Salesforce CRM Project Documentation

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

Project Overview

WhatNext Vision Motors, a growing company in the automotive industry that sells different types of vehicles, aimed to improve how it manages customers, vehicles, and dealer operations. This CRM project was conducted because the company's old manual processes caused delays, incorrect stock information, and customer dissatisfaction.

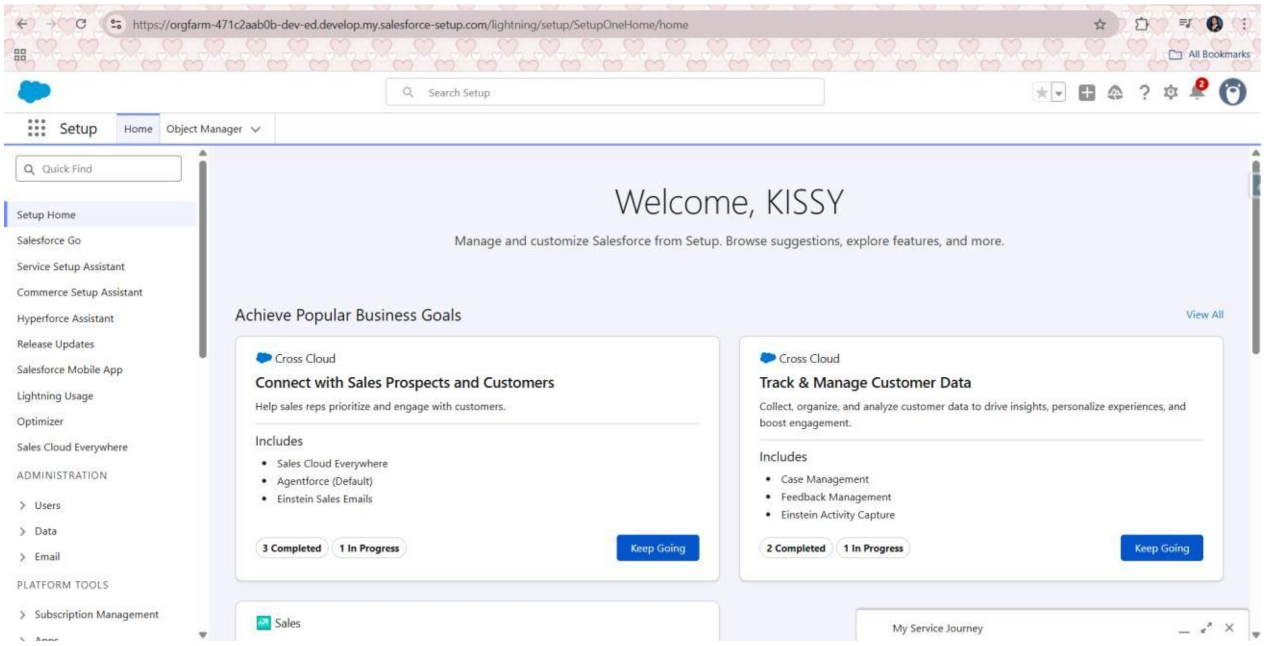
To address these issues, a customized Salesforce CRM system was created. This system automates vehicle order processing, ensures accurate stock management, assigns the nearest dealer, and automatically sends test-drive reminders.

The system also uses Lightning Apps and Dynamic Forms to provide a clean and user-friendly interface. Overall, the CRM improves operational efficiency, reduces errors, and prepares the company for future upgrades such as AI-based vehicle recommendations or an AI chatbot.

Objectives

The key objectives of the Salesforce CRM Project for WhatNext Vision Motors: Shaping the Future of Mobility with Innovation and Excellence are the following:

1. **Automate Order and Dealer Assignment**
   - Automatically assign the closest dealer based on the customer's city.
2. **Prevent Out-of-Stock Orders**
   - Use validation rules and Apex triggers to stop customers from ordering vehicles with zero stock.
3. **Send Test Drive Reminders**
   - Automatically email customers one day before their scheduled test drive.
4. **Improve User Experience**
   - Use Lightning Apps and Dynamic Forms for a simple and responsive (UI) User Interface.
5. **Maintain a Scalable Backend**
   - Use Apex classes and batch jobs to handle stock updates and order confirmations in bulk.



Welcome, KISSY

Manage and customize Salesforce from Setup. Browse suggestions, explore features, and more.

Achieve Popular Business Goals

Cross Cloud
Connect with Sales Prospects and Customers
Help sales reps prioritize and engage with customers.
Includes
- Sales Cloud Everywhere
- Agentforce (Default)
- Einstein Sales Emails

3 Completed | 1 In Progress

Cross Cloud
Track & Manage Customer Data
Collect, organize, and analyze customer data to drive insights, personalize experiences, and boost engagement.
Includes
- Case Management
- Feedback Management
- Einstein Activity Capture

2 Completed | 1 In Progress

Phase 1: Requirement Analysis & Planning

The first phase of the project focused on understanding the business needs or requirements of WhatNext Vision Motors and converting them into system requirements for Salesforce. The main goal was to create a CRM that supports the entire vehicle management process from tracking inventory, to handling customer orders, and managing post-sales interactions.

Understanding Business Requirements:

The CRM Project system must:

- Store all vehicle, dealer, and customer data in one place.
- Check stock availability during order creation.
- Assign the nearest dealer automatically.
- Track test drives and service requests.
- Automate processes to reduce manual work.

Defining Project Scope and Objectives:

To meet these needs, the system includes:

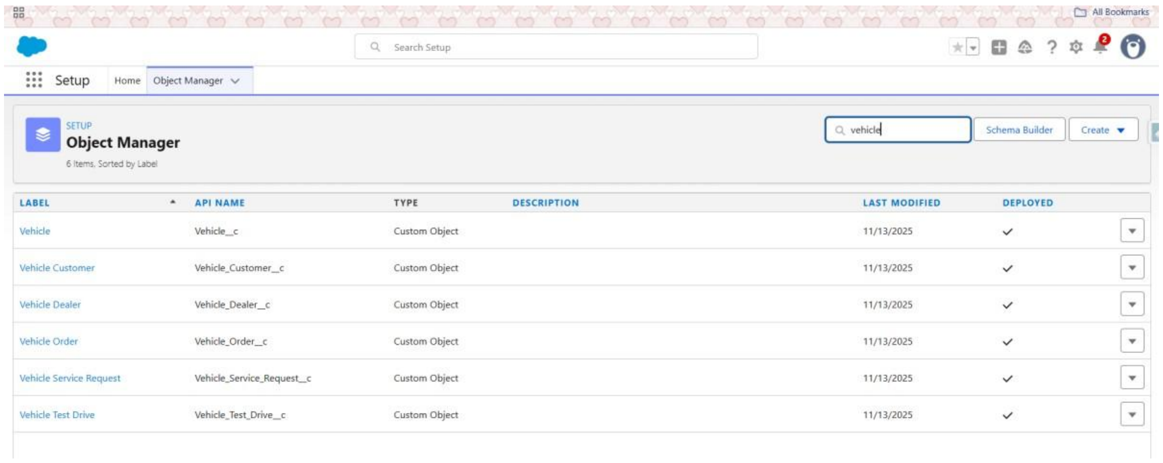
- Custom objects for vehicles, orders, customers, dealers, test drives, and service requests.
- Record-triggered flows for dealer assignment and email reminders.
- Apex triggers to validate stock and update inventory.
- Batch Apex for processing pending orders.

Data Model

Six custom objects were created to represent the business structure of the Capstone Project.

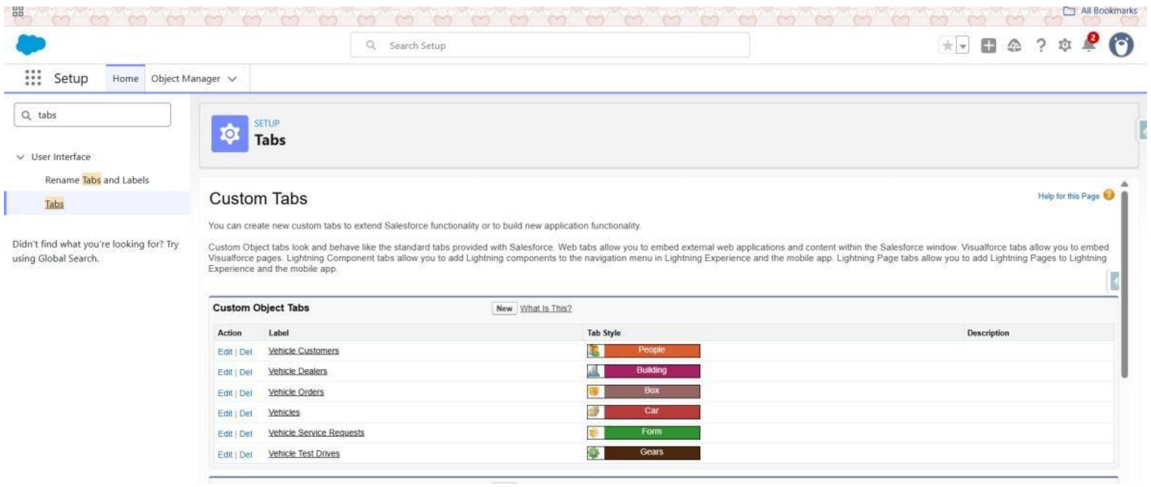
Object Name	**Purpose**
Vehicle__c	Stores vehicle details and stock informations
Vehicle_Dealer__c	Stores Dealer information
Vehicle_Customer__c	Stores customer data or informations
Vehicle_Order__c	Tracks vehicle orders
Vehicle_Test_Drive__c	Schedules and tracks test drives
Vehicle_Service_Request__c	Manages service history and issues

These objects are connected through lookup relationships to maintain accurate and consistent data.



6 Items, Sorted by Label

LABEL	API NAME	TYPE	DESCRIPTION	LAST MODIFIED	DEPLOYED
Vehicle	Vehicle__c	Custom Object		11/13/2025	✓
Vehicle Customer	Vehicle_Customer__c	Custom Object		11/13/2025	✓
Vehicle Dealer	Vehicle_Dealer__c	Custom Object		11/13/2025	✓
Vehicle Order	Vehicle_Order__c	Custom Object		11/13/2025	✓
Vehicle Service Request	Vehicle_Service_Request__c	Custom Object		11/13/2025	✓
Vehicle Test Drive	Vehicle_Test_Drive__c	Custom Object		11/13/2025	✓



Didn't find what you're looking for? Try using Global Search.

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	Box	
Edit | Del	Vehicles	Car	
Edit | Del	Vehicle Service Requests	Form	
Edit | Del	Vehicle Test Drives	Gears	

Security Model

- **Standard Salesforce profiles** were utilized, and **Permission Sets** were added to give users access to the custom objects.
- **Field-Level Security and the Role Hierarchy** were applied to make sure users can only view or edit information related to their roles.
- **Field History Tracking was turned on** for important fields, such as **Stock_Quantity__c (Vehicle) and Status__c (Order),** to support auditing and monitoring.

Phase 2: Salesforce Development – Backend & Configurations

Setup Environment & DevOps Workflow

A Salesforce Developer Org was prepared at the start of the project to develop and test all custom features and automation.

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

Customization of Objects, Fields, Validation Rules, and Automation

Custom Objects and Fields

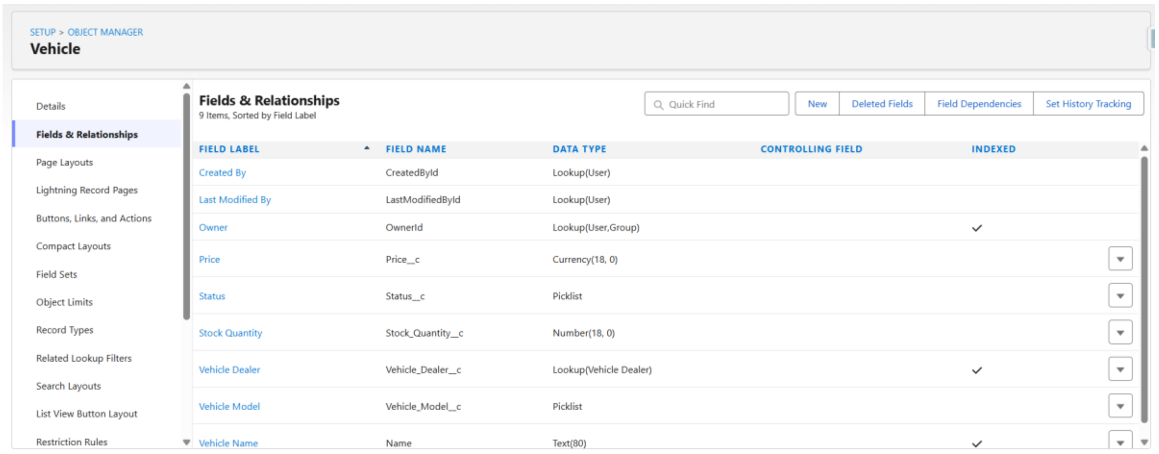
The following custom objects were created and configured to support the What Next Vision Motors business processes:

- **Vehicle** – Stores information such as vehicle name, model, and stock quantity.
- **Dealer** – Stores dealer details, including location and available vehicles.
- **Customer** – Stores customer information and address.
- **Order** – Records vehicle orders and their status.

