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

Project Overview

WhatNext Vision Motors is currently in the process of fundamentally transforming and significantly upgrading both its customer-facing interactions and internal operational workflows through the strategic deployment of a state-of-the-art, high-tech Salesforce Customer Relationship Management (CRM) system. This comprehensive modernization project is explicitly designed to optimize and streamline the entire lifecycle of vehicle ordering; it achieves this by utilizing intelligent algorithms to automatically route and assign new orders to the specific dealership situated closest to the customer's geographic location, while simultaneously enforcing strict inventory controls to effectively block and prevent the processing of any orders for vehicles that are currently out of stock or unavailable.

Furthermore, the system incorporates sophisticated automated workflows that are capable of dynamically updating the status of orders in real-time as they progress through the pipeline, as well as managing customer engagement by dispatching pre-scheduled, automated email notifications to remind clients of their upcoming test drive appointments. From a technical architecture perspective, the solution relies on several robust implementation components, specifically leveraging Apex triggers to perform immediate and rigorous validation of stock levels, utilizing batch jobs to handle large-scale volume updates to inventory data, and employing scheduled Apex classes to facilitate the fully automated processing of orders at set intervals. Ultimately, this strategic initiative serves to significantly elevate overall levels of customer satisfaction, drastically improve the precision and accuracy of order management, and substantially boost the company's total operational efficiency and productivity.

Objectives

The key objectives of this Salesforce CRM implementation are:

1. **Automate Order and Dealer Assignment**
    - Automatically assign the nearest dealer based on the customer’s city at the time of order placement.
2. **Prevent Out-of-Stock Orders**
    - Ensure customers can only place orders for vehicles currently in stock using validation rules and Apex triggers.
3. **Send Test Drive Reminders**
    - Use scheduled email flows to remind customers of their upcoming test drives, reducing missed appointments.

4. **Improve User Experience**
    - Implement Lightning Apps and Dynamic Forms to provide a clean, responsive interface for managing records.
5. **Maintain a Scalable Backend**
    - Use modular Apex classes and scheduled batch jobs to automate stock updates and order confirmations in bulk.

Phase 1: Requirement Analysis & Planning

The initial phase of the project focused on understanding the business needs of WhatNext Vision Motors and translating them into system requirements within the Salesforce ecosystem. The objective was to build a CRM that supports the complete vehicle management lifecycle—starting from inventory tracking to customer orders and post-sales interactions.

Business Requirements

The following core requirements were identified:

- Centralized storage and management of vehicle, dealer, and customer data.
- Real-time validation of vehicle stock at the time of order placement.
- Automatic assignment of the nearest dealer based on customer address.
- Tracking of test drives and vehicle service requests.
- Automation of key workflows to reduce manual intervention.

Defining Project Scope

To meet the business objectives, the system was designed to include:

- Custom objects for managing vehicles, orders, dealers, customers, test drives, and service requests.

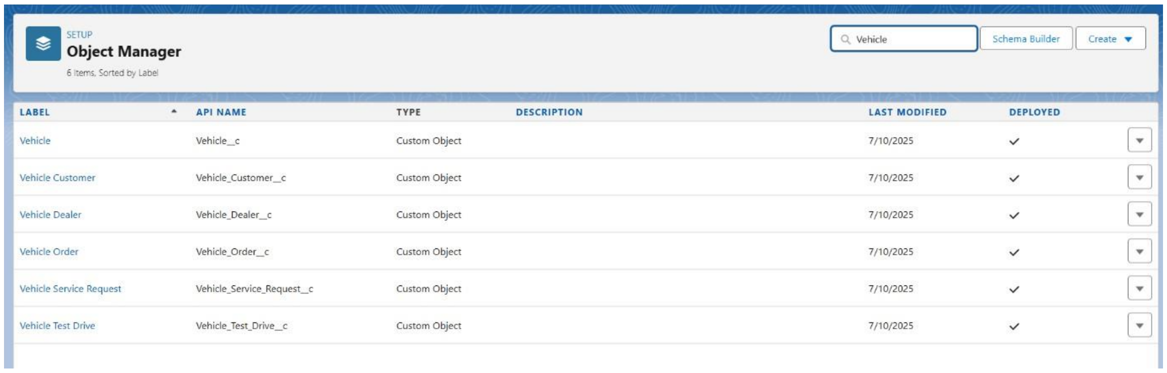
- Record-triggered flows to assign dealers and send email notifications.
- Apex triggers to validate stock availability and update inventory levels.
- Batch Apex to process pending orders based on stock replenishment.

Data Model

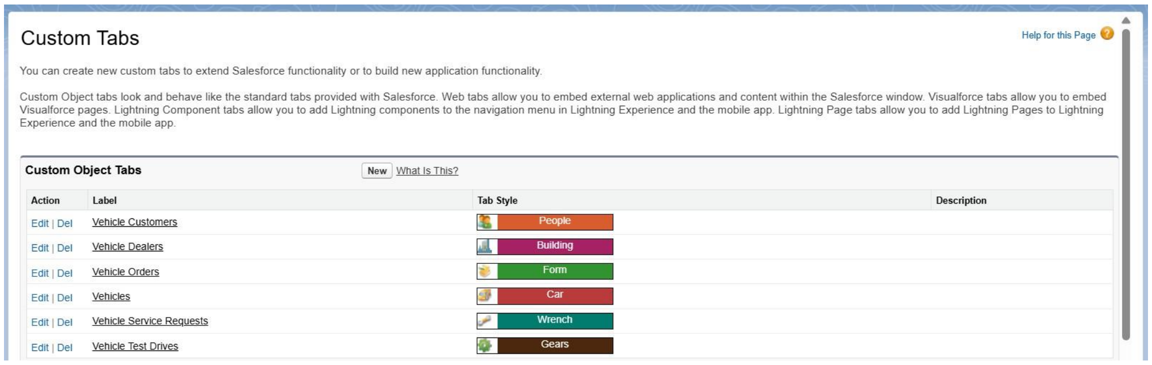
Six custom objects were created to reflect the business structure:

Object Name	**Purpose**
Vehicle__c	Stores vehicle details and stock info
Vehicle_Dealer__c	Contains dealer information
Vehicle_Customer__c	Stores customer details
Vehicle_Order__c	Tracks vehicle orders
Vehicle_Test_Drive__c	Schedules and tracks test drives
Vehicle_Service_Request__c	Manages service history and issues

These objects are interlinked using lookup relationships to ensure data integrity.


SETUP
Object Manager
6 Items, Sorted by Label

LABEL	API NAME	TYPE	DESCRIPTION	LAST MODIFIED	DEPLOYED
Vehicle	Vehicle__c	Custom Object		7/10/2025	✓
Vehicle Customer	Vehicle_Customer__c	Custom Object		7/10/2025	✓
Vehicle Dealer	Vehicle_Dealer__c	Custom Object		7/10/2025	✓
Vehicle Order	Vehicle_Order__c	Custom Object		7/10/2025	✓
Vehicle Service Request	Vehicle_Service_Request__c	Custom Object		7/10/2025	✓
Vehicle Test Drive	Vehicle_Test_Drive__c	Custom Object		7/10/2025	✓


Custom Tabs

You can create new custom tabs to extend Salesforce functionality or to build new application functionality.

Custom Object tabs look and behave like the standard tabs provided with Salesforce. Web tabs allow you to embed external web applications and content within the Salesforce window. Visualforce tabs allow you to embed Visualforce pages. Lightning Component tabs allow you to add Lightning components to the navigation menu in Lightning Experience and the mobile app. Lightning Page tabs allow you to add Lightning Pages to Lightning Experience and the mobile app.

Custom Object Tabs

Action	Label	Tab Style	Description
Edit | Del	Vehicle Customers	People	
Edit | Del	Vehicle Dealers	Building	
Edit | Del	Vehicle Orders	Form	
Edit | Del	Vehicles	Car	
Edit | Del	Vehicle Service Requests	Wrench	
Edit | Del	Vehicle Test Drives	Gears	

Security Model

- Standard Salesforce profiles were used with additional **Permission Sets** to grant access to custom objects.
- **Field-Level Security** and **Role Hierarchy** ensured that users could only view or edit data relevant to their responsibilities.
- **Field History Tracking** was enabled on critical fields such as Stock_Quantity__c (Vehicle) and Status__c (Order) for audit purposes.

Phase 2: Salesforce Development – Backend & Configurations

- **Setup Environment & DevOps Workflow**

  To begin the development process, a **Salesforce Developer Org** was set up for building and testing all customizations and automation features.

  - **Environment:** Salesforce Lightning Experience (Developer Edition)
  - **User Profiles/Roles:** Standard profiles were used for testing. No custom profiles were created.
  - **Deployment Method:** Metadata was deployed using **Change Sets** from the sandbox to production.

