & New Save

Figure 1: customer record creation

WhatNext Vision Motors

Vehicle Customers

Vehicle Customers Details

Vehicle Customer Name
AYESHA

Email
ayeshacherkuri@gmail.com

Phone
dqbduwfg

Address
sri sainath nagar

Preferred Vehicle Type
Sedan

Created By
AYESHA CHERUKURU, 12/5/2025, 7:50 AM

Owner
AYESHA CHERUKURU

Address
sri sainath nagar

Preferred Vehicle Type
Sedan

Created By
AYESHA CHERUKURU, 12/5/2025, 7:50 AM

Last Modified By
AYESHA CHERUKURU, 12/5/2025, 7:50 AM

New Contact Edit New Opportunity

Figure 1.1: customer record created

2. Dealer Record Creation: Dealer details are created and saved.

Vehicle Dealer Name: will

Dealer Location: hyderabad

Dealer Code: DC-0001

Phone: 86239846109

Email: ayeshacherkuri@gmail.com

Owner: AYESHA CHERUKURU

Created By: AYESHA CHERUKURU, 12/5/2025, 7:50 AM

Last Modified By: AYESHA CHERUKURU, 12/5/2025, 7:50 AM

Figure 2: Dealer record creation

Vehicle Dealer Name: mike

Dealer Location: bengaluru

Dealer Code: DC-0002

Phone: 897639816

Email: ayeshacherkuri@gmail.com

Owner: AYESHA CHERUKURU

Created By: AYESHA CHERUKURU, 12/5/2025, 7:51 AM

Figure 2.1: Dealer record created

3. **Vehicle Setup:** An EV model is listed with price ₹15,00,000 and stock = 2.

The screenshot shows the Salesforce Lightning interface for creating a vehicle record. The page title is "Edit pulser". The form fields are as follows:

- Vehicle Name:** pulser
- Vehicle Model:** EV
- Stock Quantity:** 299
- Price:** \$40,000
- Vehicle Dealer:** mike
- Status:** Available

Owner: AYESHA CHERUKURU

Created By: AYESHA CHERUKURU, 12/5/2025, 7:53 AM

Last Modified By: AYESHA CHERUKURU, 12/5/2025, 8:42 AM

Buttons: Cancel, Save & New, Save

Figure 3: vehicle record creation

The screenshot shows the Salesforce Lightning interface displaying the created vehicle record. The page title is "Vehicles". The vehicle details are:

- Vehicle Name:** pulser
- Vehicle Model:** EV
- Stock Quantity:** 299
- Price:** \$40,000
- Vehicle Dealer:** mike
- Status:** Available

