

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

## ABSTRACT

The Salesforce CRM implementation project at WhatsNext Vision Motors aims to revolutionize the customer ordering and service experience in the automotive industry. By leveraging Salesforce's robust platform capabilities, the project focuses on improving order accuracy, enhancing customer satisfaction, and increasing operational efficiency. Core functionalities include automatic dealer assignment based on customer location, real-time stock validation, and scheduled automation for bulk order status updates. This initiative is designed to minimize manual intervention, reduce processing delays, and provide a seamless customer journey from inquiry to vehicle delivery.

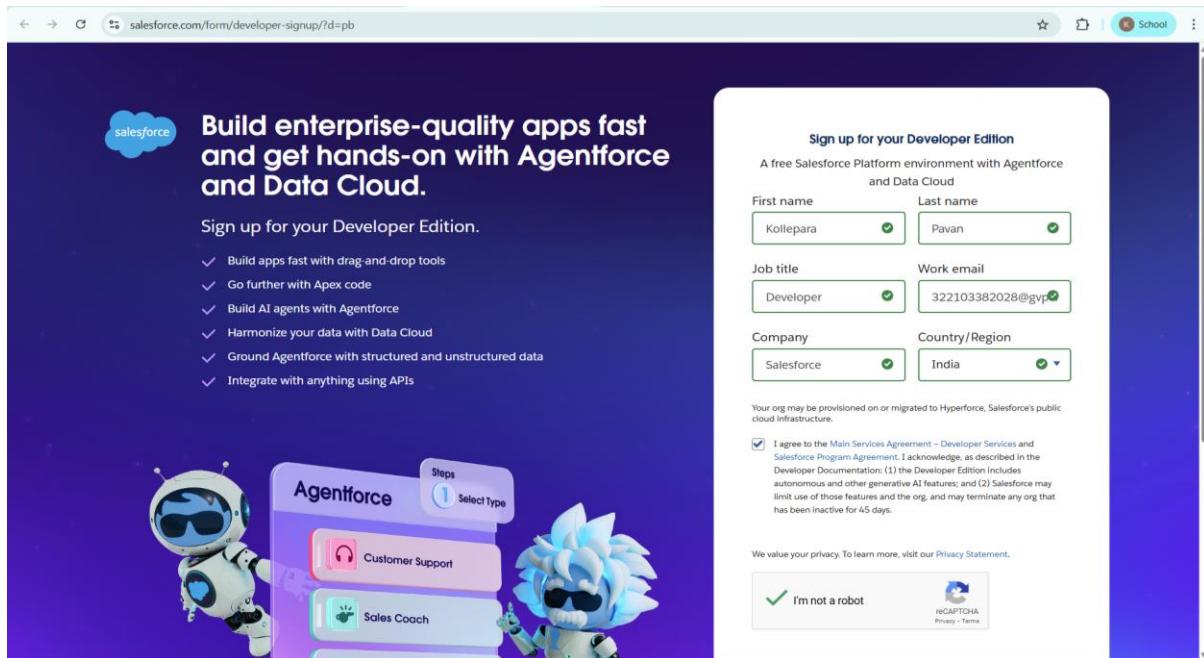
## OBJECTIVE

- To implement Salesforce CRM for centralized management of vehicle inventory, dealer details, customer orders, test drives, and service requests.
- To automate key workflows including:
  - Nearest dealer assignment based on customer address.
  - Order restriction if vehicle stock is unavailable.
  - Email notifications for test drives and stock alerts.
- To develop Apex triggers and trigger handlers that enforce stock validation and automatic dealer assignment logic in a modular and maintainable way.
- To build batch Apex jobs for periodically updating stock levels and sending scheduled notifications for inventory management and order status tracking.
- To improve overall customer satisfaction and reduce administrative overhead through efficient process automation.

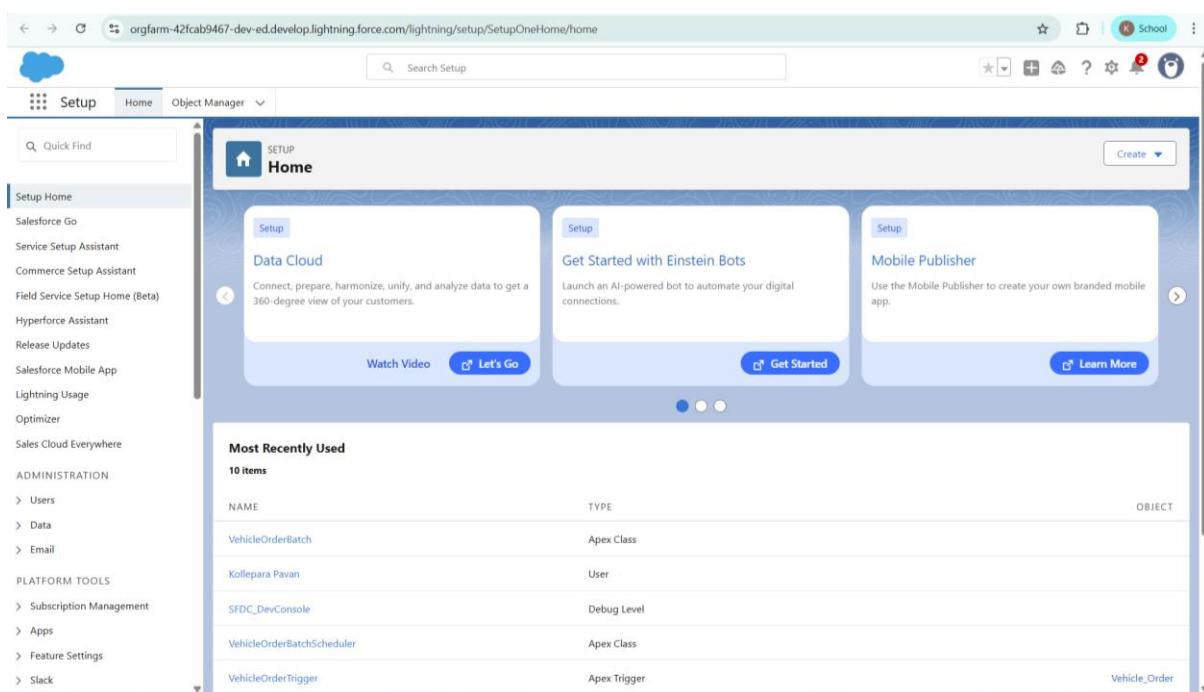
# DETAILED EXECUTION OF PROJECT PHASES

## Epic-1: Salesforce Credentials Creation

I have created a Salesforce Developer Account using my personal details and set up the necessary credentials to begin my project titled "WhatNext Vision Motors: Shaping the Future of Mobility with Innovation and Excellence." This account will serve as the development environment for building and showcasing my work. With the verifier granted access, they can now monitor and evaluate the progress and contributions I make toward this innovative mobility solution.



After Resetting the password login to the trailhead page.



