# Phase 5: Advanced Salesforce Development for WhatNext Vision Motors

Phase 5 of the WhatNext Vision Motors Salesforce CRM implementation focuses on advanced Apex programming concepts, asynchronous processing, and structured development practices. This phase enhances scalability, efficiency, and maintainability by incorporating object-oriented design, trigger frameworks, batch processing, and test-driven development.

## 1. Classes & Objects

Apex classes and objects are used to encapsulate business logic. Reusable classes ensure modularity and simplify maintenance. Custom classes represent entities such as Vehicle, Order, Dealer, and ServiceRequest, providing methods for CRUD operations and business validations.

## 2. Apex Triggers (before/after insert/update/delete)

Apex triggers automate tasks by responding to DML events on objects. In this project, triggers are used to:  
- Validate stock availability before an order is confirmed.  
- Update stock counts after order creation.  
- Maintain audit logs for test drive requests.  
Both before and after context variables are utilized to enforce data integrity.

## 3. Trigger Design Pattern

To avoid recursion and promote scalability, the Trigger Handler pattern is implemented. Each trigger delegates logic to a handler class, ensuring separation of concerns and cleaner code management.

## 4. SOQL & SOSL

Salesforce Object Query Language (SOQL) retrieves data from custom objects such as Vehicle\_\_c, Order\_\_c, and TestDrive\_\_c. SOSL is used for text-based searches across multiple objects, useful for locating customer records or vehicle details. Queries are optimized to avoid governor limit exceptions.

## 5. Collections: List, Set, Map

Collections store and manipulate groups of records efficiently:  
- List: Used for ordered records like pending service requests.  
- Set: Ensures uniqueness, e.g., avoiding duplicate dealer assignments.  
- Map: Facilitates key-value storage, e.g., mapping Dealer IDs to assigned Orders.

## 6. Control Statements

Apex control statements (if-else, switch, loops) streamline decision-making logic, such as routing orders to nearest dealers, validating inputs, or iterating over pending requests for updates.

## 7. Batch Apex

Batch Apex processes large volumes of data asynchronously. In this project, it is used to:  
- Update stock availability in bulk.  
- Send daily summaries of orders to dealers.  
- Process delayed service requests for escalation.

## 8. Queueable Apex

Queueable Apex provides flexible asynchronous processing. It is used for tasks requiring complex logic and chaining, such as sequentially processing orders and sending related notifications.

## 9. Scheduled Apex

Scheduled Apex automates recurring tasks by running classes at defined intervals. For instance:  
- Clean up inactive service requests every weekend.  
- Auto-update order statuses daily.  
- Send weekly test drive reminders to customers.

## 10. Future Methods

Future methods handle lightweight asynchronous tasks, such as sending confirmation emails or updating external systems without delaying the main transaction.

## 11. Exception Handling

Proper exception handling ensures system stability. Try-catch-finally blocks are implemented to manage DML exceptions, callout failures, and null pointer issues, with error logs maintained for administrators.

## 12. Test Classes

Test classes validate Apex logic before deployment. They ensure code coverage of at least 75% by simulating scenarios such as stock unavailability, successful order placement, and test drive scheduling. Assertions verify expected outcomes.

## 13. Asynchronous Processing

This phase leverages asynchronous Apex techniques (Batch, Queueable, Scheduled, and Future methods) to optimize performance. Asynchronous jobs ensure that high-volume data operations and time-intensive tasks do not impact user experience.

## Conclusion

Phase 5 strengthens WhatNext Vision Motors’ Salesforce CRM by introducing advanced development techniques, improving performance, and maintaining scalability. With a combination of object-oriented design, trigger frameworks, SOQL/SOSL, and asynchronous Apex, the system is now robust, efficient, and ready for future enhancements.