

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

Project Overview

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

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

Objectives

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

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

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

Phase 1: Requirement Analysis and Planning

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

Business Requirements

The following are the business requirements:

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

Defining Project Scope

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

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

Data Model

Six custom objects were created to reflect business structure:

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

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

LABEL	API NAME	TYPE	DESCRIPTION	LAST MODIFIED	DEPLOYED
Vehicle	Vehicle_c	Custom Object		11/10/2025	✓
Vehicle Customer	Vehicle_Customer_c	Custom Object		11/10/2025	✓
Vehicle Dealer	Vehicle_Dealer_c	Custom Object		11/10/2025	✓
Vehicle Order	Vehicle_Order_c	Custom Object		11/10/2025	✓
Vehicle Service Request	Vehicle_Service_Request_c	Custom Object		11/10/2025	✓
Vehicle Test Drive	Vehicle_Test_Drive_c	Custom Object		11/10/2025	✓

Custom Tabs

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

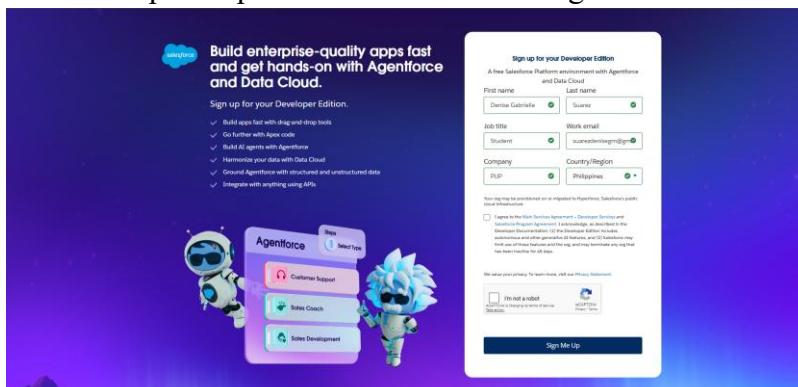
Custom Object tabs look and behave like the standard tabs provided with Salesforce. Web tabs allow you to embed external web applications and content within the Salesforce window. Visualforce tabs allow you to embed Visualforce pages. Lightning Component tabs allow you to add Lightning components to 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.

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	Phone	
Edit Del	Vehicle Test Drives	Gears	

Phase 2: Salesforce Development - Backend & Configurations

- **Setup Environment and DevOps Workflow**

The development process starts with creating Salesforce Developer Org.



- **Environment:** Salesforce Lightning Experience (Developer Edition)
 - **User Profiles/Roles:** Standard profiles were used.
- **Customization of Objects, Fields, Validation Rules, Automation**
 - Six custom objects were created to store relevant business data. This includes:
 - **Vehicle** - Stores vehicle details.
 - **Vehicle Dealer** - Stores authorized dealer info.
 - **Vehicle Customer** - Stores customer details.
 - **Vehicle Order** - Tracks vehicle purchases.
 - **Vehicle Test Drive** - Tracks test drive schedule.
 - **Vehicle Service Request** - Tracks vehicle servicing requests.
 - The following custom objects were configured to support the business flow:
 - **Vehicle** – Store vehicle name, model, quantity, price, and status.
 - Dealer → Lookup Relationship
 - **Vehicle Dealer** – Store dealer name, location, code, phone, and email.
 - **Vehicle Order** – Store order date and status.
 - Customer → Lookup Relationship
 - Vehicle → Lookup Relationship
 - **Vehicle Customer** – Store customer name, email, phone, address, and preferred vehicle type.
 - **Vehicle Test Drive** – Store test drive date and status.
 - Customer → Lookup Relationship
 - Vehicle → Lookup Relationship
 - **Vehicle Service Request** – Store service date, issue description, and status.
 - Customer → Lookup Relationship
 - Vehicle → Lookup Relationship

The image contains two side-by-side screenshots of the Salesforce Object Manager. Both screenshots show the 'Fields & Relationships' section for a specific object. The left screenshot is for 'Vehicle Dealer' and the right is for 'Vehicle Order'. Each screenshot displays a table with columns for Field Label, Field Name, and Data Type. The tables include standard Salesforce fields like Created By, Last Modified By, and Owner, along with custom fields such as Dealer Code, Dealer Location, and Vehicle.

