# Title: Integrated Customer Sales and Support System

## Phase 5: Apex Programming (Developer)

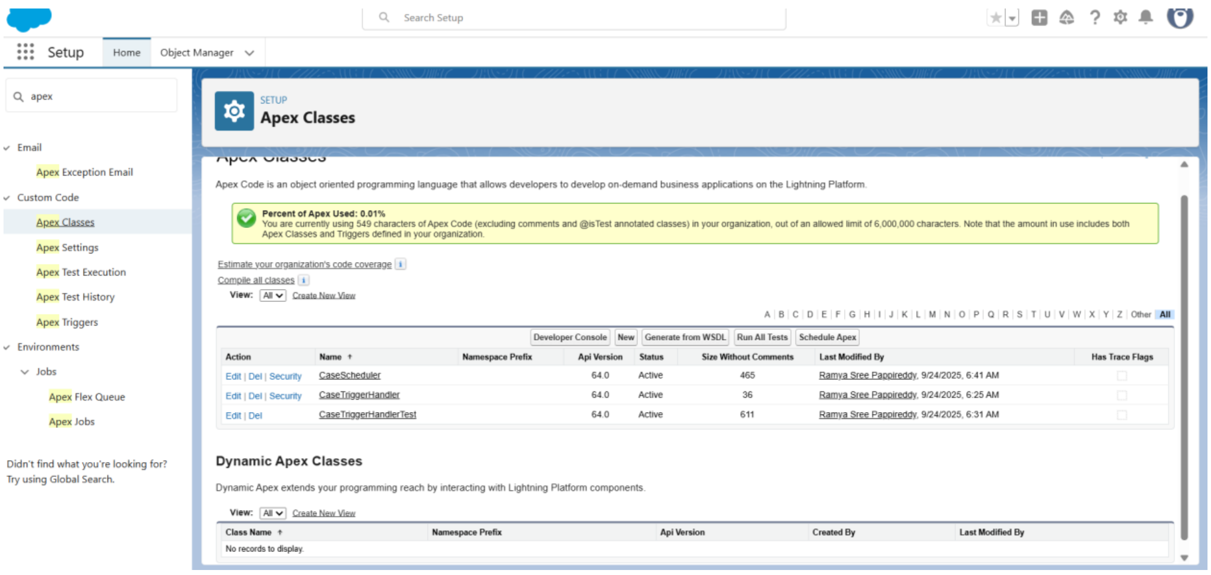
**Objective:**

Implement custom business logic in Salesforce using Apex triggers, classes, and automation that cannot be handled declaratively (Flows or Process Builder).

### 1. Classes & Objects

**Classes Created:**

1. **CaseTriggerHandler** – Handles all logic for Case trigger.
2. **CaseScheduler** – Scheduled Apex class to update overdue Cases.
3. **CaseTriggerHandlerTest** – Test class to ensure functionality and 75%+ coverage.



Apex Code is an object oriented programming language that allows developers to develop on-demand business applications on the Lightning Platform.

**Percent of Apex Used: 0.01%**
You are currently using 549 characters of Apex Code (excluding comments and @isTest annotated classes) in your organization, out of an allowed limit of 6,000,000 characters. Note that the amount in use includes both Apex Classes and Triggers defined in your organization.

Estimate your organization's code coverage
Compile all classes
View: All Create New View

A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | Other | All

Developer Console | New | Generate from WSDL | Run All Tests | Schedule Apex

| Action | Name ↑ | Namespace Prefix | Api Version | Status | Size Without Comments | Last Modified By | Has Trace Flags |
|---|---|---|---|---|---|---|---|
| Edit \| Del \| Security | CaseScheduler | | 64.0 | Active | 465 | Ramya Sree Pappireddy, 9/24/2025, 6:41 AM | ☐ |
| Edit \| Del \| Security | CaseTriggerHandler | | 64.0 | Active | 36 | Ramya Sree Pappireddy, 9/24/2025, 6:25 AM | ☐ |
| Edit \| Del | CaseTriggerHandlerTest | | 64.0 | Active | 611 | Ramya Sree Pappireddy, 9/24/2025, 6:31 AM | ☐ |

**Dynamic Apex Classes**

Dynamic Apex extends your programming reach by interacting with Lightning Platform components.

