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

ABSTRACT

WhatsNext Vision Motors is undertaking a transformative Salesforce CRM implementation aimed at enhancing customer experience and operational efficiency within the automotive sector. Central to the solution is a data model integrating vehicle inventory, dealer locations, and customer management. The CRM leverages automated workflows and Apex triggers to enforce business logic—blocking orders for out-of-stock vehicles and auto-assigning new orders to the nearest dealer based on customer geolocation.

Complementing these real-time safeguards, the project integrates scheduled processes: batch Apex updates stock availability across bulk orders and scheduled Apex dispatches email reminders for test drives and stock replenishment. Together, these components form a cohesive system that offers transparent, accurate order tracking—marking statuses as "Pending" or "Confirmed" depending on inventory levels.

The technical implementation includes structured data modeling of vehicle and dealer records, Lightning App Builder for streamlined UI interactions, record-triggered flows for dynamic logic execution, and robust Apex code for validation and automation. By reducing manual intervention, minimizing customer friction, and improving responsiveness to stock changes, the project is set to significantly elevate customer satisfaction and internal productivity— propelling WhatsNext Vision Motors toward more agile and customer-centric operations.

OBJECTIVE

The objective of this project is to enhance the customer ordering experience by leveraging Salesforce automation and intelligent data handling. It aims to streamline dealer assignment, ensure real-time stock validation, and provide accurate order status updates. By doing so, WhatsNext Vision Motors seeks to improve customer satisfaction and operational efficiency across its mobility services.

- **Enhanced Customer Experience:** Simplifying the ordering process with automated dealer suggestions and real-time updates.
- **Real-Time Stock Management:** Preventing out-of-stock orders through accurate inventory validation.
- **Automated Dealer Assignment**: Assigning orders to the nearest dealer based on customer location.
- **Efficient Order Processing:** Using scheduled processes to update order statuses and streamline fulfillment.
- **Smart Salesforce Integration:** Implementing Apex, Flows, and Batch Jobs to automate and scale operations.

TECHNOLOGY DESCRIPTION

Salesforce:-

Salesforce is a cloud-based Customer Relationship Management (CRM) platform that enables businesses to manage customer interactions, data, and processes efficiently. In this project, it is used to store and manage vehicle details, dealer locations, and customer orders. The platform supports automation through Flows, Apex triggers, and batch jobs to ensure accurate stock validation and smart dealer assignment. By leveraging Salesforce, WhatsNext Vision Motors enhances operational efficiency and delivers a seamless, customer-centric ordering experience.

Custom Objects:-

Objects in salesforce are like tables in a database. Custom Objects are created to store specific data.

Example:

- Vehicle_c - Stores vehicle details
- Vehicle_Dealer_c- Stores authorized dealer info
- Vehicle_Customer_c- Stores customer details
- Vehicle_Order_c- Tracks vehicle purchases
- Vehicle_Test_Drive_c- Tracks test drive bookings
- Vehicle_Service_Request_c- Tracks vehicle servicing requests

Tabs:-

Tabs are used to display data in the Salesforce UI

Example: A tab for vehicle_c allows users to easily view, create, and manage vehicle

Custom APP:-

An App in Salesforce is a collection of tabs grouped together for a specific business purpose, allowing users to access related data and functionality in one place.

Fields:-

A **fields** in Salesforce is a specific piece of data stored in an object, much like a column in a spreadsheet. Each field holds a particular type of information such as text, date, number, or picklist values.

Example: Vehicle_Model_c – A **Picklist** field in the Vehicle_c object with options like *Sedan*, *SUV*, *EV*, etc.

Flows:-

Flows automate business logic without writing code. They can create or update records, and send notifications automatically.

Example:-

- A flow triggers an email alert whenever a new order is created.

Apex:-

Apex is Salesforce's Object-Oriented programming language. It allows developers to write custom logic .

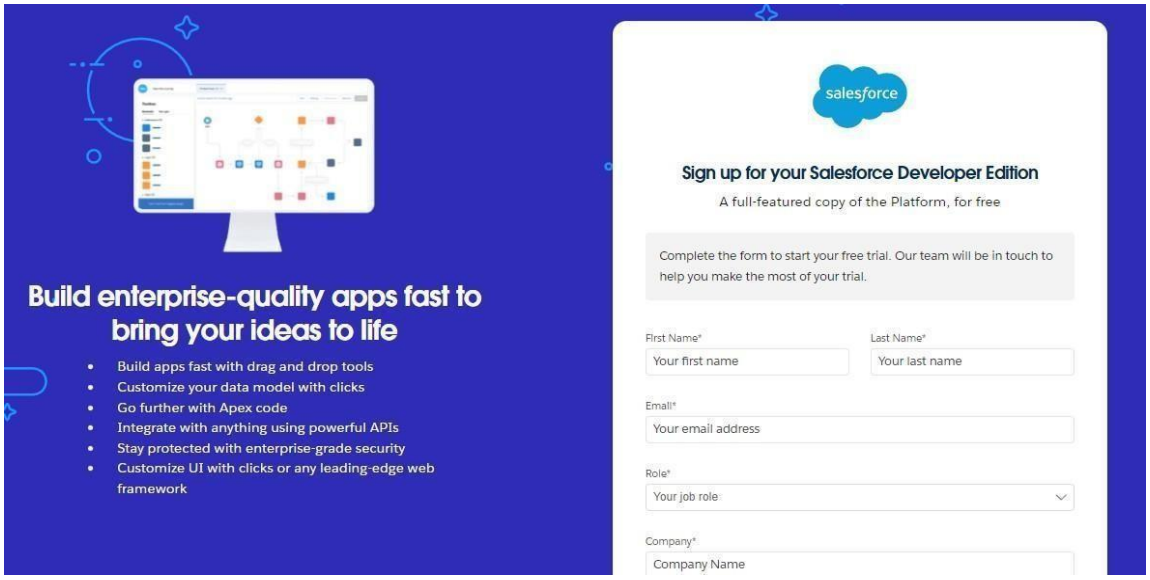
Example Triggers:

- A trigger automatically validates stock availability when a new vehicle order is created.
- It prevents orders if the vehicle is out of stock.
- Once the order is confirmed, the trigger updates the stock quantity by reducing it.

DETAILED EXECUTION OF PROJECT PHASES

1. Developer Org Setup

- A Salesforce Developer Org was Created Using **https://developer.salesforce.com/signup**



- The account was verified, password set , and access was granted to the Salesforce Setup page.