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

FIELD LABEL	FIELD NAME	DATA TYPE
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
Order Number	Name	Auto Number
Owner	OwnerId	Lookup(User/Group)
Status	Status__c	Picklist
Vehicle	Vehicle__c	Lookup(Vehicle)
Vehicle Customer	Vehicle_Customer__c	Lookup(Vehicle Customer)

Fields & Relationships		
9 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	Name	Text(30)

Fields & Relationships		
8 items, Sorted by Field Label		
FIELD LABEL	FIELD NAME	DATA TYPE
Address	Address__c	Text(60)
Created By	CreatedById	Lookup(User)
Customer Name	Name	Text(80)
Email	Email__c	Email
Last Modified By	LastModifiedById	Lookup(User)
Owner	OwnerId	Lookup(User,Group)
Phone	Phone__c	Phone
Preferred Vehicle Type	Prefer__c	Picklist

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(60)
Last Modified By	LastModifiedById	Lookup(User)
Owner	OwnerId	Lookup(User,Group)
Service Date	Service_Date__c	Date
Service Request Name	Name	Text(80)
Status	Status__c	Picklist
Vehicle	Vehicle__c	Lookup(Vehicle)
Vehicle Customer	Vehicle_Customer__c	Lookup(Vehicle Customer)

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
Test Drive Name	Name	Text(80)
Vehicle	Vehicle__c	Lookup(Vehicle)
Vehicle Customer	Vehicle_Customer__c	Lookup(Vehicle Customer)

Validation Rules

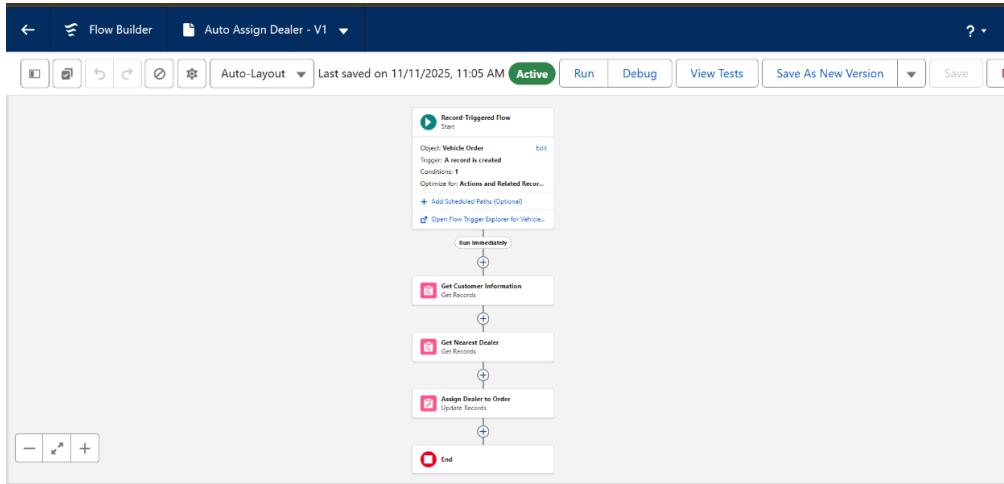
- Out-of-stock Blocker:** Prevent placing an order for out-of-stock vehicle.

The screenshot shows the Salesforce 'New Vehicle Order' page. The 'Information' section contains fields for 'Vehicle Customer' (Jane Air) and 'Vehicle' (Honda). A validation error message box is displayed, stating: 'We hit a snag.' and 'Review the errors on this page.' with the note: 'This vehicle is out of stock. Order cannot be placed.'