- **Customization of Objects, Fields, Validation Rules, and Automation**

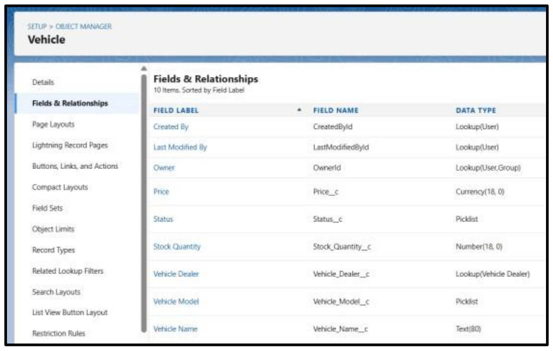
Custom Objects and Fields

The following custom objects were created and configured to support the business flow:

- **Vehicle** – Stores vehicle name, stock count, model, etc.
- **Dealer** – Stores dealer location and vehicle availability
- **Customer** – Stores customer details and address
- **Order** – Captures vehicle orders and order status

Relationships:

- Order → Vehicle: Lookup
- Order → Dealer: Lookup
- Order → Customer: Master-Detail or Lookup (based on implementation)



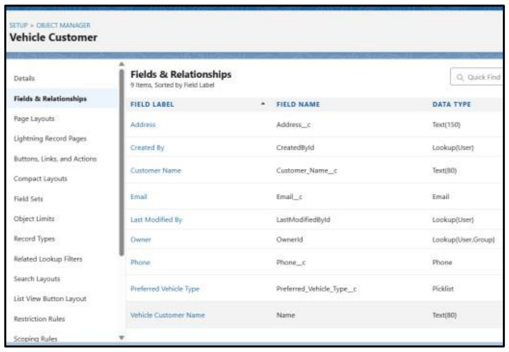
SETUP > OBJECT MANAGER

Vehicle

Fields & Relationships

10 Items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE
Created By	CreatedById	Lookup(User)
Last Modified By	LastModifiedById	Lookup(User)
Owner	OwnerId	Lookup(User,Group)
Price	Price__c	Currency(18, 0)
Status	Status__c	Picklist
Stock Quantity	Stock_Quantity__c	Number(18, 0)
Vehicle Dealer	Vehicle_Dealer__c	Lookup(Vehicle Dealer)
Vehicle Model	Vehicle_Model__c	Picklist
Vehicle Name	Vehicle_Name__c	Text(80)



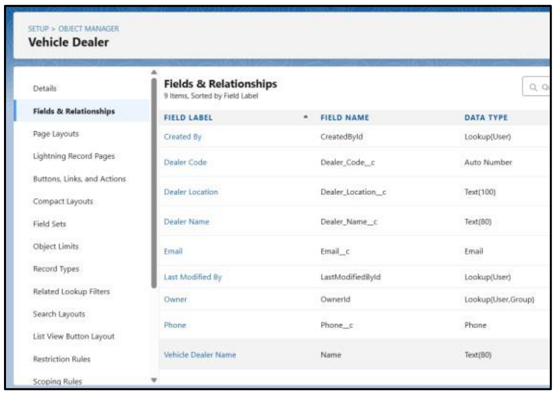
SETUP > OBJECT MANAGER

Vehicle Customer

Fields & Relationships

9 Items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE
Address	Address__c	Text(150)
Created By	CreatedById	Lookup(User)
Customer Name	Customer_Name__c	Text(80)
Email	Email__c	Email
Last Modified By	LastModifiedById	Lookup(User)
Owner	OwnerId	Lookup(User,Group)
Phone	Phone__c	Phone
Preferred Vehicle Type	Preferred_Vehicle_Type__c	Picklist
Vehicle Customer Name	Name	Text(80)



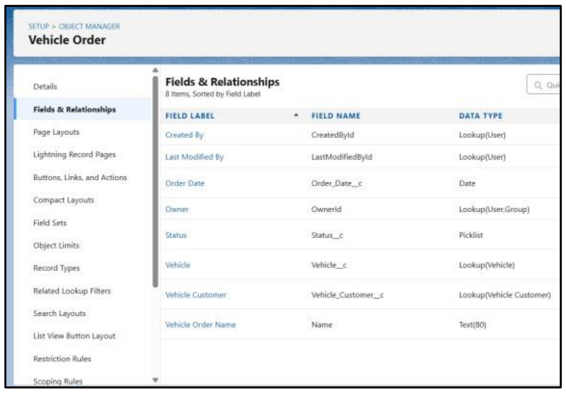
SETUP > OBJECT MANAGER

Vehicle Dealer

Fields & Relationships

9 Items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE
Created By	CreatedById	Lookup(User)
Dealer Code	Dealer_Code__c	Auto Number
Dealer Location	Dealer_Location__c	Text(100)
Dealer Name	Dealer_Name__c	Text(80)
Email	Email__c	Email
Last Modified By	LastModifiedById	Lookup(User)
Owner	OwnerId	Lookup(User,Group)
Phone	Phone__c	Phone
Vehicle Dealer Name	Name	Text(80)



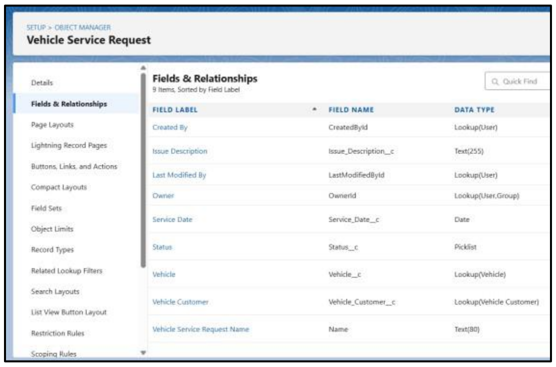
SETUP > OBJECT MANAGER

Vehicle Order

Fields & Relationships

8 Items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE
Created By	CreatedById	Lookup(User)
Last Modified By	LastModifiedById	Lookup(User)
Order Date	Order_Date__c	Date
Owner	OwnerId	Lookup(User,Group)
Status	Status__c	Picklist
Vehicle	Vehicle__c	Lookup(Vehicle)
Vehicle Customer	Vehicle_Customer__c	Lookup(Vehicle Customer)
Vehicle Order Name	Name	Text(80)



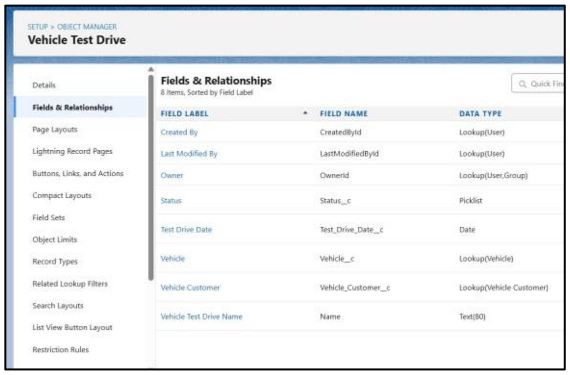
SETUP > OBJECT MANAGER

Vehicle Service Request

Fields & Relationships

9 Items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE
Created By	CreatedById	Lookup(User)
Issue Description	Issue_Description__c	Text(255)
Last Modified By	LastModifiedById	Lookup(User)
Owner	OwnerId	Lookup(User,Group)
Service Date	Service_Date__c	Date
Status	Status__c	Picklist
Vehicle	Vehicle__c	Lookup(Vehicle)
Vehicle Customer	Vehicle_Customer__c	Lookup(Vehicle Customer)
Vehicle Service Request Name	Name	Text(80)



SETUP > OBJECT MANAGER

Vehicle Test Drive

Fields & Relationships

8 Items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE
Created By	CreatedById	Lookup(User)
Last Modified By	LastModifiedById	Lookup(User)
Owner	OwnerId	Lookup(User,Group)
Status	Status__c	Picklist
Test Drive Date	Test_Drive_Date__c	Date
Vehicle	Vehicle__c	Lookup(Vehicle)
Vehicle Customer	Vehicle_Customer__c	Lookup(Vehicle Customer)
Vehicle Test Drive Name	Name	Text(80)

Validation Rules

- **Out-of-Stock Order Blocker:**
  Prevents the creation of an order if the selected vehicle has zero stock.

Automation: Workflow Tools

o **Flows (Record-Triggered):**

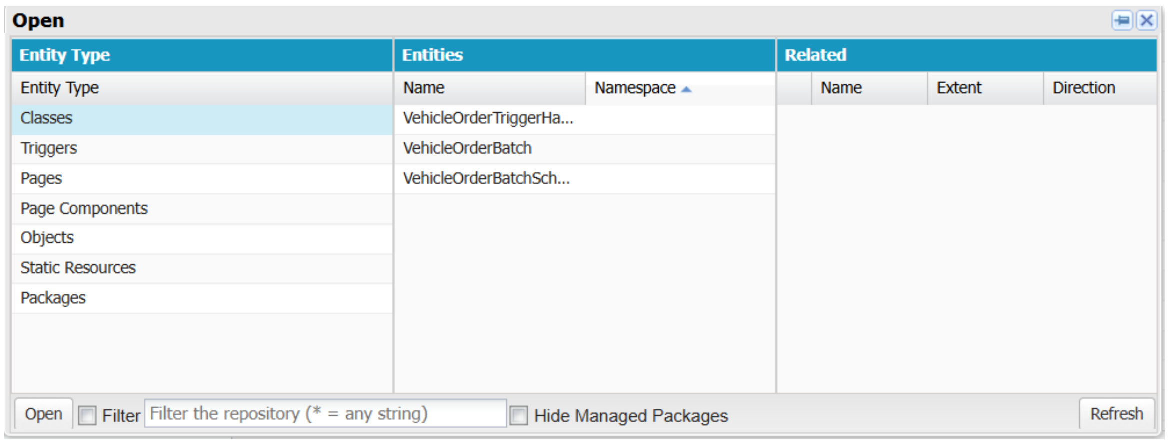
- Auto-assign the **nearest dealer** based on the customer's address using a **Record-Triggered Flow** on Order object.
- Send **test drive reminders** via **Scheduled Flows**.