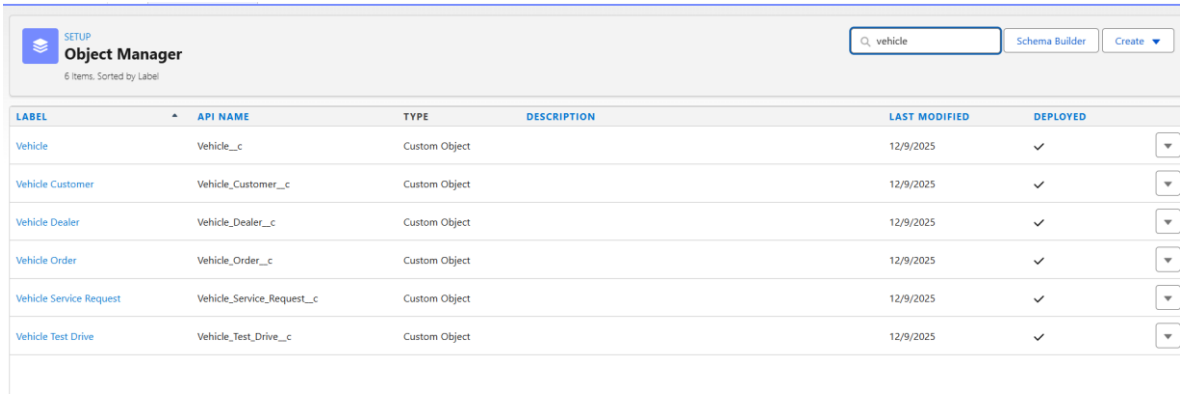
2. Custom Object Creation

Six Custom Objects were created to store business-critical data

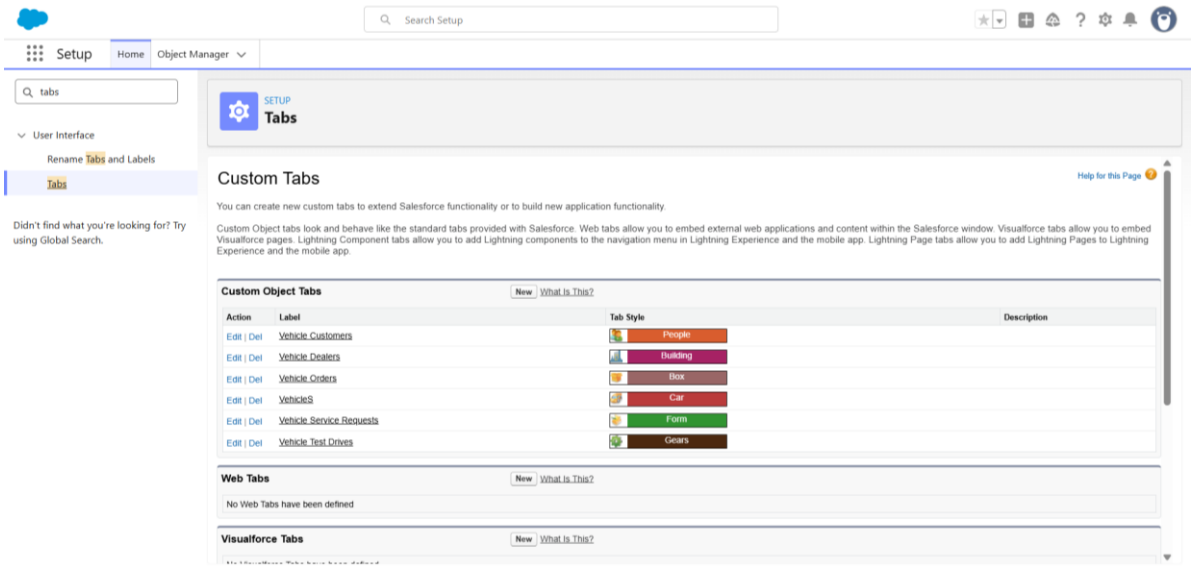
- **Vehicle** – Stores vehicle details like model, stock, price, and status.
- **Vehicle Dealer** – Contains information about dealers such as location and contact details.
- **Vehicle Order** – Tracks customer orders, order dates, and order status.
- **Vehicle Customer** – Maintains customer details including contact and preferred vehicle type.
- **Vehicle Test Drive** – Records test drive schedules and status.
- **Vehicle Service Request** – Logs service requests, dates, issues, and progress status.

Steps followed:

- Navigated to Setup → Object Manager → Create → Custom Object.
- Provided Label, Name, and enabled options for Reports and Search.
- Saved and created Tabs for each object.

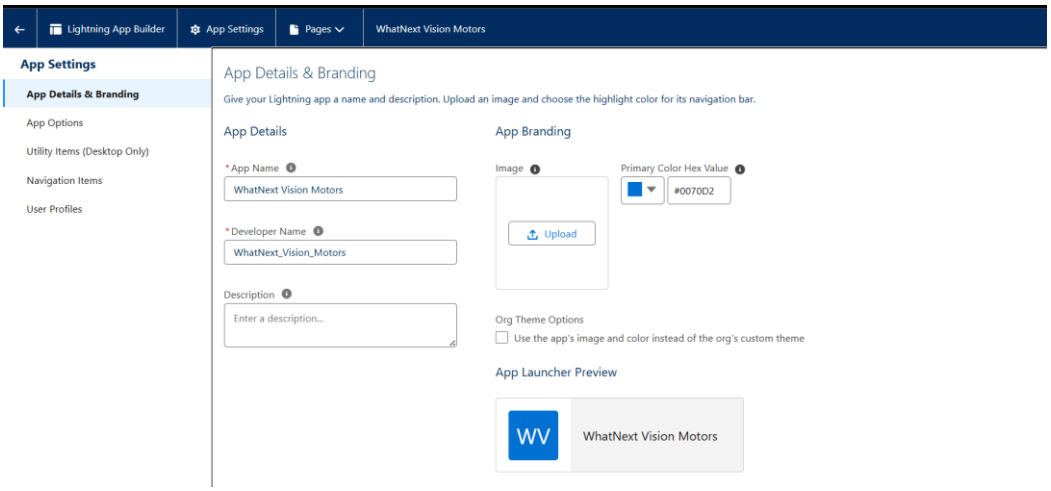


3. Creating a Custom Tab



4. Creating the Lighting App

- A custom Lightning App named WhatNext Vision Motors was created.
- Included tabs: Vehicle, Vehicle Dealer, Vehicle Order, Vehicle Customer, Vehicle Test Drive, Vehicle Service
- Assigned to the System Administrator profile.



