

## Phase 5 - Apex Programming (Developer)

---

### 1. Objective

To implement **business logic, validations, and automation** in Salesforce for the Smart Lost & Found system using **Apex classes, triggers, batch, and scheduled jobs**.

This phase ensures:

- Lost and Found items have valid dates.
- Old pending claim requests are automatically rejected.
- Notifications are triggered for new claim requests.
- Full test coverage is achieved for deployment.

### 2. Apex Class — LostFoundService

#### Description:

Centralized service class containing all Apex methods for validating items and auto-rejecting old claims.

#### Key Methods:

- **validateLostItem(List<Lost\_Item\_\_c> newList)**
  - Ensures *Lost\_Date\_\_c* is not a future date.
  - Adds an error if validation fails.
- **validateFoundItem(List<FoundItem\_\_c> newList)**
  - Ensures *Found\_Date\_\_c* is not a future date.
  - Adds an error if validation fails.
- **autoRejectOldClaims()**
  - Finds all pending *ClaimRequest\_\_c* records older than 14 days.
  - Updates their *Approval\_Status\_\_c* to **Rejected**.

**Apex Classes**

Apex Class: **LostFoundService**

**Apex Class Detail**

Name	LostFoundService	Status	Active
Namespace Prefix		Code Coverage	0% (0/15)
Created By	CHAYVAKULA LASYAVALLI	Last Modified By	CHAYVAKULA LASYAVALLI
	9/26/2025, 5:41 AM		9/26/2025, 5:41 AM

**Class Body** | Class Summary | Version Settings | Trace Flags

```

1 public class LostFoundService {
2
3     // Validate Lost Items
4     public static void validateLostItem(List<Lost_Item__c> newList) {
5         for (Lost_Item__c l : newList) {
6             if (l.Lost_Date__c != null && l.Lost_Date__c > Date.today()) {
7                 l.addError('Lost Date cannot be in the future.');
8             }
9         }
10    }
11
12    // Validate Found Items
13    public static void validateFoundItem(List<FoundItem__c> newList) {
14        for (FoundItem__c f : newList) {
15            if (f.Found_Date__c != null && f.Found_Date__c > Date.today()) {
16                f.addError('Found Date cannot be in the future.');
17            }
18        }
19    }
20
21    // Auto reject old claims
22    public static void autoRejectOldClaims() {
23        // Logic to reject old claims
24    }
25 }

```

### 3. Triggers

**Purpose:** Execute Apex logic automatically on object operations.

#### Lost Item Trigger

**Apex Triggers**

Apex Trigger: **LostItemTrigger**

**Apex Trigger Detail**

Name	LostItemTrigger	sObject Type	Lost Item
Code Coverage	0% (0/2)	Status	Active
Created By	CHAYVAKULA LASYAVALLI	Last Modified By	CHAYVAKULA LASYAVALLI
	9/26/2025, 5:43 AM		9/26/2025, 5:43 AM

**Apex Trigger** | Version Settings | Trace Flags

```

1 trigger LostItemTrigger on Lost_Item__c (before insert, before update) {
2     if (Trigger.isBefore) {
3         LostFoundService.validateLostItem(Trigger.new);
4     }
5 }

```

[https://org1arm-56d070263-dev-ed.develop.lightning.force.com/lightning/setup/ApexTriggers/page?address=\\_\\_%2F01qgU00000280u1%3FsidclframeOrigin%3Dhttps%253A%252F%252Forg1arm-56d070263-dev-ed.develop.lightning.force.com%26cd%3D01](https://org1arm-56d070263-dev-ed.develop.lightning.force.com/lightning/setup/ApexTriggers/page?address=__%2F01qgU00000280u1%3FsidclframeOrigin%3Dhttps%253A%252F%252Forg1arm-56d070263-dev-ed.develop.lightning.force.com%26cd%3D01)

# Found Item Trigger

The screenshot shows the Salesforce Setup interface for the 'FoundItemTrigger' Apex Trigger. The left sidebar contains navigation links for Setup, Home, Object Manager, and various setup assistants. The main content area displays the trigger details, including its name, code coverage, and creation/modification information. The trigger code is visible in the 'Apex Trigger' tab, showing a trigger on 'FoundItem\_\_c' that calls 'validateFoundItem' when a new record is inserted or updated.