Apex Classes and Triggers

Apex Classes:

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. All classes follow best practices using bulk-safe operations and reusable methods.



Open

Entity Type
Entity Type
Classes
Triggers
Pages
Page Components
Objects
Static Resources
Packages

Name	Namespace
VehicleOrderTriggerHa...	
VehicleOrderBatch	
VehicleOrderBatchSch...	

	Name	Extent	Direction

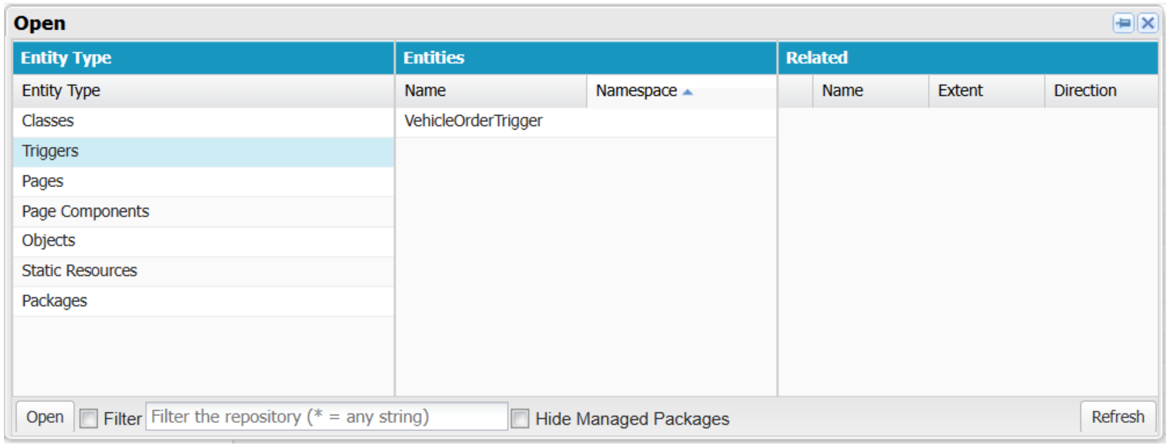
Open ☐ Filter Filter the repository (* = any string) ☐ Hide Managed Packages Refresh

Apex Trigger:

Apex Trigger was written on the **Order** object to perform:

- Stock availability validation
- Auto-dealer assignment (if not handled by Flow)
- Order status update logic (Pending or Confirmed)

Trigger follows best practices using a **Trigger Handler pattern**.



Entity Type	Entities		Related		
Entity Type	Name	Namespace	Name	Extent	Direction
Classes	VehicleOrderTrigger				
Triggers					
Pages					
Page Components					
Objects					
Static Resources					
Packages					

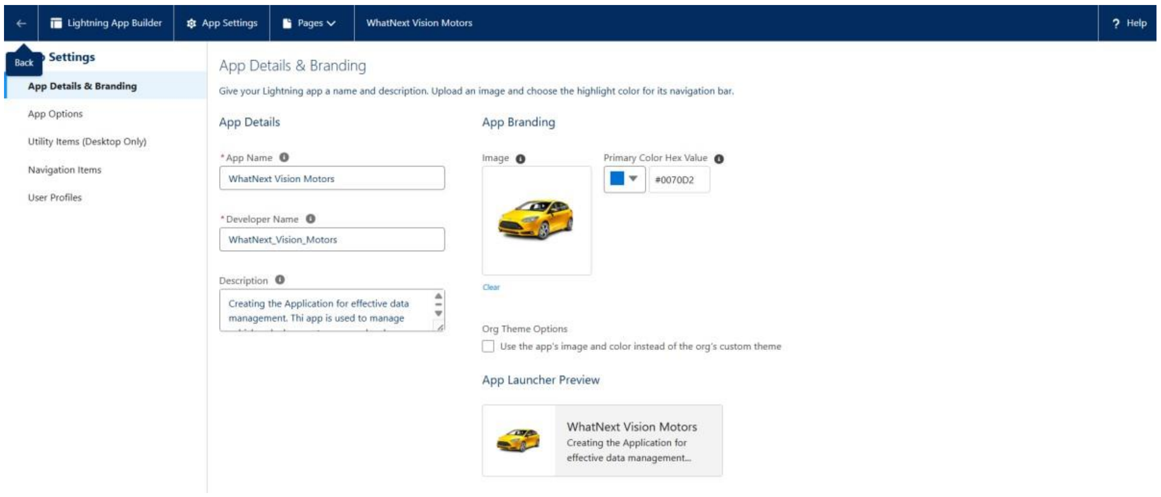
Filter the repository (* = any string)

Phase 3: UI/UX Development & Customization

Lightning App Setup via App Manager

A custom Lightning App named “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 created: *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



App Details & Branding

Give your Lightning app a name and description. Upload an image and choose the highlight color for its navigation bar.

App Details

*App Name: WhatNext Vision Motors

*Developer Name: WhatNext_Vision_Motors

Description: Creating the Application for effective data management. Thi app is used to manage

App Branding

Image

Primary Color Hex Value: #0070D2

Org Theme Options

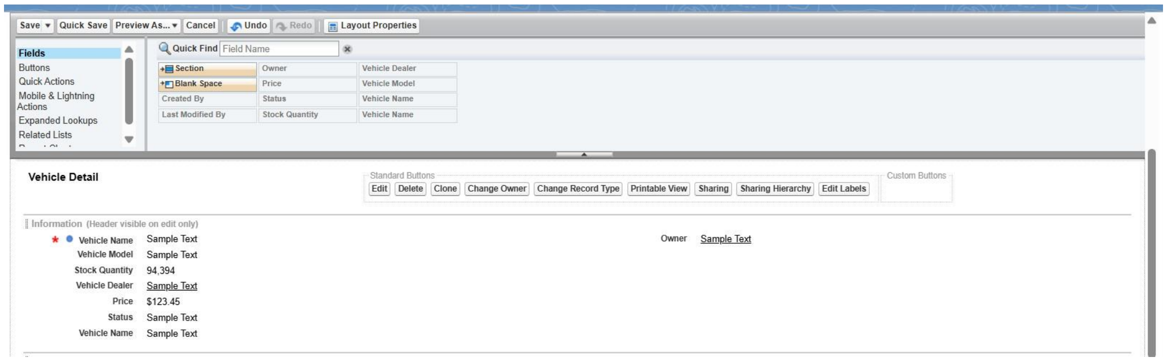
☐ Use the app's image and color instead of the org's custom theme

App Launcher Preview

WhatNext Vision Motors
Creating the Application for effective data management...

Page Layouts and Dynamic Forms:

Page layouts were customized for key objects such as Vehicle__c, Vehicle_Order__c, and Vehicle_Test_Drive__c to ensure clean UI and contextual field visibility. Dynamic Forms were used to place fields directly on the Lightning Record Page and conditionally show fields based on values like order status or vehicle availability.



Vehicle Detail

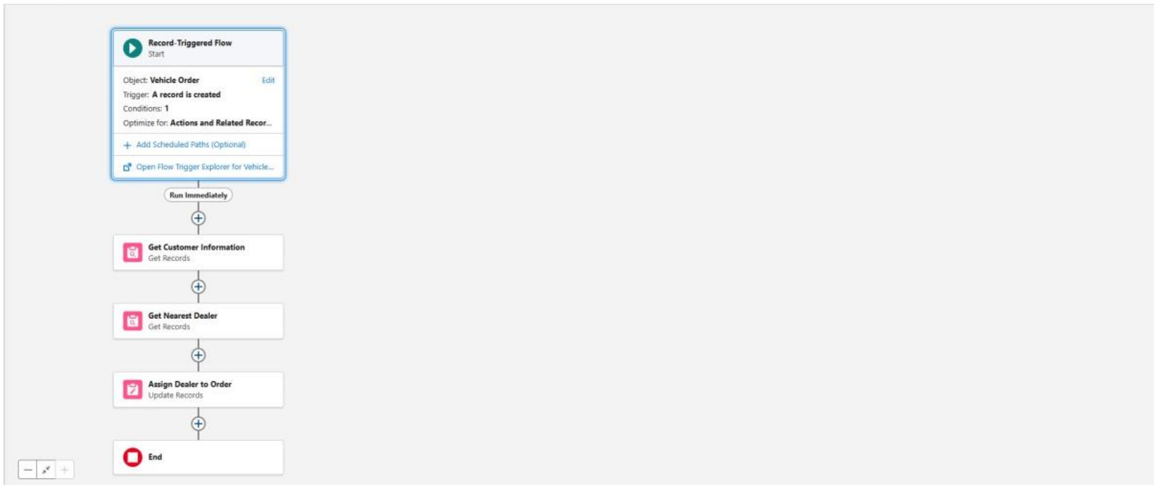
Information (Header visible on edit only)