## Epic-2: Data Management-Objects

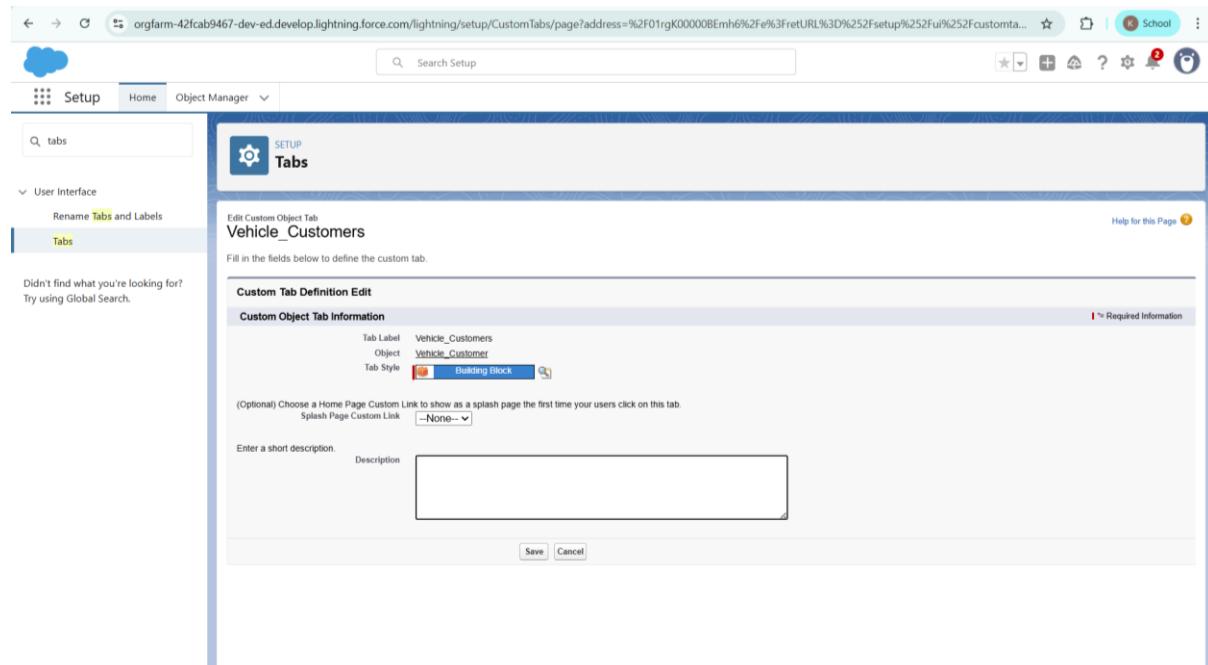
As part of my project "WhatNext Vision Motors: Shaping the Future of Mobility with Innovation and Excellence," I have successfully created six essential custom objects within Salesforce to manage and structure project data effectively. These objects include Vehicle\_\_c for storing vehicle details, Vehicle\_Dealer\_\_c for managing authorized dealer information, Vehicle\_Customer\_\_c for capturing customer records, Vehicle\_Order\_\_c for tracking vehicle purchases, Vehicle\_Test\_Drive\_\_c for handling test drive bookings, and Vehicle\_Service\_Request\_\_c for managing service-related queries. I have established appropriate relationships among these objects to ensure smooth data flow and integration—for example, linking vehicles and dealers to orders, customers to both test drives and orders, and vehicles to service requests. This object structure forms the backbone of the entire project, supporting seamless data management and real-time interaction across all components.

The screenshot displays two screenshots of the Salesforce Object Manager interface. The top screenshot shows the 'Edit Custom Object' page for 'Vehicle'. The 'Custom Object Information' section includes fields for Label (Vehicle), Plural Label (Vehicles), Object Name (Vehicle), and Description (Stores vehicle details). The bottom screenshot shows the 'Object Manager' page listing six custom objects: Vehicle, Vehicle\_Dealer\_\_c, Vehicle\_Customer\_\_c, Vehicle\_Order\_\_c, Vehicle\_Service\_Request\_\_c, and Vehicle\_Test\_Drive\_\_c. The objects are sorted by Label and include columns for Label, API Name, Type, Description, Last Modified, and Deployed status.

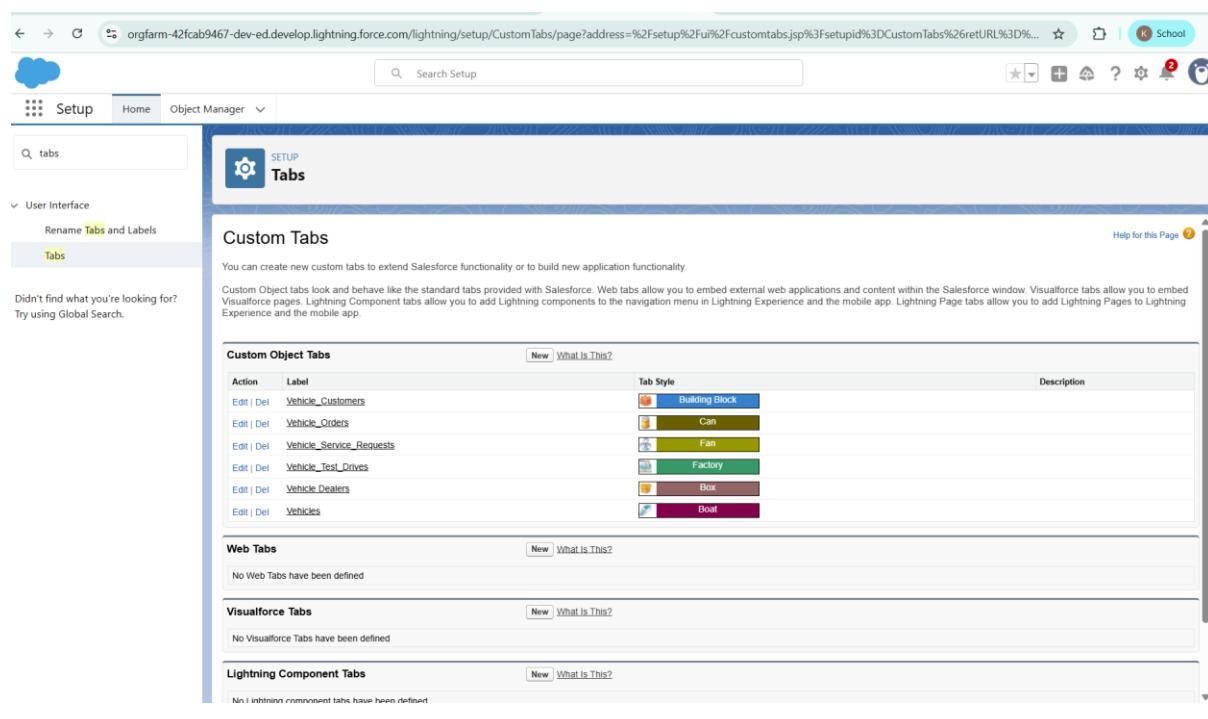
Label	API Name	Type	Description	Last Modified	Deployed
Vehicle	Vehicle__c	Custom Object	Stores vehicle details	7/23/2025	✓
Vehicle Dealer	Vehicle_Dealer__c	Custom Object	Stores authorized dealer info	7/23/2025	✓
Vehicle_Customer	Vehicle_Customer__c	Custom Object	Stores customer details	7/22/2025	✓
Vehicle_Order	Vehicle_Order__c	Custom Object	Tracks vehicle purchases	7/22/2025	✓
Vehicle_Service_Request	Vehicle_Service_Request__c	Custom Object	Tracks vehicle servicing requests	7/23/2025	✓
Vehicle_Test_Drive	Vehicle_Test_Drive__c	Custom Object	Tracks test drive bookings	7/23/2025	✓

## Epic-3: Data Management-Tabs

To enhance accessibility and usability within the Salesforce environment, I created custom tabs for all six custom objects used in the project: Vehicle\_\_c, Vehicle\_Dealer\_\_c, Vehicle\_Customer\_\_c, Vehicle\_Order\_\_c, Vehicle\_Test\_Drive\_\_c, and Vehicle\_Service\_Request\_\_c. These tabs allow users to easily navigate, view, and manage records associated with each object directly from the Salesforce interface. By setting up these custom tabs, I ensured that all relevant data is readily available for stakeholders, streamlining the user experience and improving overall efficiency in managing vehicle-related processes within the project.