**Apex Trigger Detail**

Name	FoundItemTrigger	sObject Type	FoundItem
Code Coverage	0% (0/2)	Status	Active
Created By	CHAVVAKULA LASYAVALLI	Last Modified By	CHAVVAKULA LASYAVALLI
Created	9/26/2025, 5:45 AM	Last Modified	9/26/2025, 5:45 AM
Namespace Prefix			

**Apex Trigger** | Version Settings | Trace Flags

```
1 trigger FoundItemTrigger on FoundItem__c (before insert, before update) {
2   if (Trigger.isBefore) {
3     LossFoundService.validateFoundItem(Trigger.new);
4   }
5 }
```

# Claim Request Trigger

The screenshot shows the Salesforce Setup interface for the 'ClaimRequestTrigger' Apex Trigger. The left sidebar contains navigation links for Setup, Home, Object Manager, and various setup assistants. The main content area displays the trigger details, including its name, code coverage, and creation/modification information. The trigger code is visible in the 'Apex Trigger' tab, showing a trigger on 'ClaimRequest\_\_c' that calls 'sendEmailNotification' when a new record is inserted.

**Apex Trigger Detail**

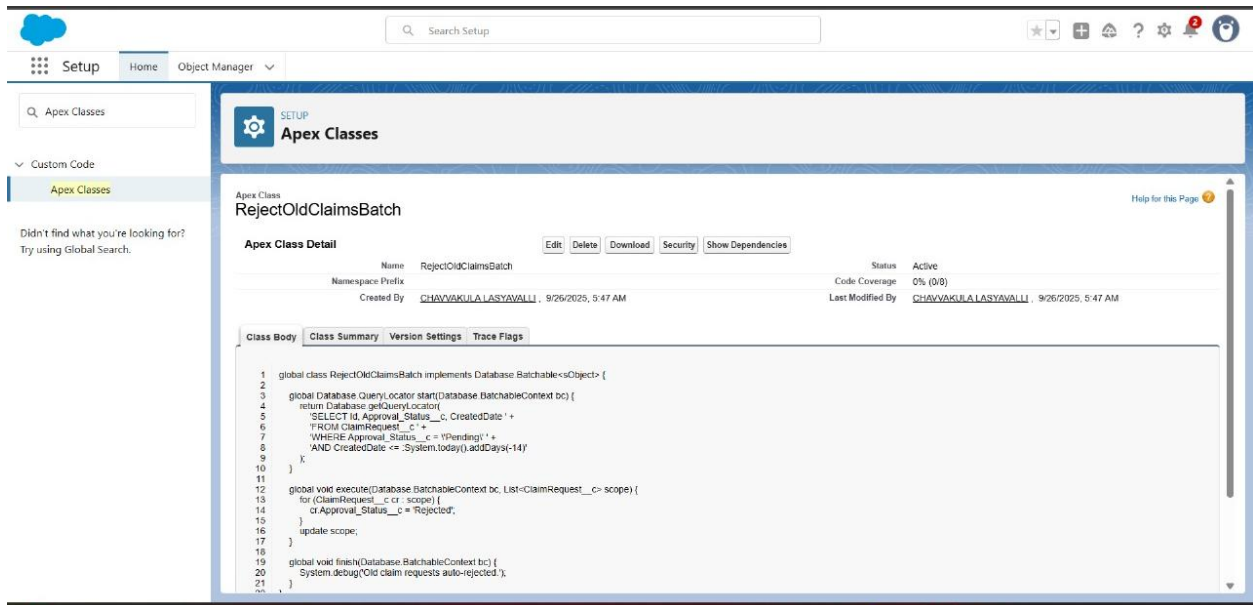
Name	ClaimRequestTrigger	sObject Type	ClaimRequest
Code Coverage	0% (0/1)	Status	Active
Created By	CHAVVAKULA LASYAVALLI	Last Modified By	CHAVVAKULA LASYAVALLI
Created	9/26/2025, 5:46 AM	Last Modified	9/26/2025, 5:46 AM
Namespace Prefix			

