

Phase 5: Apex Programming (Developer)

Objective

To implement custom business logic using **Apex Triggers and Classes** in Salesforce. The goal is to automate updates in the **Production Batch** whenever related **Bag** records are created or deleted. This ensures real-time accuracy of production data.

Implementation

1. Apex Trigger – BagRollup

- **Purpose:** Automatically updates Total_Bags_Produced__c on Production_Batch__c whenever Bag records are inserted or deleted.
- **Key Concepts Used:**
 - **SOQL:** Query related Bag records.
 - **Collections:** Set<Id> and Map<Id, Integer>.
 - **Control Statements:** if, for, Trigger.isInsert, Trigger.isDelete.

Code (Example):

```
trigger BagRollup on Bag__c (after insert, after delete) {
```

```
    Set<Id> batchIds = new Set<Id>();
```

```
    if (Trigger.isInsert) {
```

```
        for (Bag__c b : Trigger.new) {
```

```
            if (b.Production_Batch__c != null) {
```

```
                batchIds.add(b.Production_Batch__c);
```

```
            }
```

```
        }
```

```
    }
```

```
    if (Trigger.isDelete) {
```

```
        for (Bag__c b : Trigger.old) {
```

```
            if (b.Production_Batch__c != null) {
```

```

        batchIds.add(b.Production_Batch__c);
    }
}
}

```

```

Map<Id, Integer> batchCounts = new Map<Id, Integer>();
for (AggregateResult ar : [
    SELECT Production_Batch__c batchId, COUNT(Id) cnt
    FROM Bag__c
    WHERE Production_Batch__c IN :batchIds
    GROUP BY Production_Batch__c
]) {
    batchCounts.put((Id)ar.get('batchId'), (Integer)ar.get('cnt'));
}

```

```

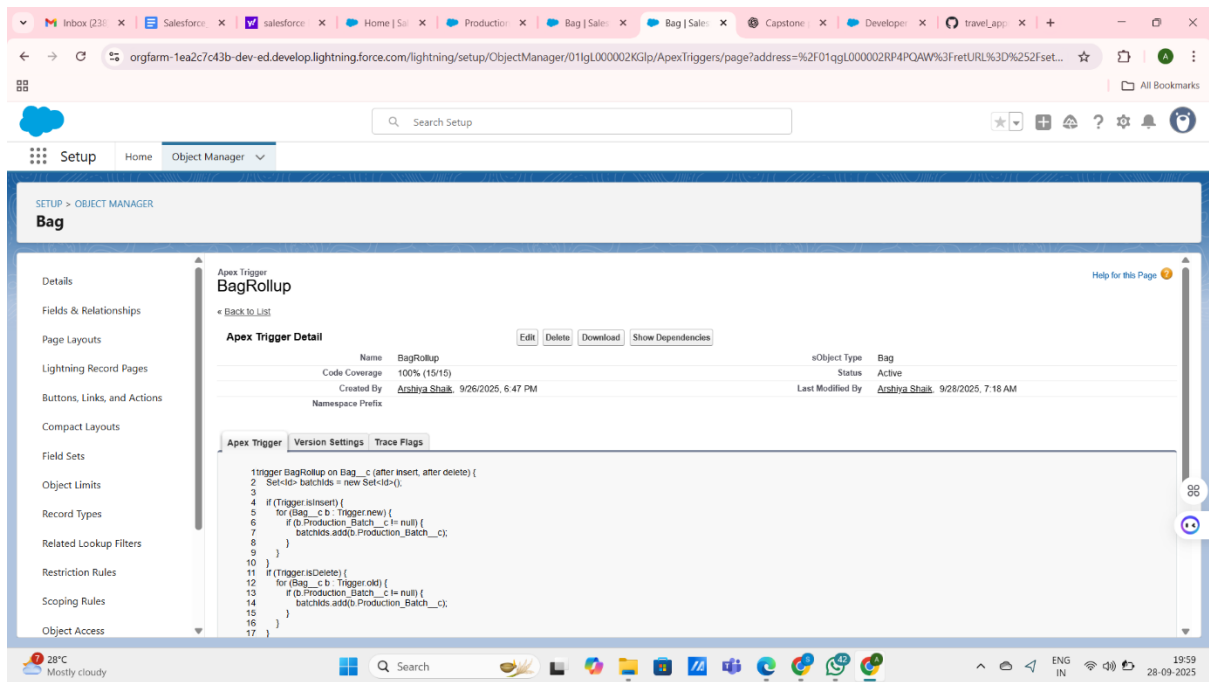
List<Production_Batch__c> updates = new List<Production_Batch__c>();
for (Id batchId : batchCounts.keySet()) {
    updates.add(new Production_Batch__c(
        Id = batchId,
        Total_Bags_Produced__c = batchCounts.get(batchId)
    ));
}

```

```

if (!updates.isEmpty()) {
    update updates;
}
}

```



2. Apex Test Class – BagRollupTest

- **Purpose:** Ensures trigger works correctly by testing insert and delete of Bag records.
- **Key Concepts Used:**
 - **SOQL** inside test.
 - **DML** operations: insert, delete.
 - **Assertions** to check correctness.