Vehicle Name	Sample Text	Owner	Sample Text
Vehicle Model	Sample Text		
Stock Quantity	94,394		
Vehicle Dealer	Sample Text		
Price	$123.45		
Status	Sample Text		
Vehicle Name	Sample Text		

Flow 1: Auto Dealer Assignment

This flow runs on Vehicle_Order__c creation and:

- Fetches customer's address
- Finds a dealer in the same city
- Assigns that dealer to the order



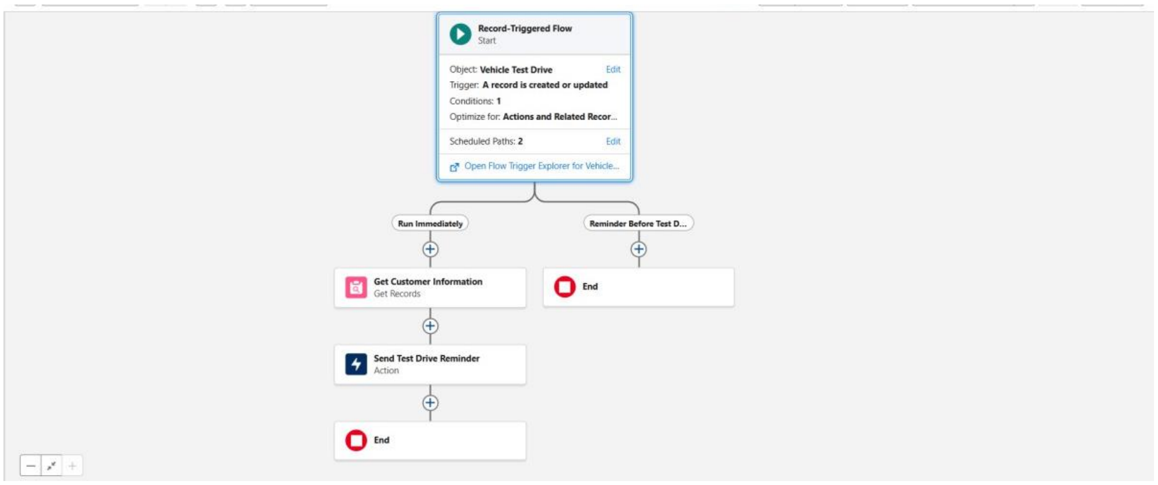
Object: Vehicle Order
Trigger: A record is created
Conditions: 1
Optimize for: Actions and Related Recor...

Flow 2: Test Drive Reminder

This Record-Triggered Flow:

- Runs on Vehicle_Test_Drive__c creation/update

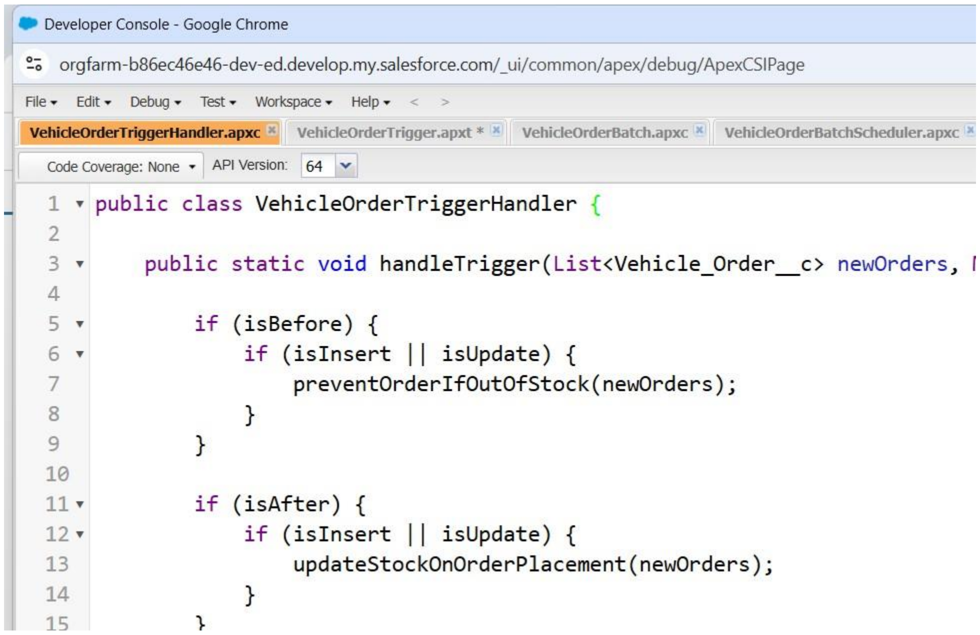
- Sends email 1 day before scheduled test drive



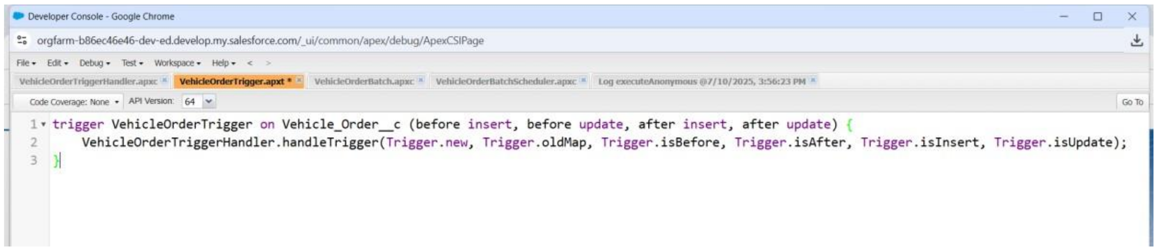
Object: Vehicle Test Drive
Trigger: A record is created or updated
Conditions: 1
Optimize for: Actions and Related Recor...
Scheduled Paths: 2

Apex Trigger & Handler

- Trigger: VehicleOrderTrigger
- Handler: VehicleOrderTriggerHandler
    - Prevents out-of-stock orders
    - Updates stock when order is confirmed


Developer Console - Google Chrome
orgfarm-b86ec46e46-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File
Edit
Debug
Test
Workspace
Help
VehicleOrderTriggerHandler.apxc
VehicleOrderTrigger.apxt *
VehicleOrderBatch.apxc
VehicleOrderBatchScheduler.apxc
Code Coverage: None
API Version:
64

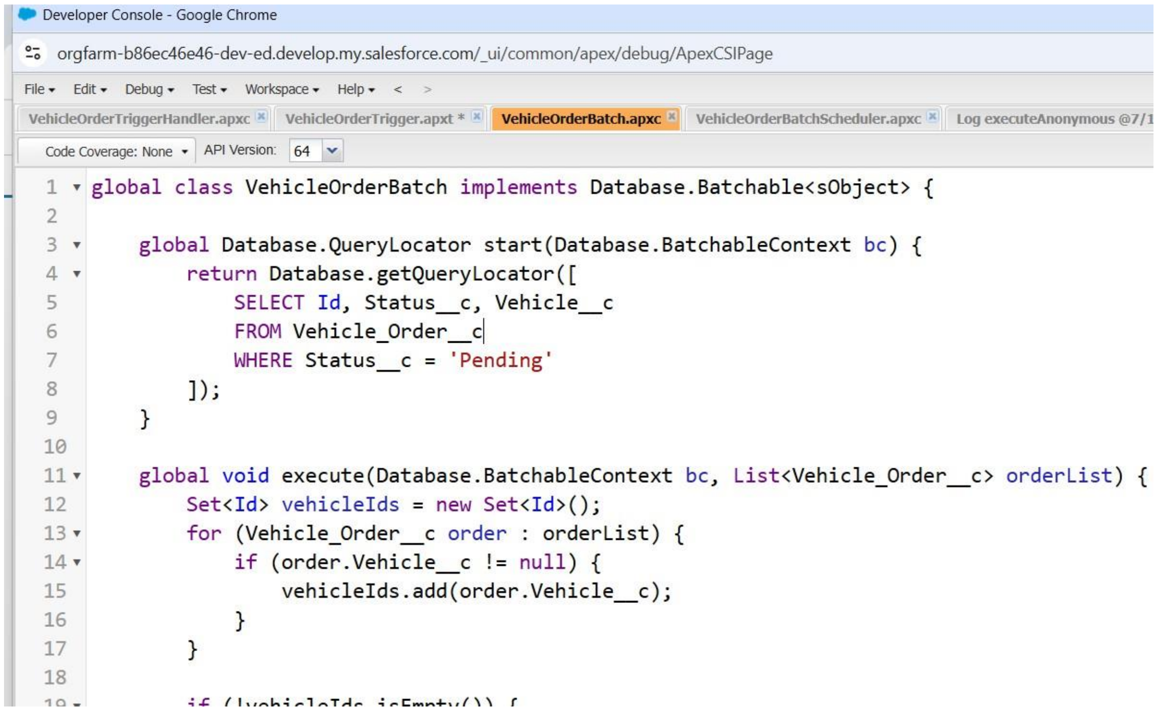
```
public class VehicleOrderTriggerHandler {

    public static void handleTrigger(List<Vehicle_Order__c> newOrders, 

        if (isBefore) {
            if (isInsert || isUpdate) {
                preventOrderIfOutOfStock(newOrders);
            }
        }

        if (isAfter) {
            if (isInsert || isUpdate) {
                updateStockOnOrderPlacement(newOrders);
            }
        }
```