The screenshot shows the Salesforce Setup interface under the 'User Interface' section, specifically the 'Tabs' page. A new custom tab is being created for the 'Vehicle\_Customer\_\_c' object. The tab is labeled 'Vehicle\_Customers'. The 'Tab Style' is set to 'Building Block'. There is a note about choosing a splash page custom link. A description field is present but empty. The 'Save' button is visible at the bottom.

The screenshot shows the 'Custom Tabs' page in the Salesforce Setup interface. It lists various custom tabs categorized by type: Action, Label, Tab Style, and Description. The tabs include:

Action	Label	Tab Style	Description
Edit   Del	Vehicle_Customers	Building Block	
Edit   Del	Vehicle_Orders	Can	
Edit   Del	Vehicle_Service_Requests	Fan	
Edit   Del	Vehicle_Test_Drives	Factory	
Edit   Del	Vehicle_Dealers	Box	
Edit   Del	Vehicles	Boat	

Below this, sections for 'Web Tabs', 'Visualforce Tabs', and 'Lightning Component Tabs' show that no tabs have been defined for these categories.

## Epic-4: Data Management-App Manager

As part of organizing and streamlining the user experience in my Salesforce project, I created a Lightning App named "WhatNext Vision Motors" to serve as a centralized workspace for managing all aspects of the vehicle business process. This app integrates all key components of the project—including Vehicles, Dealers, Customers, Orders, Test Drives, and Service Requests—into a single, user-friendly interface. I included essential navigation items and custom objects to ensure quick access to relevant records and functionalities. Additionally, I assigned the System Administrator profile to ensure full access and control over the app's features. This app provides a cohesive and efficient way for users to interact with the system, improving workflow and data visibility across all departments involved in the mobility solution.

The screenshot shows the Lightning App Builder interface with the URL [https://orgfarm-42fcab9467-dev-ed.lightning.force.com/visualEditor/appBuilder.app?id=02ugK000004Uie1QAC&retUrl=https%3A%2F%2Forgfarm-42fcab9467-dev-ed.lightning.force.com/lightning/o/Vehicle\\_c/list?filterName=\\_Recent](https://orgfarm-42fcab9467-dev-ed.lightning.force.com/visualEditor/appBuilder.app?id=02ugK000004Uie1QAC&retUrl=https%3A%2F%2Forgfarm-42fcab9467-dev-ed.lightning.force.com/lightning/o/Vehicle_c/list?filterName=_Recent). The left sidebar shows 'App Settings' with sections for 'App Details & Branding', 'App Options', and 'Utility Items (Desktop Only)'. The main area is titled 'Navigation Items' with a sub-instruction: 'Choose the items to include in the app, and arrange the order in which they appear. Users can personalize the navigation to add or move items, but users can't remove or rename the items that you add. Some navigation items are available only for phone or only for desktop. These items are dropped from the navigation bar when the app is viewed in a format that the item doesn't support.' On the left, under 'Available Items', there is a list of various Salesforce objects and components. On the right, under 'Selected Items', the 'Vehicles' object is listed. Navigation arrows between the two lists allow for moving items between them.

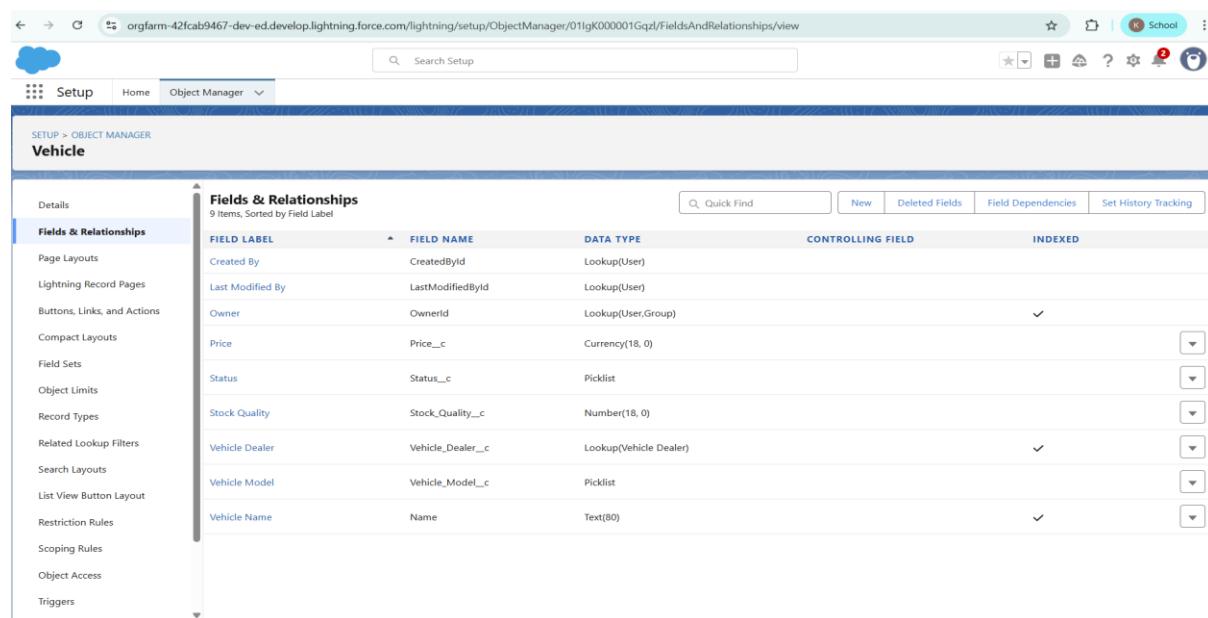
The screenshot shows the 'Vehicles' list view within the 'WhatNext Vision Motors' app. The URL is [https://orgfarm-42fcab9467-dev-ed.lightning.force.com/lightning/o/Vehicle\\_c/list?filterName=\\_Recent](https://orgfarm-42fcab9467-dev-ed.lightning.force.com/lightning/o/Vehicle_c/list?filterName=_Recent). The top navigation bar includes links for 'Vehicles', 'Vehicle Dealers', 'Vehicle\_Customers', 'Vehicle\_Orders', 'Vehicle\_Test\_Drives', 'Vehicle\_Service\_Requests', 'Reports', 'Dashboards', and 'Setup'. A 'Recently Viewed' section on the left shows 0 items updated 16 minutes ago. The main list area has a search bar and a toolbar with 'New', 'Import', 'Change Owner', and 'Assign Label' buttons. Below the toolbar is another search bar and a set of icons for 'New', 'Import', 'Change Owner', 'Assign Label', and other actions. A decorative graphic of a landscape with clouds and mountains is centered above the list. The message 'Nothing to see here' is displayed, followed by the sub-instruction: 'There's nothing in your list yet. Try adding a new record.'

## Epic-5: Data Management-Fields

As part of building the "**WhatNext Vision Motors**" Salesforce application, I created six custom objects with all the necessary fields to handle business operations like vehicle management, customer tracking, dealer coordination, orders, test drives, and service requests. Each object was carefully structured with appropriate field types such as Text, Picklist, Lookup, Date, Currency, and Auto Number to ensure accurate and efficient data storage.