Relationships:

- Order → Vehicle: Lookup

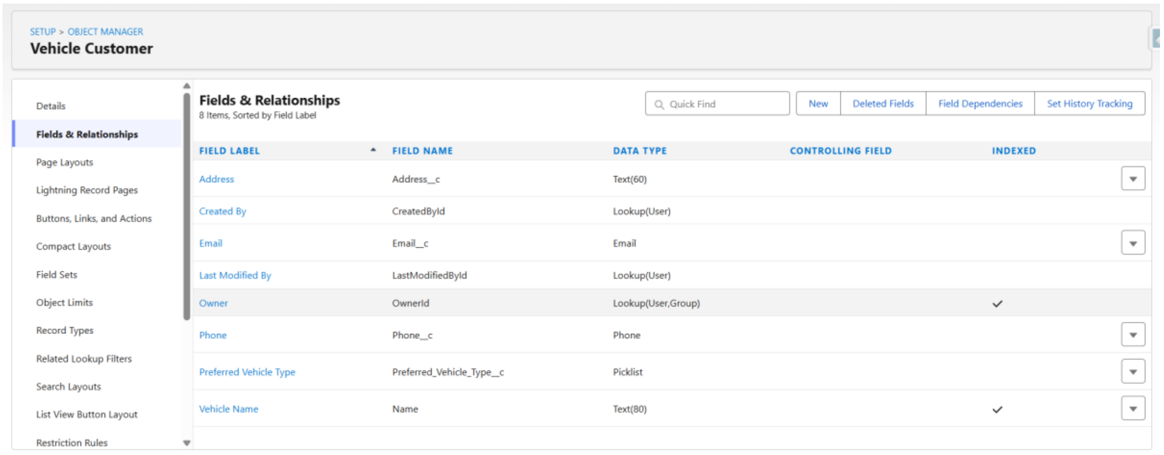
- Order → Dealer: Lookup
- Order → Customer: Lookup or Master-Detail (depending on the implementation)



Vehicle

Fields & Relationships
9 Items, Sorted by Field Label

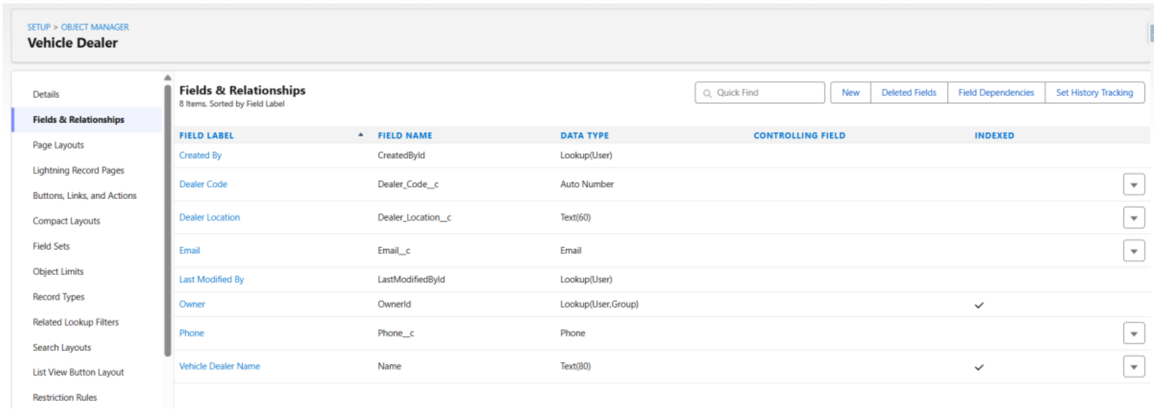
FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓
Price	Price__c	Currency(18, 0)		
Status	Status__c	Picklist		
Stock Quantity	Stock_Quantity__c	Number(18, 0)		
Vehicle Dealer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



Vehicle Customer

Fields & Relationships
8 Items, Sorted by Field Label

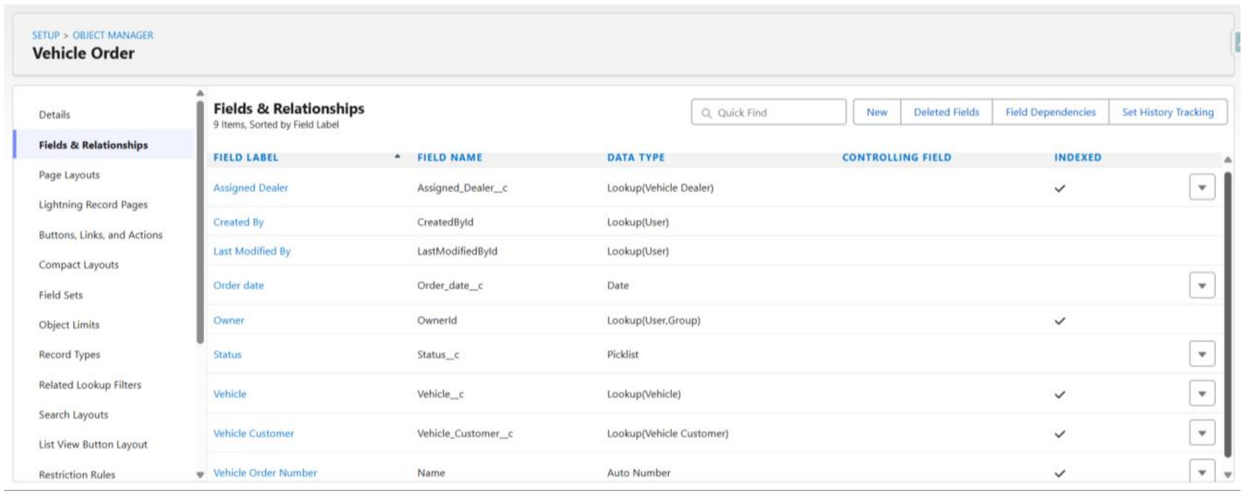
FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Address	Address__c	Text(60)		
Created By	CreatedById	Lookup(User)		
Email	Email__c	Email		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓
Phone	Phone__c	Phone		
Preferred Vehicle Type	Preferred_Vehicle_Type__c	Picklist		
Vehicle Name	Name	Text(80)		✓



Vehicle Dealer

Fields & Relationships
8 Items, Sorted by Field Label

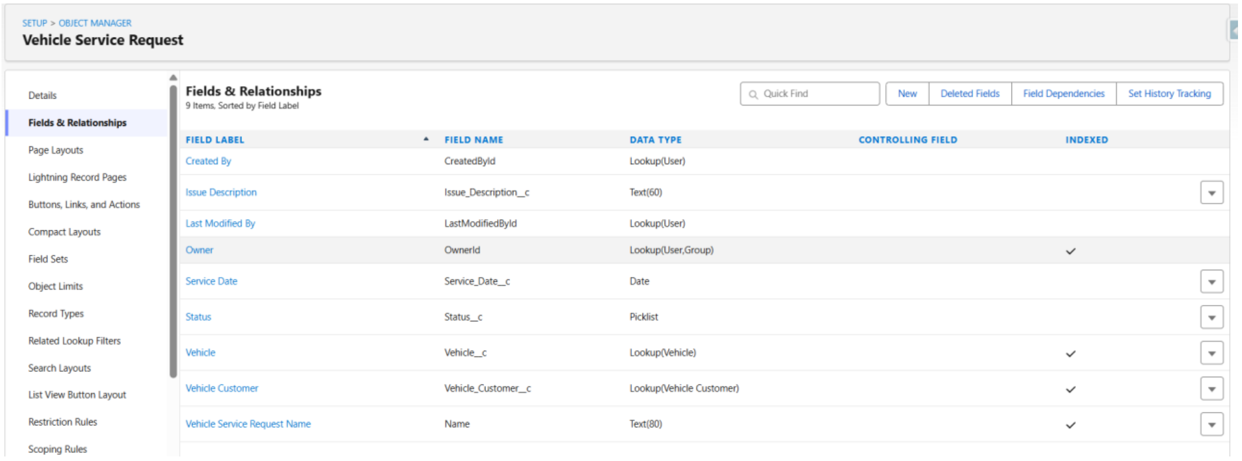
FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
Dealer Code	Dealer_Code__c	Auto Number		
Dealer Location	Dealer_Location__c	Text(60)		
Email	Email__c	Email		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓
Phone	Phone__c	Phone		
Vehicle Dealer Name	Name	Text(80)		✓



Vehicle Order

Fields & Relationships
9 Items, Sorted by Field Label

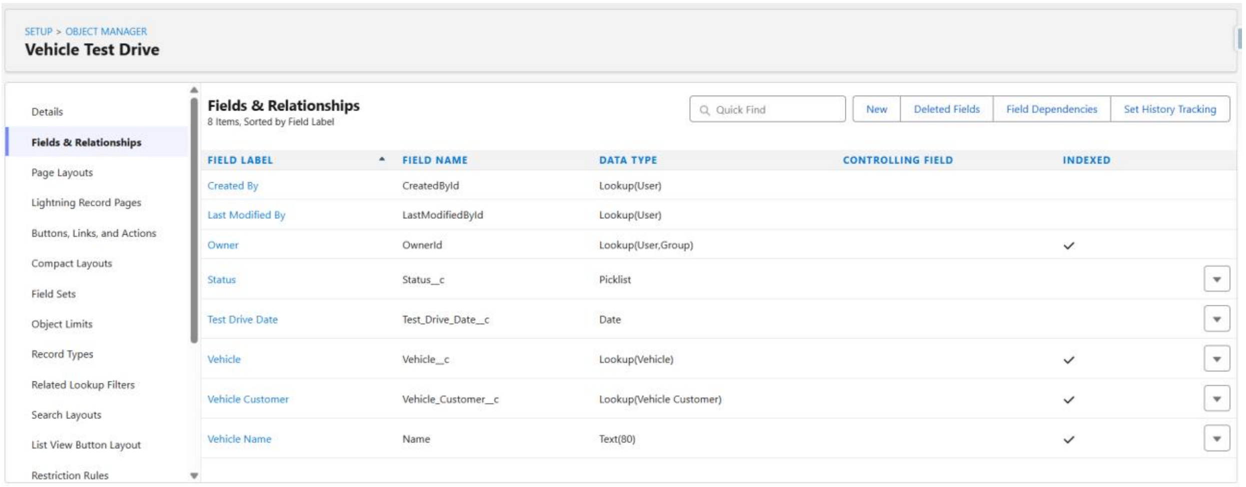
FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Assigned Dealer	Assigned_Dealer__c	Lookup(Vehicle Dealer)		✓
Created By	CreatedById	Lookup(User)		
Last Modified By	LastModifiedById	Lookup(User)		
Order date	Order_date__c	Date		
Owner	OwnerId	Lookup(User,Group)		✓
Status	Status__c	Picklist		
Vehicle	Vehicle__c	Lookup(Vehicle)		✓
Vehicle Customer	Vehicle_Customer__c	Lookup(Vehicle Customer)		✓
Vehicle Order Number	Name	Auto Number		✓



Vehicle Service Request

Fields & Relationships
9 Items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
Issue Description	Issue_Description__c	Text(60)		
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)		✓



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 Name	Name	Text(80)		✓

Validation Rules

- **Out-of-Stock Order Blocker:**
  This rule stops users from creating an order when the selected vehicle has zero stock.