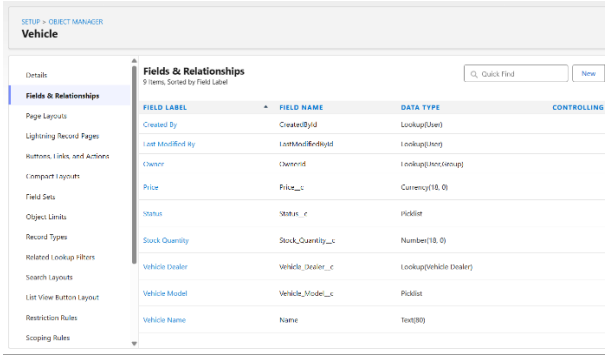
5. Creating Fields

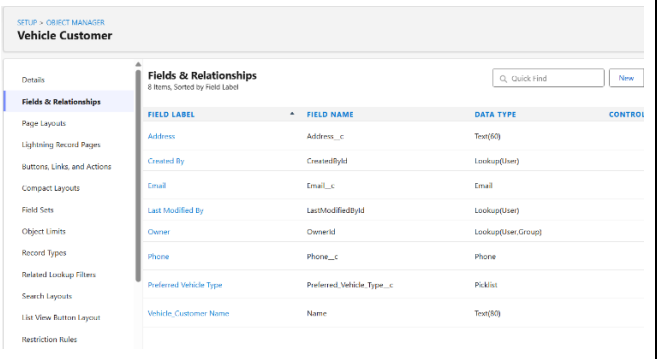
Custom fields were created on each object to store specific business data.

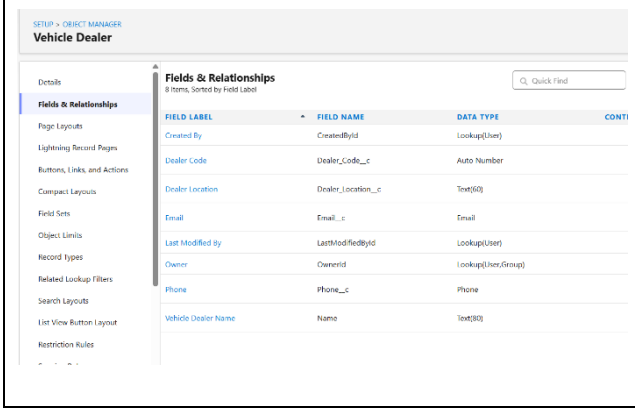
For example, on the Vehicle__c object, a Picklist field named Vehicle_Model__c was created with values like *Sedan*, *SUV*, and *EV*.

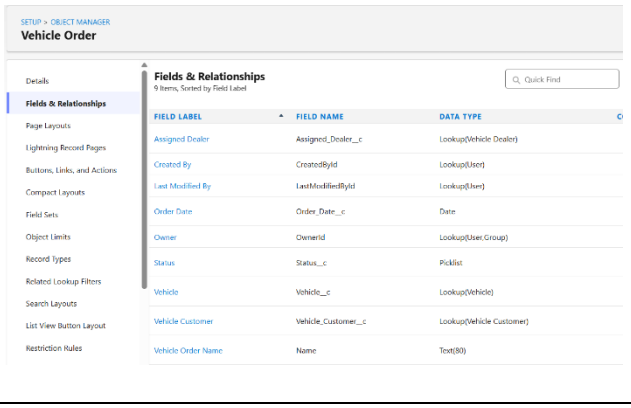
Steps followed:

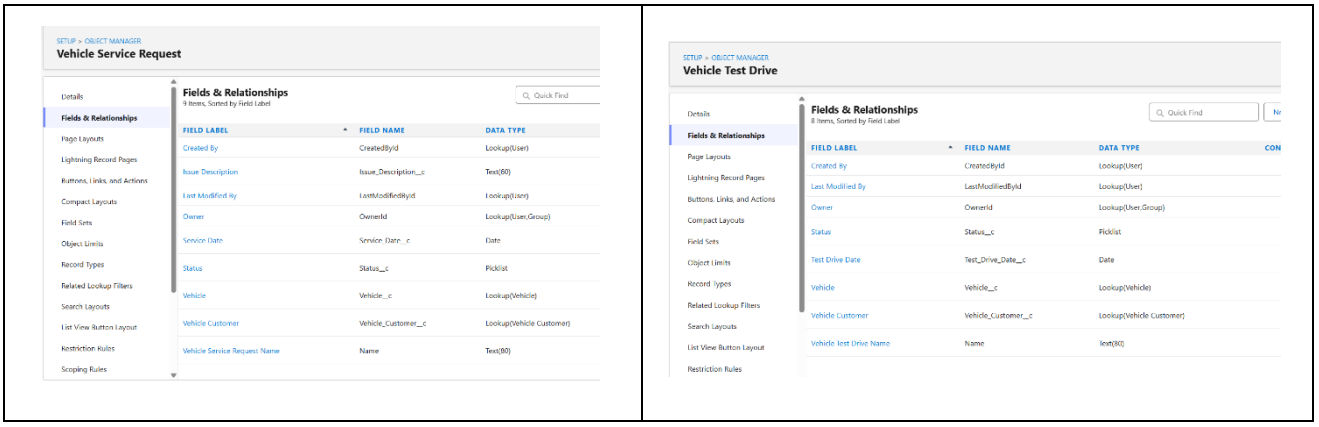
- Navigated to Setup → Object Manager → [Object Name] → Fields & Relationships → New.