Owner: AYESHA CHERUKURU

Created By: AYESHA CHERUKURU, 12/5/2025, 7:53 AM

Last Modified By: AYESHA CHERUKURU, 12/5/2025, 8:42 AM

Buttons: New Contact, Edit, New Opportunity

Figure 3.1: vehicle record created

4. Order Creation: Kunal books the EV → Status set to "Pending".

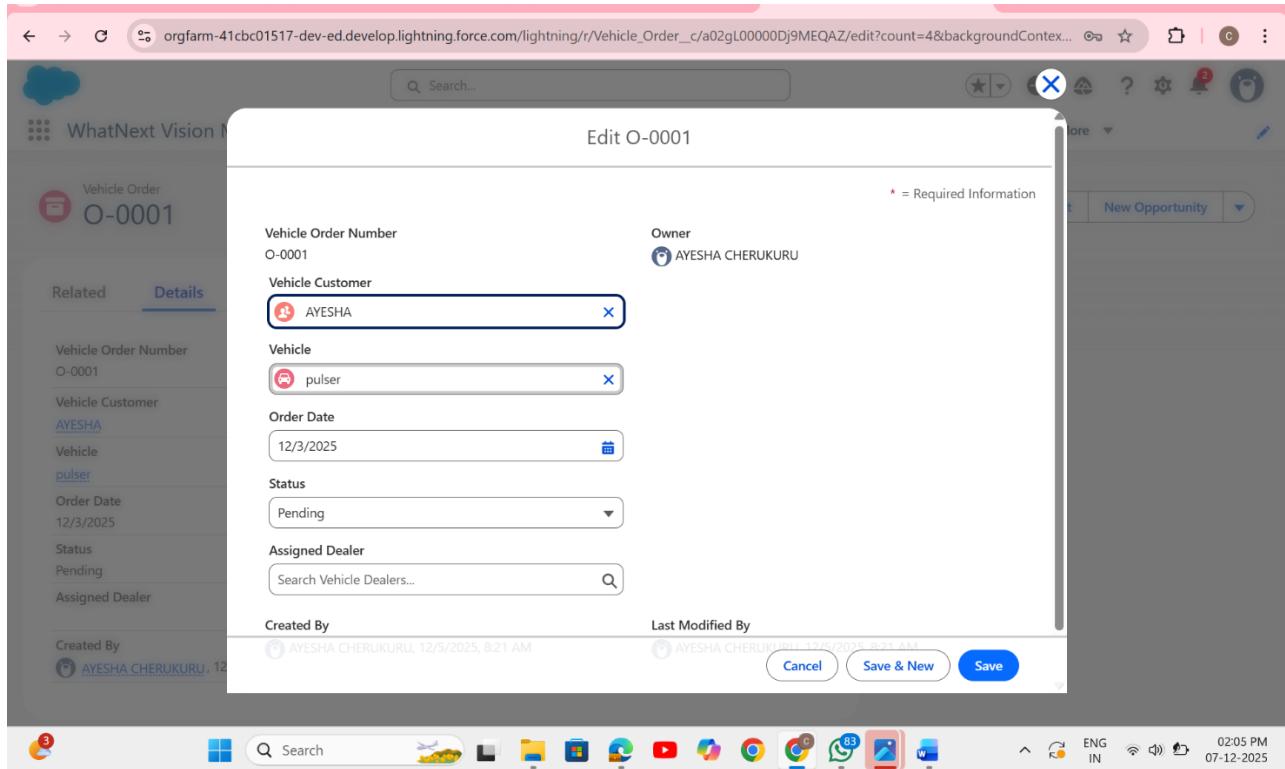


Figure 4: order creation

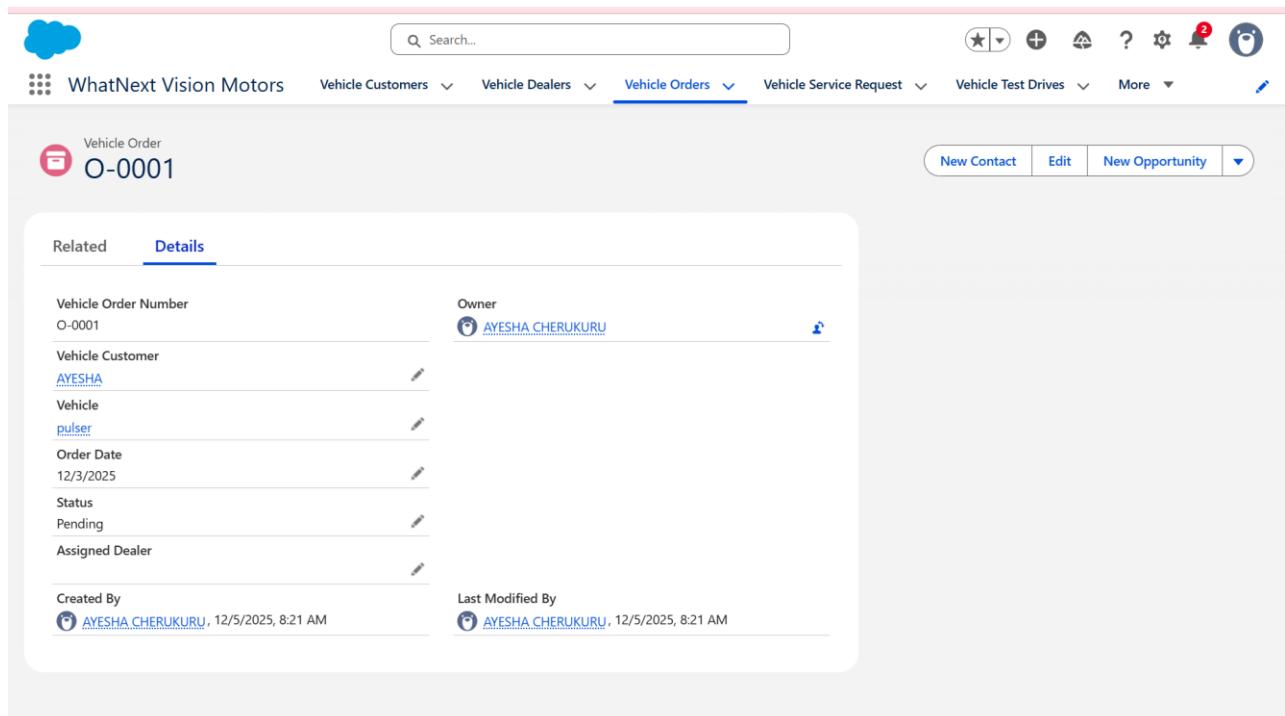


Figure 4.1: order created

5. **Dealer Assigned:** Based on Kunal's address, the nearest dealer is assigned via flow.

New Vehicle Order

Information

Vehicle Order Number: O-0007

Vehicle Customer: Kunal

Vehicle: Royal enfield

Order date: 8/4/2025

Status: Confirmed

Assigned Dealer: Guravaiah

Owner: CHITIRALA SAI MADHU KEERTHI

Figure 5: dealer assigning

Vehicle Order O-0002

Related Details

Vehicle Order Number: O-0002

Vehicle Customer: AYESHA

Vehicle: pulser

Order Date: 12/3/2025

Status: Confirmed

Assigned Dealer: (not explicitly shown)