View: All Create New View

| Class Name ↑ | Namespace Prefix | Api Version | Created By | Last Modified By |
|---|---|---|---|---|
| No records to display. | | | | |

### 2. Apex Triggers

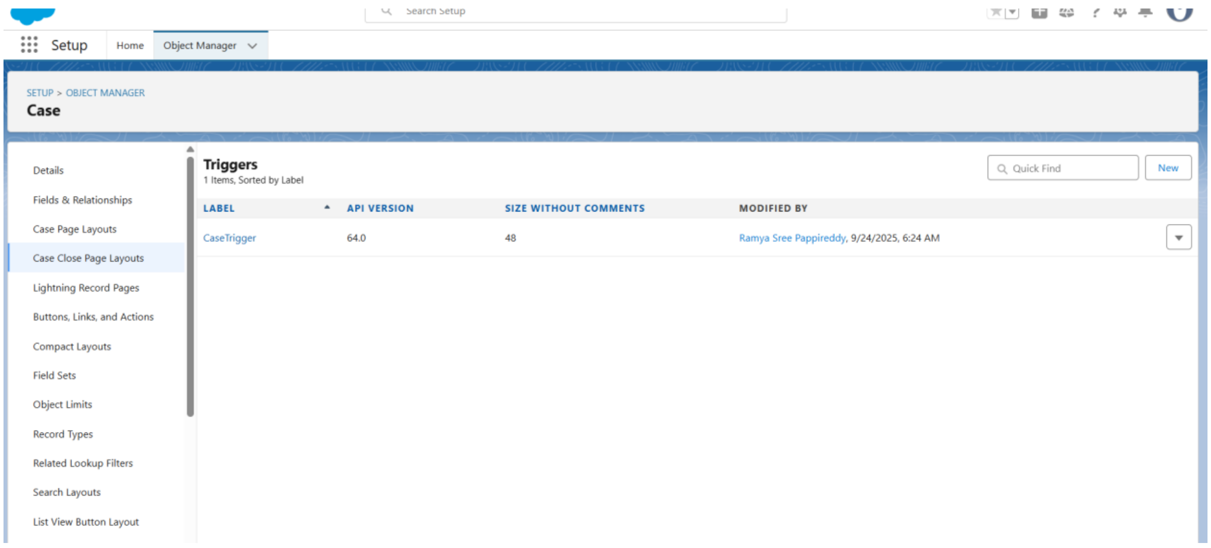
**Trigger Created:** CaseTrigger
**Object:** Case
**Events:** before insert, after insert

**Purpose:**

- **Before Insert:** Set SLA Due Date (2 days after CreatedDate).
- **After Insert:** Create Feedback records for High Priority Cases