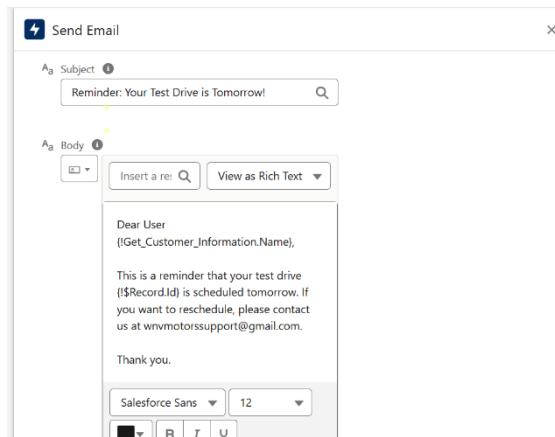
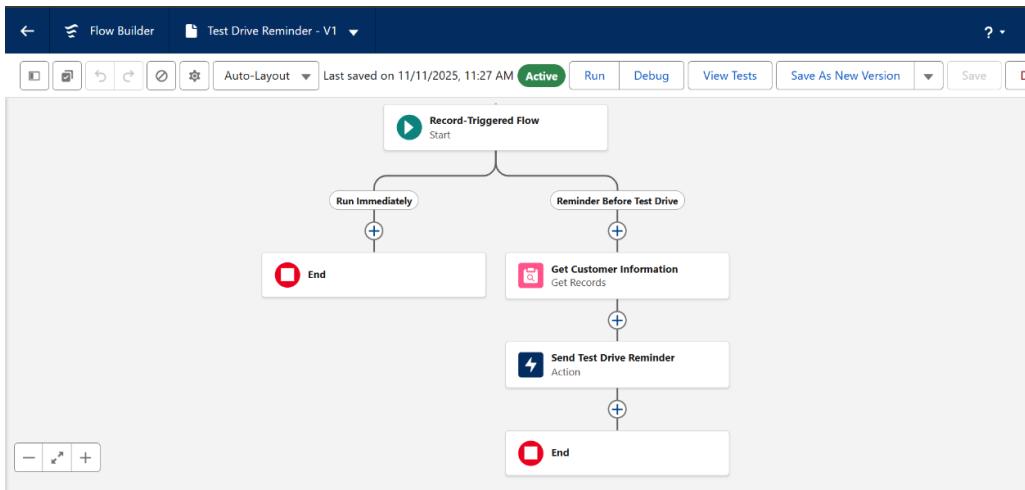
Automation

- **Flows (Record-Triggered)**

- Auto Assign the nearest dealer based on customer's location using record-triggered flow.



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



Apex Classes and Triggers

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

Open		Entities		Related		
Entity Type		Name	Namespace	Name	Extent	Direction
Entity Type						
Classes		DeveloperEditionUtils	devedapp			
Triggers		DeveloperEditionUtilsT...	devedapp			
Pages		PostInstallScript	devedapp			
Page Components		PostInstallScriptTest	devedapp			
Objects		VehicleOrderTriggerHa...				
Static Resources		VehicleOrderBatch				
Packages		VehicleOrderBatchSch...				

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

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

Phase 3: UI/UX Development & Customization

Lightning App Setup through App Manager

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

- Lightning App: *WhatNext Vision Motors*
- Tabs: Vehicles, Vehicle Dealers, Vehicle Customers, Vehicle Orders, Vehicle Test Drives, Vehicle Service Requests
- Used Dynamic Forms to store records

The screenshot shows the WhatNext Vision Motors Lightning App interface. At the top, there's a navigation bar with tabs: Vehicles (which is selected), Vehicle Dealers, Vehicle Customers, Vehicle Orders, Vehicle Test Drives, Vehicle Service Requests, and Reports. Below the navigation bar, the title "Vehicles" is displayed, followed by a "Recently Viewed" section with a dropdown arrow and a refresh icon. A search bar with placeholder text "Search this list..." is located to the right of the recently viewed items. The main content area shows a list of vehicles with checkboxes next to their names: Honda and Mitsubishi. There are also "New", "Import", "Change Owner", and "Assign Label" buttons at the top right of the list area. The bottom of the screen has a footer with standard Salesforce navigation icons.

Page Layouts and Dynamic Forms

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

The screenshot shows the "Vehicle Layout" configuration page in the Salesforce UI builder. The top navigation bar includes "Save", "Quick Save", "Preview As...", "Cancel", "Undo", "Redo", and "Layout Properties". Below the navigation bar, there's a "Fields" sidebar with options like Buttons, Quick Actions, Mobile & Lightning Actions, Expanded Lookups, Related Lists, and Report Charts. The main area displays a "Quick Find" field and a table of fields. The table has three columns: "Field Name", "Owner", and "Vehicle Dealer". The rows are: "Section", "Price", "Vehicle Model"; "Blank Space", "Status", "Vehicle Name"; and "Created By", "Stock Quantity", "Last Modified By". The "Section" row is expanded, showing "Section" and "Blank Space" under the "Field Name" column. The "Vehicle Model" and "Vehicle Name" fields have "Vehicle Dealer" assigned as the owner.