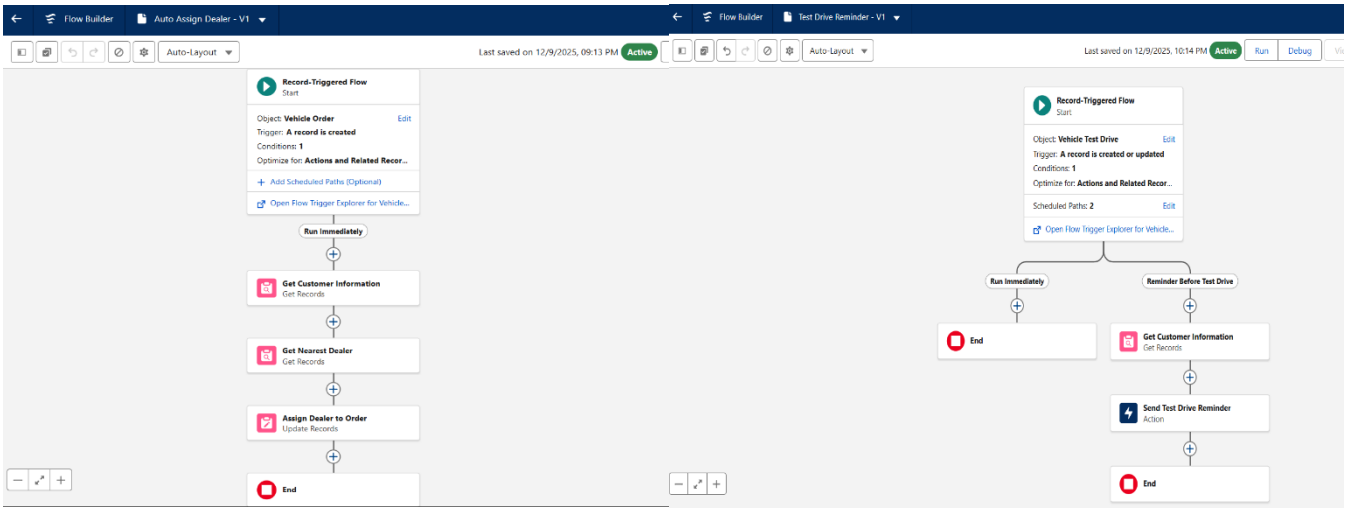
6. Creating Flows

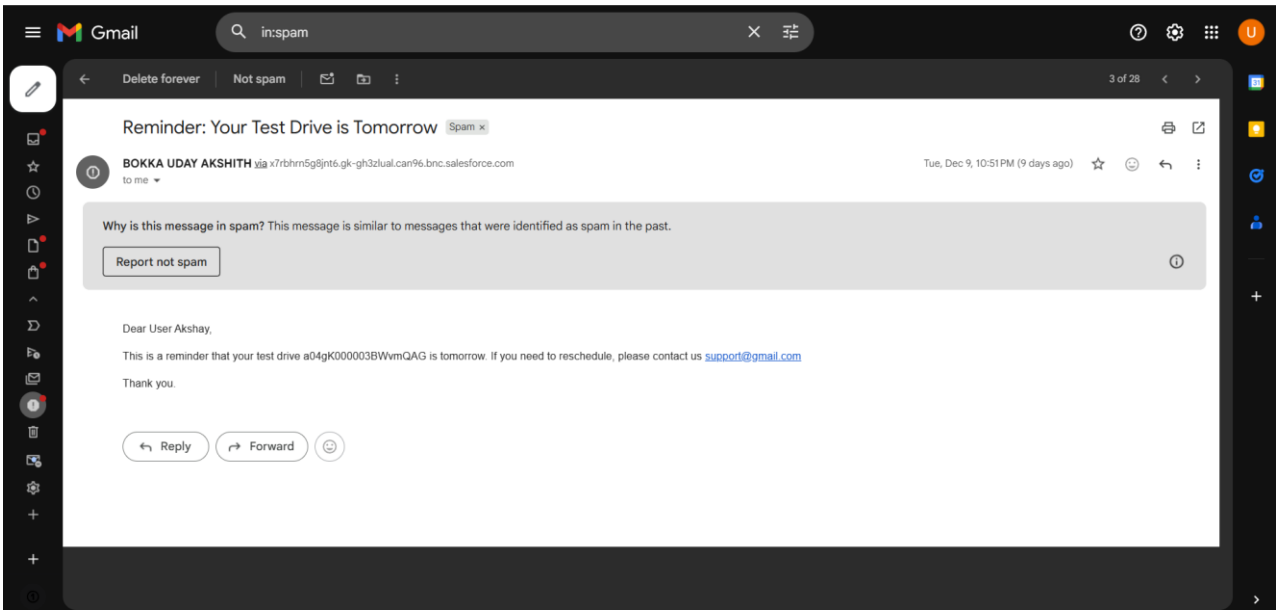
Record-triggered flows were created to automate key business processes without code. For example, a flow was built to **auto-assign the nearest dealer** when a vehicle order is created with status *Pending*.

Another flow sends **email reminders** to customers **one day before** their scheduled test drive.

Steps followed:

- Setup → Flows → New Flow → Record-Triggered Flow.





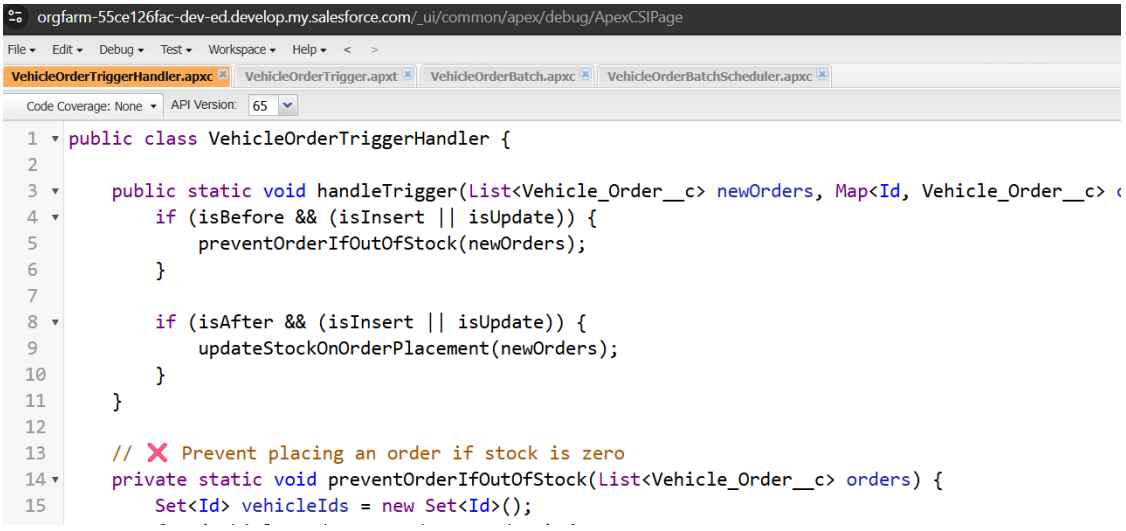
7. Create Apex and Trigger Batch Jobs:

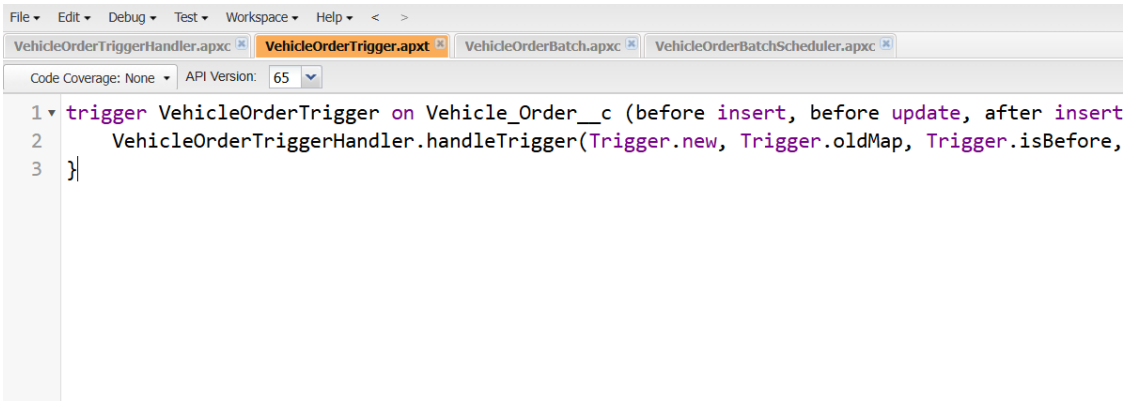
Apex classes and triggers were created to enforce business logic during order processing.

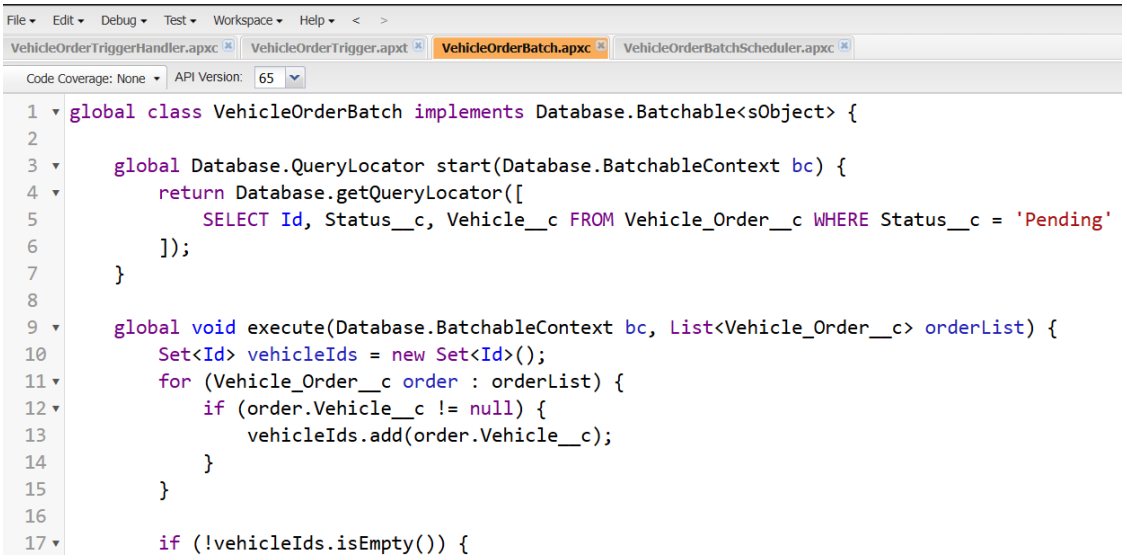
- A **trigger** checks vehicle stock when an order is placed and updates stock on confirmation.
- A **batch class** regularly scans pending orders and updates them to *Confirmed* if stock becomes available.
- A **scheduled class** runs the batch job daily to automate order updates.

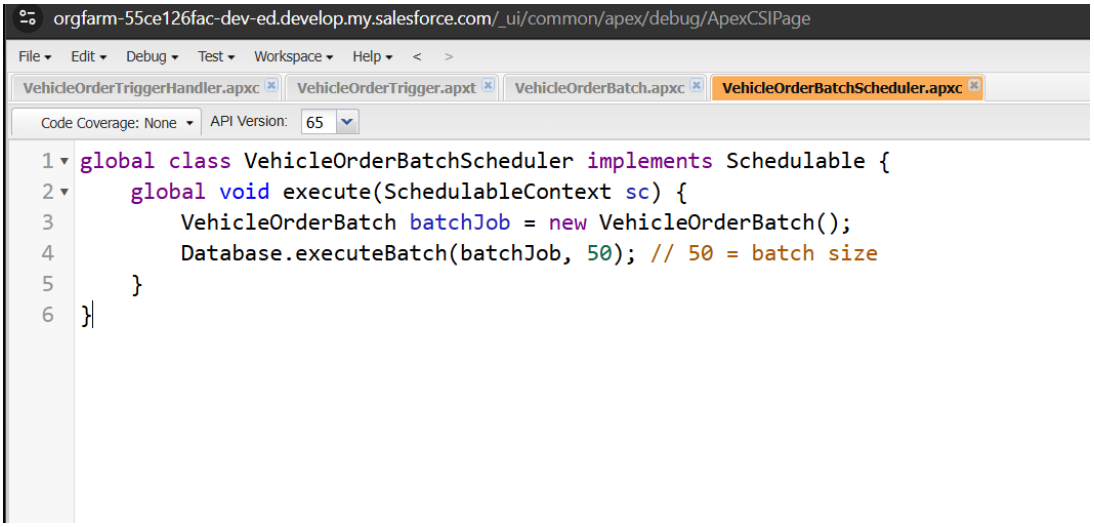
Steps followed:

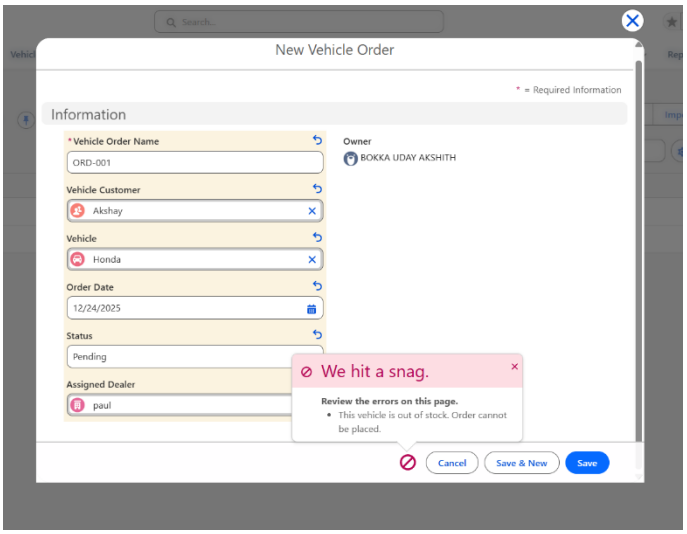
- Developer Console → File → New → Apex Class.