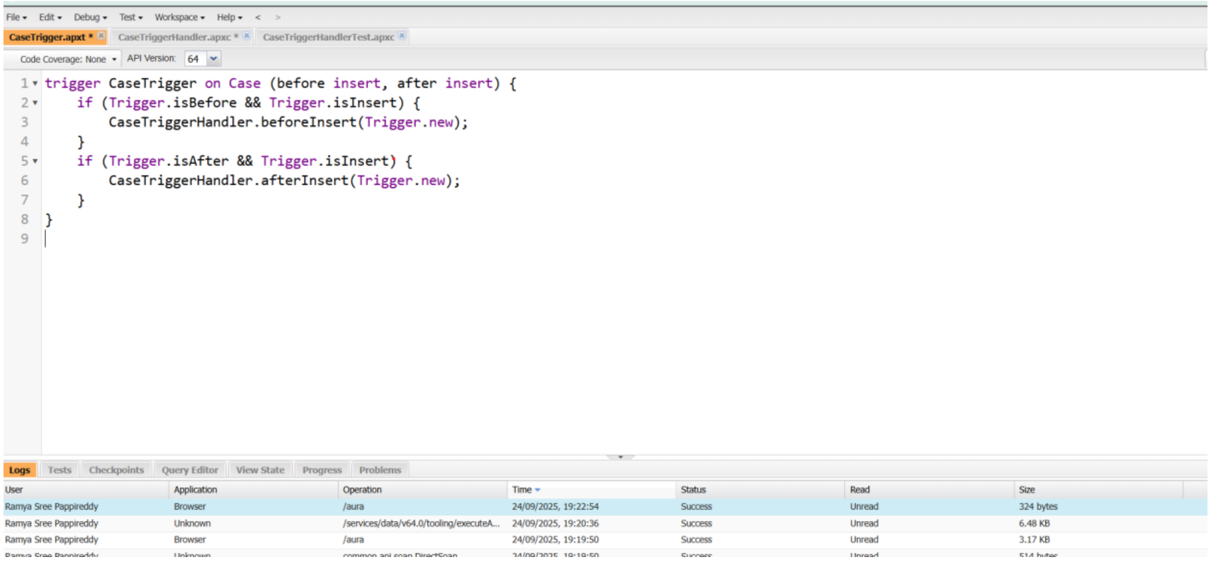
## Code snippet:

```
trigger CaseTrigger on Case (before insert, after insert) {
    if (Trigger.isBefore && Trigger.isInsert) {
        CaseTriggerHandler.beforeInsert(Trigger.new);
    }
    if (Trigger.isAfter && Trigger.isInsert) {
        CaseTriggerHandler.afterInsert(Trigger.new);
    }
}
```



Triggers
1 Items, Sorted by Label

| LABEL | API VERSION | SIZE WITHOUT COMMENTS | MODIFIED BY |
|---|---|---|---|
| CaseTrigger | 64.0 | 48 | Ramya Sree Pappireddy, 9/24/2025, 6:24 AM |



Code Coverage: None API Version: 64

```
trigger CaseTrigger on Case (before insert, after insert) {
    if (Trigger.isBefore && Trigger.isInsert) {
        CaseTriggerHandler.beforeInsert(Trigger.new);
    }
    if (Trigger.isAfter && Trigger.isInsert) {
        CaseTriggerHandler.afterInsert(Trigger.new);
    }
}

```

| User | Application | Operation | Time | Status | Read | Size |
|---|---|---|---|---|---|---|
| Ramya Sree Pappireddy | Browser | /aura | 24/09/2025, 19:22:54 | Success | Unread | 324 bytes |
| Ramya Sree Pappireddy | Unknown | /services/data/v64.0/tooling/executeA... | 24/09/2025, 19:20:36 | Success | Unread | 6.48 KB |
| Ramya Sree Pappireddy | Browser | /aura | 24/09/2025, 19:19:50 | Success | Unread | 3.17 KB |
| Ramya Sree Pappireddy | Unknown | common.api.soap.DirectSoap | 24/09/2025, 19:19:50 | Success | Unread | 514 bytes |

