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

ABSTRACT

The "WhatNext Vision Motors" Capstone Project focuses on automating vehicle order processing within Salesforce to enhance efficiency and customer experience. This project addresses key business challenges such as intelligently assigning orders to the nearest dealers, preventing orders for out-of-stock vehicles, and ensuring timely test drive reminders. By leveraging Salesforce's declarative tools like Flows and programmatic capabilities with Apex, the project aims to streamline operations and provide a robust solution for future mobility needs.

OBJECTIVE

The main objectives of the WhatNext Vision Motors project are:

- To automate vehicle order processing using Salesforce
- To assign orders to the nearest dealers based on customer location
- To create logic to prevent orders for out-of-stock vehicles
- To send email reminders for test drives one day before the scheduled date

TECHNOLOGY DESCRIPTION

The project primarily utilizes the **Salesforce platform**, employing both its declarative automation features and programmatic development capabilities:

- **Salesforce Flows:** Used for implementing business logic such as auto-assigning dealers and sending test drive reminders. Flows provide a visual interface for building complex automations without extensive coding.
- **Salesforce Apex:** A strongly-typed, object-oriented programming language used for building more complex business logic that cannot be achieved with Flows. In this project, Apex is used for handling vehicle stock quantity reduction upon order confirmation and preventing orders for out-of-stock vehicles, as well as for batch processing.

DETAILED EXECUTION OF PROJECT PHASES

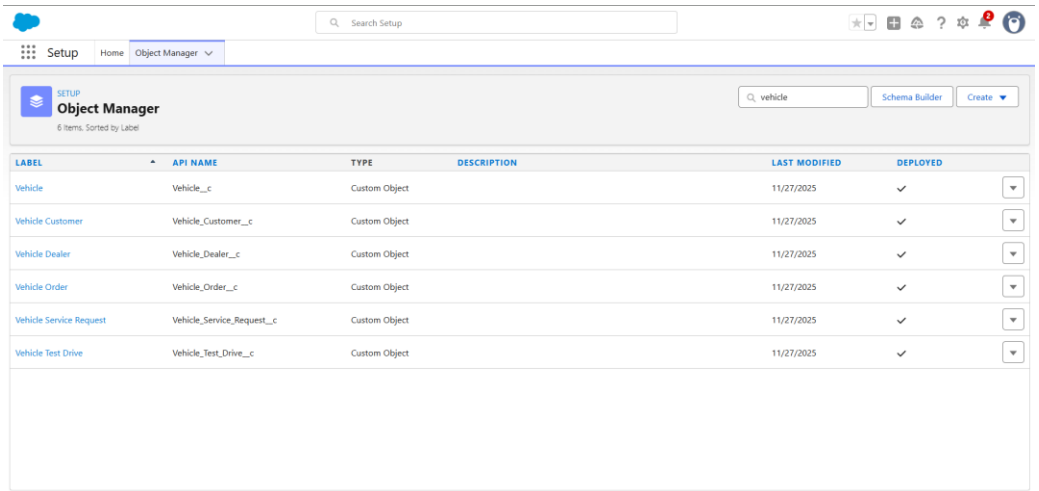
The project execution followed a structured approach, starting from Salesforce environment setup to advanced automation:

1. Salesforce Credentials Creation

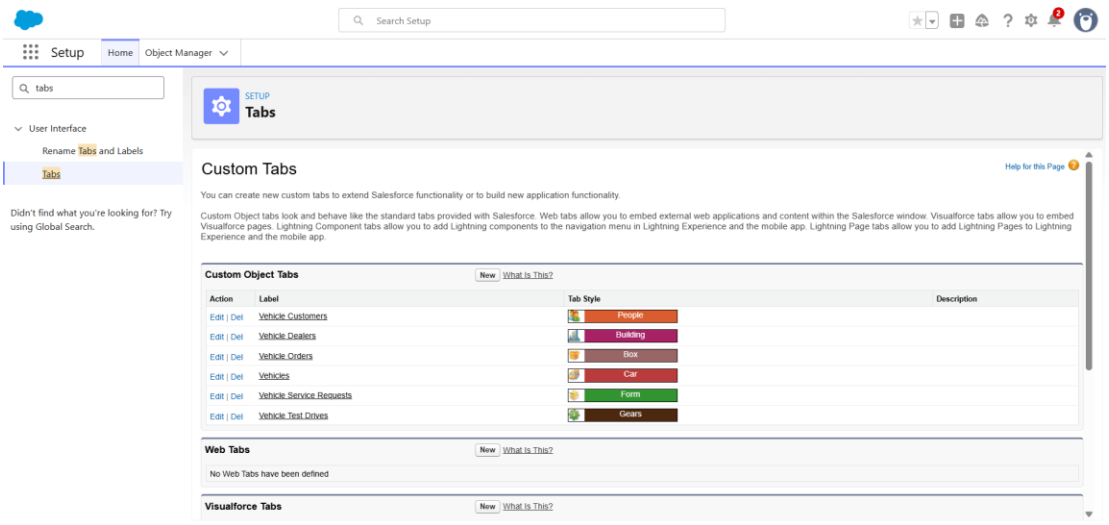
- **Sign-up for a Developer Account:** A new Salesforce Developer Edition Org was created to serve as the development environment for the project. This involves filling out personal details and verifying the account to receive login credentials.

2. Data Management

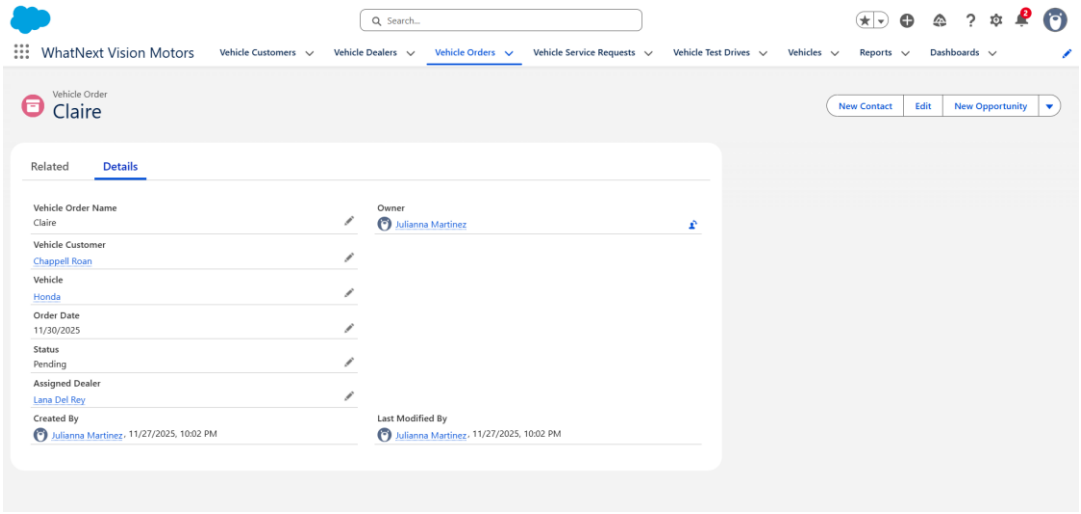
- **Custom Objects Creation:** Six custom objects were created to manage specific data entities within the vehicle ordering process:
    - **Vehicle**
    - **Vehicle Dealer**
    - **Vehicle Customer**
    - **Vehicle Order** - *Note: Vehicle Order Number was set as an Auto Number).*
    - **Vehicle Test Drive**
    - **Vehicle Service Request**



- **Custom Tabs Creation:** Custom tabs were created for each of the six custom objects, allowing users to easily navigate and access records within the Salesforce UI. Appropriate tab styles (e.g., 'car' for Vehicle, 'people' for Vehicle Customer) were selected to enhance visual representation