Developer Console - Google Chrome
orgfarm-b86ec46e46-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File Edit Debug Test Workspace Help
VehicleOrderTriggerHandler.apxc
VehicleOrderTrigger.apxt *
VehicleOrderBatch.apxc
VehicleOrderBatchScheduler.apxc
Log executeAnonymous @7/10/2025, 3:56:23 PM
Code Coverage: None
API Version: 64
Go To

```
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

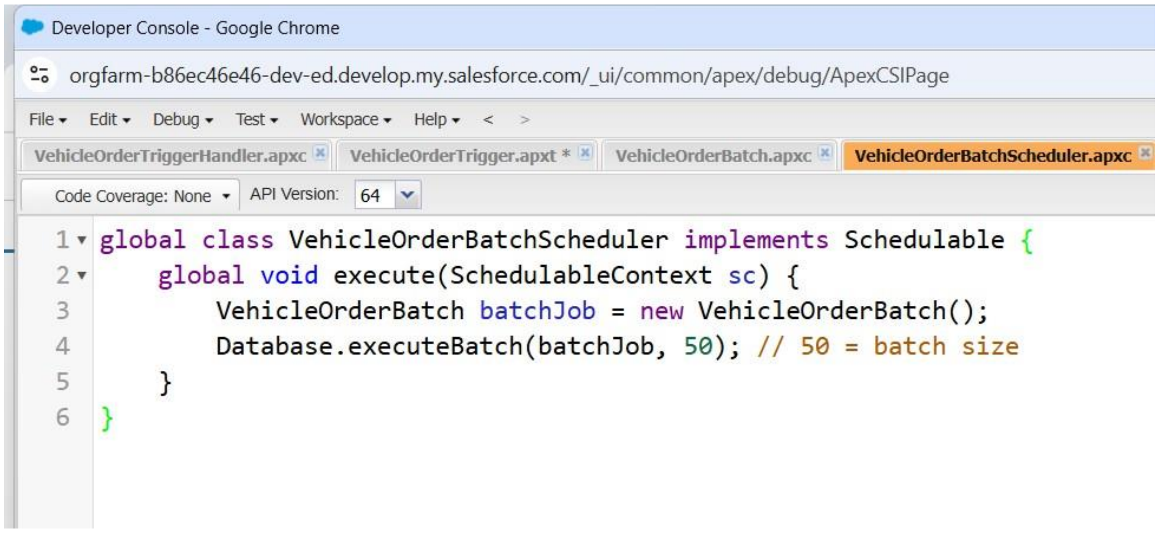
- Class: VehicleOrderBatch
- Runs daily
- Checks for pending orders and available stock
- Updates status to *Confirmed* and adjusts stock


Developer Console - Google Chrome
orgfarm-b86ec46e46-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File Edit Debug Test Workspace Help
VehicleOrderTriggerHandler.apxc
VehicleOrderTrigger.apxt *
VehicleOrderBatch.apxc
VehicleOrderBatchScheduler.apxc
Log executeAnonymous @7/1
Code Coverage: None
API Version: 64