File Edit Debug Test Workspace Help
VehicleOrderTriggerHandler.apxc
VehicleOrderTrigger.apxt
VehicleOrderBatch.apxc
VehicleOrderBatchScheduler.apxc
Code Coverage: None
API Version: 65
trigger VehicleOrderTrigger on Vehicle_Order__c (before insert, before update, after insert
VehicleOrderTriggerHandler.handleTrigger(Trigger.new, Trigger.oldMap, Trigger.isBefore,
}

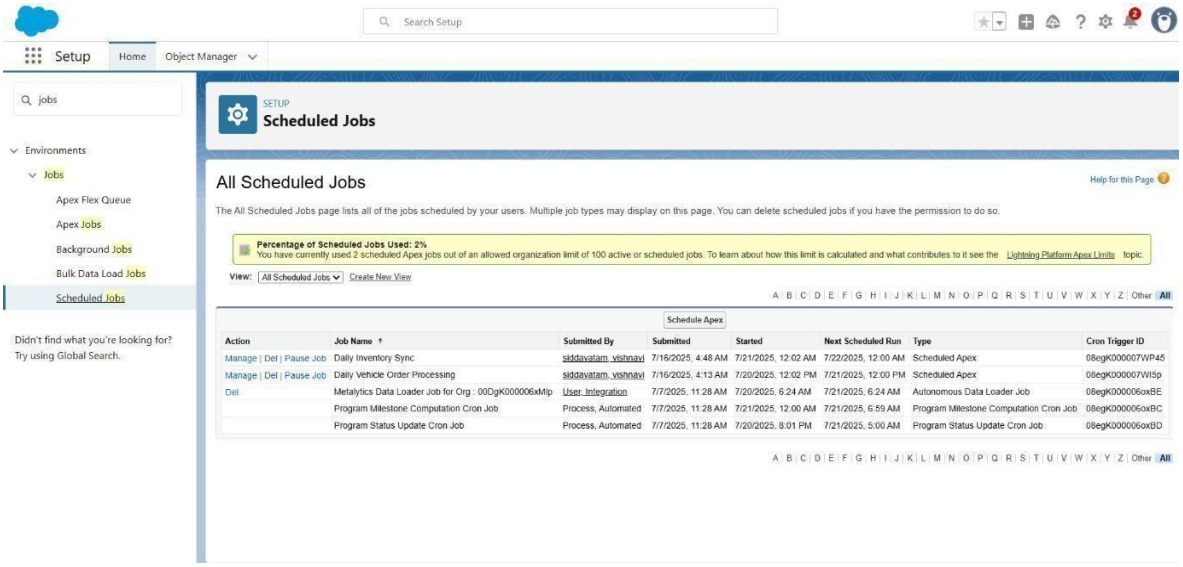

File Edit Debug Test Workspace Help
VehicleOrderTriggerHandler.apxc
VehicleOrderTrigger.apxt
VehicleOrderBatch.apxc
VehicleOrderBatchScheduler.apxc
Code Coverage: None
API Version: 65
global class VehicleOrderBatch implements Database.Batchable<sObject> {
global Database.QueryLocator start(Database.BatchableContext 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()) {


orgfarm-55ce126fac-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File Edit Debug Test Workspace Help
VehicleOrderTriggerHandler.apxc
VehicleOrderTrigger.apxt
VehicleOrderBatch.apxc
VehicleOrderBatchScheduler.apxc
Code Coverage: None
API Version: 65
global class VehicleOrderBatchScheduler implements Schedulable {
global void execute(SchedulableContext sc) {
VehicleOrderBatch batchJob = new VehicleOrderBatch();
Database.executeBatch(batchJob, 50); // 50 = batch size
}
}



8. Scheduled Jobs

The **Scheduled Apex Jobs page** in Salesforce displays all Apex jobs that are scheduled to run automatically, such as batch jobs, schedulable classes, and queueable jobs. It provides key details like job name, type, status (Queued, Processing, Completed, Failed), submission time, next scheduled run, and the user who submitted it. From this page, administrators can monitor job progress, check for errors, and manage jobs by aborting, pausing, or deleting them when needed. This helps ensure that automated processes run smoothly and any issues can be quickly identified and resolved.



PROJECT EXPLANATION WITH REAL-WORLD EXAMPLE

1. Customer Registration

Real-Life Example:
A customer, **Elijah Mikaelson**, visits the automobile showroom or website.

In Salesforce:

- A new record is created in the **Vehicle_Customer_c** object with his details such as Name, Phone, Email, and Address.
- **Validation Rule:** Ensures the email is valid (e.g., must contain "@gmail.com"). If invalid, an error is shown. **2. Vehicle & Inventory Management**

Real-Life Example:
The store admin adds **cars like BMW X5, Toyota Fortuner, etc.**, with stock quantity (e.g., 5 BMWs and 3 Fortuners).

In Salesforce:

- Each vehicle is stored in the **Vehicle_c** object with details like Price, Model, and Stock Quantity.
- **Validation Rule:** Prevents adding a vehicle with negative stock values. **3. Order Placement**

Real-Life Example:
Elijah places an order for **1 BMW X5** (Price ₹50,00,000).

In Salesforce:

- A new record is created in the **Vehicle_Order_c** object.
- **Apex Trigger (VehicleOrderTrigger):**
  - Automatically calculates **Total Amount = Price × Quantity**.
  - Checks **stock availability**:
    - If stock > 0 → **Status_c = Confirmed.**
    - If stock = 0 → **Status_c = Pending.**

4. Inventory Update

Real-Life Example:
When Elijah's order is confirmed, the BMW stock decreases by 1 (from 5 → 4).

In Salesforce:

- **Apex Trigger (VehicleOrderTriggerHandler):**
    - Updates **Vehicle_c.Stock_Quantity_c** by subtracting

the quantity ordered. ○ Prevents stock from going below zero. **5. Pending**

→ Confirmed Orders (Stock Refill)

Real-Life Example:
If BMW stock is 0 and Elijah's order was pending, when the admin refills **5 BMWs**, the system **automatically confirms pending orders** (if stock is available).

In Salesforce:

- **VehicleTrigger (After Update):**
    - Detects stock refill. ○ Calls confirmPendingOrders() in

the **VehicleOrderTriggerHandler**. ○ Updates all pending orders to **Confirmed** until stock runs out. **6. Scheduled Batch Processing**

Real-Life Example:
At the end of each day, a **batch job** runs to update **any remaining pending orders** automatically.