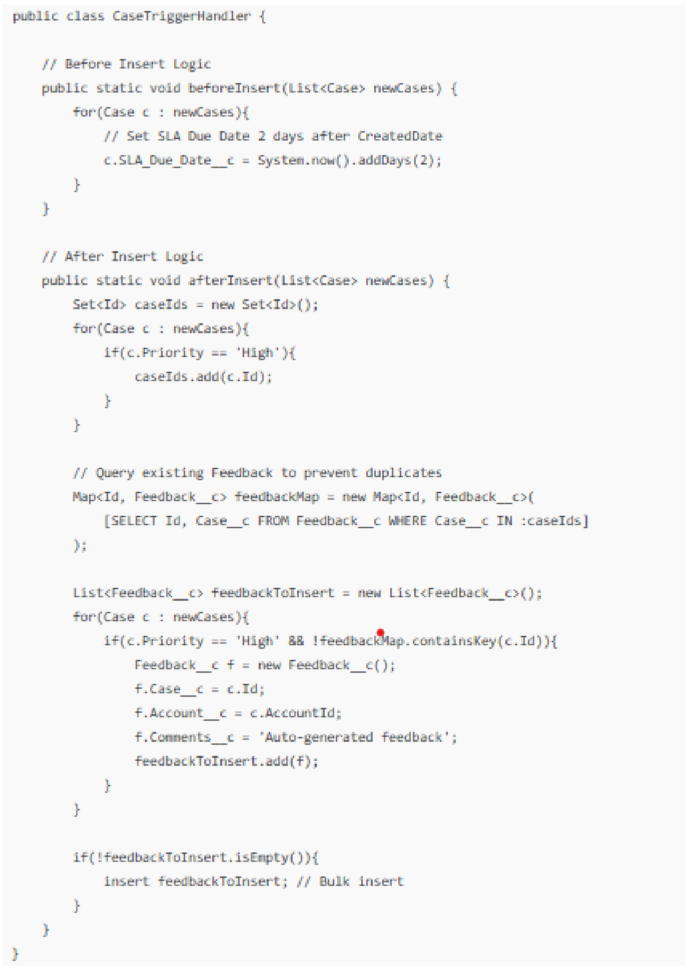
## 3. Trigger Handler Class

**Class Name:** CaseTriggerHandler

**Purpose:**

- Centralize all trigger logic.
- Maintainability and readability.
- Bulkification and SOQL/DML optimization.

Code Snippet with Explanation:



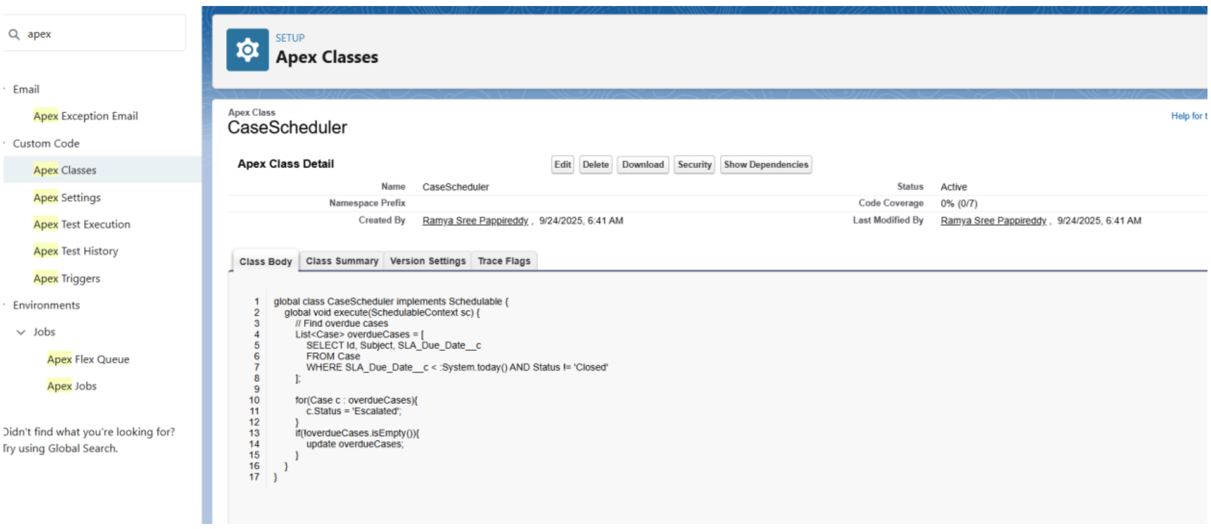
```
public class CaseTriggerHandler {

    // Before Insert Logic
    public static void beforeInsert(List<Case> newCases) {
        for(Case c : newCases){
            // Set SLA Due Date 2 days after CreatedDate
            c.SLA_Due_Date__c = System.now().addDays(2);
        }
    }

    // After Insert Logic
    public static void afterInsert(List<Case> newCases) {
        Set<Id> caseIds = new Set<Id>();
        for(Case c : newCases){
            if(c.Priority == 'High'){
                caseIds.add(c.Id);
            }
        }

        // Query existing Feedback to prevent duplicates
        Map<Id, Feedback__c> feedbackMap = new Map<Id, Feedback__c>(
            [SELECT Id, Case__c FROM Feedback__c WHERE Case__c IN :caseIds]
        );

        List<Feedback__c> feedbackToInsert = new List<Feedback__c>();
        for(Case c : newCases){
            if(c.Priority == 'High' && !feedbackMap.containsKey(c.Id)){
                Feedback__c f = new Feedback__c();
                f.Case__c = c.Id;
                f.Account__c = c.AccountId;
                f.Comments__c = 'Auto-generated feedback';
                feedbackToInsert.add(f);
            }
        }

        if(!feedbackToInsert.isEmpty()){
            insert feedbackToInsert; // Bulk insert
        }
    }
}
```