```
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()) {
```

Scheduled Apex

- Class: VehicleOrderBatchScheduler
- Cron job runs daily at 12 PM
- Executes batch class automatically


Developer Console - Google Chrome
orgfarm-b86ec46e46-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File Edit Debug Test Workspace Help < >
VehicleOrderTriggerHandler.apxc
VehicleOrderTrigger.apxt *
VehicleOrderBatch.apxc
VehicleOrderBatchScheduler.apxc
Code Coverage: None
API Version: 64

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

Phase 4: Data Migration, Testing & Security

Data Loading Process

To load initial data into Salesforce (such as vehicles, dealers, and customers), the following tools were used:

Tools Used:

- **Data Import Wizard**:
  Used for importing standard object data (like Accounts, Contacts).
- **Data Loader**:
  Used for large volumes and for custom objects like Vehicle__c, Dealer__c, Order__c.

Steps:

1. Exported CSV files with sample records.
2. Mapped columns to corresponding Salesforce fields.
3. Used Data Loader to insert records for:
   - Vehicle__c
   - Dealer__c
   - Customer__c
   - Order__c (with valid relationships)

Field History Tracking, Duplicate Rules, and Matching Rules

Field History Tracking:

Enabled for the following objects to track changes:

- **Vehicle__c:** Stock__c field
- **Order__c:** Status__c and Dealer__c fields

Duplicate & Matching Rules:

- **Matching Rule:** Custom rule defined on Customer_c based on Email_c and Phone_c
- **Duplicate Rule:** Prevents duplicate customers from being inserted

Profiles, Roles, Permission Sets, and Sharing Rules

Profiles and Roles:

- Standard profiles like **Standard User** and **System Administrator** were used.
- **Role Hierarchy** established:
  - CEO
    └── Sales Manager
    └── Sales Rep

Permission Sets:

- Created **Order Management Access** permission set
- Assigned to users who need create/read access to Orders and Vehicles

Sharing Rules:

- **Public Read/Write** for most custom objects
- **Manual Sharing** allowed for sensitive customer records

Preparation of test cases for each and every salesforce features like booking creation, Approval Process, Automatic Task creation, flows, triggers etc.

1. Create a Vehicle :

INPUT:

Vehicle Name: Test Car

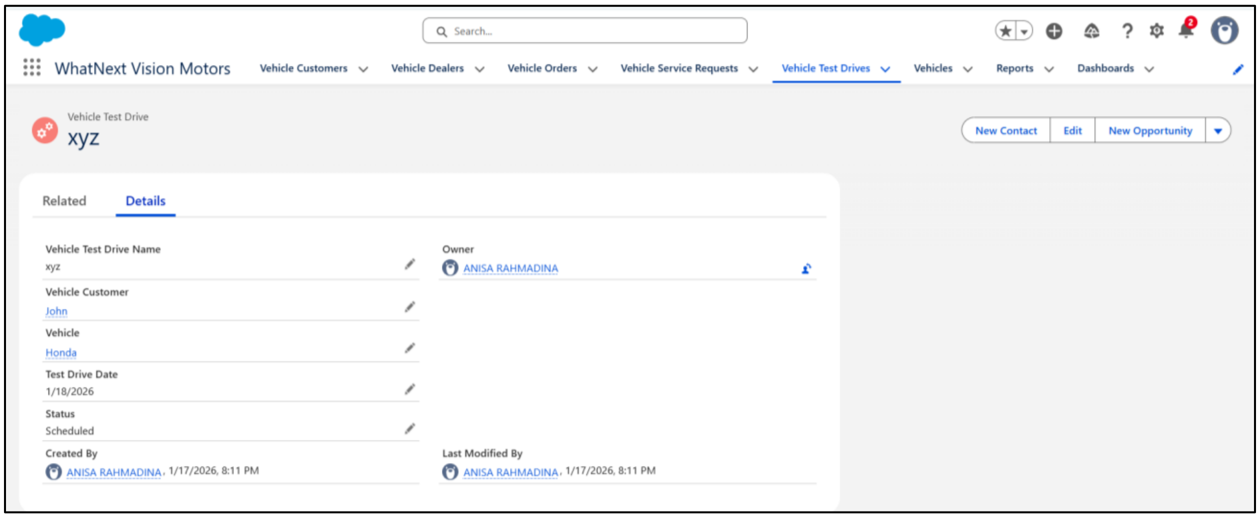
Vehicle Model: Sedan

Stock Quantity: 1

Price: 1020000

Status: Available

Dealer: Select existing Vehicle Dealer



Vehicle Test Drive
xyz

Vehicle Test Drive Name	xyz
Vehicle Customer	John
Vehicle	Honda
Test Drive Date	1/18/2026
Status	Scheduled
Created By	ANISA RAHMADINA, 1/17/2026, 8:11 PM
Owner	ANISA RAHMADINA
Last Modified By	ANISA RAHMADINA, 1/17/2026, 8:11 PM

2. Test Stock = 0 (Error Case):

INPUT:

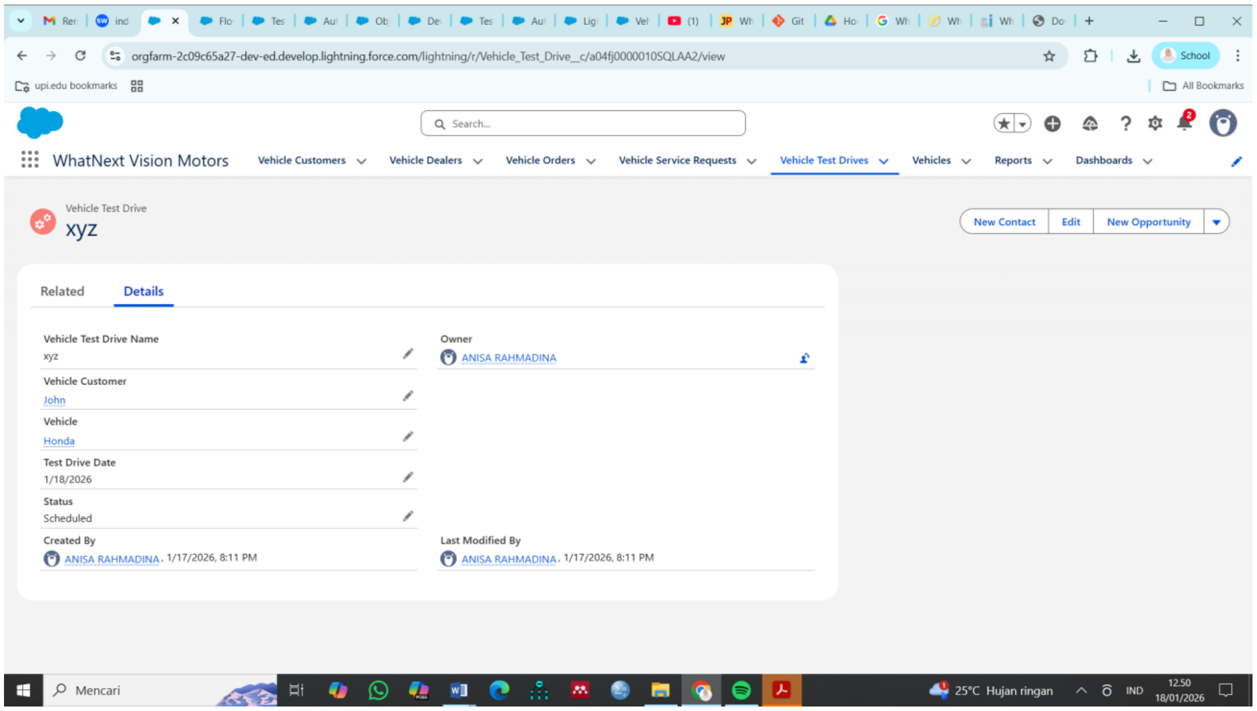
Edit the Stock Quantity of the above vehicle → Set it to 0.

Go to Vehicle Orders tab → Click New.

● **Vehicle:** Test Car

● **Status:** Confirmed

● **Customer:** Select any existing customer OUTPUT


Vehicle Test Drive
xyz

Vehicle Test Drive Name	xyz
Vehicle Customer	John
Vehicle	Honda
Test Drive Date	1/18/2026
Status	Scheduled
Created By	ANISA RAHMADINA, 1/17/2026, 8:11 PM
Owner	ANISA RAHMADINA
Last Modified By	ANISA RAHMADINA, 1/17/2026, 8:11 PM

3. Test Stock > 0 (Confirmed Order)

INPUT:

Steps:

1. Set vehicle Stock Quantity back to 1.