### 1. Vehicle\_\_c:

Vehicle\_Name\_\_c, Vehicle\_Model\_\_c, Stock\_Quantity\_\_c, Price\_\_c, Dealer\_\_c, Status\_\_c

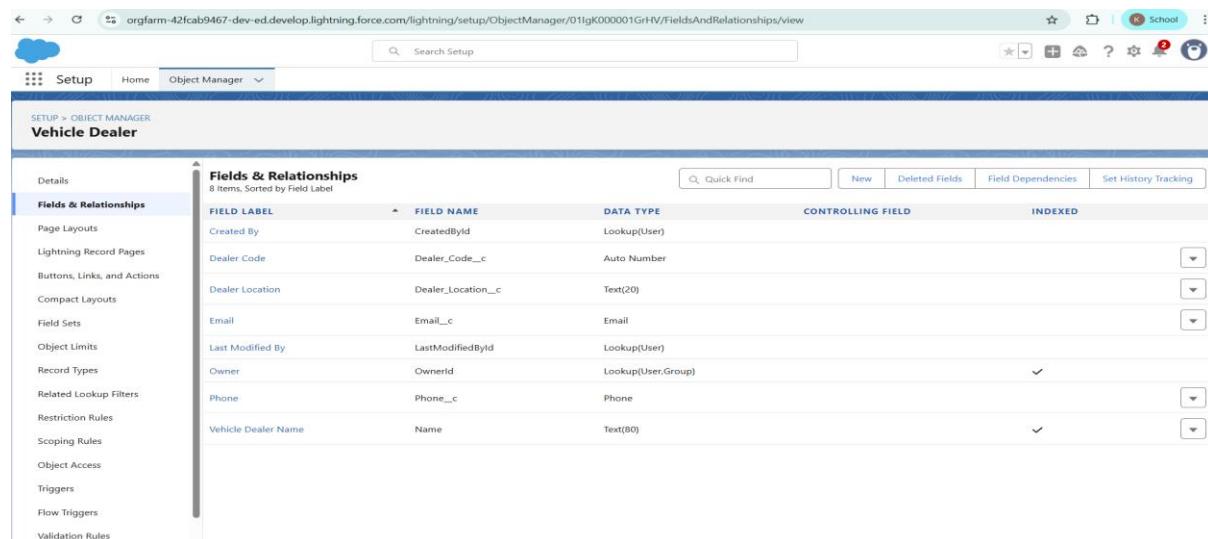


The screenshot shows the Salesforce Object Manager interface for the 'Vehicle' object. The left sidebar lists various setup options like Page Layouts, Lightning Record Pages, and Field Sets. The main content area is titled 'Fields & Relationships' and displays a table of fields. The table has columns for FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED. The fields listed are: Created By (CreatedBy), Last Modified By (LastModifiedBy), Owner (OwnerId), Price (Price\_\_c), Status (Status\_\_c), Stock Quality (Stock\_Quality\_\_c), Vehicle Dealer (Vehicle\_Dealer\_\_c), Vehicle Model (Vehicle\_Model\_\_c), and Vehicle Name (Name). Most fields are of type 'Lookup', except for Price which is 'Currency', Status which is 'Picklist', and Stock Quality which is 'Number'.

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

### 2. Vehicle\_Dealer\_\_c:

Dealer\_Name\_\_c, Dealer\_Location\_\_c, Dealer\_Code\_\_c, Phone\_\_c, Email\_\_c



The screenshot shows the Salesforce Object Manager interface for the 'Vehicle Dealer' object. The left sidebar lists various setup options. The main content area is titled 'Fields & Relationships' and displays a table of fields. The table has columns for FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED. The fields listed are: Created By (CreatedBy), Dealer Code (Dealer\_Code\_\_c), Dealer Location (Dealer\_Location\_\_c), Email (Email\_\_c), Last Modified By (LastModifiedBy), Owner (OwnerId), Phone (Phone\_\_c), and Vehicle Dealer Name (Name). Most fields are of type 'Lookup', except for Dealer Code which is 'Auto Number', Email which is 'Email', and Phone which is 'Phone'.

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedBy	Lookup(User)		
Dealer Code	Dealer_Code__c	Auto Number		
Dealer Location	Dealer_Location__c	Text(20)		
Email	Email__c	Email		
Last Modified By	LastModifiedBy	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓
Phone	Phone__c	Phone		
Vehicle Dealer Name	Name	Text(80)		✓

### **3. Vehicle\_Order\_c:**

Customer\_c, Vehicle\_c, Order\_Date\_c, Status\_c

The screenshot shows the Salesforce Object Manager interface for the 'Vehicle Order' object. The left sidebar lists various setup options like Page Layouts, Lightning Record Pages, and Field Sets. The main content area is titled 'Fields & Relationships' and displays eight items sorted by field label. The table includes columns for Field Label, Field Name, Data Type, Controlling Field, and Indexed status.

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
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)		✓

### **4. Vehicle\_Customer\_c:**

Customer\_Name\_c, Email\_c, Phone\_c, Address\_c, Preferred\_Vehicle\_Type\_c

The screenshot shows the Salesforce Object Manager interface for the 'Vehicle\_Customer' object. The left sidebar lists various setup options. The main content area is titled 'Fields & Relationships' and displays nine items sorted by field label. The table includes columns for Field Label, Field Name, Data Type, Controlling Field, and Indexed status.

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Address	Address_c	Text(100)		
Created By	CreatedById	Lookup(User)		
Customer Name	Customer_Name_c	Text(25)		
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)		✓

## 5. Vehicle\_Test\_Drive\_c:

Customer\_c, Vehicle\_c, Test\_Drive\_Date\_c, Status\_c

The screenshot shows the Salesforce Object Manager for the 'Vehicle\_Test\_Drive' object. The left sidebar has a 'Fields & Relationships' section selected. The main area displays a table titled 'Fields & Relationships' with 8 items, sorted by Field Label. The columns are FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED.

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)		✓

## 6. Vehicle\_Service\_Request\_c:

Customer\_c, Vehicle\_c, Service\_Date\_c, Issue\_Description\_c

The screenshot shows the Salesforce Object Manager for the 'Vehicle\_Service\_Request' object. The left sidebar has a 'Fields & Relationships' section selected. The main area displays a table titled 'Fields & Relationships' with 9 items, sorted by Field Label. The columns are FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED.

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
Issue Description	Issue_Description__c	Text(100)		
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)		✓

These custom fields enable full tracking of all interactions and transactions within the system. By designing them logically and linking them through relationships, the app provides a seamless experience for managing every aspect of the vehicle sales and service lifecycle.