Created By: AYESHA CHERUKURU, 12/5/2025, 8:42 AM

Last Modified By: AYESHA CHERUKURU, 12/5/2025, 8:42 AM

Owner: AYESHA CHERUKURU

Figure 5.1: dealer assigned

6. Order Confirmation: When stock is verified → Status auto-updated to "Confirmed".

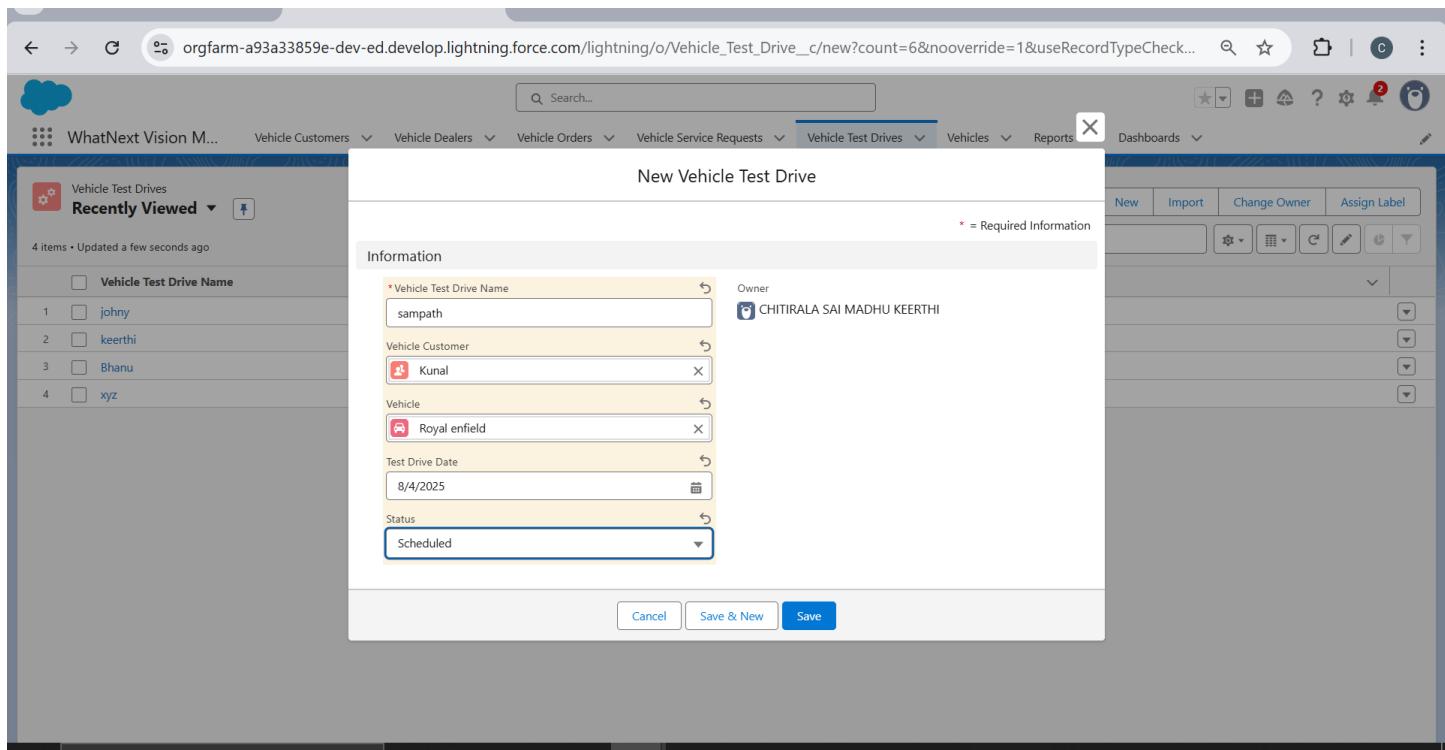


Figure 6: order confirming

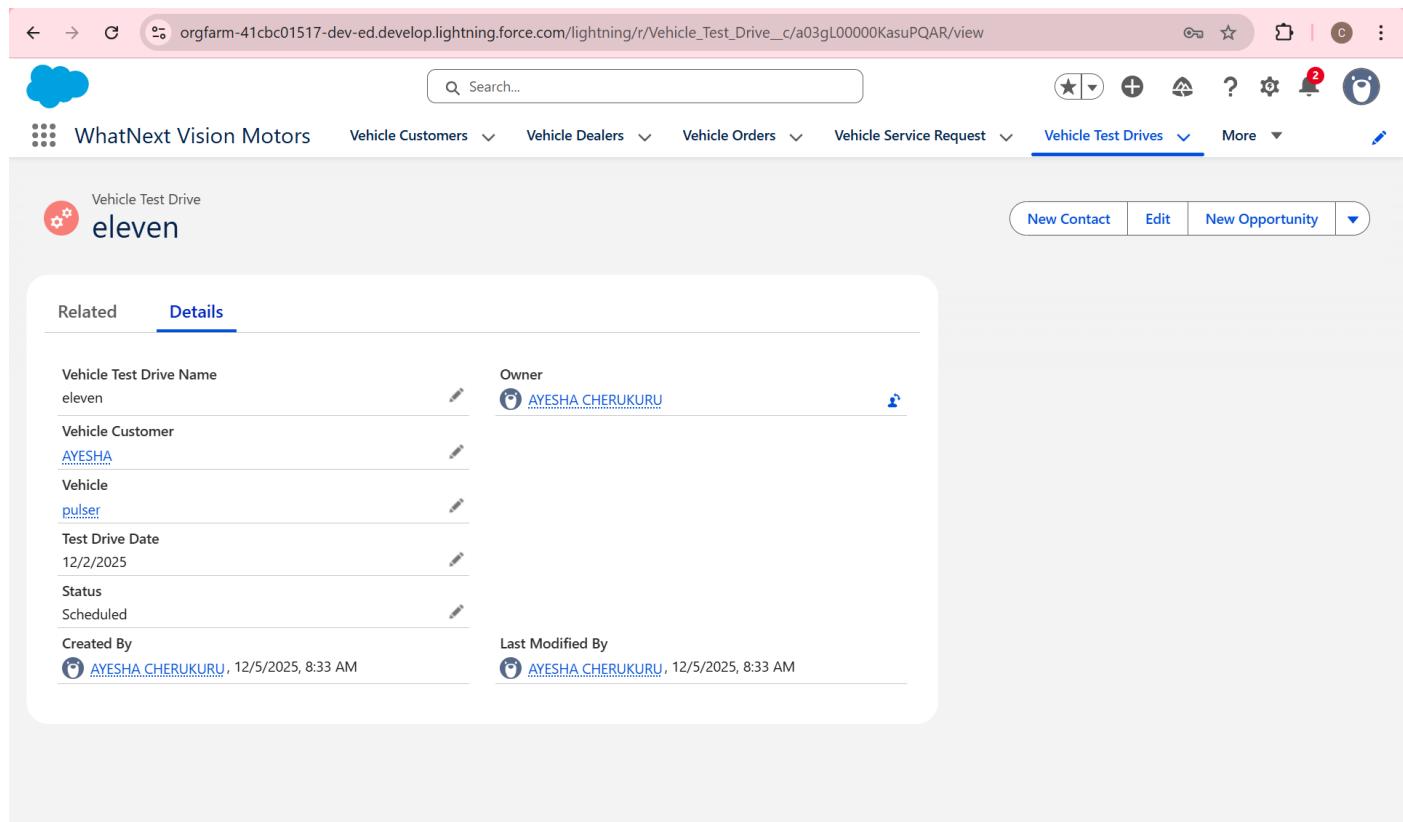


