

# **Salesforce Project Phase-5 :- Phase 5: Apex Programming (Developer)**

## **Title :- Dairy Management System Private Limited Using Salesforce CRM**

### **1. Classes & Objects**

**Purpose:** Encapsulate business logic for reusability and maintainability.

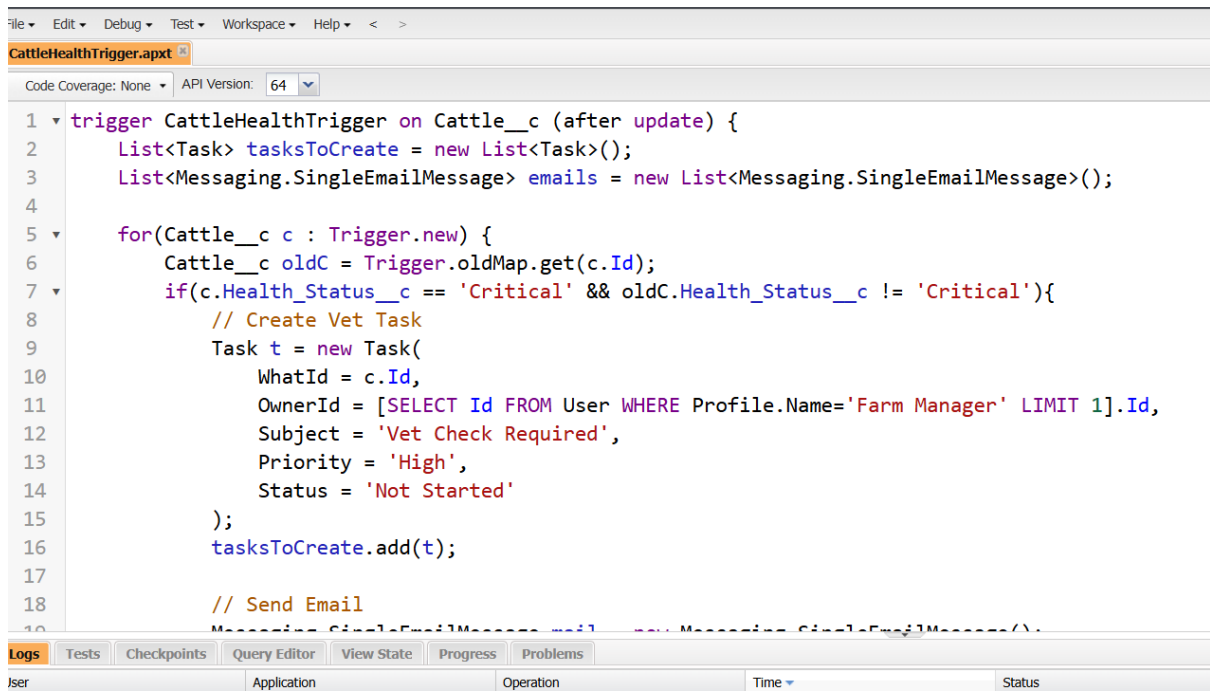
- **OrderService.cls**
    - Handles order-related logic.
    - Responsibilities:
      - Apply discounts (festival, bulk orders).
      - Update inventory stock after order placement.
      - Validate order quantity & limits.
  - **FestivalOfferService.cls**
    - Handles all festival offer logic.
    - Responsibilities:
      - Check if an offer is active/valid.
      - Calculate discount for applicable orders.
      - Handle exceptions if offer is invalid.
- 

### **2. Apex Triggers**

**Purpose:** Execute logic automatically on DML events (insert, update, delete).

- **Before Insert (Order Trigger)**
  - Validate order data (e.g., quantity > 0, valid customer).
  - Example: prevent orders exceeding stock.
- **After Insert (Order Trigger)**

- Apply festival discounts.
- Update inventory quantities.
- Notify relevant parties if needed.
- **Trigger Design Pattern**
  - Use a **handler class** (e.g., OrderTriggerHandler) to keep triggers slim.
  - Trigger only delegates to the handler:

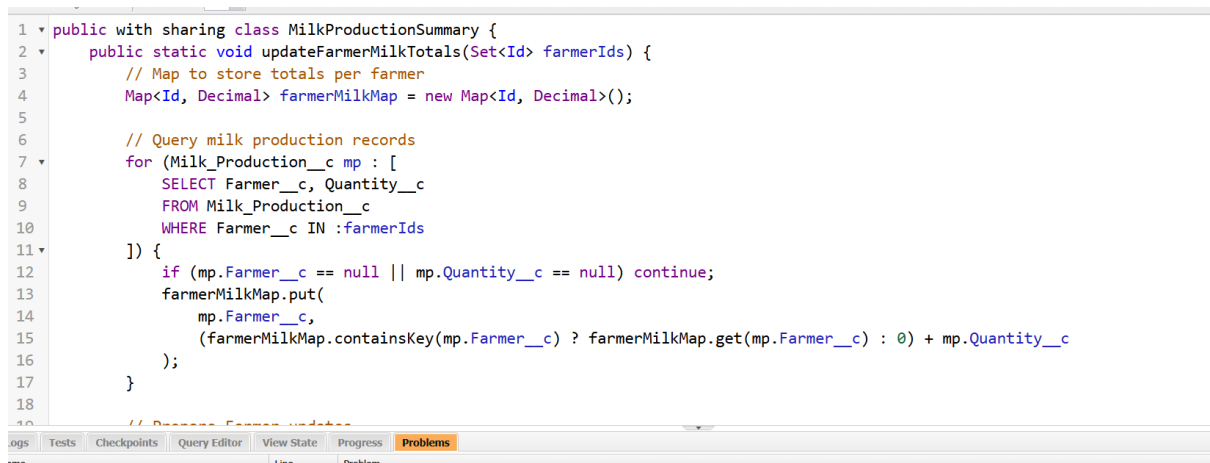


```

1 trigger CattleHealthTrigger on Cattle__c (after update) {
2     List<Task> tasksToCreate = new List<Task>();
3     List<Messaging.SingleEmailMessage> emails = new List<Messaging.SingleEmailMessage>();
4
5     for(Cattle__c c : Trigger.new) {
6         Cattle__c oldC = Trigger.oldMap.get(c.Id);
7         if(c.Health_Status__c == 'Critical' && oldC.Health_Status__c != 'Critical'){
8             // Create Vet Task
9             Task t = new Task(
10                 WhatId = c.Id,
11                 OwnerId = [SELECT Id FROM User WHERE Profile.Name='Farm Manager' LIMIT 1].Id,
12                 Subject = 'Vet Check Required',
13                 Priority = 'High',
14                 Status = 'Not Started'
15             );
16             tasksToCreate.add(t);
17
18             // Send Email
19             Messaging.SingleEmailMessage mail = new Messaging.SingleEmailMessage();
20

```

## Milk Production Summary



```

1 public with sharing class MilkProductionSummary {
2     public static void updateFarmerMilkTotals(Set<Id> farmerIds) {
3         // Map to store totals per farmer
4         Map<Id, Decimal> farmerMilkMap = new Map<Id, Decimal>();
5
6         // Query milk production records
7         for (Milk_Production__c mp : [
8             SELECT Farmer__c, Quantity__c
9             FROM Milk_Production__c
10            WHERE Farmer__c IN :farmerIds
11        ]) {
12            if (mp.Farmer__c == null || mp.Quantity__c == null) continue;
13            farmerMilkMap.put(
14                mp.Farmer__c,
15                (farmerMilkMap.containsKey(mp.Farmer__c) ? farmerMilkMap.get(mp.Farmer__c) : 0) + mp.Quantity__c
16            );
17        }
18
19        // Process Farmer updates
20

```

Debug

Enter Apex Code

1

Id ordId = [SELECT Id FROM Order LIMIT 1].Id;

2

System.enqueueJob(new OrderProcessingQueueable(ordId));

3

☒ Open Log

Execute

Execute Highlighted

CattleHealthTrigger.apxt \* | MilkProductionSummary.apxc \* | OrderDiscountTrigger.apxt \* | MilkProductionBatch.apxc \* | OrderProcessingQueueable.apxc \* | MilkProductionBatchTest.apxc \* | Log execut