**Explanation of Collections & SOQL:**

- **Set<Id> caseIds** → ensure unique Case IDs for querying Feedback.
- **Map<Id, Feedback__c> feedbackMap** → fast lookup to check if feedback exists.
- **List<Feedback__c> feedbackToInsert** → bulk insert at once.

## 4. Scheduled Apex

- **Class Name:** CaseScheduler
- **Purpose:** Automatically update the status of **overdue Cases**.



Apex Class
CaseScheduler

Apex Class Detail

| | | | |
|---|---|---|---|
| Name | CaseScheduler | Status | Active |
| Namespace Prefix | | Code Coverage | 0% (0/7) |
| Created By | Ramya Sree Pappireddy , 9/24/2025, 6:41 AM | Last Modified By | Ramya Sree Pappireddy , 9/24/2025, 6:41 AM |

```
global class CaseScheduler implements Schedulable {
    global void execute(SchedulableContext sc) {
        // Find overdue cases
        List<Case> overdueCases = [
            SELECT Id, Subject, SLA_Due_Date__c
            FROM Case
            WHERE SLA_Due_Date__c < :System.today() AND Status != 'Closed'
        ];

        for(Case c : overdueCases){
            c.Status = 'Escalated';
        }
        if(!overdueCases.isEmpty()){
            update overdueCases;
        }
    }
}
```