In Salesforce:

- **VehicleOrderBatch & VehicleOrderBatchScheduler:**
    - Confirm pending orders if stock is available. ○ Sends

admin a log of processed orders.

7. Email Notifications (Flow)

Real-Life Example:
When an order is **confirmed** or **pending**, Elijah gets an **email notification**.

In Salesforce:

- **Record-Triggered Flow:**
    - Sends an email alert:
      "Your BMW X5 order is confirmed!" or
      "Your order is pending due to low stock."

8. Users and Roles

Real-Life Example:

- **Niklaus Mikaelson** (Sales Manager) handles customer orders.
- **Kol Mikaelson** (Inventory Manager) manages vehicle stock.

In Salesforce:

- Roles and Profiles:
  - **Sales Role:** Access to Vehicle_Order_c and Vehicle_Customer_c.
  - **Inventory Role:** Access to Vehicle_c and stock management.

9. Real-Life Benefit

This system ensures:

- **No manual stock checks** (triggers handle it automatically).
- **Automatic order status updates** (Confirmed or Pending).
- **Smooth communication with customers** via email alerts.
- **Dealer assignment** (optional via Flow) ensures orders are assigned to the nearest dealer.

SCREENSHOTS

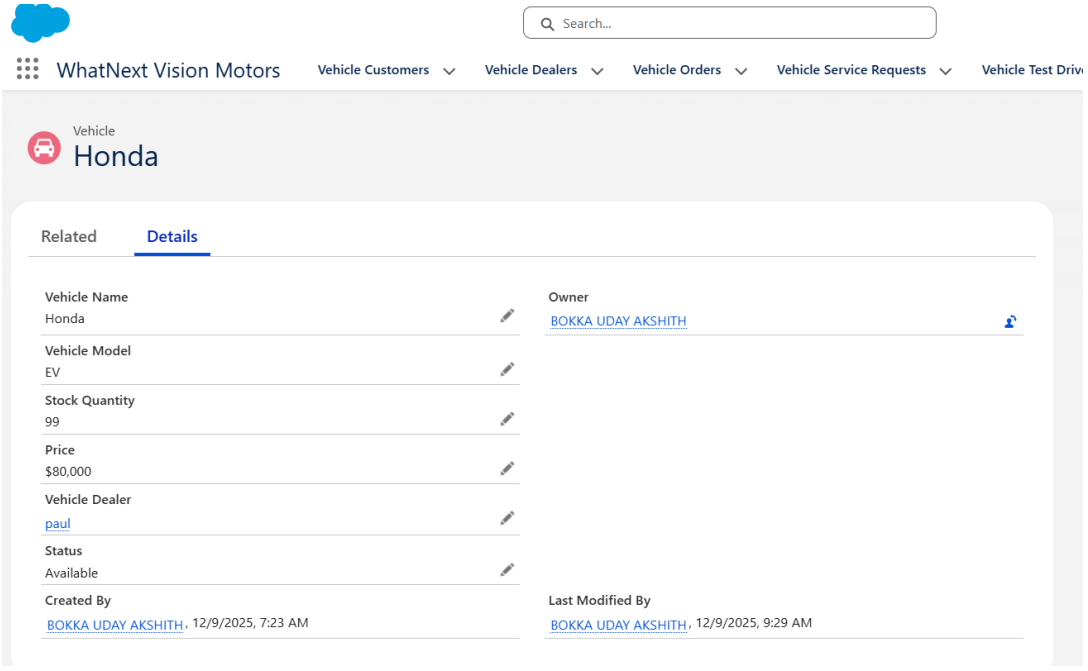


Fig: Vehicles in WhatNext Vision Motors

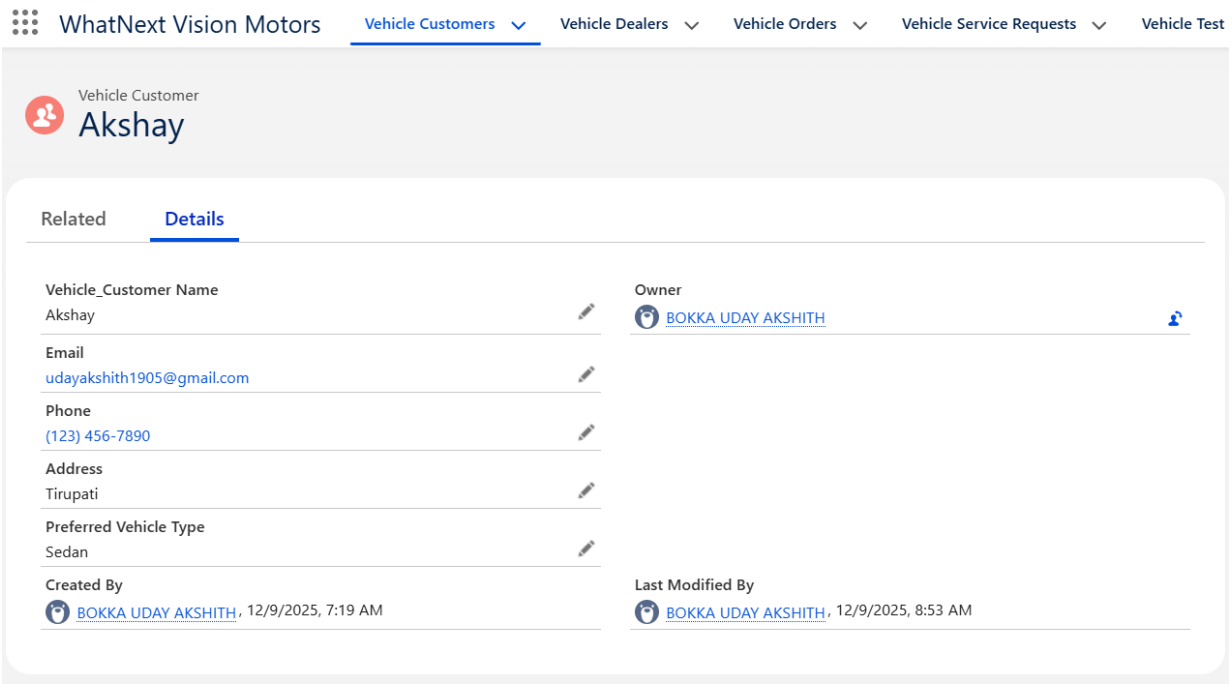


Fig: Customers in WhatNext Vision Motors

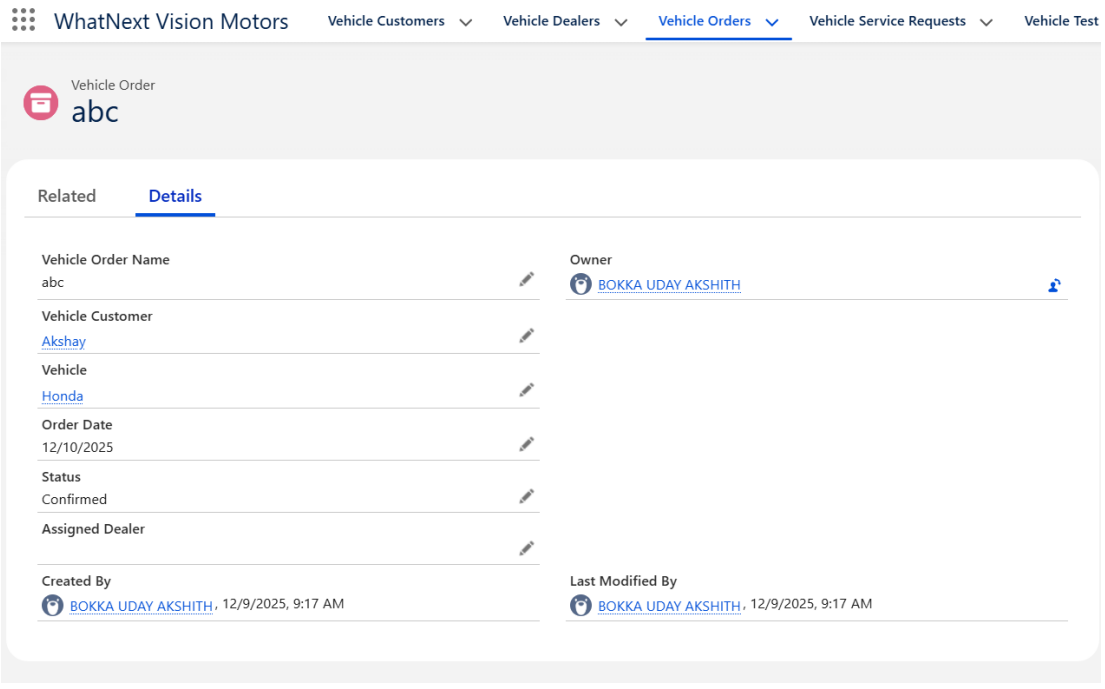


Fig: Vehicle Orders in WhatNext Vision Motors

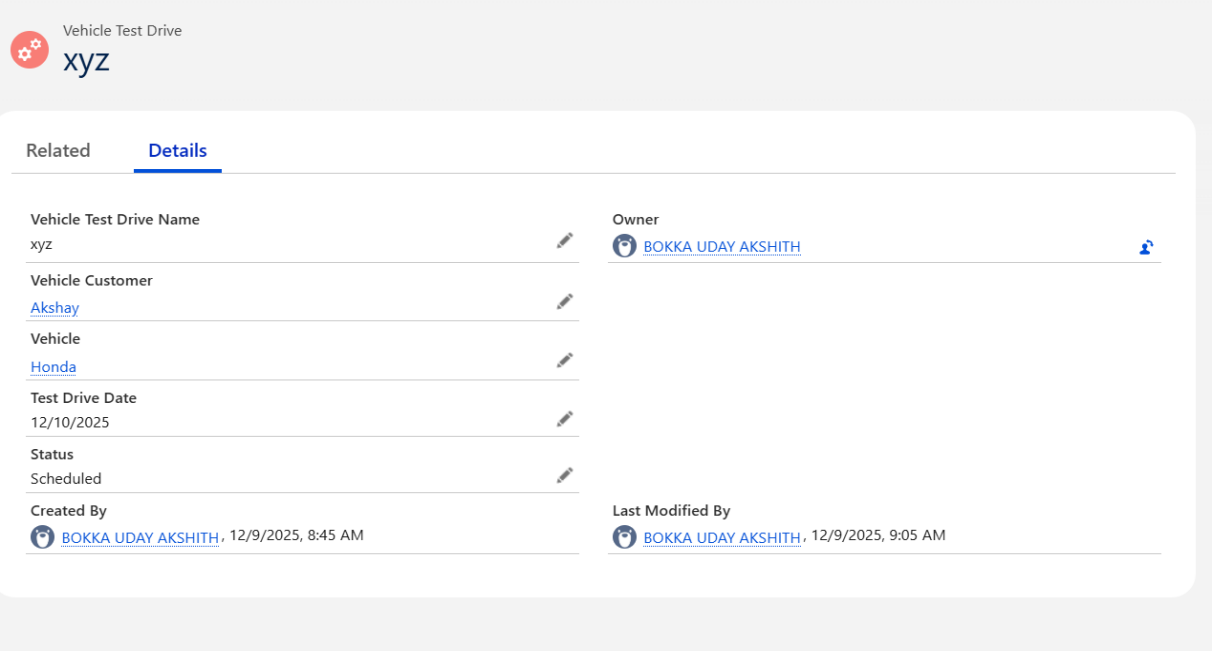


Fig: Test Drives in WhatNext Vision Motors

CONCULSION