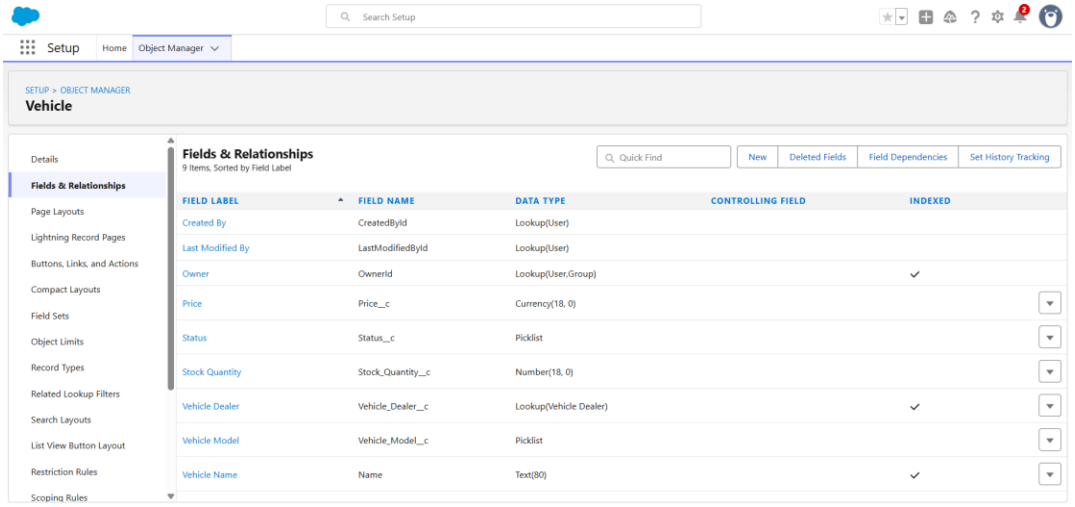


- **App Manager - Lightning App Creation:** A custom Lightning App named "WhatNext Vision Motors" was created. This app serves as a centralized hub for all project-related objects and functionalities. Key custom objects (Vehicle, Vehicle Customer, Vehicle Dealer, Vehicle Order, Vehicle Service Request, Vehicle Test Drive) and standard objects (Reports, Dashboards) were added to the app. Access was granted to the 'System Administrator' profile