**Apex Trigger** | Version Settings | Trace Flags

```
1 trigger ClaimRequestTrigger on ClaimRequest__c (after insert) {
2   if (Trigger.isAfter && Trigger.isInsert) {
3     // Future: send email notification to Security
4     System.debug('Claim Request created. Notify Security');
5   }
6 }
```

## 4. Batch Apex — RejectOldClaimsBatch

**Purpose:** Automatically reject pending claims older than 14 days.



## 5. Scheduled Apex — RejectOldClaimsScheduler

**Purpose:** Runs the batch job daily to maintain data consistency.

**Setup:**

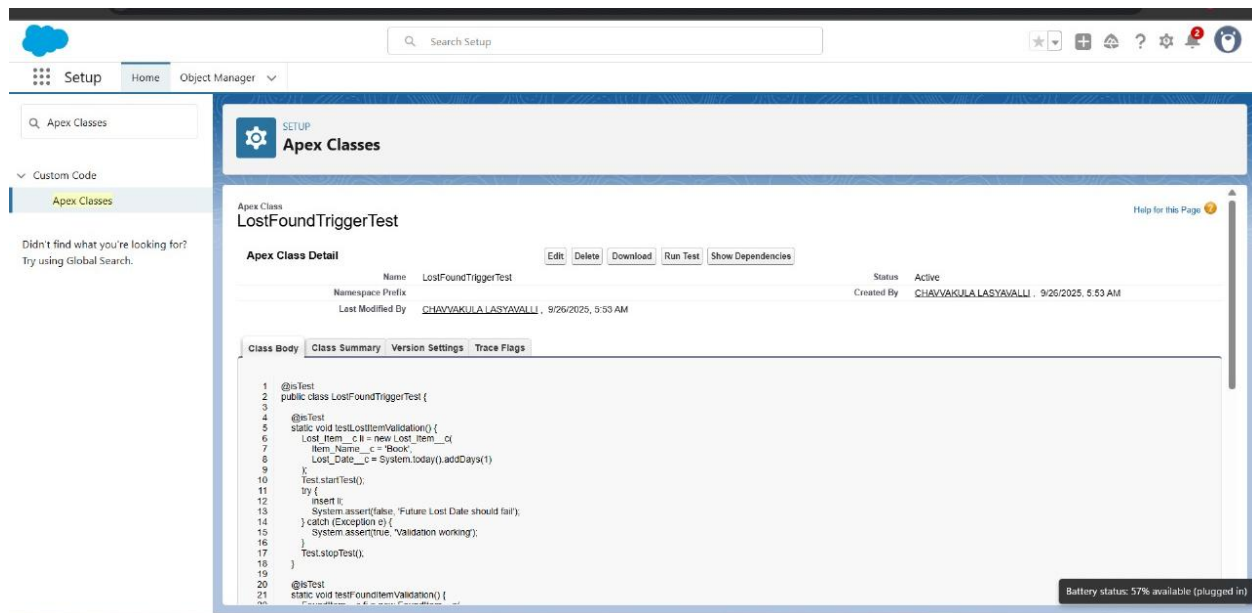
- Setup → Apex Classes → Schedule Apex
- Run daily (e.g., 12:00 AM IST)

## 6. Test Class — LostFoundTriggerTest

**Purpose:** Validate all Apex logic and ensure **75%+ code coverage**.

**Key Points Tested:**

- Future *Lost\_Date\_\_c* triggers validation error.
- Future *Found\_Date\_\_c* triggers validation error.
- Old pending claims are auto-rejected using `LostFoundService.autoRejectOldClaims()`.



## 7. Verification in Org

- Create a **Lost Item** with a future date → Should throw error.
- Create a **Found Item** with a future date → Should throw error.
- Create a **Claim Request** → Check debug logs (simulate notification).
- Run batch manually:
- Database.executeBatch(new RejectOldClaimsBatch());

→ Old pending claims updated to **Rejected**.

## 8. Outcomes

- All future dates for Lost/Found items are blocked.
- Old pending claim requests are automatically rejected.
- Notifications for new claims are simulated (can integrate email later).
- Apex programming enforces **business rules** beyond declarative automation.
- Full test coverage achieved (>75%).