Automation: Workflow Tools

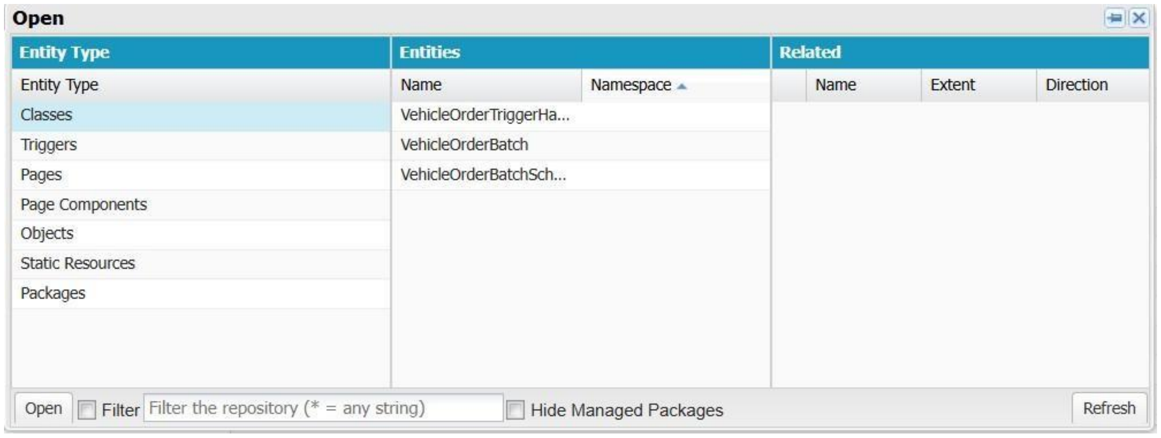
- A **Record-Triggered Flow** on the Order object automatically assigns the nearest dealer based on the customer's address.
- **Scheduled Flows** were used to send test-drive reminder notifications.

Apex Classes and Triggers

Apex Classes:
Apex classes were developed to organize the trigger logic and support automation in the backend:

- VehicleOrderTriggerHandler – manages stock validation and updates inside the trigger.
- VehicleOrderBatch – checks pending orders and confirms them when stock becomes available.
- VehicleOrderBatchScheduler – schedules the batch job to run every day at 12 PM.

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



Open

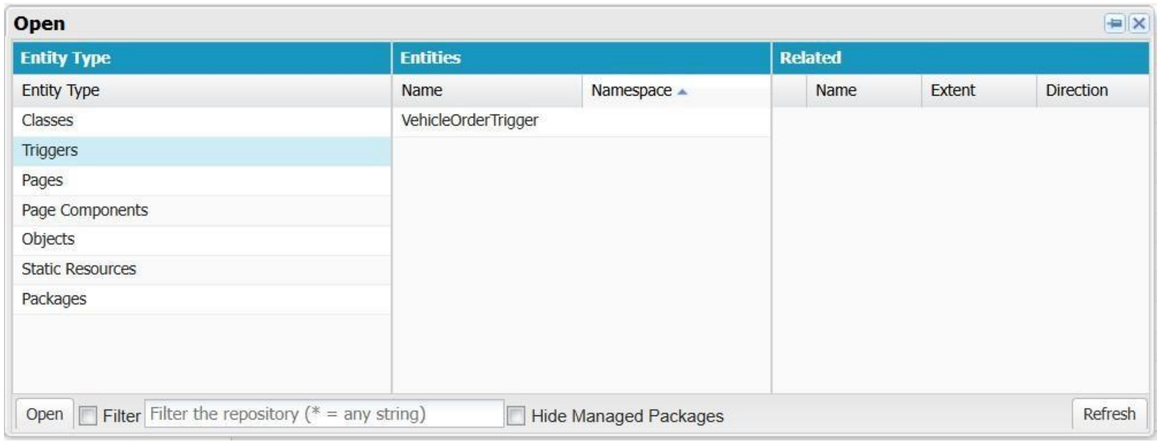
Entity Type	Entities		Related		
Entity Type	Name	Namespace	Name	Extent	Direction
Classes	VehicleOrderTriggerHa...				
Triggers	VehicleOrderBatch				
Pages	VehicleOrderBatchSch...				
Page Components					
Objects					
Static Resources					
Packages					

Apex Trigger:

An Apex Trigger was created on the **Order** object to perform the following functions:

- Validate vehicle stock availability.
- Automatically assign a dealer (if this task is not already done by a Flow).
- Order status update logic (**Pending or Confirmed**).

The trigger uses a **Trigger Handler** to follow Salesforce best practices.



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

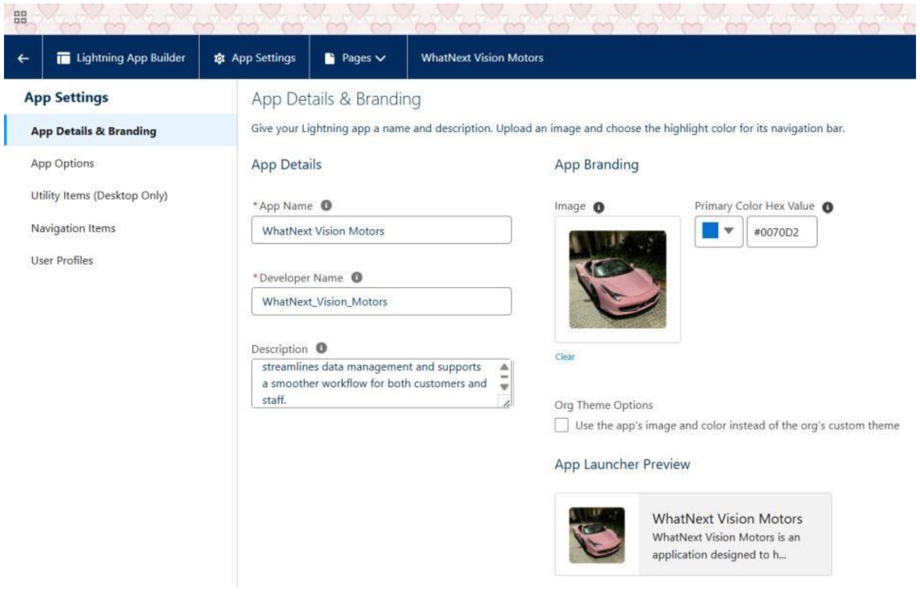
Filter the repository (* = any string)

Phase 3: UI/UX Development & Customization

Lightning App Setup through App Manager

A custom Lightning App called **"WhatNext Vision Motors"** was created using the App Manager. This app includes important custom tabs such as Vehicles, Dealers, Orders, Customers, Test Drives, and Service Requests to make navigation easier.

- Lightning App created: ***WhatNext Vision Motors***
- Tabs: Vehicles, Vehicle Dealers, Vehicle Customers, Vehicle Orders, Vehicle Test Drives, Services
- **Dynamic Forms** were used to show fields based on the record's status and 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: streamlines data management and supports a smoother workflow for both customers and staff.

App Branding

Image

Primary Color Hex Value: #0070D2

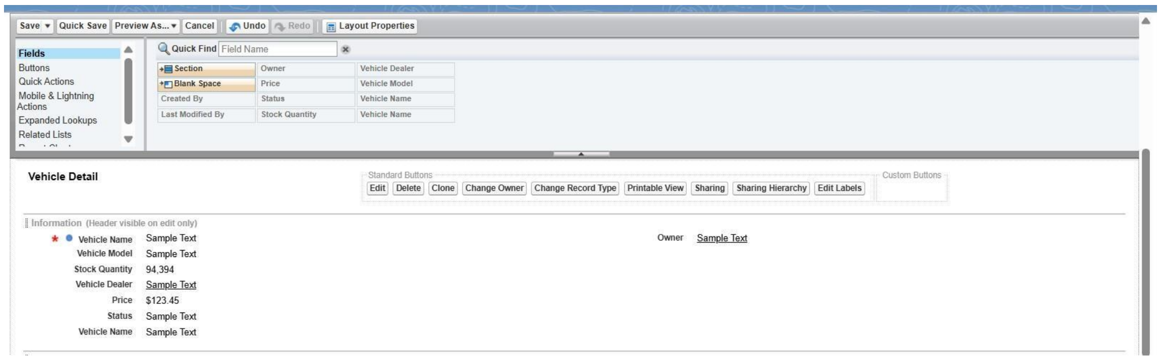
Org Theme Options

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

WhatNext Vision Motors
WhatNext Vision Motors is an application designed to h...

Page Layouts and Dynamic Forms:

Page layouts were customized for the key objects **Vehicle__c**, **Vehicle_Order__c**, and **Vehicle_Test_Drive__c** to provide a clear interface and show only the fields that users need. Dynamic Forms were applied so that fields appear directly on the Lightning Record Page and are shown or hidden based on conditions such as order status or vehicle availability.



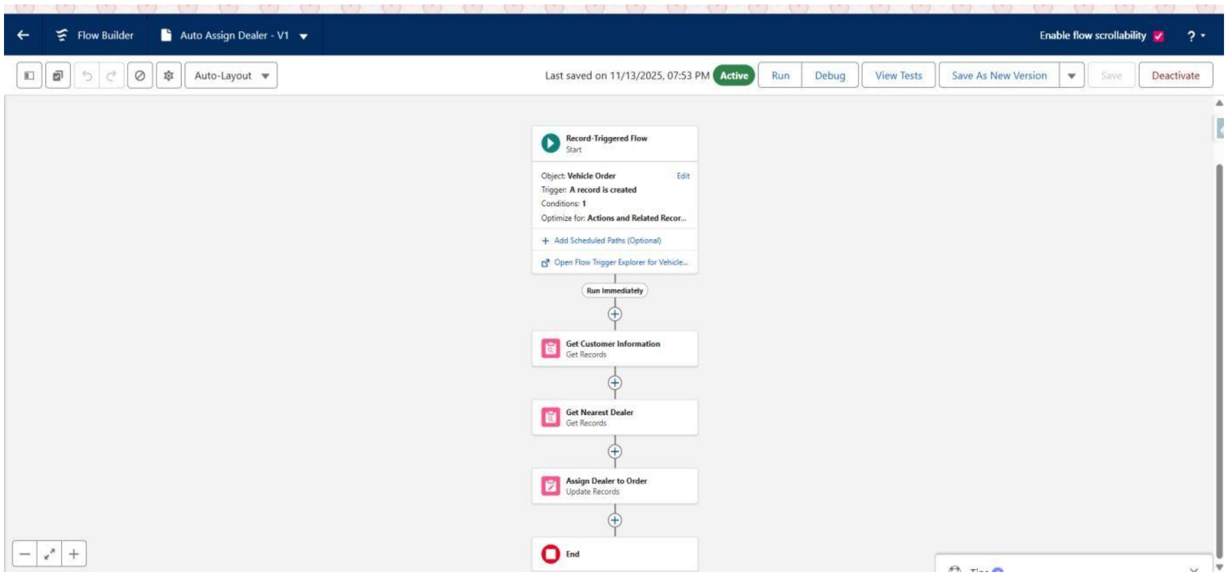
Information (Header visible on edit only)

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:

- Retrieves the customer's address.
- Identifies a dealer located in the same city or near location.
- Assigns that vehicle dealer to the order.



Last saved on 11/13/2025, 07:53 PM

Record-Triggered Flow
Start

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

Get Customer Information
Get Records

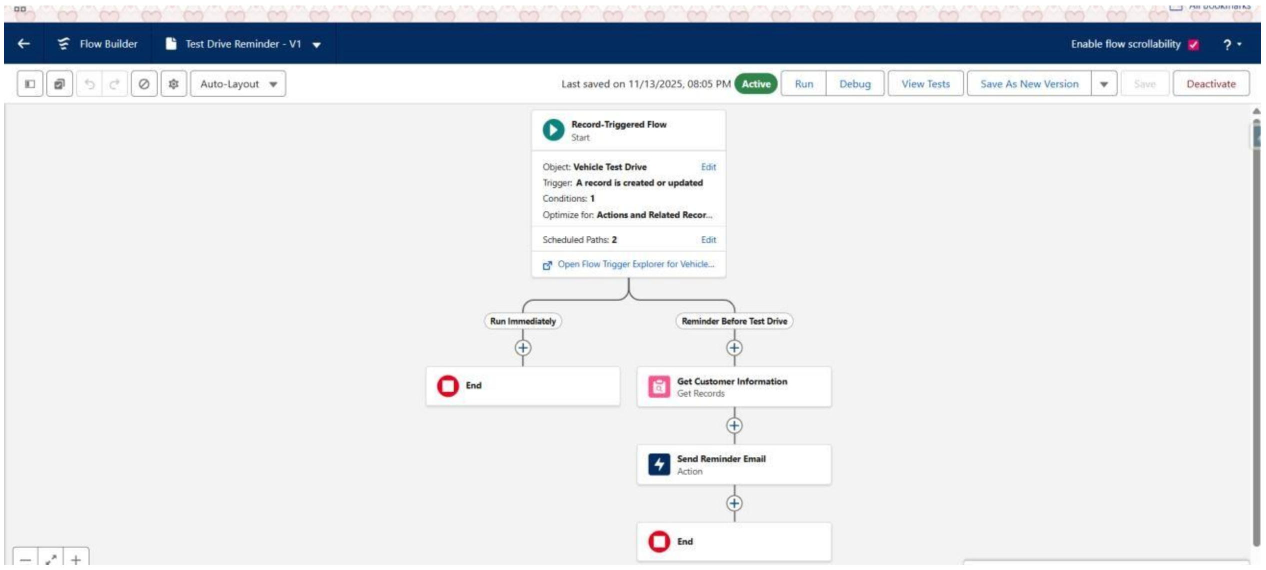
Get Nearest Dealer
Get Records

Assign Dealer to Order
Update Records

End

Flow 2: Test Drive Reminder

- This **Record-Triggered Flow** runs when a **Vehicle_Test_Drive__c** record is created or updated.
- Sends an email reminder **one day before** the scheduled test drive.