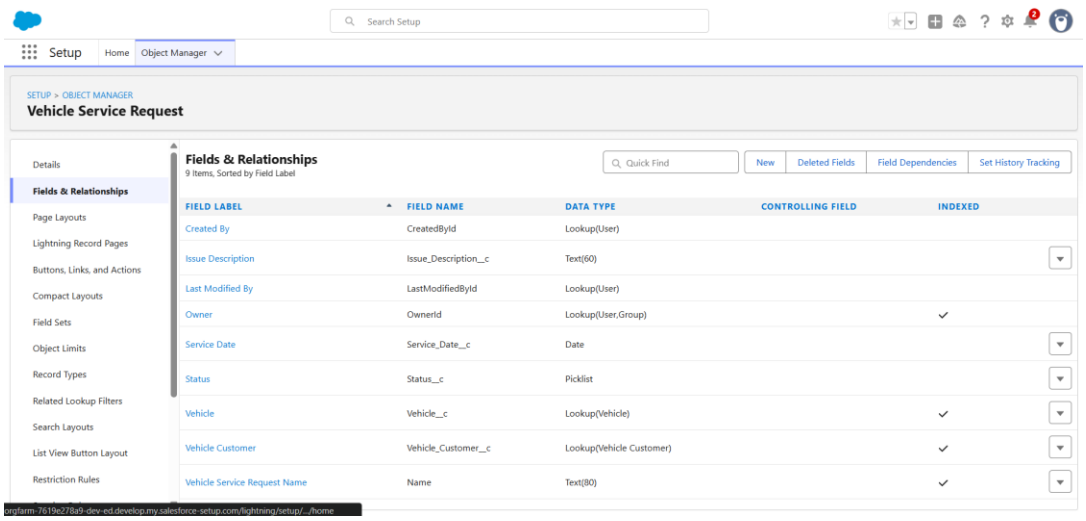


- **Custom Fields Creation:** Various custom fields were created for each object to store specific data points, including:
    - **Vehicle Object:** Vehicle Model (picklist: Sedan, SUV, EV, etc.), Stock Quantity (number), Price (currency), Dealer (lookup to Vehicle Dealer), Status (picklist: Available, Out of Stock, Discontinued)

- **Vehicle Dealer Object:** Dealer Location (text), Dealer Code (auto number), Phone (phone), Email (email)
- **Vehicle Order Object:** Customer (lookup to Vehicle Customer), Vehicle (lookup to Vehicle), Order Date (date), Status (picklist: Pending, Confirmed, Delivered, Cancel). An additional Assigned Dealer (lookup to Vehicle Dealer) field was created for the flow's functionality
- **Vehicle Customer Object:** Email (email), Phone (phone), Address (text), Preferred Vehicle Type (picklist: Sedan, SUV, EV, etc.)
- **Vehicle Test Drive Object:** Customer (lookup to Vehicle Customer), Vehicle (lookup to Vehicle), Test Drive Date (date), Status (picklist: Scheduled, Completed, Cancelled

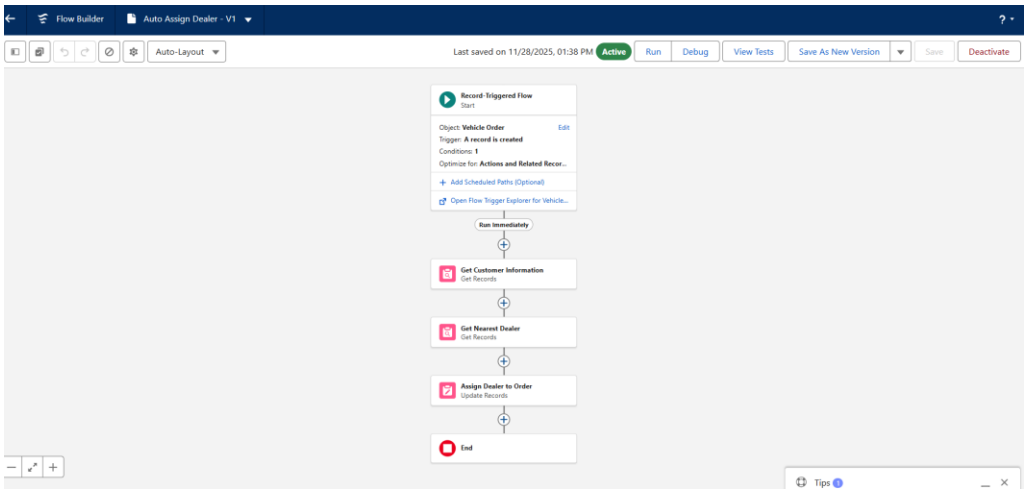


- **Vehicle Service Request Object:** Customer (lookup to Vehicle Customer), Vehicle (lookup to Vehicle), Service Date (date), Issue Description (text), Status (picklist: Requested, In Progress, Completed)