Execution Log

Timestamp	Event	Details
10:12:56:000	USER_INFO	[EXTERNAL]005gL00000831AH 30rsunnykumar573@agentforce.com (GMT-07:00) Pacific Daylight Time (America/Los_Angeles) (GMT-07:00)
10:12:56:000	EXECUTION_ST...	
10:12:56:000	CODE_UNIT_ST...	[EXTERNAL]01pgL000005rU3V OrderProcessingQueueable
10:12:56:001	HEAP_ALLOCATE	[95] Bytes:3
10:12:56:001	HEAP_ALLOCATE	[100] Bytes:152
10:12:56:001	HEAP_ALLOCATE	[417] Bytes:408
10:12:56:001	HEAP_ALLOCATE	[430] Bytes:408
10:12:56:001	HEAP_ALLOCATE	[317] Bytes:6
10:12:56:001	HEAP_ALLOCATE	[EXTERNAL] Bytes:5
10:12:56:001	SYSTEM_METH...	[14] QueueableContextImpl.QueueableContextImpl()
10:12:56:001	STATEMENT_EX...	[14]
10:12:56:001	SYSTEM_METH...	[14] QueueableContextImpl
10:12:56:001	HEAP_ALLOCATE	[EXTERNAL] Bytes:8
10:12:56:001	HEAP_ALLOCATE	[EXTERNAL] Bytes:4
10:12:56:001	VARIABLE_SCO...	[21] this[System.QueueableContextImpl true false
10:12:56:001	VARIABLE_ASSI...	[21] this {} 0x7109506c
10:12:56:001	VARIABLE_SCO...	[21] jobId false false
10:12:56:001	VARIABLE_ASSI...	[21] jobId "707gL00000EmpBvQAJ"
10:12:56:009	HEAP_ALLOCATE	[EXTERNAL] Bytes:3
10:12:56:009	METHOD_ENTRY	(11)01not 000005rU3V OrderProcessingQueueable.OrderProcessingQueueable()

☐ This Frame ☐ Executable ☐ Debug Only ☐ Filter

# Batch Apex (For Bulk Operations)

## Use Case in Your Project

- **Milk Production Data Processing:** Farmers upload daily/weekly cattle milk production in bulk. Batch Apex processes these records (quality check, average yield, anomalies).
- **Inventory Stock Updates:** At the end of each day, Batch Apex recalculates stock quantities across warehouses.
- **Order History Archival:** Move old order records (>1 year) to a custom “Archived Orders” object.



The screenshot shows an IDE window with several tabs. The active tab is 'MilkProductionBatchTest.apxc'. The code is an Apex test class named 'MilkProductionBatchTest' with a static test method 'testMilkBatch()'. The test method is divided into three steps: 1. Create test cattle, 2. Run batch, and 3. Query updated cattle. Step 1 creates a list of two cattle records, 'Cow A' and 'Cow B', with different milk quantities. Step 2 runs the 'MilkProductionBatch' class with a batch size of 2. Step 3 queries the database for the two cattle records and returns them as a map.

```
1  @isTest
2  private class MilkProductionBatchTest {
3      @isTest static void testMilkBatch() {
4          // Step 1: Create test cattle
5          List<Cattle__c> cattleList = new List<Cattle__c>();
6          cattleList.add(new Cattle__c(Name='Cow A', Milk_Quantity__c=1)); // Low
7          cattleList.add(new Cattle__c(Name='Cow B', Milk_Quantity__c=9)); // High
8          insert cattleList;
9
10         // Step 2: Run batch
11         Test.startTest();
12         Database.executeBatch(new MilkProductionBatch(), 2);
13         Test.stopTest();
14
15         // Step 3: Query updated cattle
16         Map<String, Cattle__c> cows = new Map<String, Cattle__c>{
17             [SELECT Name, Milk_Quantity__c, Health_Status__c FROM Cattle__c WHERE Name IN ('Cow A', 'Cow B')]
18         };
19     }
20 }
```