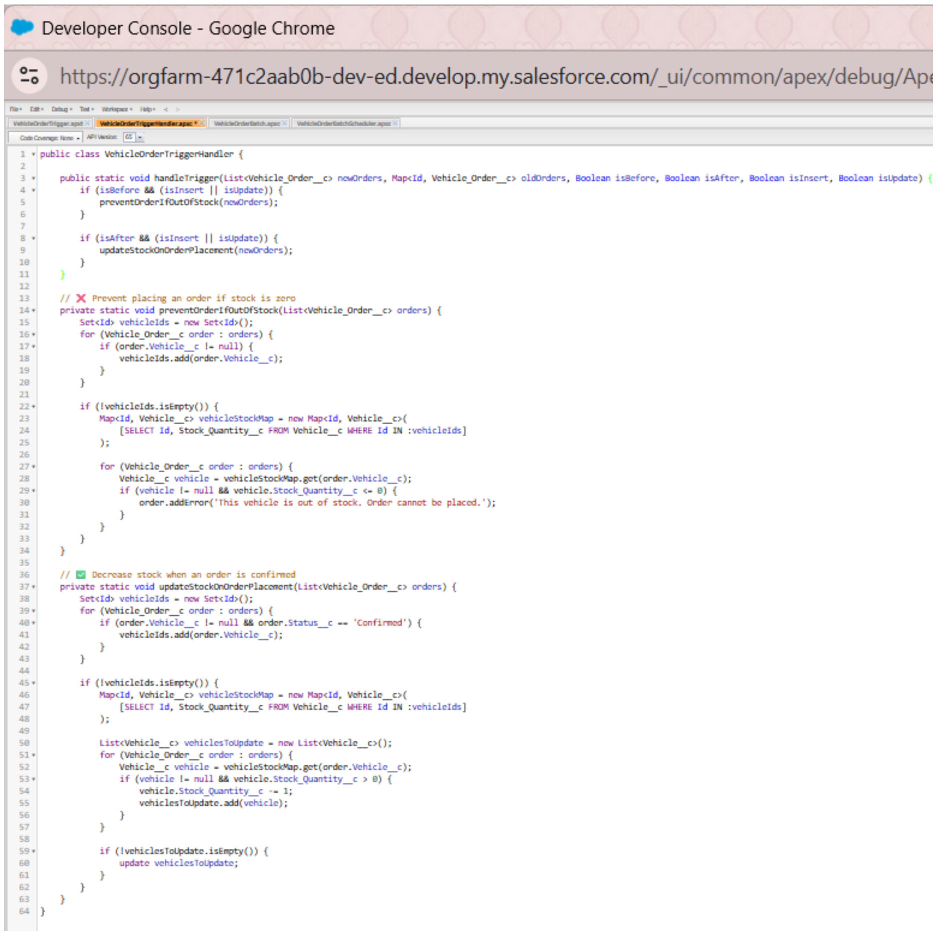
Last saved on 11/13/2025, 08:05 PM

Record-Triggered Flow
Start

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

Apex Trigger & Handler

- Trigger: VehicleOrderTrigger
- Handler: VehicleOrderTriggerHandler
    - This prevents out-of-stock orders
    - It updates stock when order is confirmed

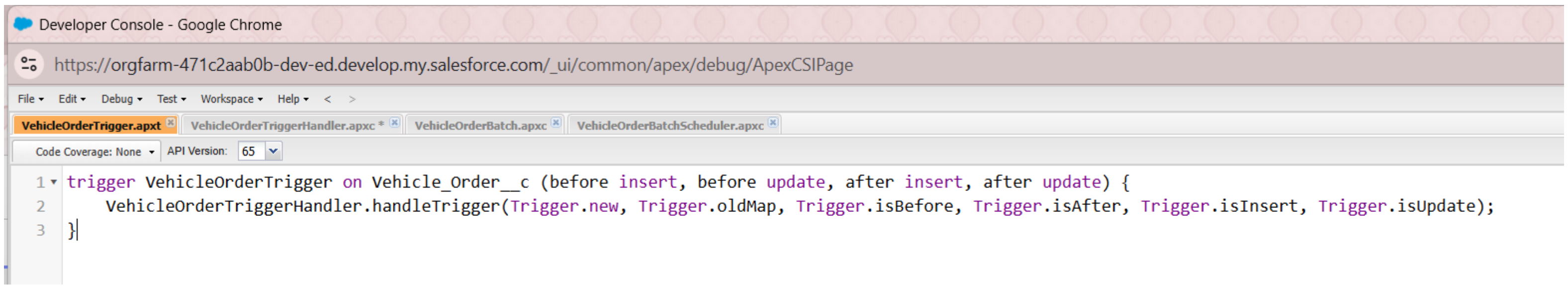


Developer Console - Google Chrome

https://orgfarm-471c2aab0b-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/Ape

File ▾ Edit ▾ Debug ▾ Test ▾ Workspace ▾ Help ▾ < >
VehicleOrderTrigger.apxt
VehicleOrderTriggerHandler.apxc *
VehicleOrderBatch.apxc
VehicleOrderBatchScheduler.apxc
Code Coverage: None
API Version: 65

```
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);
        }
    }

    // ❌ Prevent placing an order if stock is zero
    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) {
                    order.addError('This vehicle is out of stock. Order cannot be placed.');
                }
            }
        }
    }

    // ✅ Decrease stock when an order is confirmed
    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;
            }
        }
    }
}
```



Developer Console - Google Chrome

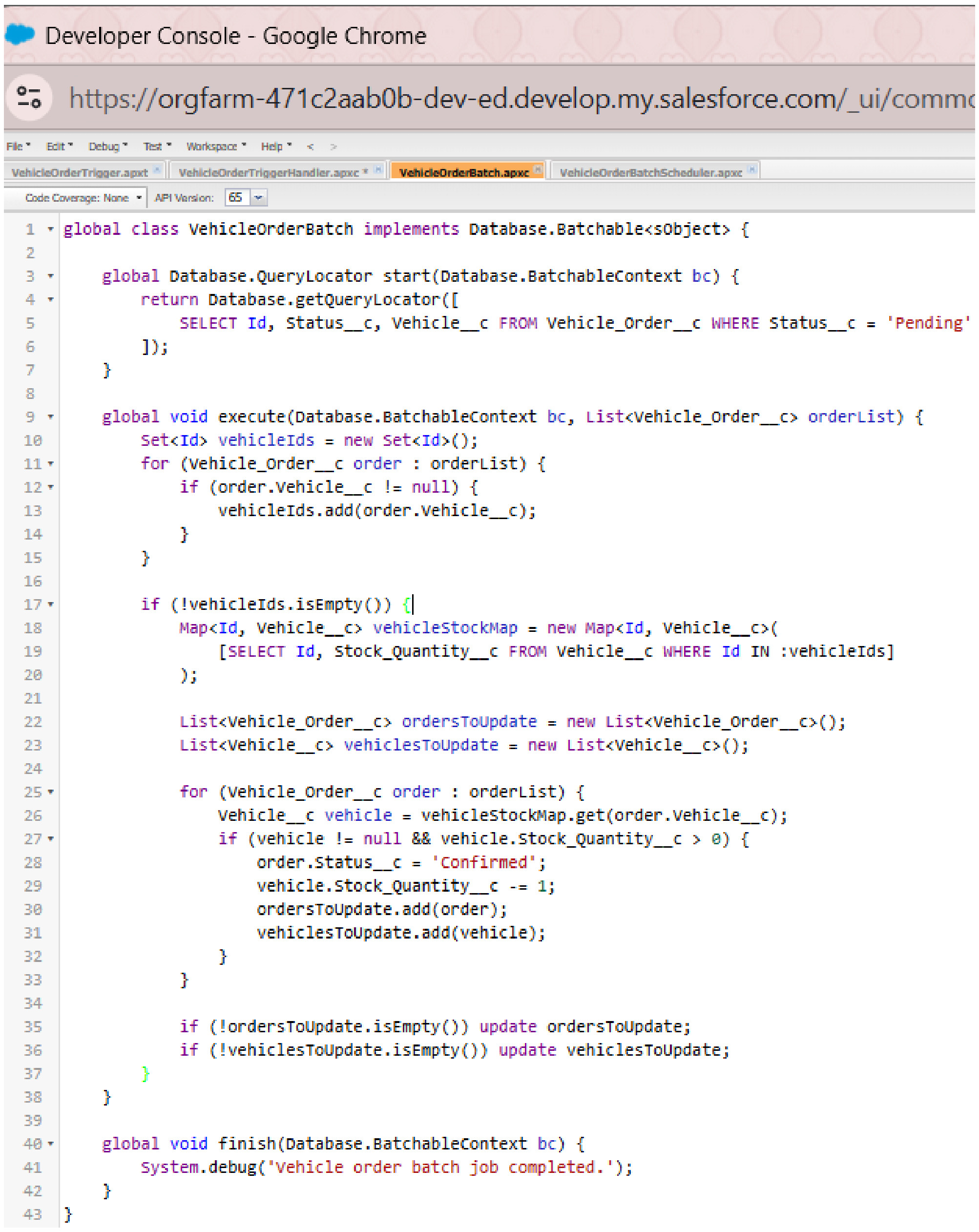
https://orgfarm-471c2aab0b-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage

File ▾ Edit ▾ Debug ▾ Test ▾ Workspace ▾ Help ▾ < >
VehicleOrderTrigger.apxt
VehicleOrderTriggerHandler.apxc *
VehicleOrderBatch.apxc
VehicleOrderBatchScheduler.apxc
Code Coverage: None
API Version: 65

```
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

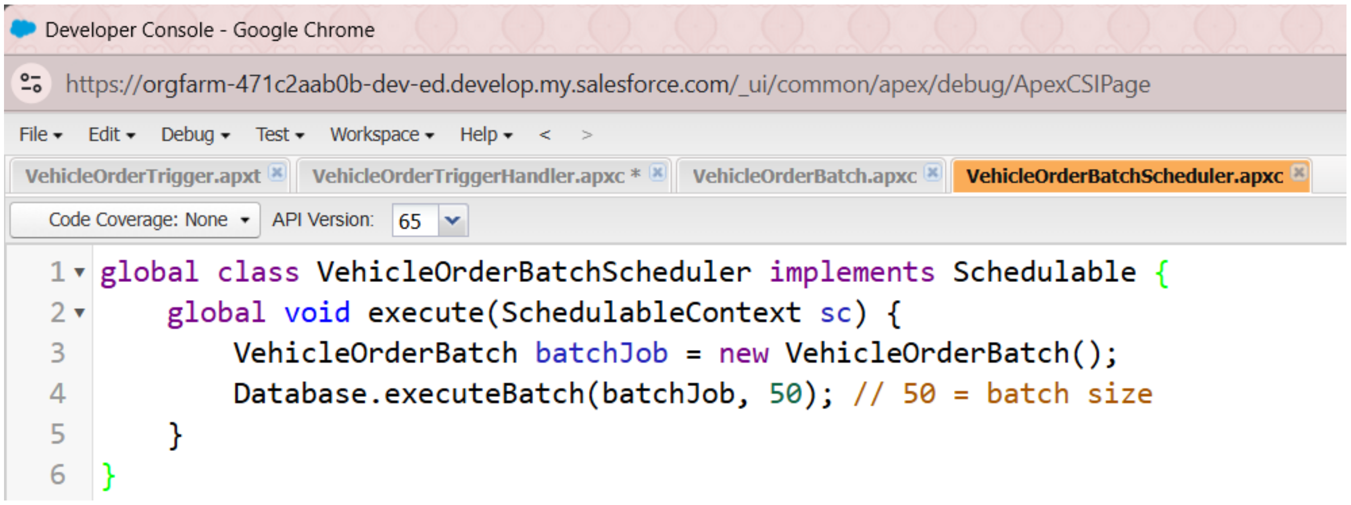
- Class: VehicleOrderBatch
- Runs daily
- Checks for pending orders and available stock
- Updates status to *Confirmed* and adjusts stock


Developer Console - Google Chrome
https://orgfarm-471c2aab0b-dev-ed.develop.my.salesforce.com/_ui/comm
File Edit Debug Test Workspace Help < >
VehicleOrderTrigger.apxt
VehicleOrderTriggerHandler.apxc *
VehicleOrderBatch.apxc
VehicleOrderBatchScheduler.apxc
Code Coverage: None
API Version: 65

```
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.');
    }
}
```

Scheduled Apex

- Class: VehicleOrderBatchScheduler
- Executes batch class automatically


Developer Console - Google Chrome
https://orgfarm-471c2aab0b-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File Edit Debug Test Workspace Help < >
VehicleOrderTrigger.apxt
VehicleOrderTriggerHandler.apxc *
VehicleOrderBatch.apxc
VehicleOrderBatchScheduler.apxc
Code Coverage: None
API Version: 65

```
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 to import data for standard objects like Accounts and Contacts.
- **Data Loader**:
  Used for large data volumes and for custom objects like **Vehicle__c**, **Dealer__c**, and **Order__c**.