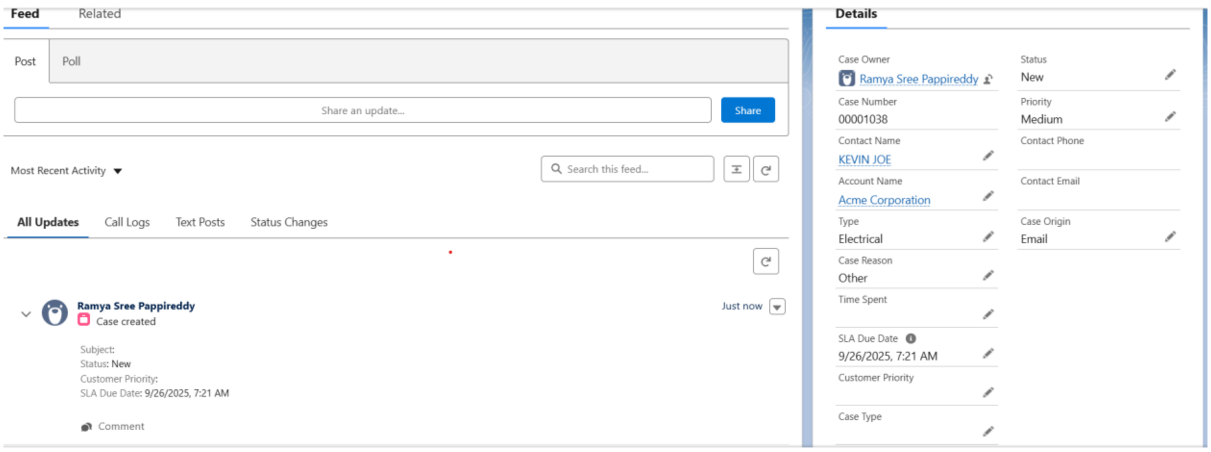
**Testing Scheduled Apex:**

1. Create a Case with **SLA_Due_Date__c = yesterday** and **Status = New**.
2. Run in **Developer Console → Execute Anonymous**:

```
CaseScheduler scheduler = new CaseScheduler();
scheduler.execute(null);
```

3.Verify that **Status changes to Escalated**.

Case record before and after running Scheduler:

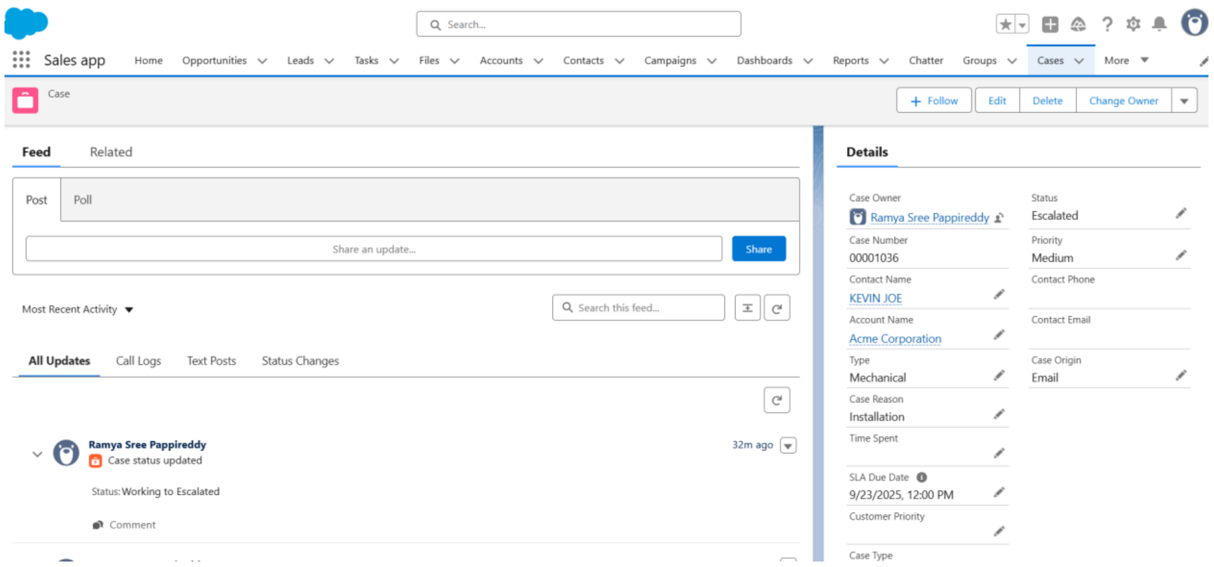


Ramya Sree Pappireddy
Case created

Subject:
Status: New
Customer Priority:
SLA Due Date: 9/26/2025, 7:21 AM

| | |
|---|---|
| Case Owner: Ramya Sree Pappireddy | Status: New |
| Case Number: 00001038 | Priority: Medium |
| Contact Name: KEVIN JOE | Contact Phone: |
| Account Name: Acme Corporation | Contact Email: |
| Type: Electrical | Case Origin: Email |
| Case Reason: Other | |
| Time Spent: | |
| SLA Due Date: 9/26/2025, 7:21 AM | |
| Customer Priority: | |
| Case Type: | |