## Epic-6: Automation

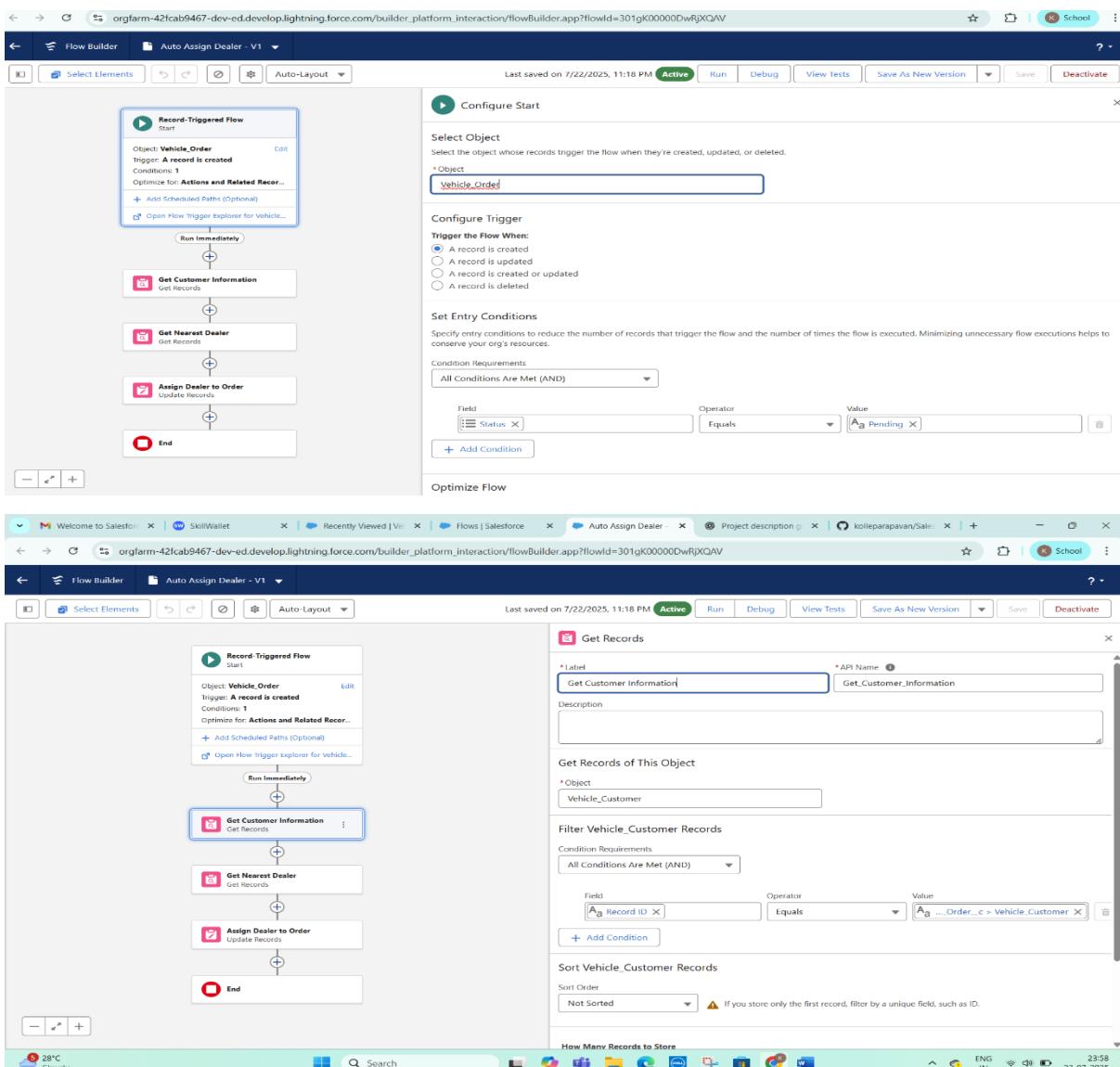
As part of automating key processes in the "WhatNext Vision Motors" Salesforce application, I created two **Record-Triggered Flows** to enhance efficiency and customer engagement.

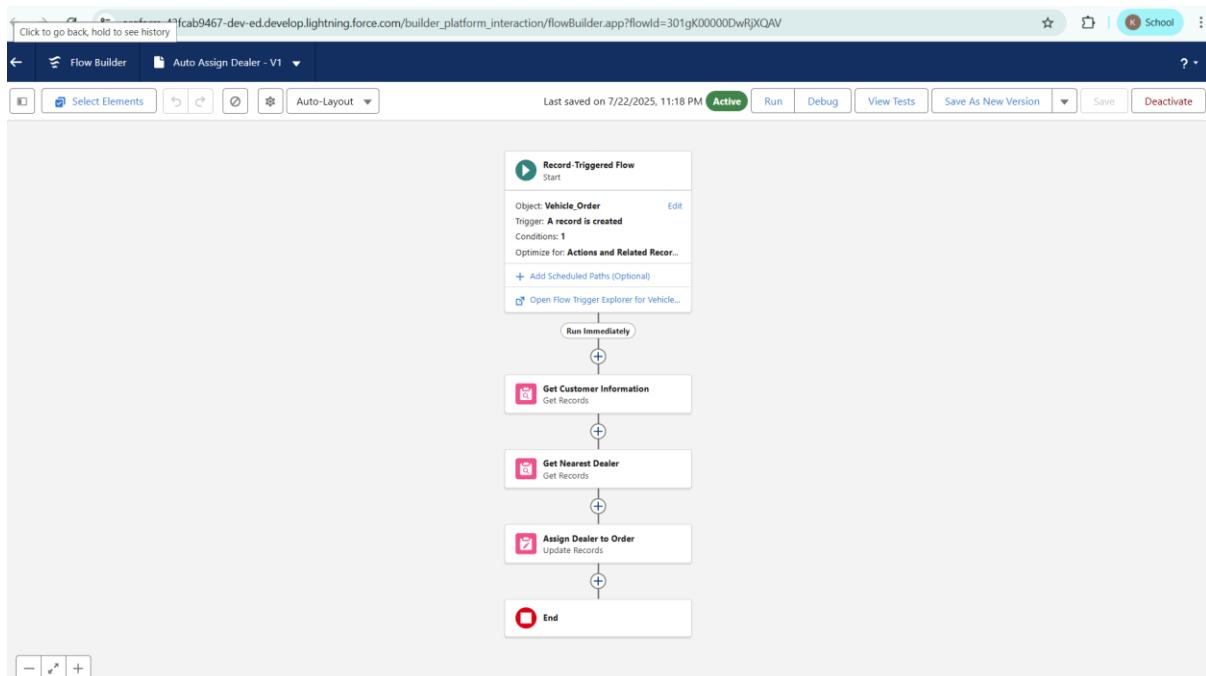
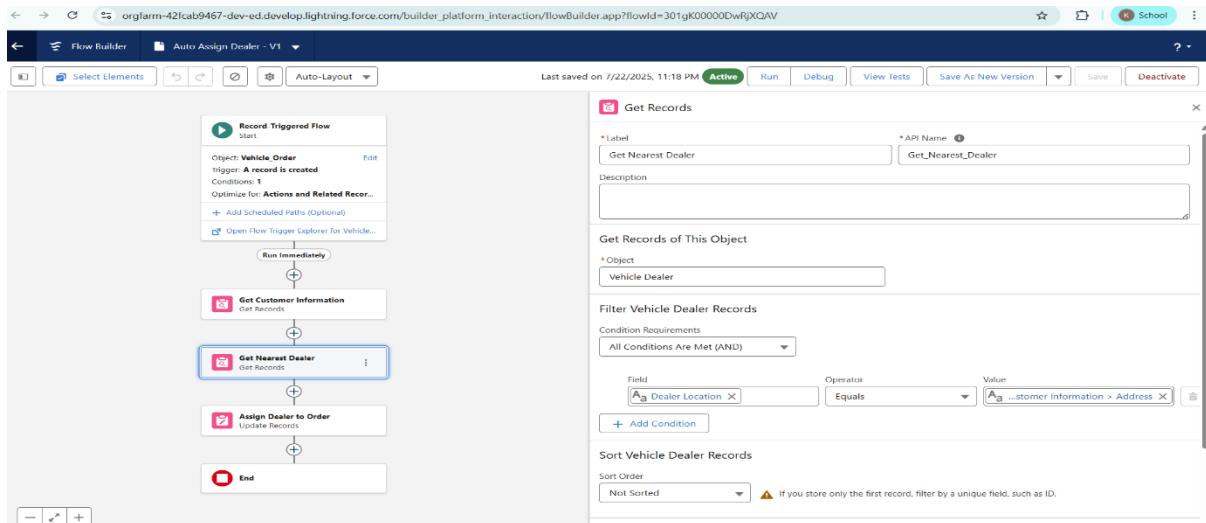
### 1. Assign Nearest Dealer to Customer:

This flow is triggered when a new customer record is created. It automatically identifies and assigns the nearest authorized dealer based on the customer's location. This ensures a more personalized experience and helps in streamlining the order and service process by connecting customers with the most relevant dealer.