Steps:

1. Uploaded CSV files containing sample records.
2. Mapped the CSV columns to the corresponding Salesforce fields.
3. Used **Data Loader** to insert records for:

- **Vehicle__c**
- **Dealer__c**
- **Customer__c**
- **Order__c** (with valid relationships to other objects)

Field History Tracking, Duplicate Rules, and Matching Rules

Field History Tracking:

Field History Tracking was enabled for the following objects to monitor changes:

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

Duplicate & Matching Rules:

- **Matching Rule:** A custom rule on **Customer__c** based on **Email__c** and **Phone__c**.
- **Duplicate Rule:** Prevents the insertion of duplicate customer records.

Profiles, Roles, Permission Sets, and Sharing Rules Profiles and Roles:

- Standard profiles, such as **Standard User** and **System Administrator**, were used.

Role Hierarchy was set up as follows:

- CEO

```
        └── Sales Manager
        └── Sales Rep
```

Permission Sets:

- An **Order Management Access** permission set was created.
- Assigned to users who need **create/read access** to Orders and Vehicles.

Sharing Rules:

- **Public Read/Write** access 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:

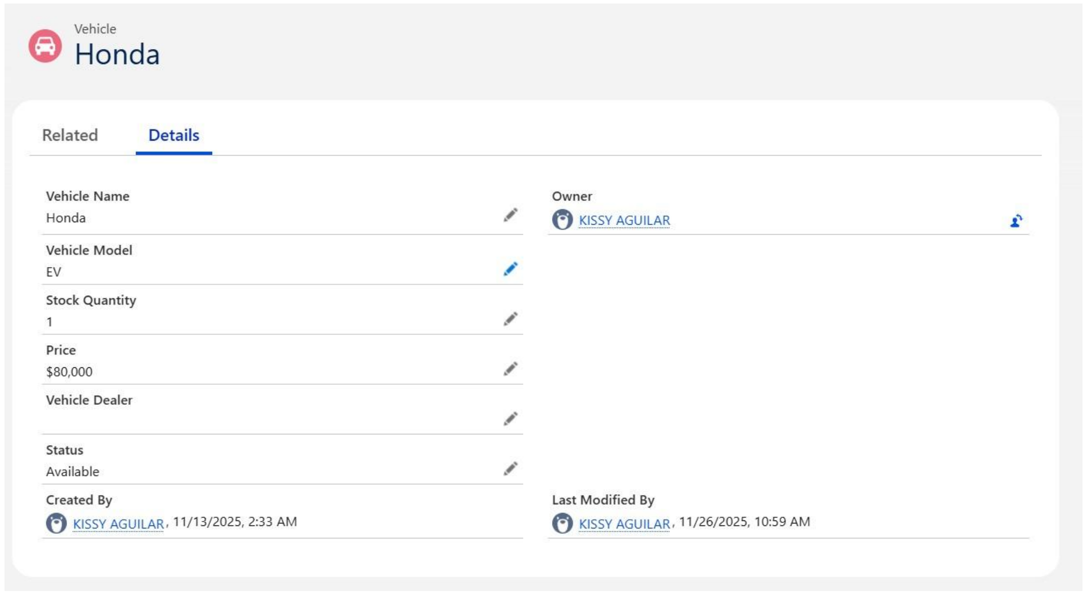
Vehicle Name: Honda

Vehicle Model: EV

Stock Quantity: 1

Price: $80,000

Status: Available



Vehicle
Honda

Related | Details

Vehicle Name Honda	Owner KISSY AGUILAR
Vehicle Model EV	
Stock Quantity 1	
Price $80,000	
Vehicle Dealer	
Status Available	
Created By KISSY AGUILAR, 11/13/2025, 2:33 AM	Last Modified By KISSY AGUILAR, 11/26/2025, 10:59 AM

2. **Create Vehicle Customer**

INPUT:

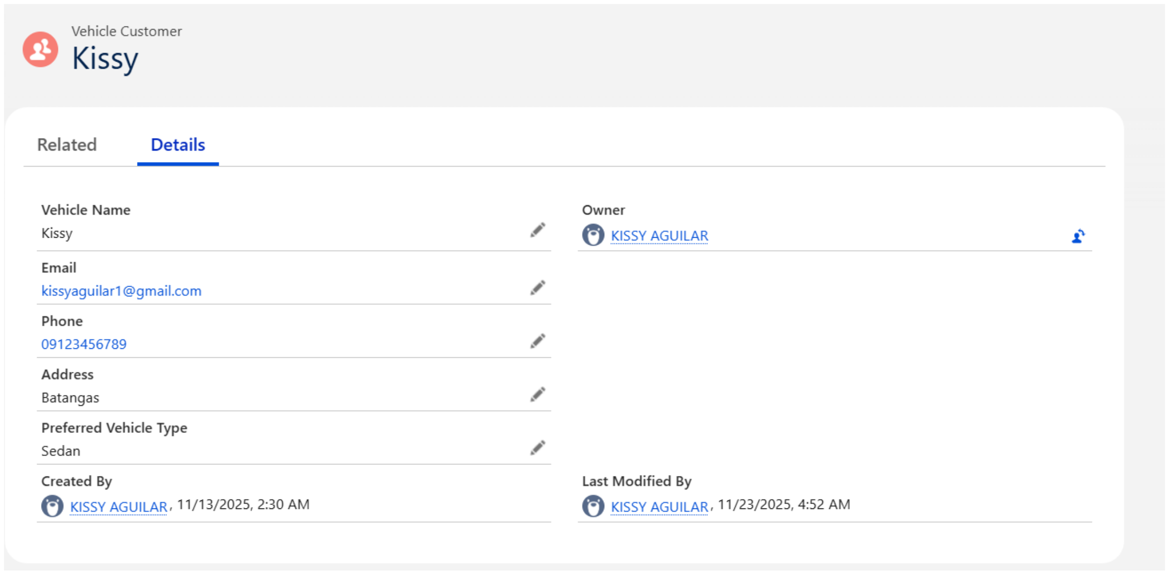
Vehicle Name: (Name)

Email: (ex:kissyaguilar1@gmail.com)

Phone: 09123456789

Address: Batangas

Preferred Vehicle Type: Sedan

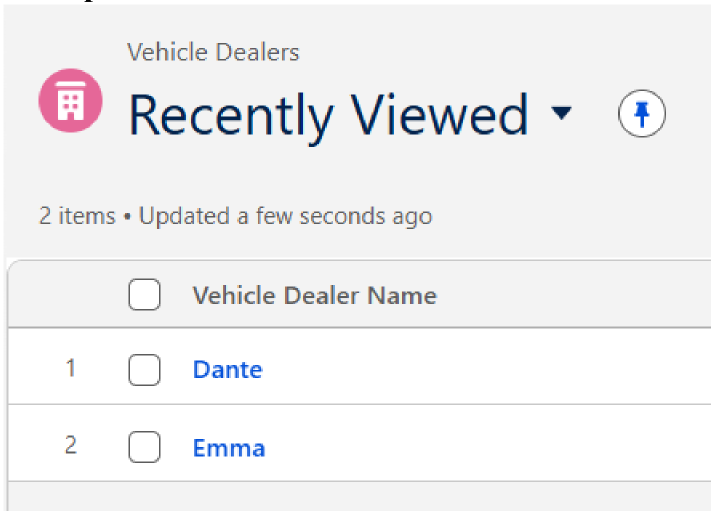

Vehicle Customer

Kissy

Related | Details

Vehicle Name Kissy	Owner KISSY AGUILAR
Email kissyaguilar1@gmail.com	
Phone 09123456789	
Address Batangas	
Preferred Vehicle Type Sedan	
Created By KISSY AGUILAR, 11/13/2025, 2:30 AM	Last Modified By KISSY AGUILAR, 11/23/2025, 4:52 AM

3. **Create Vehicle Dealers**
   Example:


Vehicle Dealers

Recently Viewed

2 items • Updated a few seconds ago

	Vehicle Dealer Name
1	Dante
2	Emma

INPUT:

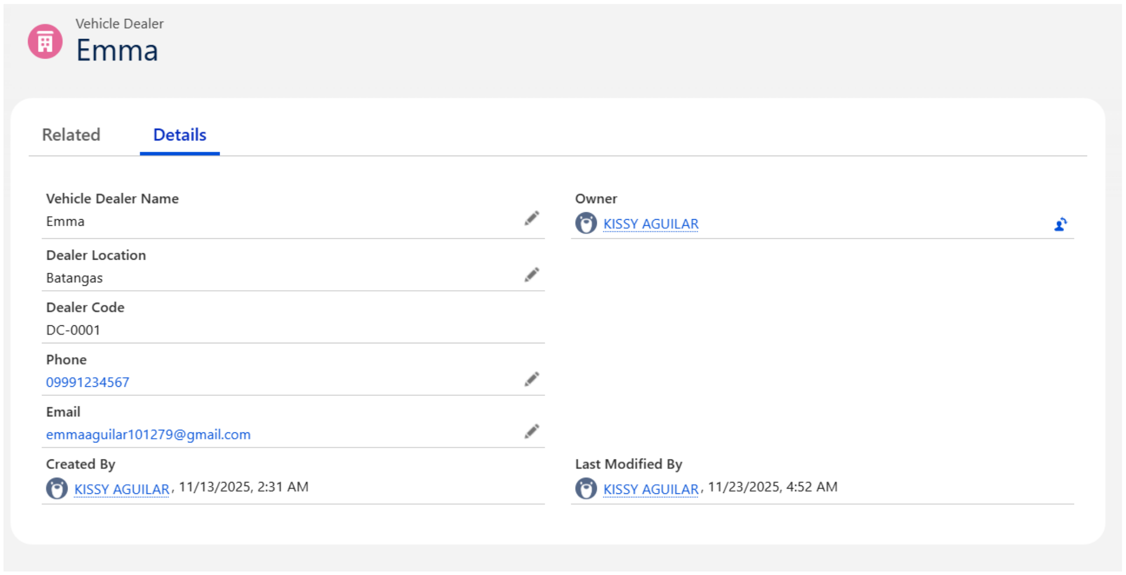
Vehicle Dealer Name: Emma

Dealer Location: Batangas

Dealer Code: DC-001

Phone: 09123456789

Email: (email)