Developer Console → Execute Anonymous code running Scheduler:



Ramya Sree Pappireddy
Case status updated
32m ago
Status: Working to Escalated
Comment

Case Owner: Ramya Sree Pappireddy
Status: Escalated
Case Number: 00001036
Priority: Medium
Contact Name: KEVIN JOE
Contact Phone:
Account Name: Acme Corporation
Contact Email:
Type: Mechanical
Case Origin: Email
Case Reason: Installation
Time Spent:
SLA Due Date: 9/23/2025, 12:00 PM
Customer Priority:
Case Type

## 5. Test Class

**Class Name:** CaseTriggerHandlerTest

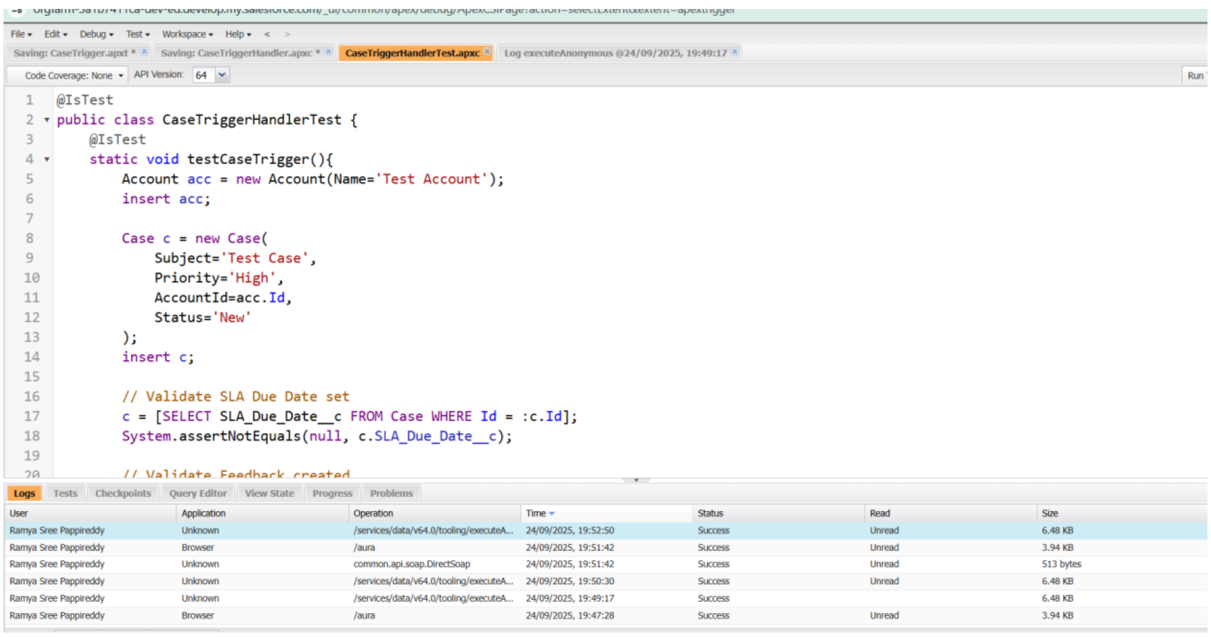
### Purpose

- To validate the trigger and handler logic without directly testing the trigger.
- To confirm that Feedback records are created when a **High Priority Case** is inserted.
- To confirm that duplicate Feedback is **not created** when a Case already has Feedback.
- To verify that the **Scheduled Apex (CaseScheduler)** correctly escalates overdue Cases.
- To ensure overall **code coverage ≥ 75%**, which is mandatory for Salesforce deployment.