 WhatNext Vision Motors

Vehicle Test Drive Test3

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

 WhatNext Vision Motors

Vehicles Honda

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

 WhatNext Vision Motors

Vehicle Customer John Doe

Related		Details	
Customer Name	John Doe	Owner	 Denise Gabrielle Suarez
Email	dsquare@gmail.com		
Phone	123-56789		
Address	California		
Preferred Vehicle Type	Sedan		
Created By	 Denise Gabrielle Suarez	11/10/2025, 6:42 PM	Last Modified By  Denise Gabrielle Suarez, 11/10/2025, 11:27 PM

 WhatNext Vision Motors

Vehicle Orders O-0012

Related		Details	
Order Number	O-0012	Owner	 Denise Gabrielle Suarez
Vehicle Customer	John Doe		
Vehicle	Honda		
Order Date	11/10/2025		
Status	Pending		
Assigned Dealer	Kristine Martinez		
Created By	 Denise Gabrielle Suarez	11/11/2025, 2:46 AM	Last Modified By  Denise Gabrielle Suarez, 11/11/2025, 2:46 AM

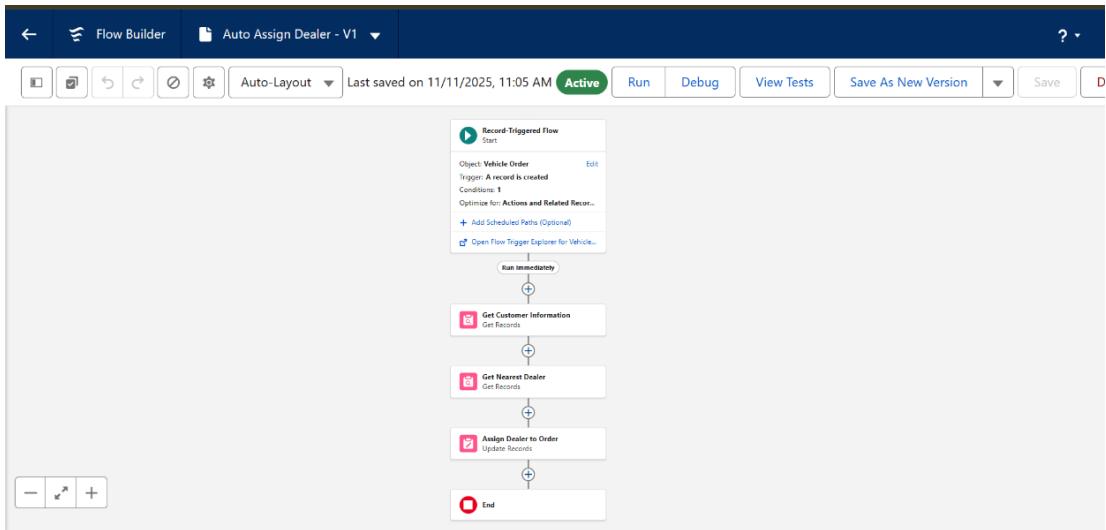
 WhatNext Vision Motors

Vehicle Dealers Kristine Martinez

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

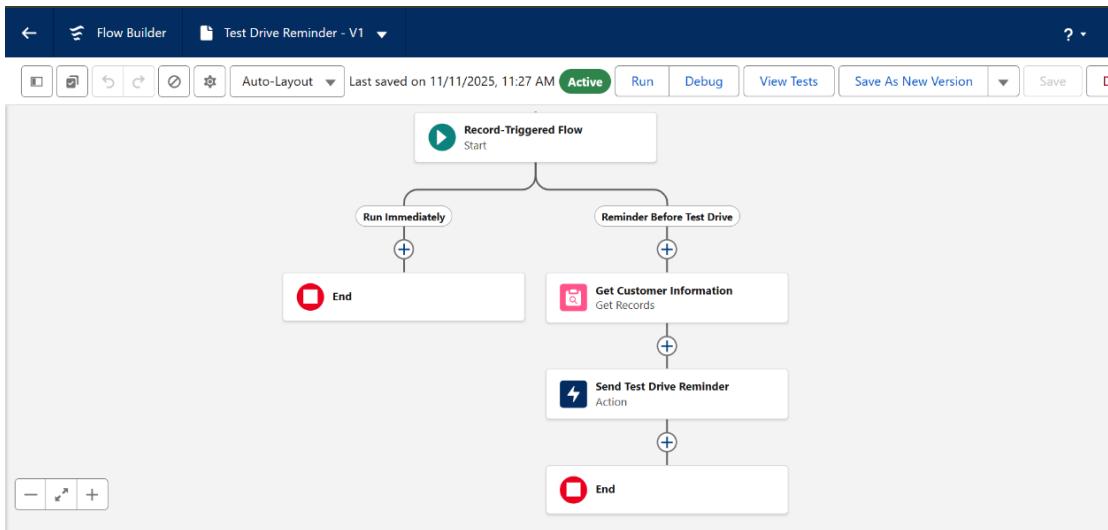
Flow 1: Auto Assign Dealer

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



Flow 2: Test Drive Reminder

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



Apex Trigger and Handler

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

```
File Edit Debug Test Workspace Help < >
VehicleOrderTriggerHandler.apxc [ ] VehicleOrderTrigger.apxc [ ] VehicleOrderBatch.apxc [ ] VehicleOrderBatchScheduler.apxc [ ]
Code Coverage: None API Version: 65

30
31     private static void preventOrderIfOutOfStock(List<Vehicle_Order__c> orders) {
32
33         Set<Id> vehicleIds = new Set<Id>();
34
35         for (Vehicle_Order__c order : orders) {
36
37             if (order.Vehicle__c != null) {
38
39                 vehicleIds.add(order.Vehicle__c);
40
41             }
42
43         }
44
45         if (!vehicleIds.isEmpty()) {
46
47             Map<Id, Vehicle__c> vehicleStockMap = new Map<Id, Vehicle__c>();
48
49             for (Vehicle__c vehicle : [SELECT Id, Stock_Quantity__c FROM Vehicle__c WHERE Id IN :vehicleIds]) {
50
51                 vehicleStockMap.put(vehicle.Id, vehicle);
52
53             }
54
55         }
56     }
57 }

File Edit Debug Test Workspace Help < >
VehicleOrderTriggerHandler.apxc [ ] VehicleOrderTrigger.apxc [ ] VehicleOrderBatch.apxc [ ] VehicleOrderBatchScheduler.apxc [ ]
Code Coverage: None API Version: 65