### 2. Send Test Drive Reminder Email:

This flow is triggered when a new test drive record is created. It sends an automated email reminder to the customer with details about their scheduled test drive. This improves communication, reduces no-shows, and ensures that customers are well-informed and engaged with the brand.





## Tested the flow by creating the customer

"Reminder: Your Test Drive is Tomorrow!"

Kollepara Pavan via mu1jwcidy1f868.gk-thnlfu0.aon96.bnc.salesforce.com

Wed, Jul 23, 12:47PM (11 hours 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.

Report spam   Looks safe

Dear pavan,

This is a reminder that your test drive for the a02gK0000037iiXQAQ is scheduled for tomorrow at.

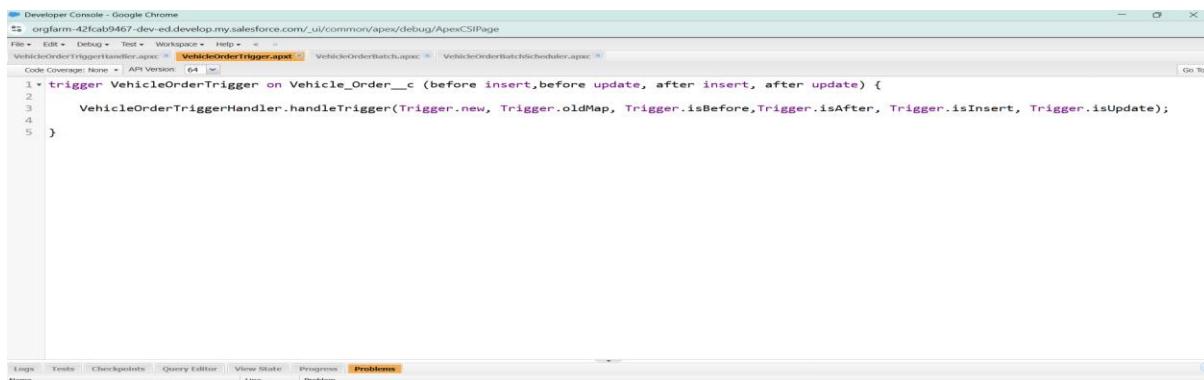
We look forward to seeing you!

## Epic-7: Apex and Batch Class

To automate the **vehicle availability tracking and order processing workflow** in the *WhatNext Vision Motors* project, I implemented Apex-based automation using **triggers, handler classes, batch jobs, and schedulers**. These components ensure that vehicle orders are processed automatically and efficiently on a daily basis, reducing manual intervention and increasing accuracy.

### ◆ VehicleOrderTrigger

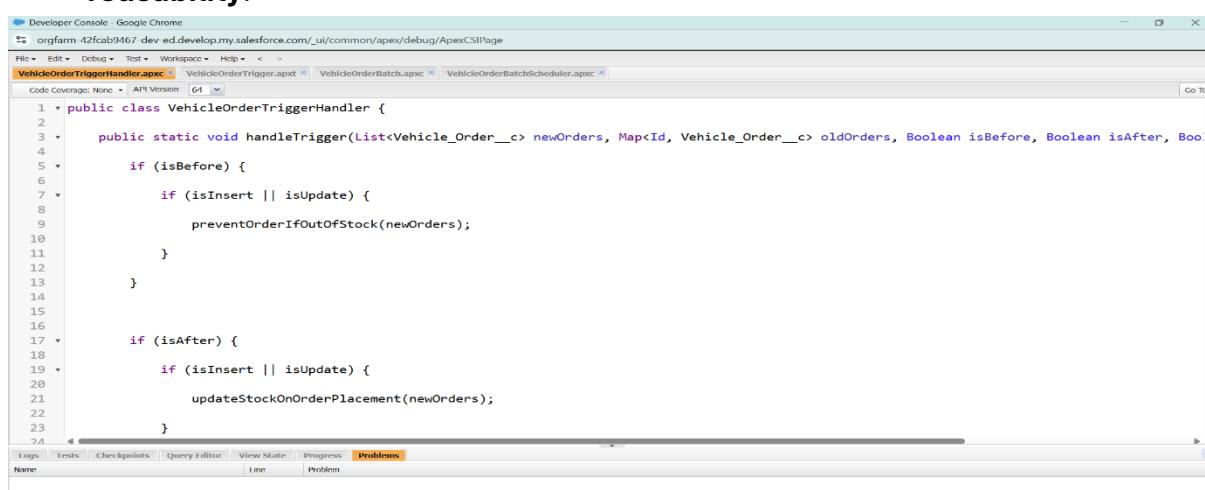
- This is the **Apex trigger** written on the `Vehicle_Order__c` object.
- It is responsible for executing logic whenever a vehicle order is created or updated.
- The trigger delegates its logic to the handler class to keep the code modular and maintainable.



```
Developer Console - Google Chrome
File Edit Debug Test Workspaces Help
VehicleOrderTriggerHandler.apxc VehicleOrderBatch.apxc VehicleOrderBatchScheduler.apxc
Code Coverage: None API Version: 64
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 }
```

### ◆ VehicleOrderTriggerHandler

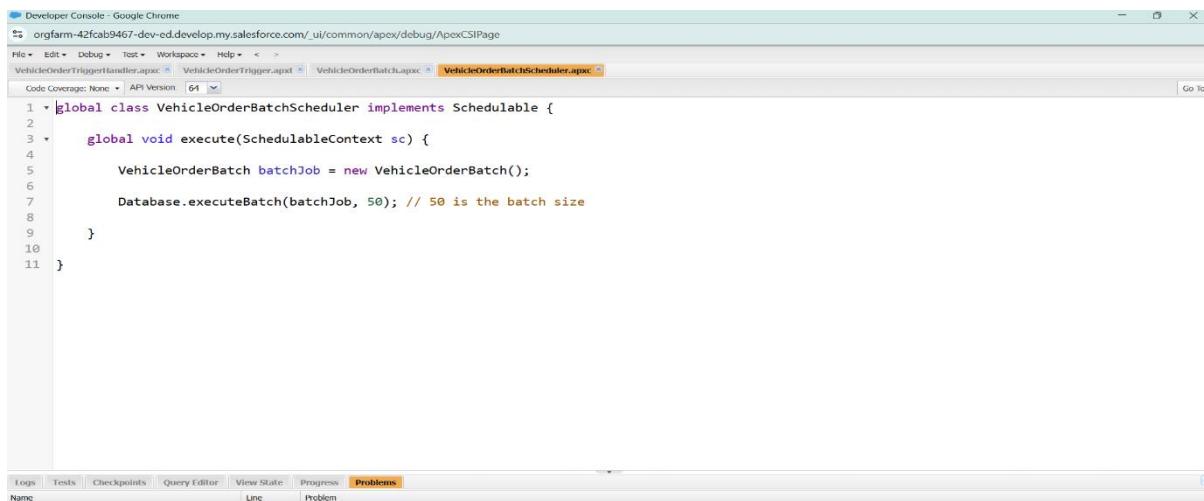
- This class contains the **business logic** related to the `VehicleOrderTrigger`.
- It checks vehicle availability, updates order status, and manages relationships with vehicles and dealers.
- Keeping logic in this separate handler promotes **clean code architecture** and **reusability**.



```
Developer Console - Google Chrome
File Edit Debug Test Workspaces Help
VehicleOrderTriggerHandler.apxc VehicleOrderTrigger.apxc VehicleOrderBatch.apxc VehicleOrderBatchScheduler.apxc
Code Coverage: None API Version: 61
1 public class VehicleOrderTriggerHandler {
2
3     public static void handleTrigger(List<Vehicle_Order__c> newOrders, Map<Id, Vehicle_Order__c> oldOrders, Boolean isBefore, Boolean isAfter, Boolean
4
5         if (isBefore) {
6
7             if (isInsert || isUpdate) {
8
9                 preventOrderIfOutOfStock(newOrders);
10
11             }
12
13         }
14
15
16         if (isAfter) {
17
18             if (isInsert || isUpdate) {
19
20                 updateStockOnOrderPlacement(newOrders);
21
22             }
23
24     }
25 }
```

## ◆ VehicleOrderBatch

- This is a **batch class** that processes large numbers of vehicle orders in bulk.
- It queries all pending vehicle orders and performs actions such as assigning vehicles, updating availability, or marking orders as processed.
- Batch processing is used here to handle **large data volumes efficiently** and without hitting governor limits.