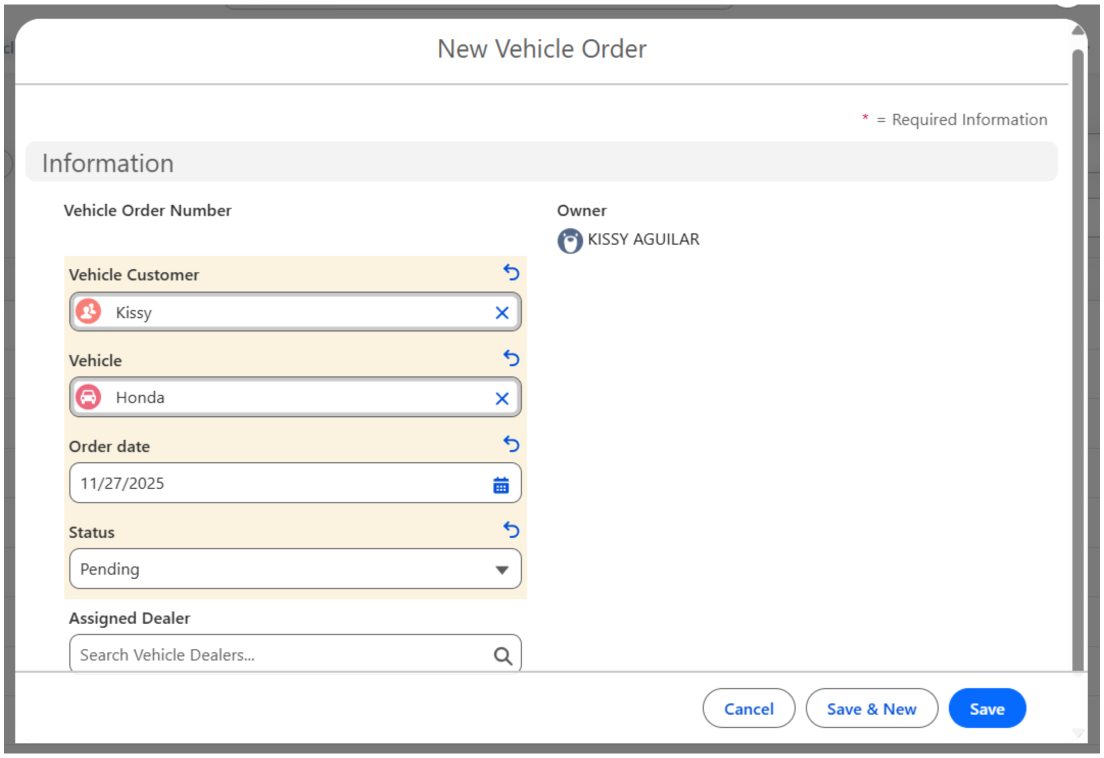

Vehicle Dealer
Emma

Related | Details

Vehicle Dealer Name Emma	Owner KISSY AGUILAR
Dealer Location Batangas	
Dealer Code DC-0001	
Phone 09991234567	
Email emmaaguilar101279@gmail.com	
Created By KISSY AGUILAR, 11/13/2025, 2:31 AM	Last Modified By KISSY AGUILAR, 11/23/2025, 4:52 AM

4. Create Vehicle Order (Test Auto-assign of Nearest or same City of Dealer)

INPUT:

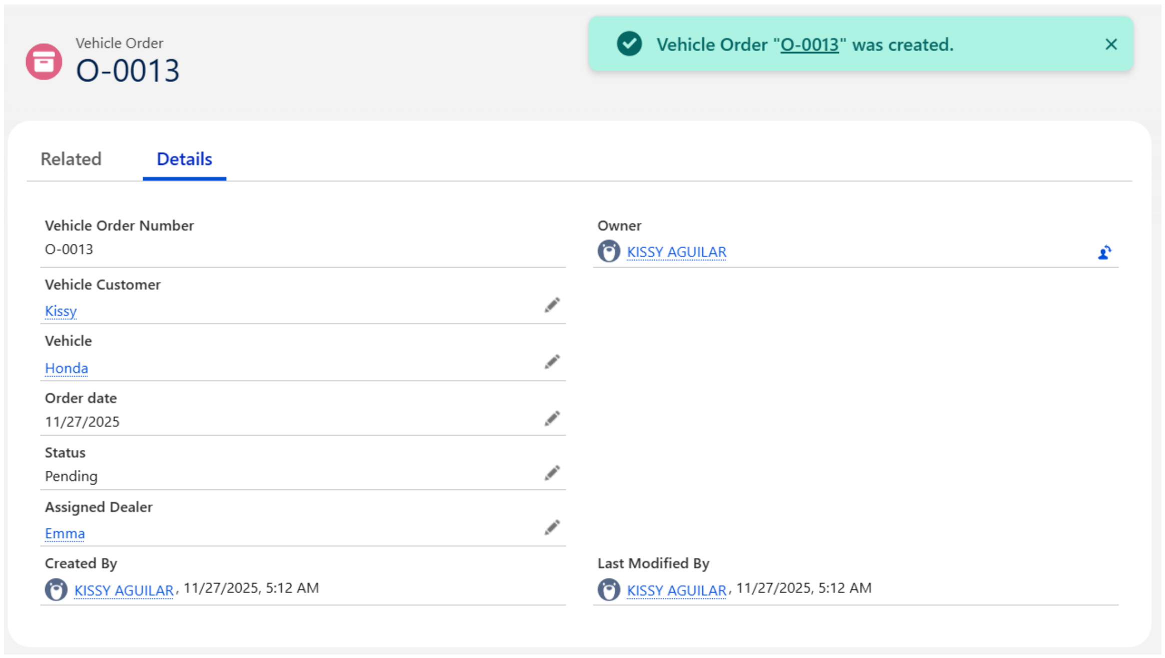

New Vehicle Order

* = Required Information

Information

Vehicle Order Number	Owner KISSY AGUILAR
Vehicle Customer Kissy	
Vehicle Honda	
Order date 11/27/2025	
Status Pending	
Assigned Dealer Search Vehicle Dealers...	

OUPUT:


Vehicle Order
O-0013

Vehicle Order "O-0013" was created.

Related Details

Vehicle Order Number
O-0013

Owner
KISSY AGUILAR

Vehicle Customer
Kissy

Vehicle
Honda

Order date
11/27/2025

Status
Pending

Assigned Dealer
Emma

Created By
KISSY AGUILAR, 11/27/2025, 5:12 AM

Last Modified By
KISSY AGUILAR, 11/27/2025, 5:12 AM

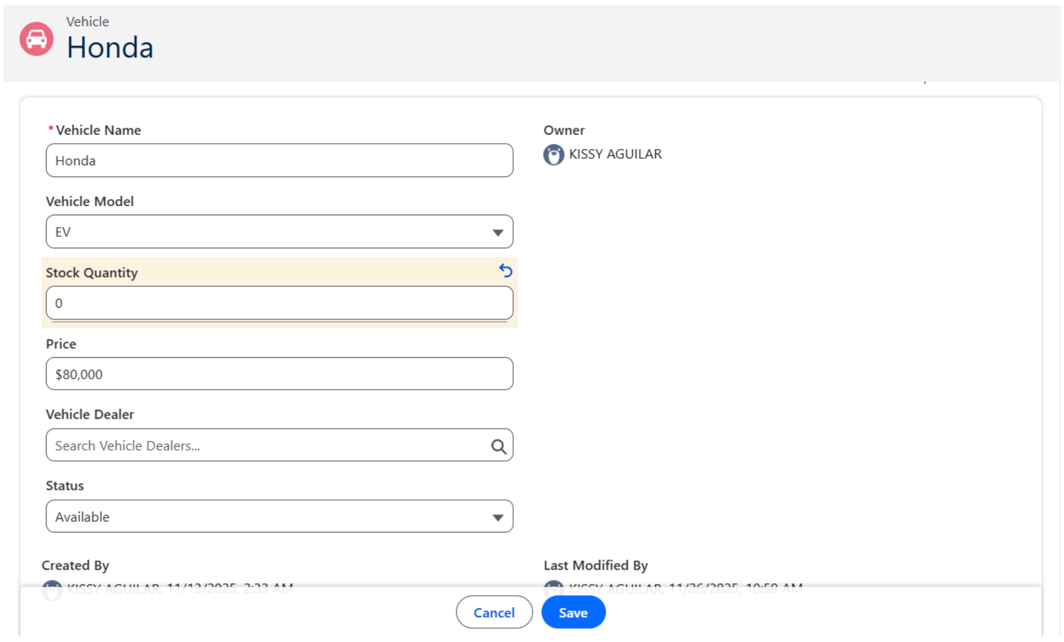
EXPLANATION:

Since the Vehicle Customer (Kissy) and the Vehicle Dealer (Emma) are in the same location (Batangas), **the system automatically assigns Emma as the dealer**.

5. Test OUT-OF-STOCK Vehicle Order

INPUT:

Stock Quantity: Set to 0 (zero)


Vehicle
Honda

*Vehicle Name
Honda

Owner
KISSY AGUILAR

Vehicle Model
EV

Stock Quantity
0

Price
$80,000

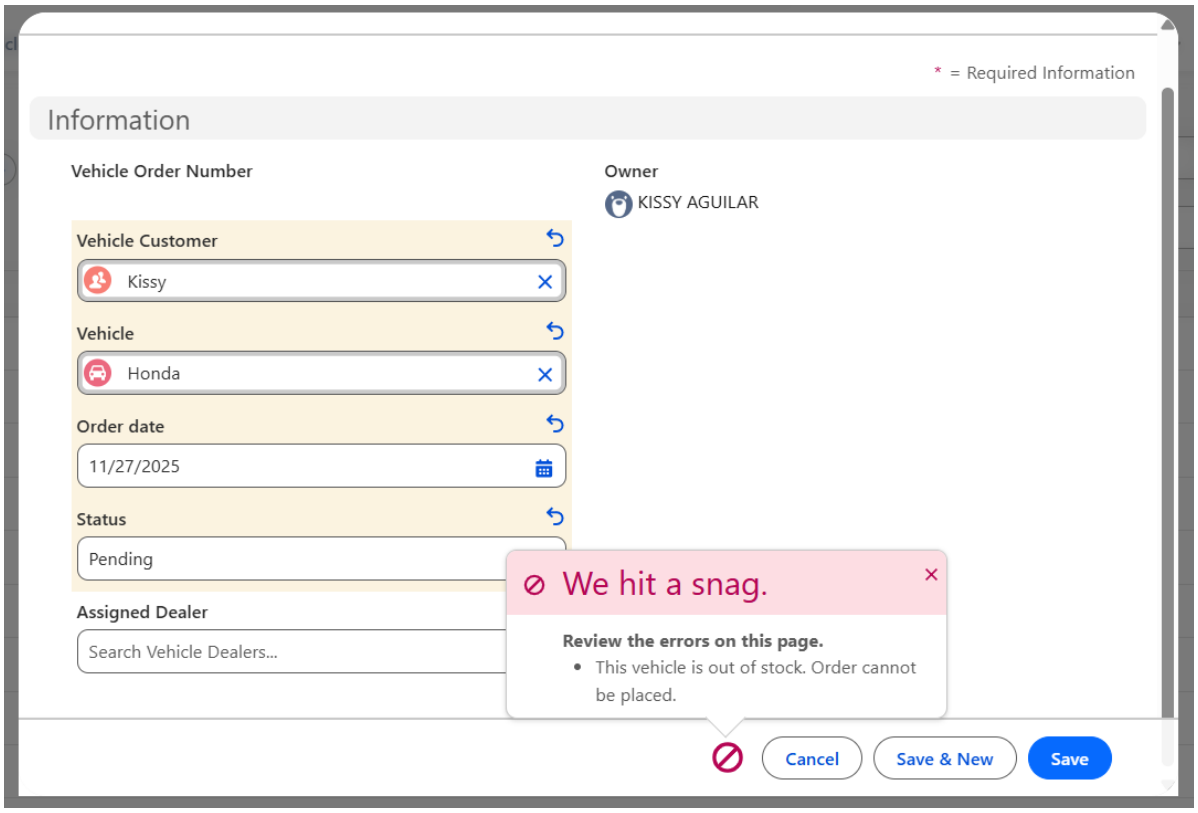
Vehicle Dealer
Search Vehicle Dealers...

Status
Available

Created By
KISSY AGUILAR, 11/13/2025, 2:33 AM

Last Modified By
KISSY AGUILAR, 11/26/2025, 10:59 AM

OUTPUT:


* = Required Information

Information

Vehicle Order Number

Owner
KISSY AGUILAR

Vehicle Customer
Kissy

Vehicle
Honda

Order date
11/27/2025

Status
Pending

Assigned Dealer
Search Vehicle Dealers...