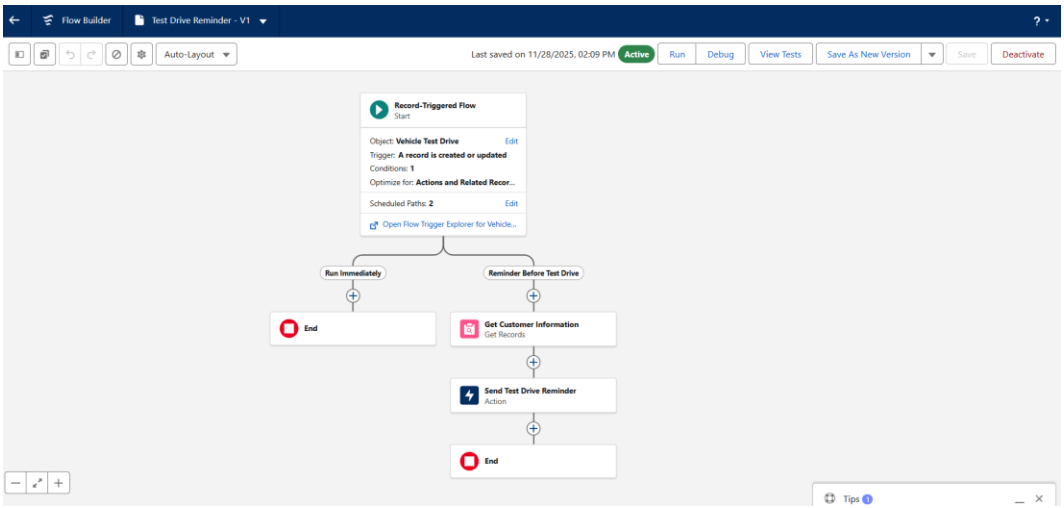


3. Automation - Salesforce Flows **Auto-Assign Dealer Flow**

- **Type:** Record-Triggered Flow
- **Object:** Vehicle Order
- **Trigger:** When a record is created or updated, specifically when Status equals 'Pending'
- **Logic:**
    - **Get Customer Information:** Retrieves the Vehicle Customer record related to the Vehicle Order
    - **Get Nearest Dealer:** Retrieves Vehicle Dealer records where the Dealer Location matches the customer's Address
    - **Assign Dealer to Order:** Updates the Assigned Dealer field on the Vehicle Order record with the ID of the nearest dealer found
- **Activation:** The flow was activated to ensure real-time automation



- **Test Drive Reminder Flow**
  - **Type:** Record-Triggered Flow
  - **Object:** Vehicle Test Drive
  - **Trigger:** When a record is created or updated, and Status equals 'Scheduled', triggering only when the record is updated to meet the condition
  - **Scheduled Path:** Configured to execute one day before the Test Drive Date
  - **Logic:**
    - **Get Customer Information:** Retrieves the Vehicle Customer record associated with the Vehicle Test Drive
    - **Send Email Action:** Sends an email to the customer's email address with a reminder about the upcoming test drive. The email body includes dynamic customer name and test drive ID
  - **Activation:** The flow was activated upon creation