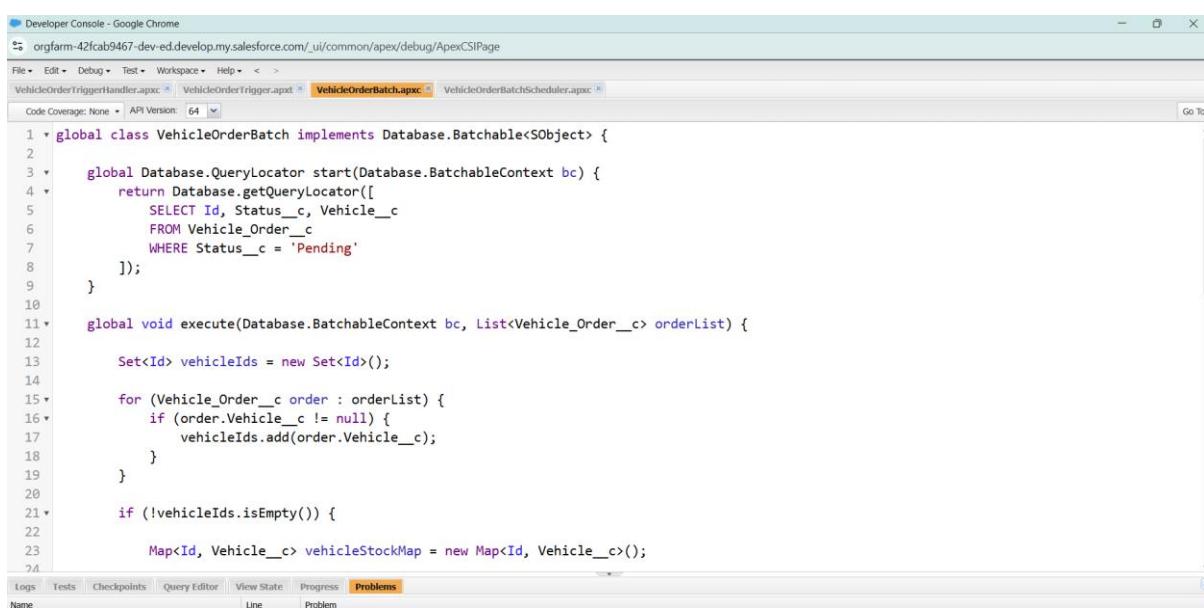
The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is orgfarm-42fcab9467-dev-ed.develop.my.salesforce.com/\_ui/common/apex/debug/ApexCSIPage. The tabs at the top are VehicleOrderTriggerHandler.apxc, VehicleOrderTrigger.apxc, VehicleOrderBatchScheduler.apxc (which is the active tab), and VehicleOrderBatch.apxc. The code editor contains the following Apex code:

```
1 * global class VehicleOrderBatchScheduler implements Schedulable {
2
3     global void execute(SchedulableContext sc) {
4
5         VehicleOrderBatch batchJob = new VehicleOrderBatch();
6
7         Database.executeBatch(batchJob, 50); // 50 is the batch size
8
9     }
10 }
```

The bottom navigation bar shows tabs for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Problems tab is selected. There are no visible errors or problems in the code.

## ◆ VehicleOrderBatchScheduler

- This class is used to **schedule the batch job to run automatically** at a specific time.
- It leverages the System.schedule method to ensure the VehicleOrderBatch job runs every day without manual triggering.



The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is orgfarm-42fcab9467-dev-ed.develop.my.salesforce.com/\_ui/common/apex/debug/ApexCSIPage. The tabs at the top are VehicleOrderTriggerHandler.apxc, VehicleOrderTrigger.apxc, and VehicleOrderBatch.apxc (which is the active tab). The code editor contains the following Apex code:

```
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
6             FROM Vehicle_Order__c
7             WHERE Status__c = 'Pending'
8         ]);
9     }
10
11     global void execute(Database.BatchableContext bc, List<Vehicle_Order__c> orderList) {
12
13         Set<Id> vehicleIds = new Set<Id>();
14
15         for (Vehicle_Order__c order : orderList) {
16             if (order.Vehicle__c != null) {
17                 vehicleIds.add(order.Vehicle__c);
18             }
19         }
20
21         if (!vehicleIds.isEmpty()) {
22
23             Map<Id, Vehicle__c> vehicleStockMap = new Map<Id, Vehicle__c>();
24         }
25     }
26 }
```

The bottom navigation bar shows tabs for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Problems tab is selected. There are no visible errors or problems in the code.

## ◆ Scheduled Job Monitoring

- Once the scheduler is deployed, you can monitor the scheduled job from **Setup > Scheduled Jobs** in Salesforce.
- This allows tracking of batch job execution, checking for errors, and ensuring it runs as intended.

The screenshot shows the Salesforce Setup interface with the 'Scheduled Jobs' page selected. The left sidebar shows various setup categories like Data, Feature Settings, and Objects and Fields. The main content area displays a table of scheduled jobs with columns for Action, Job Name, Submitted By, Submitted, Started, Next Scheduled Run, Type, and Cron Trigger ID. A message bar at the top indicates that 1% of scheduled Apex jobs have been used out of a limit of 100.

Action	Job Name *	Submitted By	Submitted	Started	Next Scheduled Run	Type	Cron Trigger ID
Manage   Del   Pause Job	Daily Vehicle Order Processing	Eavan_Kollepara	7/23/2025, 1:17 AM		7/23/2025, 12:00 PM	Scheduled Apex	06egk0000008093wV
Del	Metalytics Data Loader Job for Org : 00Dgk000007HnLF	User_Integration	7/12/2025, 7:11 PM	7/23/2025, 3:53 AM	7/24/2025, 3:53 AM	Autonomous Data Loader Job	06egk0000007FwvVu
	Program Milestone Computation Cron Job	Process_Automated	7/12/2025, 7:11 PM	7/23/2025, 7:00 AM	7/23/2025, 11:59 AM	Program Milestone Computation Cron Job	06egk0000007FwvVu
	Program Status Update Cron Job	Process_Automated	7/12/2025, 7:11 PM	7/23/2025, 5:00 AM	7/23/2025, 8:00 PM	Program Status Update Cron Job	06egk0000007FwvVu

# **Project Demonstration – Functional Walkthrough with Sample Data**

## **Sample Data Setup:**

### **Vehicle Customers:**

- Customer-1
  - Customer Name : Pavan
  - Email : [322103382028@gvpce.ac.in](mailto:322103382028@gvpce.ac.in)
  - Phone : 6303643871
  - Address : Visakhapatnam
- Customer-2
  - Customer Name : Madhu
  - Email : abc@gmail.com
  - Phone : 6303643871
  - Address : Hyderabad