1 trigger VehicleOrderTrigger on Vehicle_Order__c (before insert,before update, after insert, after update) {
2
3     VehicleOrderTriggerHandler.handleTrigger(Trigger.new, Trigger.oldMap, Trigger.isBefore,Trigger.isAfter, Trigger.isInsert, Trigger.isUpdate);
4
5 }
```

Apex Batch Class

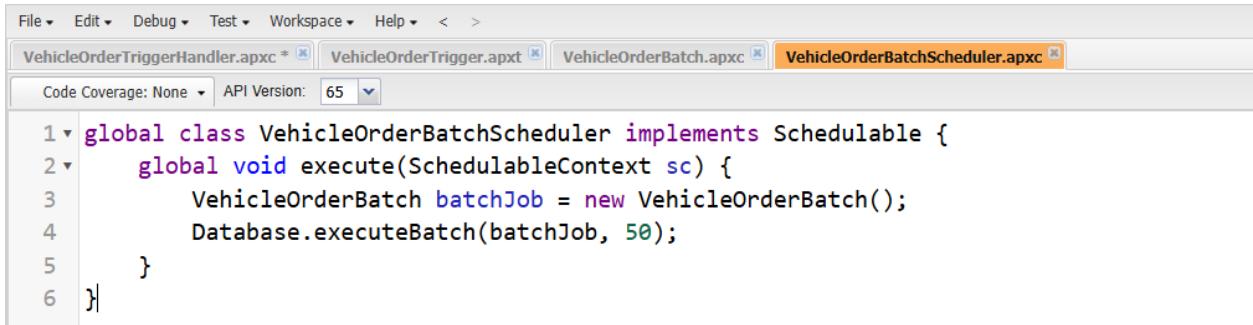
- The class *VehicleOrderBatch* runs daily as it checks for pending orders, available stocks, updates status when it is *Confirmed* and reduces the stock.

```
File Edit Debug Test Workspace Help < >
VehicleOrderTriggerHandler.apxc [ ] VehicleOrderTrigger.apxc [ ] VehicleOrderBatch.apxc [ ] VehicleOrderBatchScheduler.apxc [ ]
Code Coverage: None API Version: 65

1 global class VehicleOrderBatch implements Database.Batchable<sObject> {
2
3     global Database.QueryLocator start(Database.BatchableContext bc) {
4         return Database.getQueryLocator([
5             SELECT Id, Status__c, Vehicle__c FROM Vehicle_Order__c WHERE Status__c = 'Pending'
6         ]);
7     }
8
9     global void execute(Database.BatchableContext bc, List<Vehicle_Order__c> orderList) {
10        Set<Id> vehicleIds = new Set<Id>();
11        for (Vehicle_Order__c order : orderList) {
12            if (order.Vehicle__c != null) {
13                vehicleIds.add(order.Vehicle__c);
14            }
15        }
16
17        if (!vehicleIds.isEmpty()) {
18            Map<Id, Vehicle__c> vehicleStockMap = new Map<Id, Vehicle__c>(
19                [SELECT Id, Stock_Quantity__c FROM Vehicle__c WHERE Id IN :vehicleIds]
20            );
21
22            List<Vehicle_Order__c> ordersToUpdate = new List<Vehicle_Order__c>();
23            for (Vehicle_Order__c order : orderList) {
24                if (order.Vehicle__c != null) {
25                    Vehicle__c vehicle = vehicleStockMap.get(order.Vehicle__c);
26                    if (vehicle != null && vehicle.Stock_Quantity__c > 0) {
27                        vehicleStockMap.put(order.Vehicle__c, vehicle);
28                        if (order.Status__c == 'Pending') {
29                            order.Status__c = 'Confirmed';
30                            ordersToUpdate.add(order);
31                        }
32                    }
33                }
34            }
35
36            Database.update(ordersToUpdate);
37        }
38    }
39
40    global void finish(Database.BatchableContext bc) {
41    }
42
43    global void reExecute(Database.BatchableContext bc) {
44    }
45 }
```