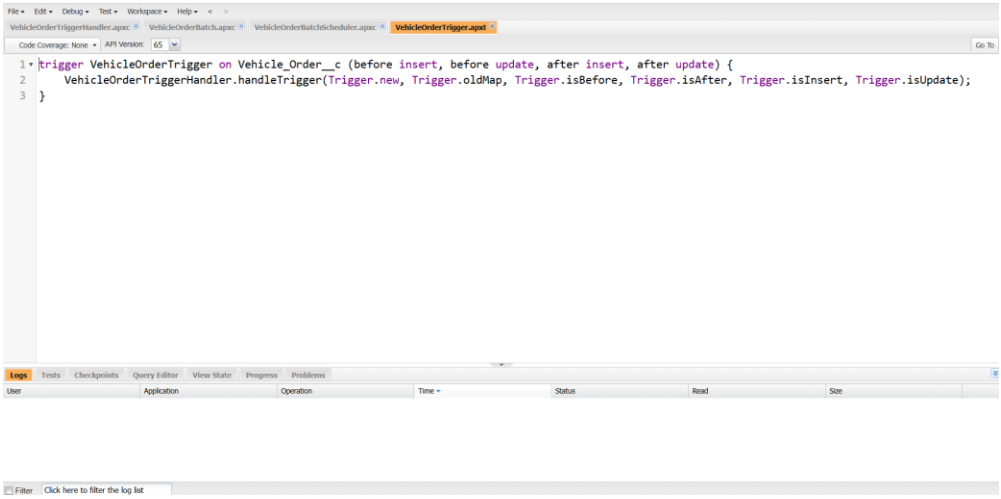


4. Apex and Batch Classes

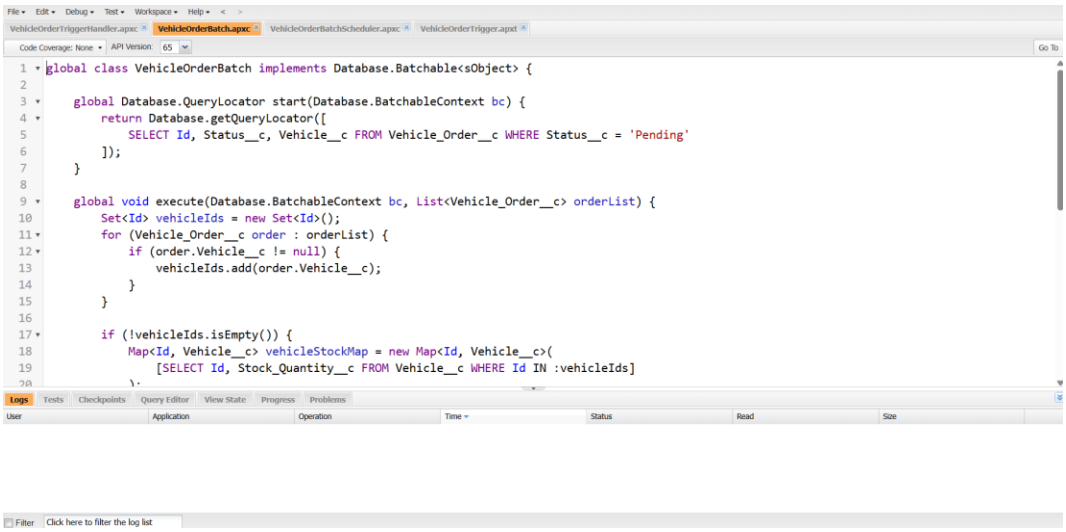
- **Apex Class: VehicleOrderTriggerHandler**
  - This class contains the core logic for the Apex trigger.
  - **Purpose:** To handle the reduction of vehicle stock quantity and prevent orders for out-of-stock vehicles.

- **Apex Trigger: VehicleOrderTrigger**
  - **Object:** Vehicle Order

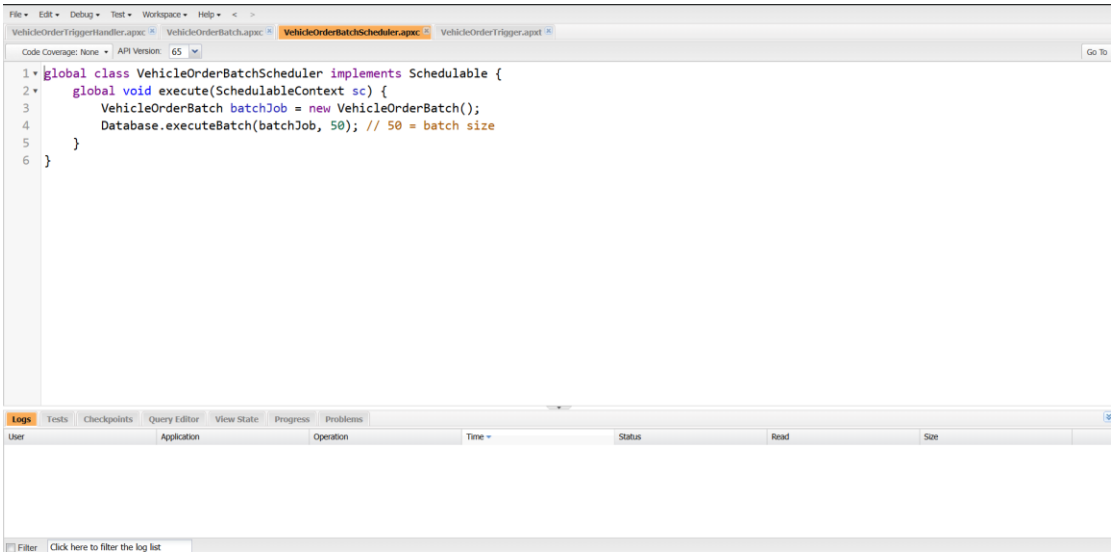
- **Purpose:** Fires on specific DML operations (e.g., after insert, after update) on the Vehicle Order object to invoke the VehicleOrderTriggerHandler class.
- **Functionality:**
    - **Reduce Stock Quantity:** When a Vehicle Order status is 'Confirmed', the stock Quantity of the associated Vehicle is automatically reduced by one
    - **Prevent Out-of-Stock Orders:** If the Stock Quantity of a Vehicle is zero, users are prevented from creating new orders for that vehicle