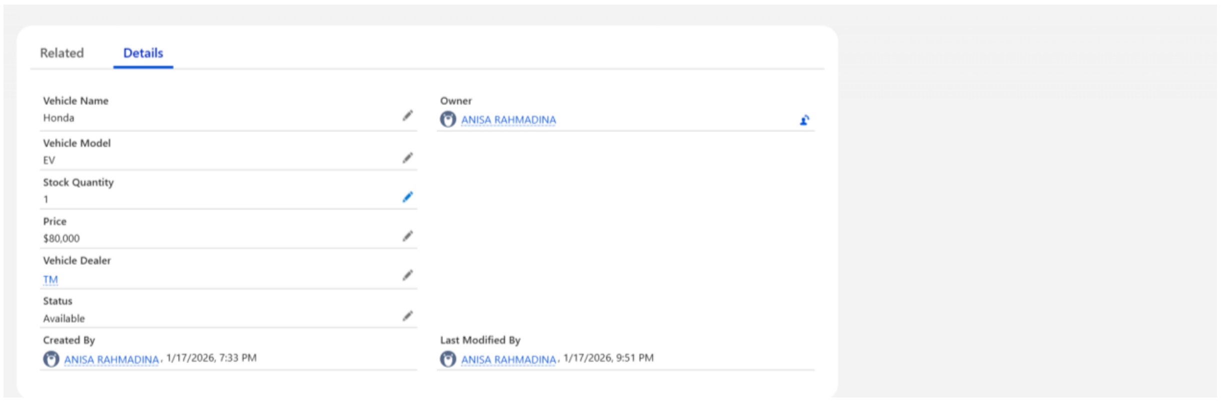
2. Create a Vehicle Order:

○ Status: Confirmed

○ Vehicle: Test Car

○ Vehicle stock should reduce from 2 → 1 automatically.

OUTPUT:



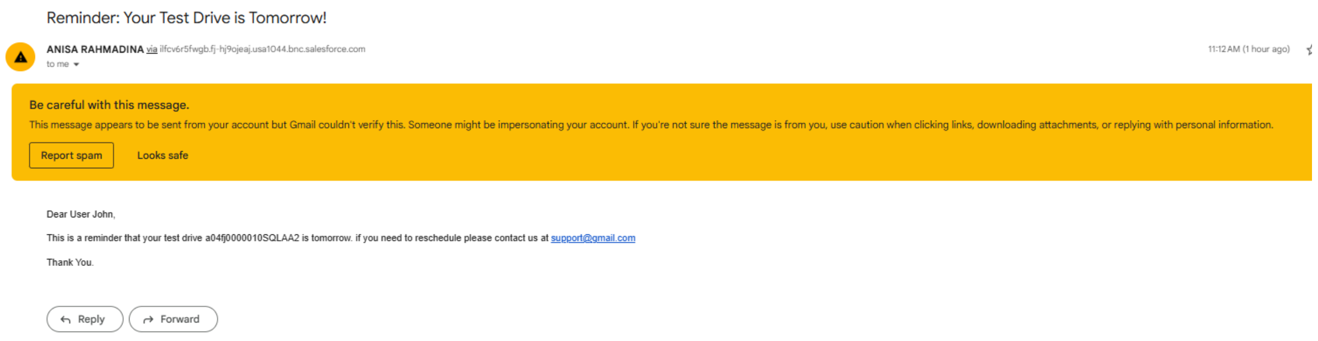
Vehicle Name	Honda
Vehicle Model	EV
Stock Quantity	1
Price	$80,000
Vehicle Dealer	TM
Status	Available
Created By	ANISA RAHMADINA, 1/17/2026, 7:33 PM
Owner	ANISA RAHMADINA
Last Modified By	ANISA RAHMADINA, 1/17/2026, 9:51 PM

4. Test Drive Reminder Email:

Customer: Select any customer with email

Status: Scheduled Test Drive Date: Tomorrow (pick tomorrow's date)

OUTPUT



Reminder: Your Test Drive is Tomorrow!

ANISA RAHMADINA via ilfcv6r5fwgb.fj-hj9ojeaj.usa1044.bnc.salesforce.com
to me

11:12 AM (1 hour ago)

Be careful with this message.
This message appears to be sent from your account but Gmail couldn't verify this. Someone might be impersonating your account. If you're not sure the message is from you, use caution when clicking links, downloading attachments, or replying with personal information.

Dear User John,

This is a reminder that your test drive a04fj0000010SQLAA2 is tomorrow. if you need to reschedule please contact us at support@gmail.com

Thank You.

Test Batch Job for Pending Orders :

INPUT:

Create a Pending Order when stock is 0:

1. Set Test Car stock to 0.

2. Create a Vehicle Order:

○ Status: Pending

Update stock:

● Set Stock Quantity = 1

Run batch manually:

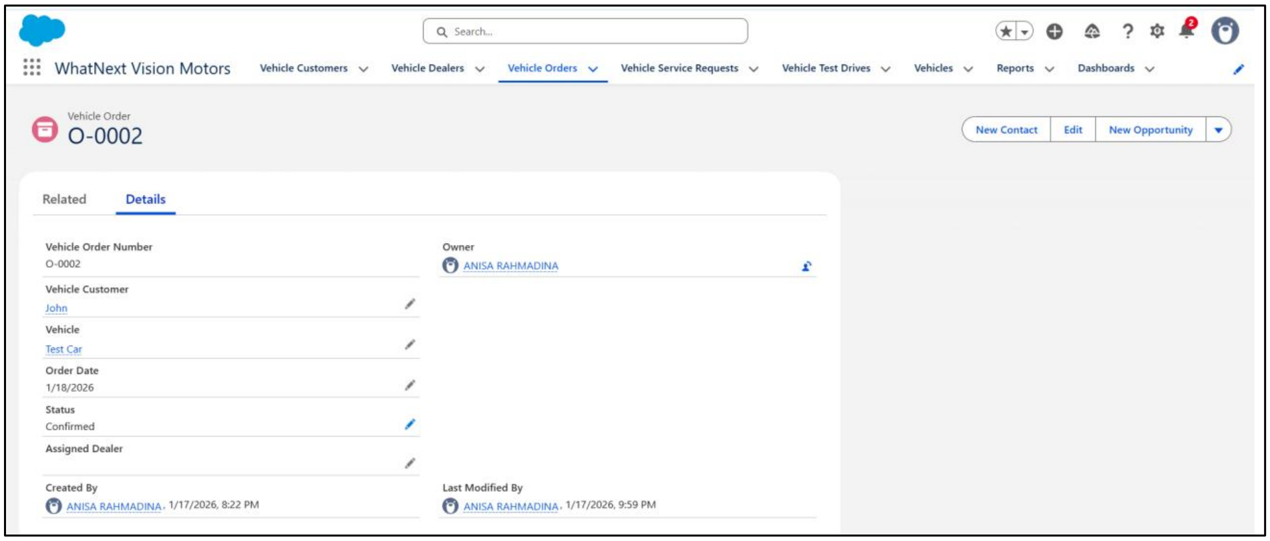
```
VehicleOrderBatch job = new VehicleOrderBatch();
Database.executeBatch(job, 50);
```

OUTPUT:

Expected Result:

● Your Pending Order should become Confirmed.

● Vehicle stock should reduce by 1.



Vehicle Order
O-0002

Vehicle Order Number
O-0002

Owner
ANISA RAHMADINA

Vehicle Customer
John

Vehicle
Test Car

Order Date
1/18/2026

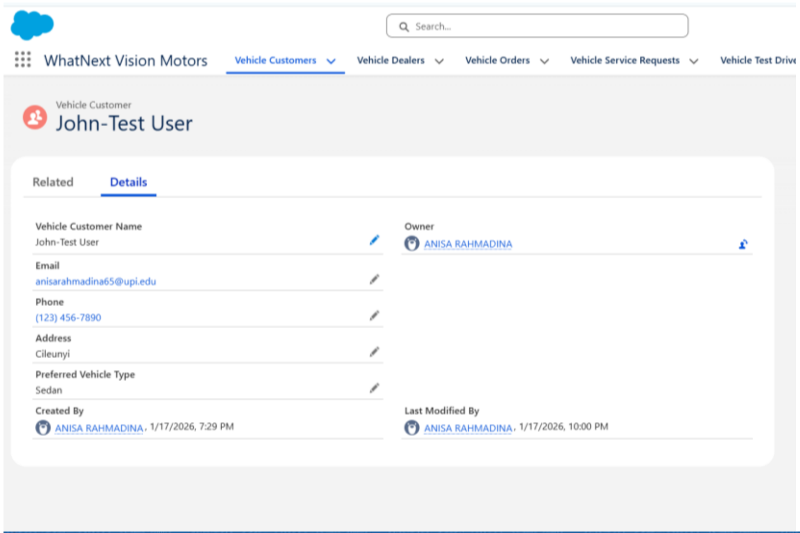
Status
Confirmed

Assigned Dealer

Created By
ANISA RAHMADINA, 1/17/2026, 8:22 PM

Last Modified By
ANISA RAHMADINA, 1/17/2026, 9:59 PM

Creation of Test Cases



Vehicle Customer
John-Test User

Vehicle Customer Name
John-Test User

Owner
ANISA RAHMADINA

Email
anisarahmadina65@upi.edu

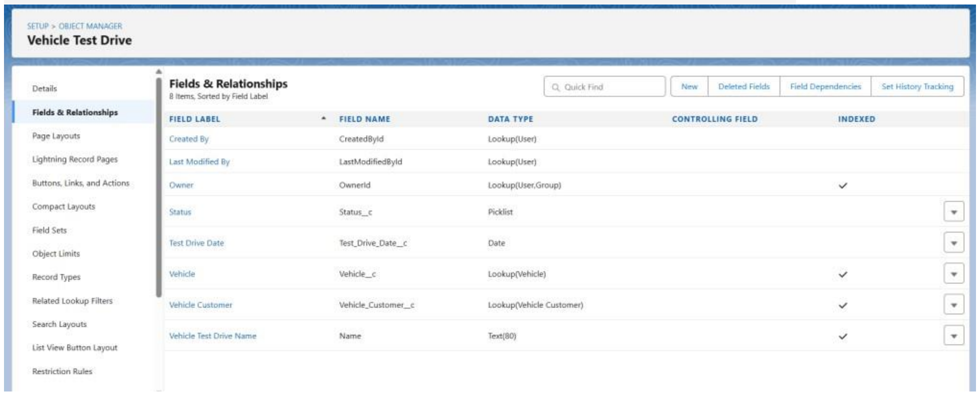
Phone
(123) 456-7890

Address
Cileunyi

Preferred Vehicle Type
Sedan

Created By
ANISA RAHMADINA, 1/17/2026, 7:29 PM

Last Modified By
ANISA RAHMADINA, 1/17/2026, 10:00 PM



Vehicle Test Drive

Fields & Relationships
8 Items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓
Status	Status__c	Picklist		
Test Drive Date	Test_Drive_Date__c	Date		
Vehicle	Vehicle__c	Lookup(Vehicle)		✓
Vehicle Customer	Vehicle_Customer__c	Lookup(Vehicle Customer)		✓
Vehicle Test Drive Name	Name	Text(80)		✓

To ensure Apex code is deployable and functional, **Test Classes** were created for:

- OrderTriggerHandler
- DealerAssignmentService
- StockValidationTrigger

Test Class Features:

- Minimum 75% coverage
- Positive and negative test cases
- Used @isTest annotation with test data setup

Phase 5: Deployment, Documentation & Maintenance

Deployment Strategy

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

Deployment Steps:

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

Testing & Sample Scenarios

Test Cases:

- Create vehicle and order with 0 stock → error
- Set stock = 2 → place order → stock becomes 1
- Create pending order → update stock → batch job confirms order