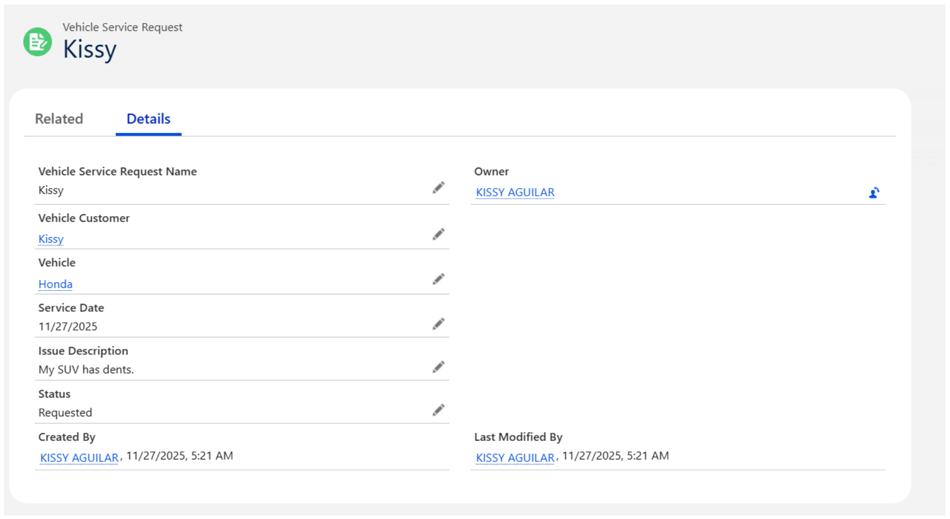
We hit a snag.

Review the errors on this page.
- This vehicle is out of stock. Order cannot be placed.

EXPLANATION:

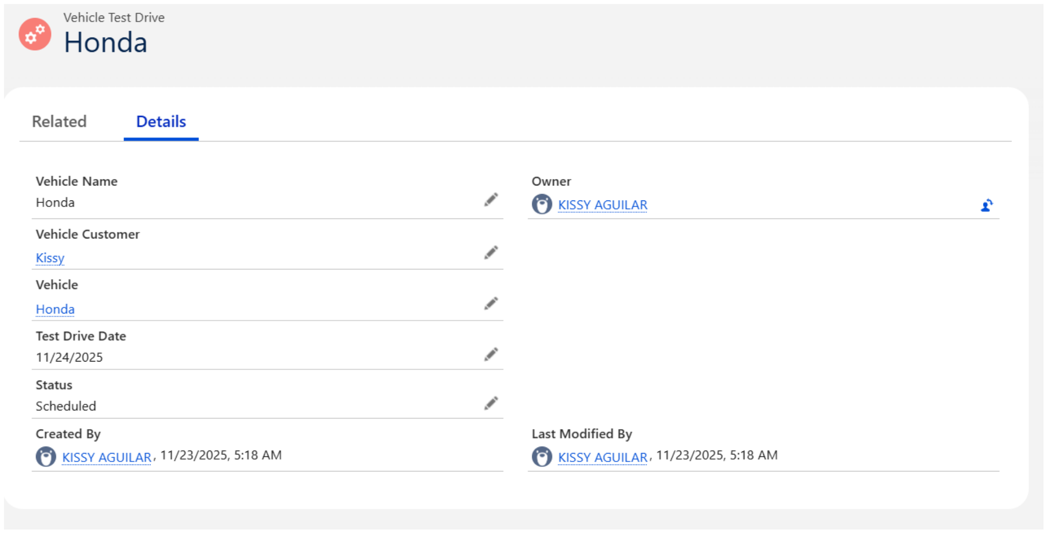
Since the Stock Quantity is 0, the system will display an alert when an order is attempted. It will show: "This vehicle is out of stock. Order cannot be placed."

6. Create Vehicle Service Request


Vehicle Service Request
Kissy

Vehicle Service Request Name: Kissy	Owner: KISSY AGUILAR
Vehicle Customer: Kissy	
Vehicle: Honda	
Service Date: 11/27/2025	
Issue Description: My SUV has dents.	
Status: Requested	
Created By: KISSY AGUILAR, 11/27/2025, 5:21 AM	Last Modified By: KISSY AGUILAR, 11/27/2025, 5:21 AM

7. Create Vehicle Test Drive


Vehicle Test Drive
Honda

Related | Details

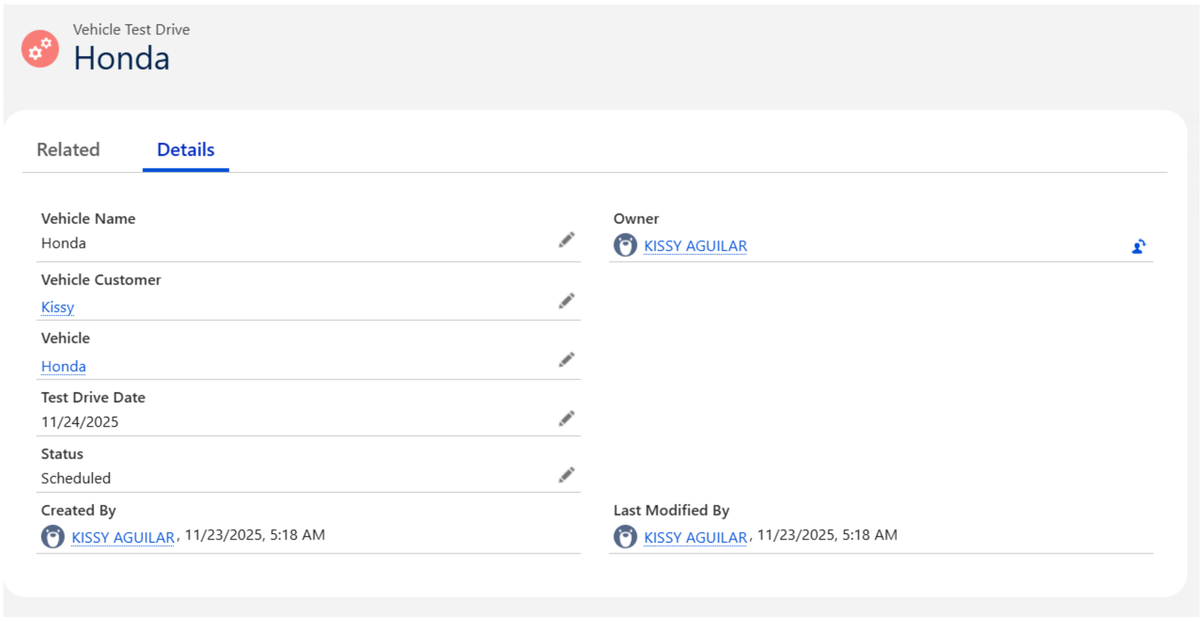
Vehicle Name Honda	Owner KISSY AGUILAR
Vehicle Customer Kissy	
Vehicle Honda	
Test Drive Date 11/24/2025	
Status Scheduled	
Created By KISSY AGUILAR, 11/23/2025, 5:18 AM	Last Modified By KISSY AGUILAR, 11/23/2025, 5:18 AM

8. Test Drive Reminder Email:

Customer: Select any customer with email (example: Kissy)

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

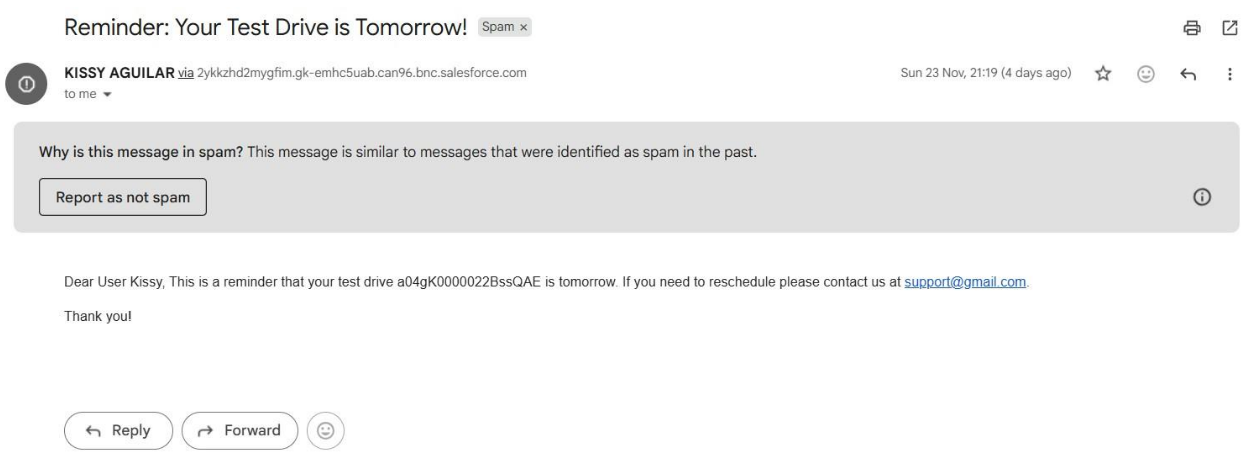
INPUT:


Vehicle Test Drive
Honda

Related | Details

Vehicle Name Honda	Owner KISSY AGUILAR
Vehicle Customer Kissy	
Vehicle Honda	
Test Drive Date 11/24/2025	
Status Scheduled	
Created By KISSY AGUILAR, 11/23/2025, 5:18 AM	Last Modified By KISSY AGUILAR, 11/23/2025, 5:18 AM

OUTPUT:



Reminder: Your Test Drive is Tomorrow!

KISSY AGUILAR via 2ykkzhd2mygfim.gk-emhc5uab.can96.bnc.salesforce.com
to me

Sun 23 Nov, 21:19 (4 days ago)

Why is this message in spam? This message is similar to messages that were identified as spam in the past.

Dear User Kissy, This is a reminder that your test drive a04gK0000022BssQAE is tomorrow. If you need to reschedule please contact us at support@gmail.com.

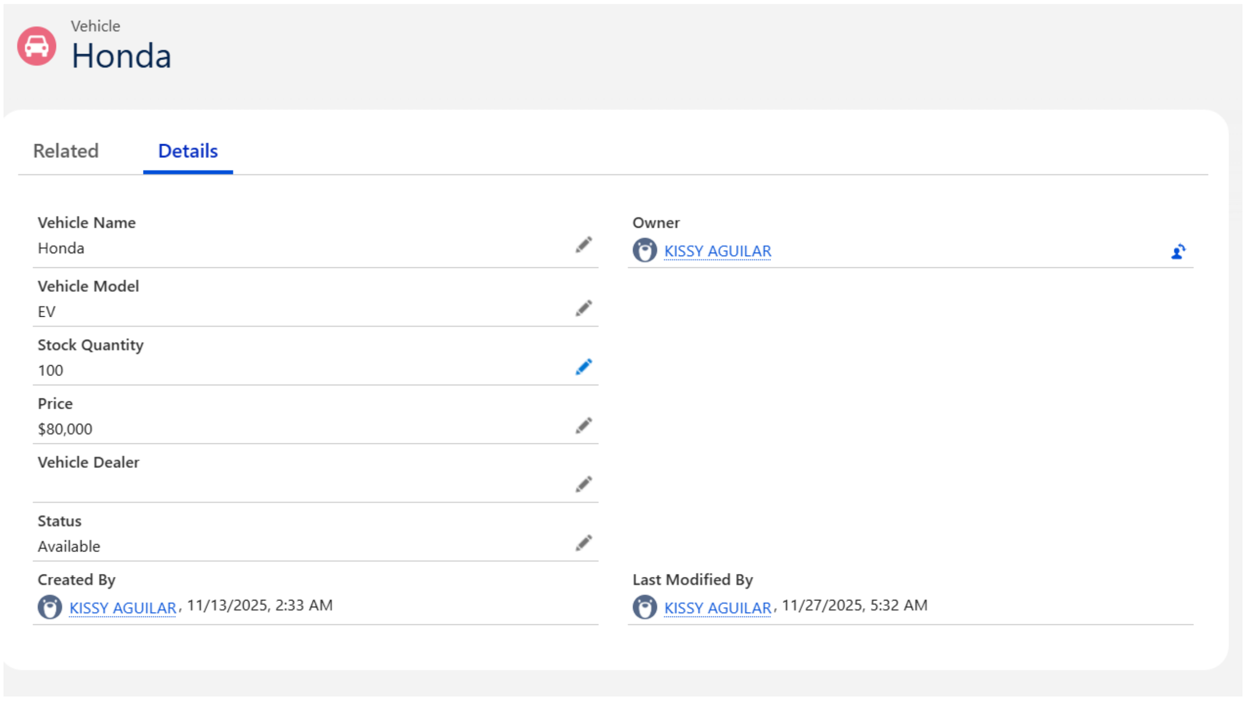
Thank you!