Figure 6.1: order confirmation

7. **Email Sent:** Kunal receives confirmation email.

The screenshot shows a Gmail inbox with 42 messages. A message from 'CHITIRALA SAI MADHU KEERTHI' is highlighted, labeled as 'Spam'. The subject is 'Reminder: Your Test Drive is Tomorrow!'. The message body contains a reminder about a test drive and contact information: 'Dear User Kunal, This is a remainder that your test drive a04gL000008Vm45QAC is tomorrow. If you need to reschedule please contact us at support@gmail.com Thank You.' Below the message are 'Reply', 'Forward', and a smiley face icon.

Figure 7: email received

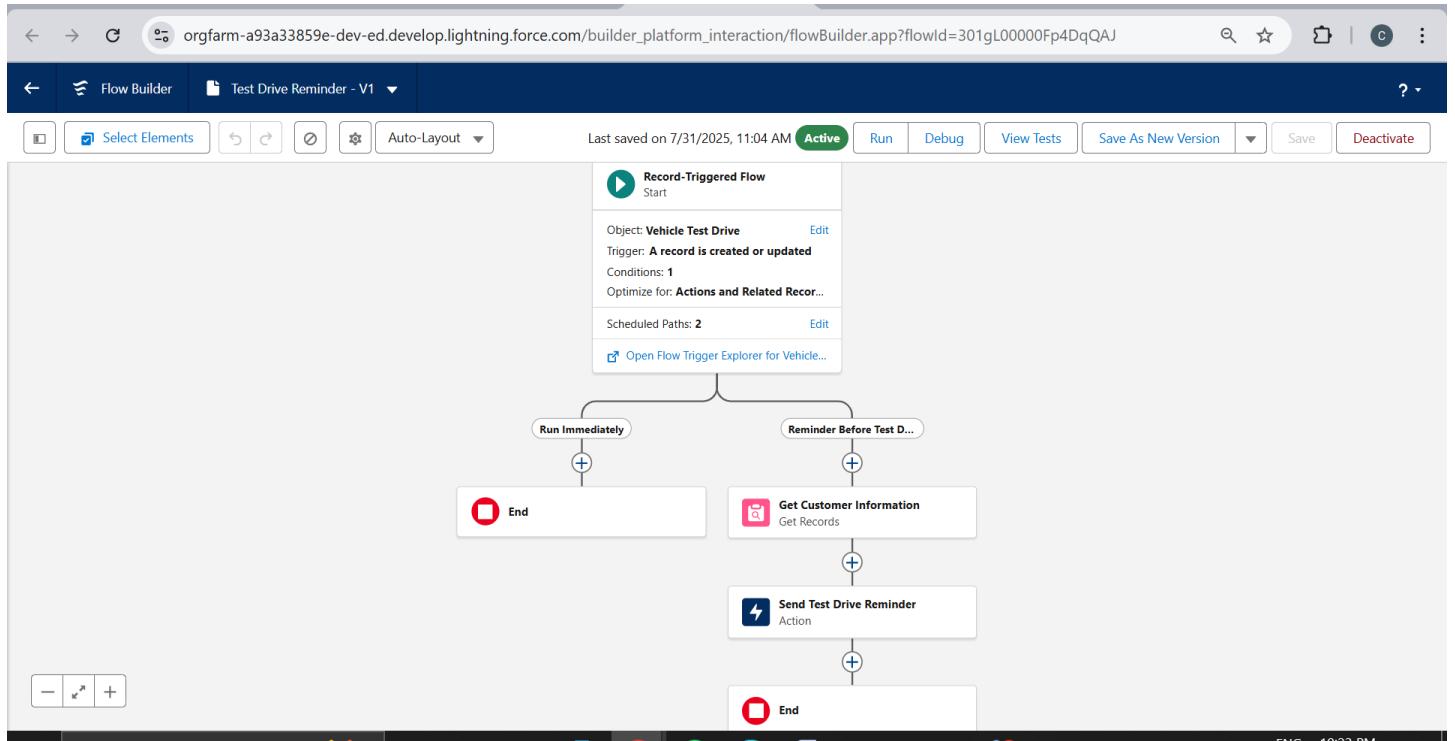
8. **Stock Updated:** Remaining stock becomes 1.