WhatNext Vision Motors leverages Salesforce to streamline its vehicle sales and service operations. By creating custom objects, automated flows, and Apex triggers, the company ensures efficient order processing, real-time stock management, and improved customer engagement. This digital transformation enhances productivity, reduces errors, and delivers a seamless experience for both staff and customers, positioning WhatNext Vision Motors for sustained growth and success.

Future Scope

1. **Customer Portal Integration** ○ Develop a **Salesforce Community Portal** where customers can log in to:
    - View their order history and current order status.
    - Check loyalty program points and rewards.
    - Raise queries or service requests directly.
2. **Mobile App using Salesforce Mobile SDK** ○ Build a **mobile application** for store staff and managers to:
    - Manage inventory (add or update stock).
    - Process customer orders in real-time.
    - Access dashboards and order reports on the go.
3. **Reports & Dashboard** ○ Create **dynamic dashboards** and reports in Salesforce for:
    - Real-time vehicle stock levels.
    - Sales trends and revenue analysis.
    - Monitoring loyalty program metrics.
4. **AI-Powered Recommendations (Einstein AI)** ○ Integrate **Salesforce Einstein** to:
    - Analyze customer purchase history.

- Provide **personalized vehicle recommendations** (e.g., suggest SUVs for customers buying family cars).

5. **WhatsApp/SMS Integration** ○ Use Salesforce **Digital Engagement** or third-party tools (like Twilio) to:
   - Send **instant notifications** about order confirmation or pending status.
   - Update customers about loyalty status or promotional offers.