### **Vehicle Dealer:**

- Dealer-1
  - Dealer Name : Satya
  - Dealer Location : Visakhapatnam
  - Phone : 6303643871
  - Email : abc@gmail.com
- Dealer-2
  - Dealer Name : Naga
  - Dealer Location : Hyderabad
  - Phone : 6303643871
  - Email : xyz@gmail.com

### **Vehicles:**

- Vehicle-1
  - Vehicle Name : RX\_100
  - Vehicle Model : SUV
  - Stock Quantity : 5
  - Vehicle Dealer : Madhu
  - Price : 500000
  - Status : Available
- Vehicle-2
  - Vehicle Name : AUDI
  - Vehicle Model : Sedan

- Stock Quantity :0
- Vehicle Dealer :Naga
- Price :1000000
- Status :Out of Stock

Vehicle Orders:

- Vehicle-1
  - Vehicle Order Name :car
  - Vehicle Customer :Pavan
  - Vehicle :RX\_100
  - Order\_Date :7/28/2025
  - Status :Pending
- Vehicle-1
  - Vehicle Order Name :Bike
  - Vehicle Customer :Pavan
  - Vehicle :RX\_100
  - Order\_Date :7/29/2025
  - Status :Pending

## Flow Execution

After the above data is created, here's what happens automatically:

### Dealer Assignment Flow:

When a new Vehicle Order is created with status Pending, the flow:

- Fetches the selected Vehicle\_\_c
- Retrieves its associated Dealer\_\_c
- Assigns that dealer to the Vehicle\_Order\_\_c (via lookup)
- Status remains Pending until confirmed

### Trigger/Batch Apex:

If a vehicle is out of stock, order cannot be placed (error shown via addError() in trigger)

- When the Status\_\_c of order changes to Confirmed, the Trigger or Batch Apex:
  - Reduces the vehicle stock by 1
  - Prevents over-ordering
  - Keeps vehicle stock synced

## Before the Flow executed the Status is Pending:

The screenshot shows a Salesforce Lightning interface for a vehicle order. The URL is [https://orgfarm-42fcab9467-dev-ed.lightning.force.com/lightning/r/Vehicle\\_Order\\_c/a03gK000005KklnQAS/view](https://orgfarm-42fcab9467-dev-ed.lightning.force.com/lightning/r/Vehicle_Order_c/a03gK000005KklnQAS/view). The page title is "Vehicle Order". The "Status" field is highlighted with a yellow box and contains the value "Pending". Other fields visible include "Vehicle Order Name" (car), "Vehicle Customer" (pavan), "Vehicle" (RX\_100), and "Order Date" (7/28/2025). The "Owner" is listed as "Kollepara Pavan". The "Created By" and "Last Modified By" fields both show "Kollepara Pavan" with the timestamp "7/25/2025, 4:21 AM".

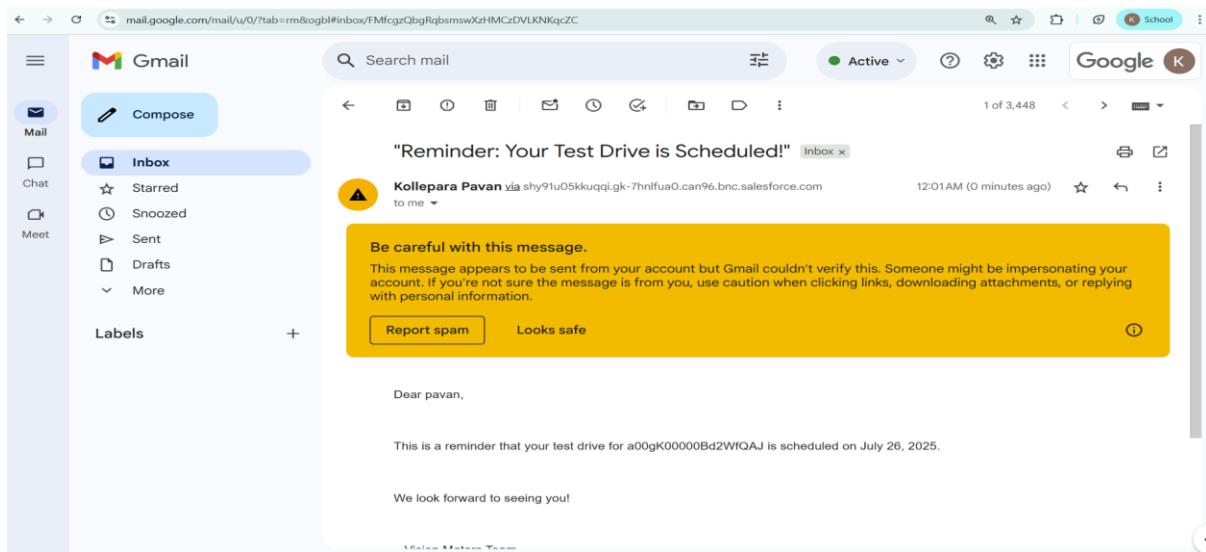
## After the Execution of Flow

The screenshot shows the same Salesforce Lightning interface after the flow has executed. The "Status" field is now highlighted with a yellow box and contains the value "Confirmed". All other fields remain the same: "Vehicle Order Name" (car), "Vehicle Customer" (pavan), "Vehicle" (RX\_100), "Order Date" (7/28/2025), "Owner" (Kollepara Pavan), and the "Created By" and "Last Modified By" fields both show "Kollepara Pavan" with the timestamp "7/25/2025, 11:20 AM".

Here Audi car is Out of stock.lets see the order is creating or not:

The screenshot shows a Salesforce Lightning interface for a vehicle record. The URL is [https://orgfarm-42fcab9467-dev-ed.lightning.force.com/lightning/r/Vehicle\\_c/a00gK00000BcvJQAZ/view](https://orgfarm-42fcab9467-dev-ed.lightning.force.com/lightning/r/Vehicle_c/a00gK00000BcvJQAZ/view). The page title is "Vehicle". The "Status" field is highlighted with a yellow box and contains the value "Out of Stock". Other fields visible include "Vehicle Name" (Audi), "Vehicle Model" (Sedan), "Stock Quality" (0), "Price" (\$1,000,000), and the "Created By" field showing "Kollepara Pavan" with the timestamp "7/25/2025, 3:52 AM". The "Owner" is listed as "Kollepara Pavan".

When a test drive is scheduled then email will be send Automatically to customer like this:



This is how a customer can raise a vehicle service request in the system:

## **Conclusion:**

Through the development of the "**WhatNext Vision Motors**" Salesforce application, I gained hands-on experience in building a complete, scalable business solution using Salesforce tools and automation features. I successfully designed and created custom objects, fields, and relationships to represent real-world entities like vehicles, customers, dealers, orders, test drives, and service requests. I enhanced user experience by setting up a custom **Lightning App**, creating **custom tabs**, and configuring **navigation items**.

On the automation side, I developed **record-triggered flows** to assign the nearest dealer to a customer and to send automated test drive reminders, improving customer engagement. I implemented Apex-based automation by creating a structured set of classes including **triggers**, **handler classes**, **batch classes**, and a **scheduler** to process vehicle orders automatically on a daily basis. This taught me how to handle real-time and scheduled processes in Salesforce effectively.

Overall, this project helped me understand and apply **Salesforce development principles**, automation best practices, and scalable design. It not only strengthened my technical skills but also gave me practical insights into building enterprise-level CRM solutions.