Apex Batch Scheduled

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



A screenshot of the Salesforce IDE interface. The top navigation bar includes File, Edit, Debug, Test, Workspace, Help, and tabs for VehicleOrderTriggerHandler.apxc, VehicleOrderTrigger.apxt, VehicleOrderBatch.apxc, and VehicleOrderBatchScheduler.apxc. The VehicleOrderBatchScheduler.apxc tab is highlighted with a yellow background. Below the tabs, the API Version is set to 65. The code editor displays the following Apex code:

```
1 global class VehicleOrderBatchScheduler implements Schedulable {  
2     global void execute(SchedulableContext sc) {  
3         VehicleOrderBatch batchJob = new VehicleOrderBatch();  
4         Database.executeBatch(batchJob, 50);  
5     }  
6 }
```

Phase 4: Data Migration, Testing & Security

Profiles and Roles

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

Sharing Rules

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

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

1. Create a Vehicle

INPUT:

- **Vehicle Name:** Honda
- **Vehicle Model:** Sedan
- **Stock Quantity:** 1
- **Price:** \$1,500,000
- **Vehicle Dealer:** Kristine Martinez
- **Status:** Available

OUTPUT:

The screenshot shows a Salesforce interface for a vehicle record. The top navigation bar includes links for Vehicles, Vehicle Dealers, Vehicle Customers, Vehicle Orders, Vehicle Test Drives, Vehicle Service Requests, and Reports. The main content area displays a vehicle record for a 'Honda'. The record has the following details:

Field	Value
Vehicle Name	Honda
Vehicle Model	Sedan
Stock Quantity	1
Price	\$1,500,000
Vehicle Dealer	Kristine Martinez
Status	Available
Created By	Denise Gabrielle Suarez
Last Modified By	Denise Gabrielle Suarez

2. Confirmed Order

INPUT:

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

OUTPUT:

The stock became 0.

The screenshot shows the 'Vehicle' details for a Honda. Key information includes:

- Vehicle Name:** Honda
- Owner:** Denise Gabrielle Suarez
- Vehicle Model:** Sedan
- Stock Quantity:** 0
- Price:** \$1,500,000
- Vehicle Dealer:** Kristine Martinez
- Status:** Out of stock
- Created By:** Denise Gabrielle Suarez, 11/10/2025, 6:48 PM
- Last Modified By:** Denise Gabrielle Suarez, 11/11/2025, 2:47 AM

3. Stock = 0 (Create new order)

INPUT:

- **Vehicle:** Honda
- **Status:** Pending
- **Customer:** Jane Air

Prevent to place an order for 0 stock vehicle.