- **Batch Apex Class: VehicleOrderBatch**
    - **Purpose:** This class is designed to process a large number of records in batches, typically for data synchronization or complex calculations that exceed governor limits in a single transaction. In this project, it is used to periodically check vehicle stock.



- **Scheduled Apex Class: VehicleOrderBatchScheduler**
    - **Purpose:** To schedule the VehicleOrderBatch class to run at specific intervals. This allows for automated, periodic checks and updates related to vehicle stock, such as setting order status to 'pending' if stock quantity falls below a threshold



PROJECT EXPLANATION WITH REAL-WORLD EXAMPLE

Let's illustrate the project's functionality with a real-world scenario:

Imagine a customer, **John**, places an order for a **Honda EV**

1. **Order Placement and Dealer Assignment:**
    - John creates a new **Vehicle Order** and sets its Status to **Pending**
    - The **Auto-Assign Dealer Flow** is triggered. It identifies John's location (e.g., Hyderabad) and finds the nearest dealer (e.g., **EM, located in USA**
    - The flow automatically updates the Assigned Dealer field on John's order to **EM**. This ensures efficient order routing based on proximity.
2. **Test Drive Reminder:**
    - John later schedules a **Vehicle Test Drive** for the Honda EV, setting the Status to **Scheduled** and selecting a Test Drive Date for tomorrow
    - The **Test Drive Reminder Flow** is triggered. Due to its scheduled path, it will wait until one day before the test drive date.

- One day prior, John receives an automated email reminder, confirming his test drive details and providing options to reschedule, enhancing customer service.

3. **Stock Management and Order Confirmation:**
    - Once John's order for the Honda EV is confirmed (e.g., by changing the Vehicle Order Status to **Confirmed**).
    - The **VehicleOrderTrigger** fires, invoking the VehicleOrderTriggerHandler Apex class.
    - This class automatically **reduces the Stock Quantity** of the Honda EV from 100 to 99. This ensures real-time inventory updates.
    - Later, if the Honda EV's Stock Quantity drops to **zero**, and another customer tries to place an order for it, the **VehicleOrderTrigger** will prevent the creation of that order, displaying an "This vehicle is out of stock" error message. This prevents selling non-existent inventory.
    - The **Batch Apex Classes** further ensure that the system can handle large-scale data processing and periodic checks for stock quantities, maintaining data integrity and business rules over time.

CONCLUSION

The "WhatNext Vision Motors" Capstone Project successfully implemented a robust Salesforce solution to automate critical aspects of vehicle order processing. By strategically utilizing Salesforce Flows for intelligent dealer assignment and customer reminders, and Apex for complex stock management and order prevention, the project achieved its core objectives. This comprehensive system streamlines operations, reduces manual effort, improves customer satisfaction through timely communications, and ensures accurate inventory management, setting a strong foundation for the future of mobility with innovation and excellence.