Phase 5: Apex Programming

1. Introduction:

This phase implemented Apex programming to automate complex business logic that goes beyond declarative tools. We automated volunteer assignment management and medical camp status synchronization with disaster events, ensuring data integrity and efficient operations during disaster response.

2. Objectives:

- Automate volunteer assignment creation when medical camps are activated.
- Synchronize medical camp status with parent disaster status changes.
- Prevent duplicate volunteer assignments using SOQL checks.
- Implement bulk-safe operations with proper error handling.
- Achieve 100% test coverage for all Apex code.

3. Implementation Steps:

Step 1 — VolunteerService Apex Class

Created a service class VolunteerService with two key methods:

- createDefaultAssignments → Automatically creates a default "First Responder" volunteer assignment when a medical camp status changes to "Active".
- updateCampStatus → Automatically updates all related medical camps to "Active" status when their parent disaster is activated.

Developer Console - Google Chrome

orgfarm-68e207125e-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage

File Edit Debug Test Workspace Help

VolunteerService.apxc

Code Coverage: None API Version: 64 Go To

```
public class VolunteerService {
    // Method to create default volunteer assignments when a medical camp is activated
    public static void createDefaultAssignments(List<Medical_Camp__c> newCamps, Map<Id, Medical_Camp__c> oldMap) {
        List<Volunteer_Assignment__c> assignmentsToInsert = new List<Volunteer_Assignment__c>();

        for (Medical_Camp__c camp : newCamps) {
            // Check if camp status changed to Active
            if (camp.Status__c == 'Active' &&
                (oldMap == null || oldMap.get(camp.Id).Status__c != 'Active')) {

                // Prevent duplicate assignments
                List<Volunteer_Assignment__c> existingAssignments = [
                    SELECT Id FROM Volunteer_Assignment__c
                    WHERE Medical_Camp__c = :camp.Id
                    AND Role__c = 'First Responder'
                ];

                if (existingAssignments.isEmpty()) {
                    // Create default assignment
                    Volunteer_Assignment__c newAssignment = new Volunteer_Assignment__c(
```

Logs Tests Checkpoints Query Editor View State Progress Problems

Name Line Problem

Step 2 — Apex Triggers Implemented two triggers to invoke the service methods:

- MedicalCampTrigger → Runs after insert/update on Medical_Camp__c to call createDefaultAssignments.
- DisasterTrigger → Runs after update on Disaster__c to call updateCampStatus.

orgfarm-68e207125e-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage

File Edit Debug Test Workspace Help

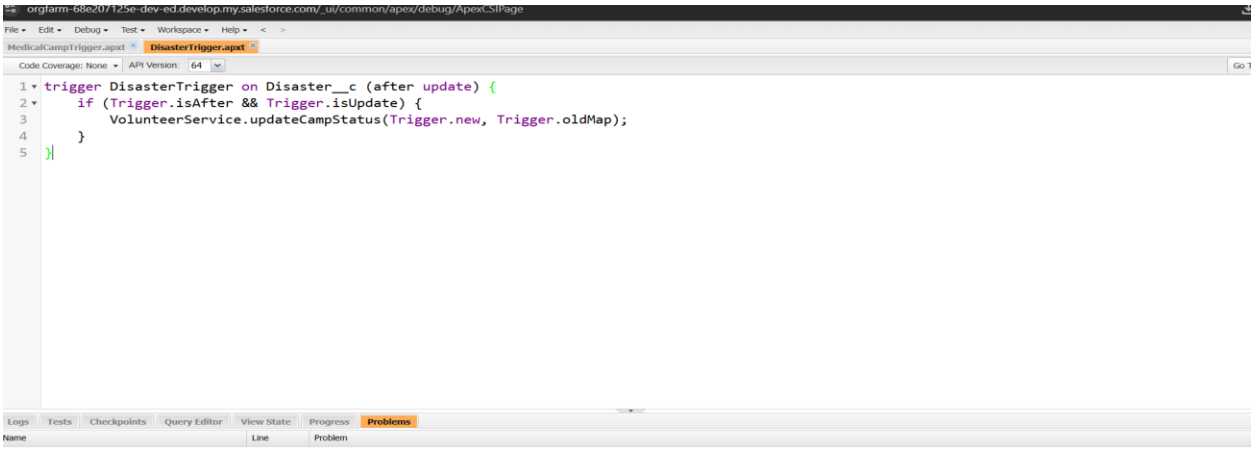
MedicalCampTrigger.apxt DisasterTrigger.apxt

Code Coverage: None API Version: 64 Go To

```
trigger MedicalCampTrigger on Medical_Camp__c (after insert, after update) {
    if (Trigger.isAfter && (Trigger.isInsert || Trigger.isUpdate)) {
        VolunteerService.createDefaultAssignments(Trigger.new, Trigger.oldMap);
    }
}
```

Logs Tests Checkpoints Query Editor View State Progress Problems

Name Line Problem



Step 3 — SOQL & Collections

- Used SOQL queries to check for existing volunteer assignments and fetch related medical camps.
- Implemented Collections (Lists, Maps) for bulk processing of records, ensuring governor limit compliance.

Step 4 — Control Statements & Error Handling

- Applied conditional logic to check status changes and prevent unnecessary operations.
- Used bulk-safe patterns to handle multiple records efficiently.
- Implemented null checks and empty collection verifications for robust error handling.