The screenshot shows the 'New Vehicle Order' form. The 'Information' section includes fields for:

- Order Number:** O-0012
- Vehicle Customer:** Jane Air
- Vehicle:** Honda
- Order Date:** 11/18/2025
- Status:** Pending
- Assigned Dealer:** Searchable field

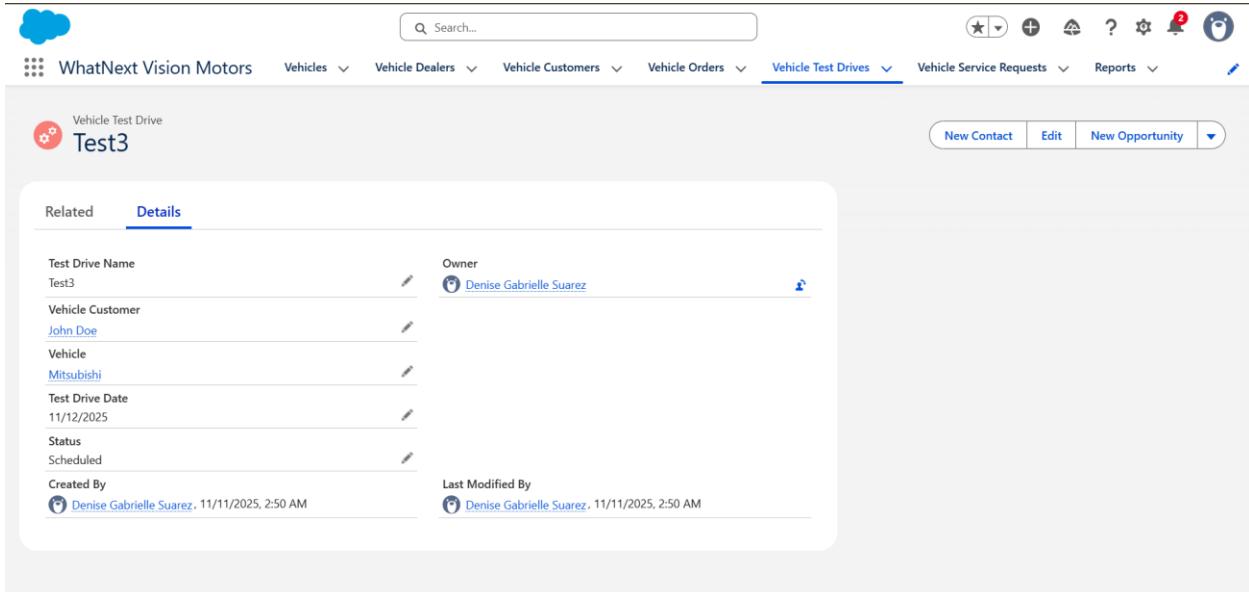
A red error message box is displayed, stating: "We hit a snag. Review the errors on this page. • This vehicle is out of stock. Order cannot be placed." The 'Save' button is highlighted in blue.

4. Test Drive Reminder Email

INPUT:

- **Vehicle Customer:** John Doe
- **Test Drive Date:** Tomorrow (pick tomorrow's date)
- **Status:** Scheduled

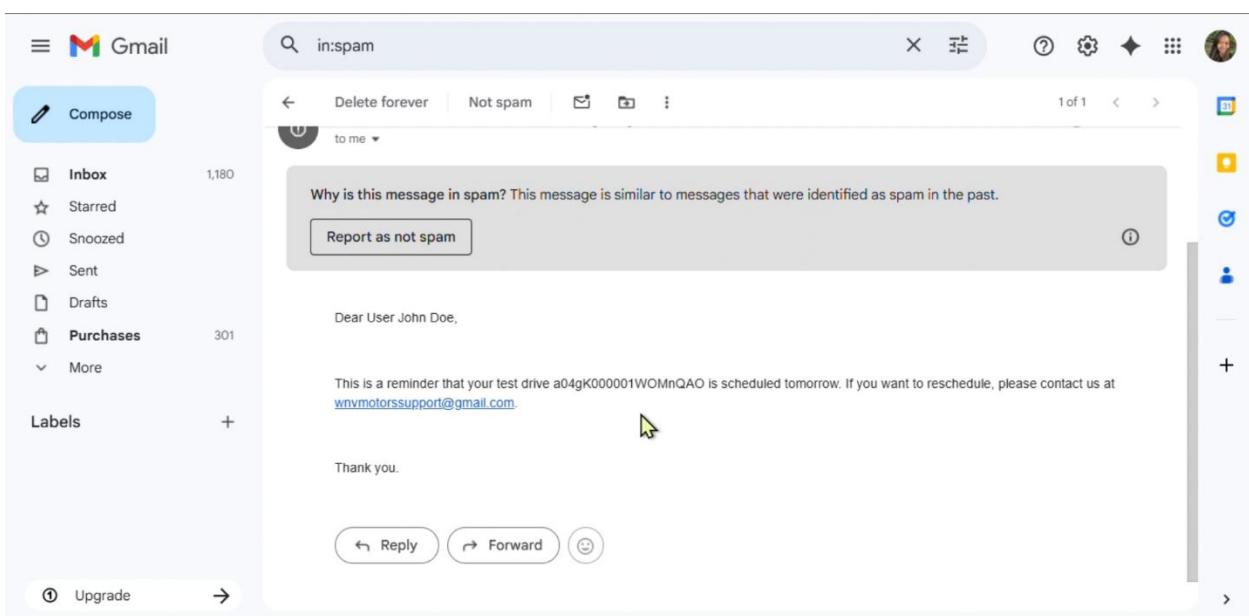
OUTPUT:



The screenshot shows a CRM application interface for 'WhatNext Vision Motors'. The top navigation bar includes links for Vehicles, Vehicle Dealers, Vehicle Customers, Vehicle Orders, Vehicle Test Drives (which is the active tab), and Vehicle Service Requests. Below the navigation is a search bar and a toolbar with icons for New Contact, Edit, and New Opportunity.

The main view displays a 'Vehicle Test Drive' record for 'Test3'. The record details are as follows:

- Test Drive Name: Test3
- Vehicle Customer: John Doe
- Vehicle: Mitsubishi
- Test Drive Date: 11/12/2025
- Status: Scheduled
- Created By: Denise Gabrielle Suarez, 11/11/2025, 2:50 AM
- Owner: Denise Gabrielle Suarez
- Last Modified By: Denise Gabrielle Suarez, 11/11/2025, 2:50 AM



The screenshot shows a Gmail inbox with the search term 'in:spam' applied. The results show a single message from 'wnvmotorssupport@gmail.com' with the subject 'Why is this message in spam? This message is similar to messages that were identified as spam in the past.' A button labeled 'Report as not spam' is visible below the message preview. The message body contains a reminder about a scheduled test drive and contact information.

Phase 5: Deployment, Documentation & Maintenance

Deployment Strategy

The Lightning App was deployed directly in the Salesforce org using App builder. The application doesn't use any external tools.

Deployment Steps:

1. The Lightning App was created and configured using app builder.
2. The app was saved and activated.
3. Visibility settings and profiles were updated to ensure the app appears in the App Launcher.
4. Testing was performed inside the same org to confirm navigation and functionality.

Testing & Sample Scenarios

Test Cases:

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

- Attempting to create a vehicle order with zero stock shouldn't place an order.
- Setting stock to 1 and placing an order should correctly decrease the stock to 1. The current stock is 0.
- Creating a pending order and then updating the stock should allow the batch job to automatically confirm the order

System Maintenance and Monitoring

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

- The app will be reviewed periodically for enhancements.
- Lightning App Builder will be used to update pages or modify layouts.
- Admin tools such as Setup Audit Trail, Flow Error Logs, and User Access Logs will be monitored for issues.
- Profile and Permission Set updates will be applied when new users need access.

Troubleshooting Approach

- Use Lightning App Builder's configuration panel to verify page assignments and visibility rules.
- Check Object Manager for missing permissions or fields.
- Use Debug Logs to investigate errors related to automation (Flows, Triggers, etc.).
- Confirm that the app is assigned to profiles if users cannot see it in the App Launcher.

Conclusion

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

Several Salesforce capabilities were used to support the system's functionality. *Validation rules* were implemented to prevent incorrect or incomplete data entry, such as restricting order creation when stock is zero and the form cannot be saved when there are missing fields. *Approval processes* were set up to ensure the effectiveness of relevant actions like order confirmation. *Automation flows or record-triggered flows* were used and developed to handle key processes, including updating stock, sending reminders, and confirming pending orders through scheduler. Additional components, such as custom objects and Apex classes, were also created to support system operations and data monitoring.

Testing was done in the Salesforce org itself. Each flow was tested by triggering the corresponding action for example, creating or updating a record to confirm field updates, email alerts, and automation behavior. Validation rules were tested by intentionally entering invalid data to ensure the rule triggers the correct error message.

Future Enhancements

1. Customer Portal Integration

- A portal or community where customer can login and view orders.

2. Mobile Application Support

- Extend CRM features to a mobile app to provide customers and dealers easier access to order updates and service information.

3. Reports and Dashboard

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

4. Chatbot and Virtual Assistant Integration

- Add automated customer support to provide real-time answers, guide users through the ordering process, and improve response times.

5. SMS Integration

- Notify customers via SMS about order updates.

6. Financial System Integration

- Connect the system using API payment gateways to streamline billing, invoicing, and loan processing.