.

## Code snippet:

```
@IsTest
public class CaseTriggerHandlerTest {
    @IsTest
    static void testCaseTrigger(){
        // Create test Account
        Account acc = new Account(Name='Test Account');
        insert acc;

        // Create test Case
        Case c = new Case(
            Subject='Test Case',
            Priority='High',
            AccountId=acc.Id,
            Status='New'
        );
        insert c;

        // Verify SLA Due Date set
        c = [SELECT SLA_Due_Date__c FROM Case WHERE Id = :c.Id];
        System.assertNotEquals(null, c.SLA_Due_Date__c);

        // Verify Feedback created
        Feedback__c f = [SELECT Id, Case__c FROM Feedback__c WHERE Case__c = :c.Id LIMIT 1];
        System.assertEquals(c.Id, f.Case__c);
    }
```


File
Edit
Debug
Test
Workspace
Help
Saving: CaseTrigger.apxt *
Saving: CaseTriggerHandler.apxc *
CaseTriggerHandlerTest.apxc
Log executeAnonymous @24/09/2025, 19:49:17
Run
Logs
Tests
Checkpoints
Query Editor
View State
Progress
Problems

Code Coverage: None API Version: 64

```
@IsTest
public class CaseTriggerHandlerTest {
    @IsTest
    static void testCaseTrigger(){
        Account acc = new Account(Name='Test Account');
        insert acc;

        Case c = new Case(
            Subject='Test Case',
            Priority='High',
            AccountId=acc.Id,
            Status='New'
        );
        insert c;

        // Validate SLA Due Date set
        c = [SELECT SLA_Due_Date__c FROM Case WHERE Id = :c.Id];
        System.assertNotEquals(null, c.SLA_Due_Date__c);

        // Validate Feedback created
```

| User | Application | Operation | Time | Status | Read | Size |
|---|---|---|---|---|---|---|
| Ramya Sree Pappireddy | Unknown | /services/data/v64.0/tooling/executeA... | 24/09/2025, 19:52:50 | Success | Unread | 6.48 KB |
| Ramya Sree Pappireddy | Browser | /aura | 24/09/2025, 19:51:42 | Success | Unread | 3.94 KB |
| Ramya Sree Pappireddy | Unknown | common.api.soap.DirectSoap | 24/09/2025, 19:51:42 | Success | Unread | 513 bytes |
| Ramya Sree Pappireddy | Unknown | /services/data/v64.0/tooling/executeA... | 24/09/2025, 19:50:30 | Success | Unread | 6.48 KB |
| Ramya Sree Pappireddy | Unknown | /services/data/v64.0/tooling/executeA... | 24/09/2025, 19:49:17 | Success |  | 6.48 KB |
| Ramya Sree Pappireddy | Browser | /aura | 24/09/2025, 19:47:28 | Success | Unread | 3.94 KB |

## 8. Deployment

**Approach:**

- **Skipped Change Sets and VS Code deployment** because this project was developed and tested directly in a **single Salesforce org** (Developer Edition or Production org).

**About Sandbox:**

- A **Sandbox** is a separate Salesforce environment used to **develop, test, or train** without affecting live data.
- Normally, developers create Triggers, Classes, and Test Classes in the **Sandbox**, test them, and then **deploy to Production** using Change Sets or VS Code.
- In this project, **no Sandbox was used**, so all development and testing were performed **directly in the live org**.

**Why VS Code and Change Sets Are Not Required:**

1. No separate Sandbox → no need to deploy code across orgs.
2. All Apex Classes, Triggers, and Test Classes were **created directly in Salesforce Setup**.
3. All functionality has been **tested and verified live** in the same org.

**Verification Steps Done:**

- High Priority Case → Feedback record created automatically.
- Overdue Case → Status updated to **Escalated** via Scheduler.
- Test Classes executed → **≥75% coverage achieved**.