EXPLANATION:
Since it is scheduled, if the test drive is set for tomorrow, the system will send an email today to remind me to attend the test drive.

9. **Test that when an order is confirmed, the vehicle stock is reduced accordingly.**

INPUT:

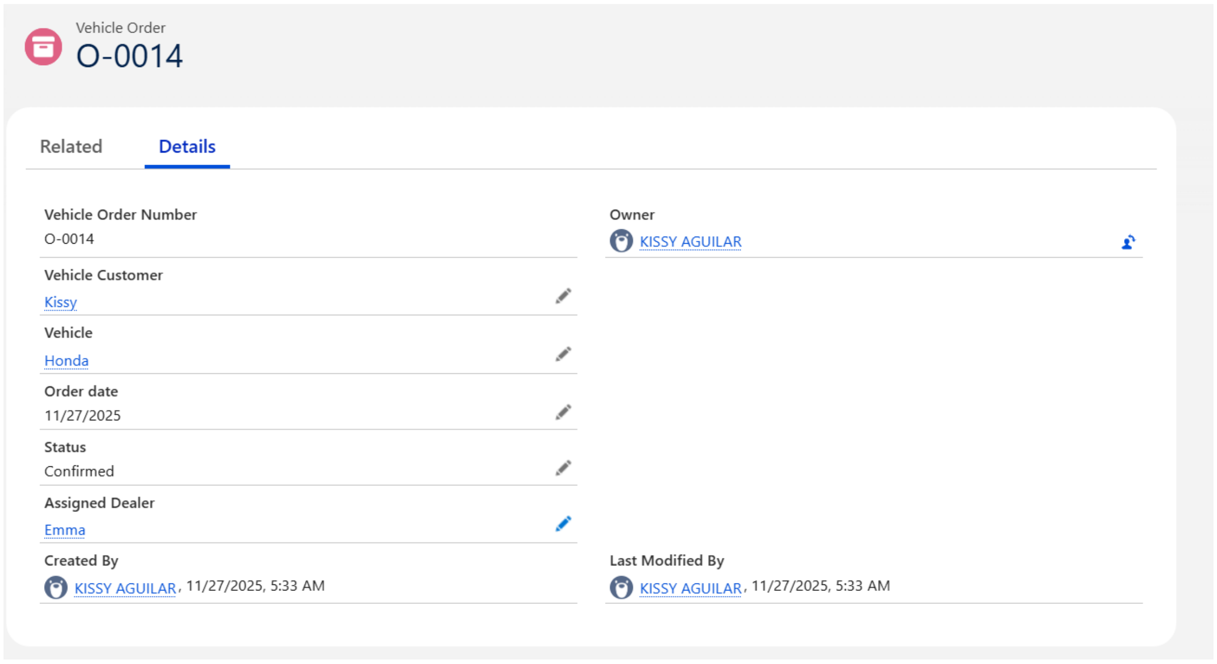
Stock Quantity: Set to 100



Vehicle
Honda

Vehicle Name	Honda
Vehicle Model	EV
Stock Quantity	100
Price	$80,000
Vehicle Dealer	
Status	Available
Created By	KISSY AGUILAR, 11/13/2025, 2:33 AM
Owner	KISSY AGUILAR
Last Modified By	KISSY AGUILAR, 11/27/2025, 5:32 AM

ORDER: 1 honda



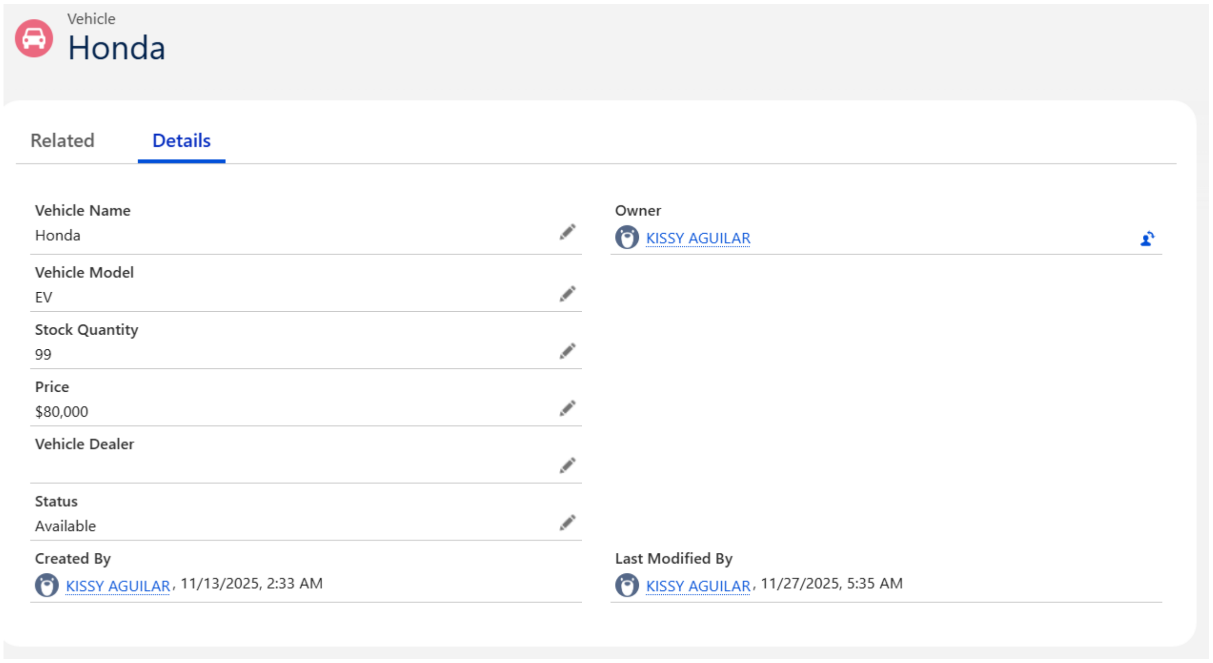
Vehicle Order
O-0014

Related | Details

Vehicle Order Number: O-0014	Owner: KISSY AGUILAR
Vehicle Customer: Kissy	
Vehicle: Honda	
Order date: 11/27/2025	
Status: Confirmed	
Assigned Dealer: Emma	
Created By: KISSY AGUILAR, 11/27/2025, 5:33 AM	Last Modified By: KISSY AGUILAR, 11/27/2025, 5:33 AM

OUTPUT:

Stock Quantity: Reduce 1 (as a result it became 99)



Vehicle
Honda

Related | Details

Vehicle Name: Honda	Owner: KISSY AGUILAR
Vehicle Model: EV	
Stock Quantity: 99	
Price: $80,000	
Vehicle Dealer:	
Status: Available	
Created By: KISSY AGUILAR, 11/13/2025, 2:33 AM	Last Modified By: KISSY AGUILAR, 11/27/2025, 5:35 AM

EXPLANATION:

The system automatically reduces stock when an order is confirmed, but it does not reduce stock if the order is pending.

To make sure the Apex code can be deployed and works correctly, Test Classes were created for the following components:

- **OrderTriggerHandler**
- **DealerAssignmentService**
- **StockValidationTrigger**

Phase 5: Deployment, Documentation & Maintenance

Deployment Strategy

The **Change Set** method was used to deploy features from the Developer Org to the production environment.

1. An Outbound Change Set was created in the source organization.
2. All required custom components were added, including:

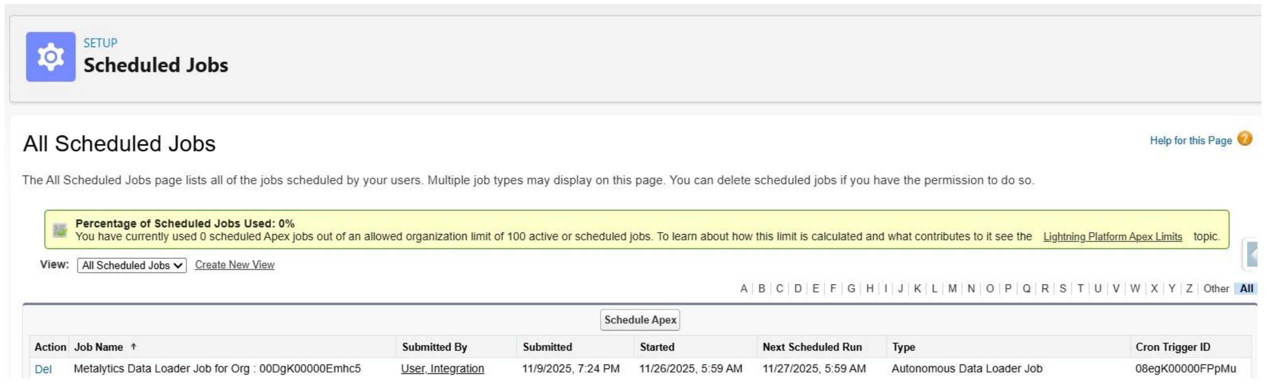
- Custom objects
- Fields
- Flows
- Validation rules
- Triggers
- Apex classes

3. The Change Set was uploaded to the target organization (production or sandbox).
4. It was validated and deployed from the Inbound Change Sets section in the target org.
5. A post-deployment manual check was performed to confirm that all features were working properly.

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



SETUP

Scheduled Jobs

All Scheduled Jobs

Help for this Page

The All Scheduled Jobs page lists all of the jobs scheduled by your users. Multiple job types may display on this page. You can delete scheduled jobs if you have the permission to do so.

Percentage of Scheduled Jobs Used: 0%
You have currently used 0 scheduled Apex jobs out of an allowed organization limit of 100 active or scheduled jobs. To learn about how this limit is calculated and what contributes to it see the Lightning Platform Apex Limits topic.

View: All Scheduled Jobs Create New View

A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | Other | All

Schedule Apex

Action	Job Name ↑	Submitted By	Submitted	Started	Next Scheduled Run	Type	Cron Trigger ID
Del	Metalytics Data Loader Job for Org : 00DgK00000Emhc5	User, Integration	11/9/2025, 7:24 PM	11/26/2025, 5:59 AM	11/27/2025, 5:59 AM	Autonomous Data Loader Job	08egK00000FPpMu

System Maintenance and Monitoring

To keep the system working smoothly after deployment, the following maintenance approach was used:

1. Monitoring

- Apex Jobs were used to monitor scheduled jobs and batch processes.
- Debug Logs were reviewed to trace errors or unexpected system behavior.
- Email Alerts were enabled for test drive reminders and any failed processes.

2. User Feedback Loop

- The sales and operations teams used the system for several days after deployment.
- Feedback was collected through manual walkthroughs to find missing features or issues.

3. Updates and Fixes

- Small updates—such as adding help text or adjusting field labels—were made in the sandbox and deployed again using Change Sets.
- Quarterly reviews were planned to introduce system enhancements and improve the user interface.

Troubleshooting Approach

If issues occur in the production environment, the following steps will be taken:

Step 1: Reproduce the Issue

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

Step 2: Enable Debug Logs

- Set **debug logs** for the affected user and analyze the **Flow** or **Apex execution**.

Step 3: Check Apex Jobs or Flows

- For issues related to background processes, review **Apex Job failures** or **Flow error emails**.

Step 4: Fix and Retest

- Update the **Flow** or **Apex logic** as needed.
- Retest the changes in the **sandbox** and redeploy using a **Change Set**.

Conclusion

The Salesforce implementation at WhatNext Vision Motors successfully achieved its goal of improving the customer ordering process and overall operational workflows.

The key accomplishments of the project include:

- Automatic assignment of the nearest dealer through Flows or Apex Triggers
- Stock validation to prevent orders for vehicles with zero availability
- Scheduled automation to update order statuses using Batch Apex
- Enhanced customer experience through streamlined and automated processes
- Reduced manual workload for internal teams

In addition, the project provided several long-term benefits:

- Improved data accuracy and consistency across vehicles, customers, and dealers
- A scalable system that can support future business growth and additional Salesforce features
- Better visibility and reporting capabilities for management, enabling informed decision-making
- Increased efficiency in handling test drives, service requests, and post-sales processes

Overall, this project not only strengthens the company’s customer-facing operations but also establishes a strong foundation for future Salesforce enhancements and automation initiatives. Through this implementation, **WhatNext Vision Motors moves closer to its vision of innovation and excellence** in the mobility sector, ensuring both operational efficiency and superior customer satisfaction.