The screenshot shows a Salesforce Lightning page for a vehicle named 'pulser'. The 'Details' tab is selected. Key fields shown include: Vehicle Name (pulser), Vehicle Model (EV), Stock Quantity (299), Price (\$40,000), Vehicle Dealer (mike), Status (Available), Created By (AYESHA CHERUKURU) on 12/5/2025, 7:53 AM, and Last Modified By (AYESHA CHERUKURU) on 12/5/2025, 8:42 AM. The top navigation bar includes links for Vehicle Customers, Vehicle Dealers, Vehicle Orders, Vehicle Service Request, Vehicles, and More.

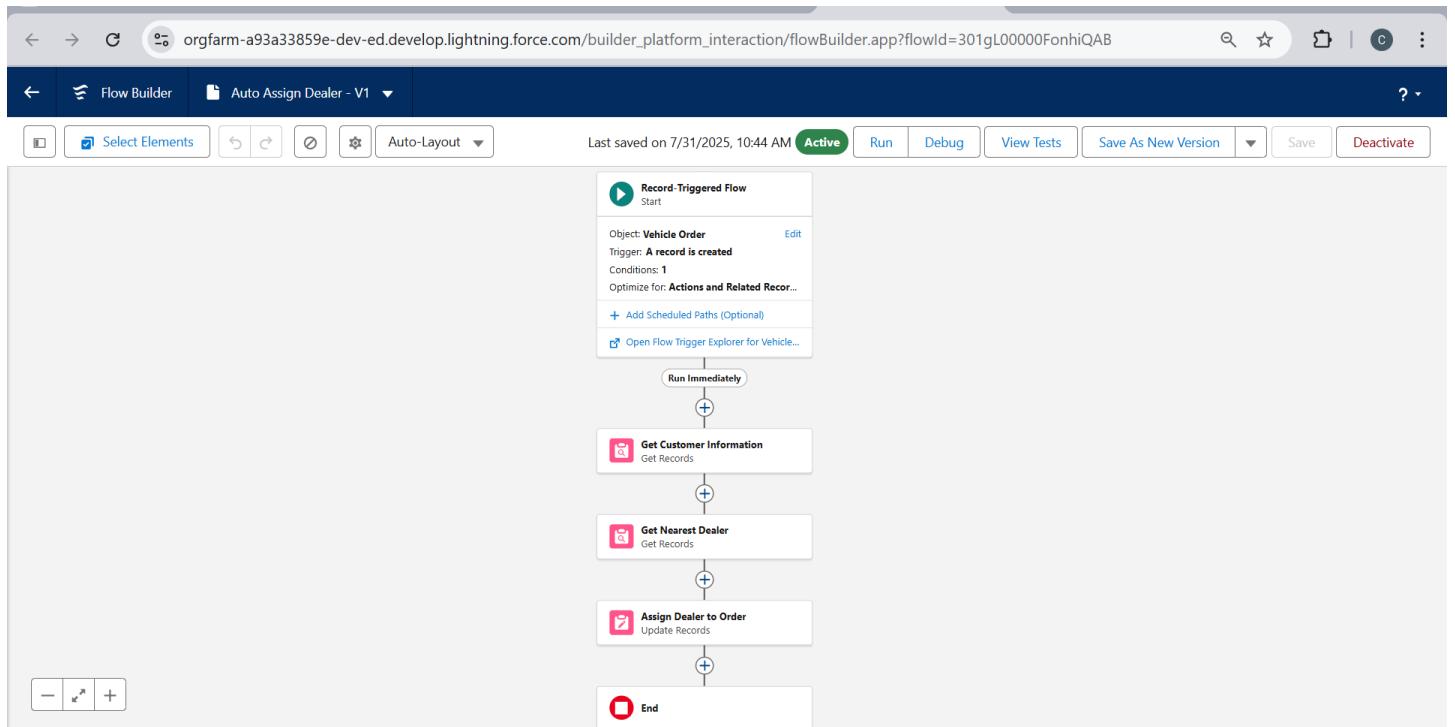
Figure 8: stock updated

FLOW SCREENSHOTS

Flow 1: Test drive remainder



Flow 2: Auto assign dealer



CODES

Code 1: VehicleOrderTriggerHandler

```
public class VehicleOrderTriggerHandler {  
    public static void handleTrigger(List<Vehicle_Order_c> newOrders, Map<Id, Vehicle_Order_c>  
        oldOrders, Boolean isBefore, Boolean isAfter, Boolean isInsert, Boolean isUpdate) {  
        if (isBefore && (isInsert || isUpdate)) {  
            preventOrderIfOutOfStock(newOrders);  
        }  
        if (isAfter && (isInsert || isUpdate)) {  
            updateStockOnOrderPlacement(newOrders);  
        }  
    }  
    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>(  
                [SELECT Id, Stock_Quantity_c FROM Vehicle_c WHERE Id IN :vehicleIds]  
            );  
            for (Vehicle_Order_c order : orders) {  
                Vehicle_c vehicle = vehicleStockMap.get(order.Vehicle_c);  
                if (vehicle != null && vehicle.Stock_Quantity_c <= 0) {  
                    orderaddError('This vehicle is out of stock. Order cannot be placed.');//  
                }  
            }  
        }  
    }  
    private static void updateStockOnOrderPlacement(List<Vehicle_Order_c> orders) {  
        Set<Id> vehicleIds = new Set<Id>();  
        for (Vehicle_Order_c order : orders) {  
            if (order.Vehicle_c != null && order.Status_c == 'Confirmed') {  
                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__c> vehiclesToUpdate = new List<Vehicle__c>();
for (Vehicle_Order__c order : orders) {
    Vehicle__c vehicle = vehicleStockMap.get(order.Vehicle__c);
    if (vehicle != null && vehicle.Stock_Quantity__c > 0) {
        vehicle.Stock_Quantity__c -= 1;
        vehiclesToUpdate.add(vehicle);
    }
}
if (!vehiclesToUpdate.isEmpty()) {
    update vehiclesToUpdate;
}
}
}
}
}

```

Code 2: VehicleOrderBatch

```

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);
                }
            }
            update vehiclesToUpdate;
        }
    }
}

```

```

        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.');
}
}

```

Code 3: VehicleOrderBatchScheduler

```

global class VehicleOrderBatchScheduler implements Schedulable {
    global void execute(SchedulableContext sc) {
        VehicleOrderBatch batchJob = new VehicleOrderBatch();
        Database.executeBatch(batchJob, 50); // 50 = batch size
    }
}

```

CONCLUSION

The WhatNext Vision Motors Salesforce CRM implementation successfully demonstrates the power of automation in delivering customer-centric, intelligent, and scalable vehicle sales and service operations. From smart dealer assignment to scheduled order processing and proactive customer engagement, the CRM supports a 360° view of the mobility sales cycle. With strong foundations in both declarative and programmatic Salesforce capabilities, the system is future-ready, capable of expanding into electric mobility, shared vehicle solutions, and digital servicing.

Through seamless integration of business logic and user experience, this CRM stands as a model of innovation in automotive tech aligning with the brand's mission to drive mobility forward with excellence.