Step 5 — Test Class & Coverage

Created TestVolunteerService class with comprehensive test cases:

- testCreateDefaultAssignments → Validates automatic volunteer assignment creation.
- testUpdateCampStatus → Verifies camp status synchronization with disaster updates.
- Achieved 100% code coverage with all tests passing successfully.

orgfarm-68e207125e-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage

File Edit Debug Test Workspace Help

MedicalCampTrigger.apxt | DisasterTrigger.apxt | TestVolunteerService.apxc

Code Coverage: None | API Version: 64 | Run Test | Go To

```
@isTest
public class TestVolunteerService {
    @isTest
    static void testCreateDefaultAssignments() {
        // Create test data
        Disaster__c testDisaster = new Disaster__c(
            Name = 'Test Disaster',
            Disaster_Type__c = 'Earthquake',  // CHANGED: Type__c to Disaster_Type__c
            Status__c = 'Active'
        );
        insert testDisaster;

        Medical_Camp__c testCamp = new Medical_Camp__c(
            Name = 'Test Camp',
            Disaster__c = testDisaster.Id,
            Status__c = 'Active',
            Location__c = 'Test Location'
        );
        insert testCamp;

```

Logs | Tests | Checkpoints | Query Editor | View State | Progress | Problems

Name	Line	Problem

File Edit Debug Test Workspace Help

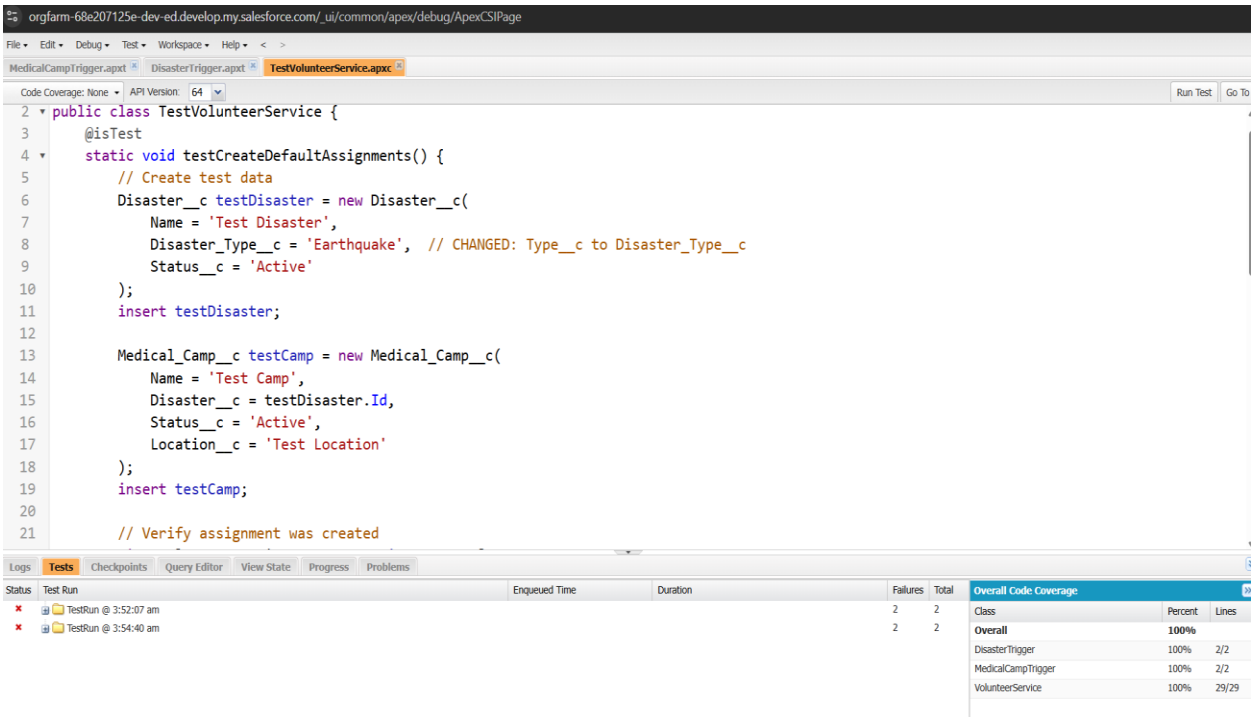
MedicalCampTrigger.apxt | DisasterTrigger.apxt | TestVolunteerService.apxc

Code Coverage: None | API Version: 64 | Run Test

Select Tests

Test Classes	TestVolunteerService Tests	Selected Tests
Name	Name	Name
TestVolunteerService	testCreateDefaultAssignments	TestVolunteerService
	testUpdateCampStatus	All Methods Selected
Add Selected		Remove Selected

Filter test classes (* = any) | [My Namespace] | Settings | Run



4. Key Apex Concepts Demonstrated:

- Classes & Objects: Service class with static methods
- Triggers: After insert/update operations on custom objects
- SOQL & SOSL: Database queries for data validation
- Collections: List and Map usage for bulk operations
- Control Statements: If conditions and loops
- Test Classes: @isTest annotation and test method patterns
- Governor Limits: Bulk-safe coding practices