Code (Example):

@isTest

```
public class BagRollupTest {
```

```
    @isTest
```

```
    static void testBagRollup_Insert() {
```

```
        Production_Batch__c batch = new Production_Batch__c(
```

```
            Name = 'Batch Test',
```

```
            Quantity__c = 200,
```

```
            Supervisor__c = 'Test Supervisor'
```

```
        );
```

```
insert batch;
```

```
Bag__c bag1 = new Bag__c(  
    Name = 'Bag 1',  
    Packet_Count__c = 50,  
    Production_Batch__c = batch.Id,  
    Production_Date__c = Date.today()  
);
```

```
Bag__c bag2 = new Bag__c(  
    Name = 'Bag 2',  
    Packet_Count__c = 50,  
    Production_Batch__c = batch.Id,  
    Production_Date__c = Date.today()  
);
```

```
insert new List<Bag__c>{ bag1, bag2 };
```

```
batch = [SELECT Id, Total_Bags_Produced__c FROM Production_Batch__c WHERE Id =  
:batch.Id];
```

```
System.assertEquals(2, batch.Total_Bags_Produced__c, 'Bag count should be 2');
```

```
delete bag1;
```

```
batch = [SELECT Id, Total_Bags_Produced__c FROM Production_Batch__c WHERE Id =  
:batch.Id];
```

```
System.assertEquals(1, batch.Total_Bags_Produced__c, 'Bag count should be updated to  
1');
```

```
}
```

}

The screenshot shows the Salesforce Setup interface for the Apex Class **BagRollupTest**. The left sidebar contains navigation links for Setup Home, Salesforce Go, Service Setup Assistant, Commerce Setup Assistant, Field Service Setup Home (Beta), Hyperforce Assistant, Release Updates, Salesforce Mobile App, Lightning Usage, Optimizer, and Sales Cloud Everywhere. The main content area displays the **Apex Class Detail** for **BagRollupTest**, including its Name, Namespace Prefix, Status (Active), Created By (Anshiya Shaik), and Last Modified By (Anshiya Shaik, 9/28/2025, 7:22 AM). Below this, the **Class Body** is shown with the following code:

```
1 @isTest
2 public class BagRollupTest {
3     @isTest
4     static void testBagRollup_InsertDelete() {
5         // Create a Production Batch
6         Production_Batch__c pb = new Production_Batch__c(
7             Name = 'Test Batch',
8             Date__c = Date.today(),
9             Quantity__c = 200
10        );
11        insert pb;
12
13        // Insert Bag linked to this batch
14        Bag__c b = new Bag__c(
15            Production_Batch__c = pb.id,
16            Packet_Count__c = 50,
17            Production_Date__c = Date.today());
18    }
```

3. Test Execution

- Ran the test in **Developer Console** → **Test** → **New Run** → **BagRollupTest**.
- Result: ☒ Test passed successfully.
- Code coverage achieved (must be 75% or higher for deployment).

| Status | Test Run | Enqueued Time | Duration | Failures | Total | Overall Code Coverage | | |
|--------|----------------------------|---------------|----------|----------|-------|-----------------------|--|--|
| ✖ | TestRun @ 7:36:50 pm | | | 1 | 1 | | | |
| ✖ | BagTriggerHandlerTest | | | 1 | 1 | | | |
| ✖ | testBagRollup_InsertDelete | | 0:00 | 1 | 1 | | | |
| ✖ | TestRun @ 7:50:33 pm | | | 1 | 1 | | | |
| ✖ | BagTriggerHandlerTest | | | 1 | 1 | | | |
| ✖ | testBagRollup_InsertDelete | | 0:00 | 1 | 1 | | | |
| ✔ | TestRun @ 7:52:52 pm | | | 0 | 1 | | | |

| Class | Percent | Lines |
|-------------------|---------|-------|
| Overall | 96% | |
| BagRollup | 100% | 15/15 |
| BagTriggerHandler | 94% | 16/17 |