System Maintenance and Monitoring

To ensure smooth system performance after deployment, the following basic maintenance strategy was defined:

1. Monitoring

- Use **Apex Jobs** to monitor scheduled jobs or batch classes.
- Use **Debug Logs** to trace errors or unexpected behavior.
- Enable **Email Alerts** for test drive reminders or failed processes.

2. User Feedback Loop

- Sales and operations team were asked to use the system for a few days post-deployment.
- Collected feedback via manual walkthroughs to identify any missing features or issues.

3. Updates and Fixes

- Minor updates (like adding help text or updating field labels) were handled in sandbox and redeployed via Change Sets.
- Scheduled quarterly reviews for enhancements or UI improvements.

Troubleshooting Approach

If any issues arise in the production environment, the following steps will be followed:

Step 1: Reproduce the Issue

- Try to replicate the problem in a sandbox or developer org.

Step 2: Enable Debug Logs

- Set debug logs for the impacted user and analyze the flow or Apex execution.

Step 3: Check Apex Jobs or Flows

- If it's related to background processing, check Apex Job failures or Flow error emails.

Step 4: Fix and Retest

- Modify the logic (Flow or Apex).
- Retest in sandbox and re-deploy using Change Set.

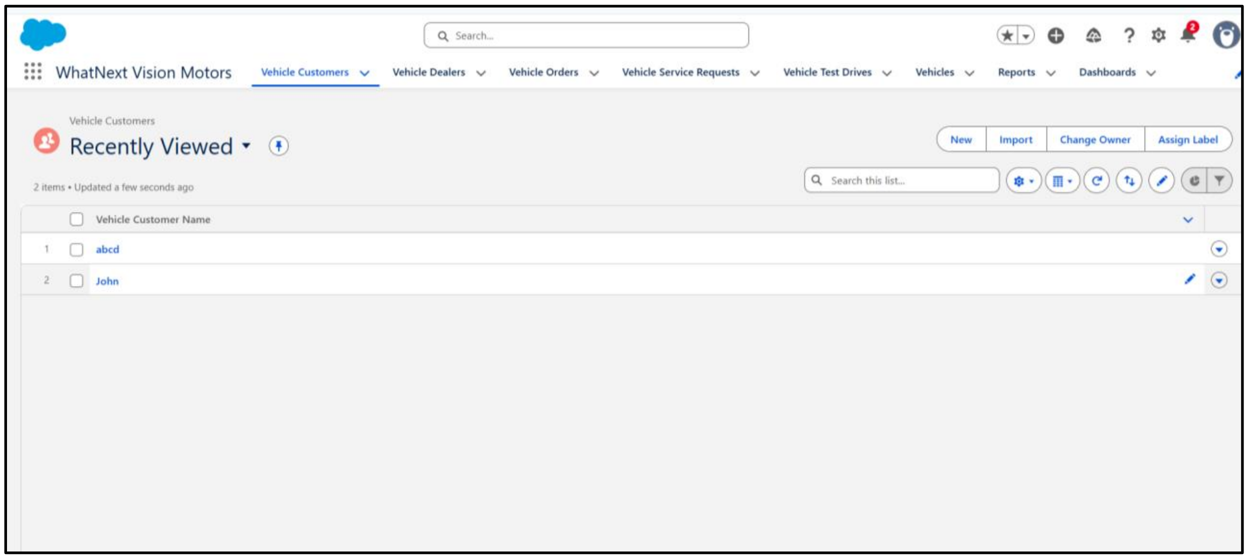
Conclusion

The Salesforce implementation at **WhatsNext Vision Motors** successfully achieved its objective of streamlining the customer ordering process and improving operational workflows. Key achievements include:

- Automated **nearest dealer assignment** using Flows or Triggers
- **Stock validation** to prevent out-of-stock orders
- **Scheduled logic** to update order statuses (if Batch Apex was implemented)
- Enhanced **customer experience** through automation
- Reduced **manual intervention** for internal teams

This project not only enhances the company's customer-facing processes but also establishes a strong foundation for future Salesforce expansion and automation. Through this initiative, WhatsNext Vision Motors has moved a step closer toward its vision of **innovation and excellence in mobility.**

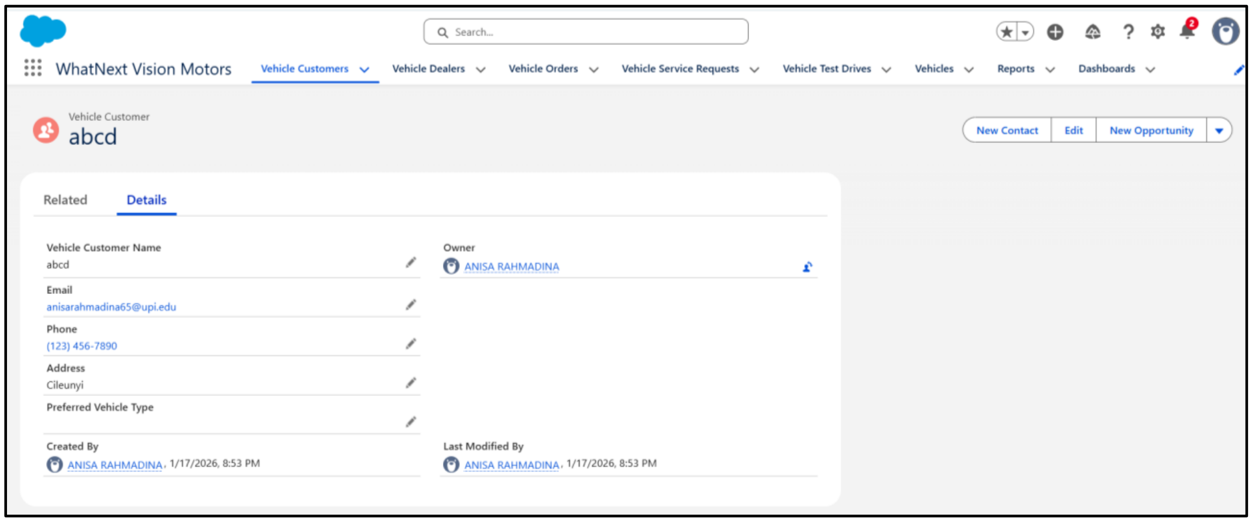
Screenshots



2 items • Updated a few seconds ago

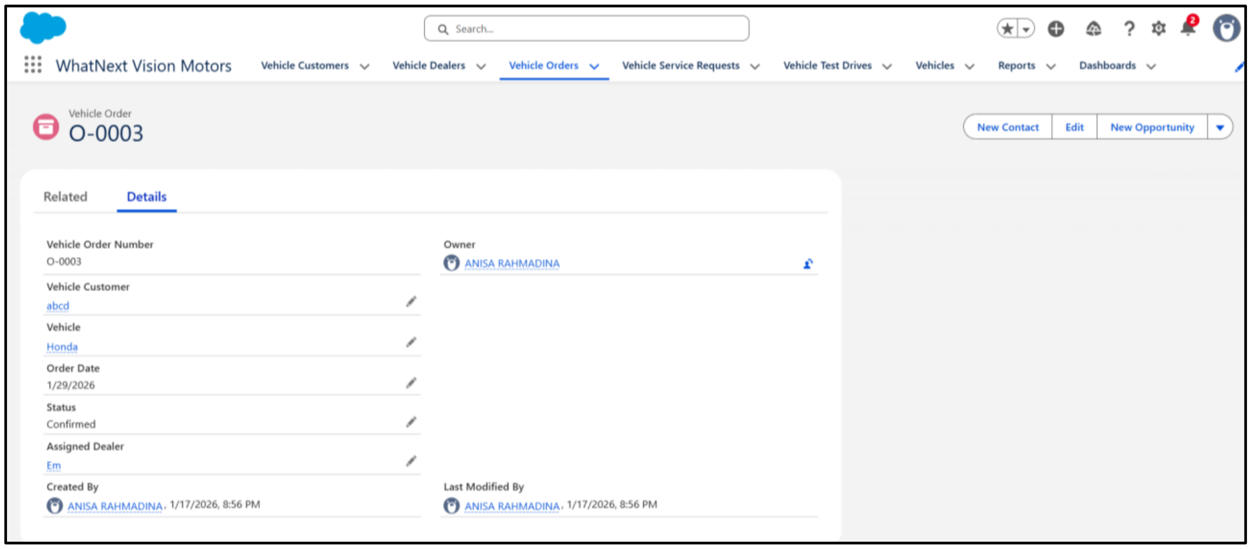
	Vehicle Customer Name
1	abcd
2	John

Fig: Custom App for WhatNext Vision Motors



Vehicle Customer Name abcd	Owner ANISA RAHMADINA
Email anisarahmadina65@upi.edu	
Phone (123) 456-7890	
Address Cileunyi	
Preferred Vehicle Type	
Created By ANISA RAHMADINA, 1/17/2026, 8:53 PM	Last Modified By ANISA RAHMADINA, 1/17/2026, 8:53 PM

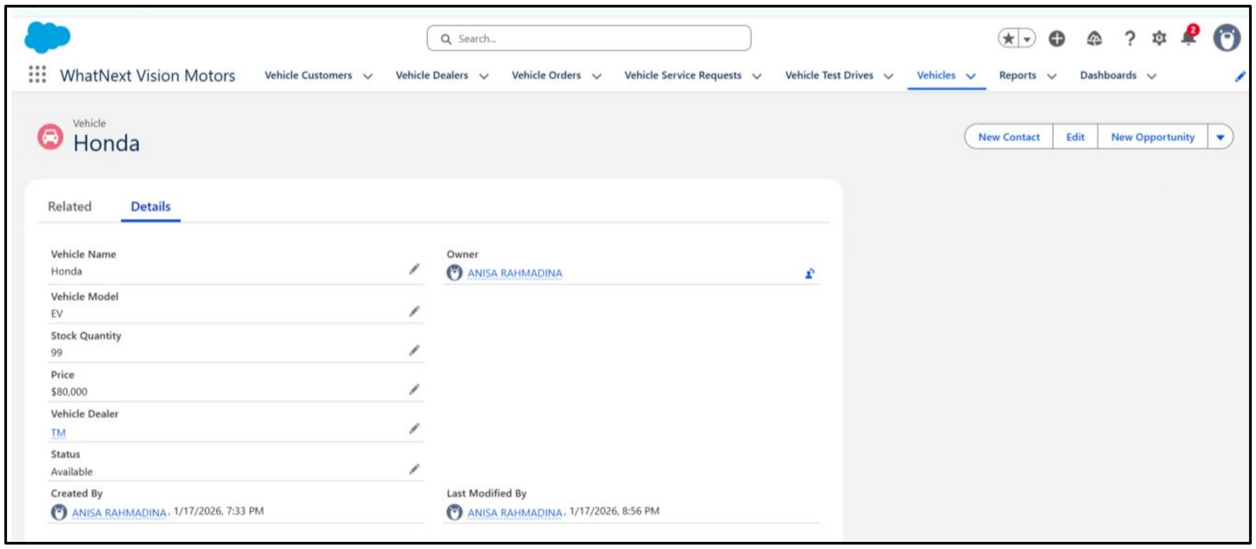
Fig: Customer Creation in WhatNext Vision Motors



Vehicle Order
O-0003

Vehicle Order Number	O-0003
Owner	ANISA RAHMADINA
Vehicle Customer	abcd
Vehicle	Honda
Order Date	1/29/2026
Status	Confirmed
Assigned Dealer	Em
Created By	ANISA RAHMADINA, 1/17/2026, 8:56 PM
Last Modified By	ANISA RAHMADINA, 1/17/2026, 8:56 PM

Fig: Vehicle Dealer in WhatNext Vision Motors



Vehicle
Honda

Vehicle Name	Honda
Owner	ANISA RAHMADINA
Vehicle Model	EV
Stock Quantity	99
Price	$80,000
Vehicle Dealer	TM
Status	Available
Created By	ANISA RAHMADINA, 1/17/2026, 7:33 PM
Last Modified By	ANISA RAHMADINA, 1/17/2026, 8:56 PM

Fig: The Stock Vehicle in WhatNext Vision Motors