Phase 5: Apex Programming (Developer)

Objective

To extend Salesforce functionality beyond declarative tools (Flows/Validation Rules) using **Apex Triggers, Classes, and Test Methods**. Apex ensures scalability, complex logic execution, and handling of bulk data operations that cannot be fully managed via point-and-click automation.

Key Apex Use Cases in This Project

1. Donation Auto-Assignment Trigger

- **Purpose:** Automatically link donations to the most recent active Event if donor doesn't specify an event.
- **Logic:**
  - On **before insert** of Donation__c.
  - If Related_Event__c is blank → auto-populate with the latest active Event.

2. Volunteer Skills Matching (Future Enhancement)

- **Purpose:** Recommend suitable events for volunteers based on skills + availability.
- **Logic:**
  - Apex Class evaluates volunteer's skill set and compares it with Event requirements.
  - Suggests best-fit events (stored in a custom field or shown via LWC).

3. Bulk Email Handler (Fallback to Flow Limitations)

- **Purpose:** When more than 200 donors need reminders (Flow governor limit exceeded).
- **Logic:**
  - Apex Batch Class processes donations in bulk.
  - Uses Messaging.sendEmail() to send reminders without hitting Flow limits.

4. Custom Validation via Apex (Edge Cases)

- **Purpose:** Advanced business rules beyond simple validation rules.

- **Example:** Prevent duplicate donations (same donor, same amount, same date).

Apex Snippets

Trigger Example – Auto-Assign Donation to Event

```
trigger DonationTrigger on Donation__c (before insert) {
   List<Event__c> events = [SELECT Id FROM Event__c WHERE Event_Date__c >= TODAY ORDER BY Event_Date__c ASC LIMIT 1];
   for(Donation__c don : Trigger.new) {
      if(don.Related_Event__c == null && events.size() > 0) {
         don.Related_Event__c = events[0].Id;
      }
   }
}
```

Batch Apex Example – Monthly Reminder

```
global class MonthlyReminderBatch implements Database.Batchable<sObject> {
   global Database.QueryLocator start(Database.BatchableContext bc) {
      return Database.getQueryLocator([SELECT Id, Donor_Email__c FROM Donation__c WHERE Donation_Type__c = 'Monthly']);
   }
   global void execute(Database.BatchableContext bc, List<Donation__c> donations) {
      List<Messaging.SingleEmailMessage> emails = new List<Messaging.SingleEmailMessage>();
      for(Donation__c don : donations) {
         Messaging.SingleEmailMessage mail = new Messaging.SingleEmailMessage();
         mail.setToAddresses(new String[] {don.Donor_Email__c});
         mail.setSubject('Monthly Donation Reminder');
         mail.setPlainTextBody('This is a friendly reminder for your monthly donation.');
         emails.add(mail);
      }
```

```
        Messaging.sendEmail(emails);
    }
    global void finish(Database.BatchableContext bc) {}
}
```

Tabular Summary

Apex Component	Type	Purpose
DonationTrigger	Trigger	Auto-assigns Event if missing during Donation creation.
VolunteerSkillMatcher	Apex Class	Matches volunteers with events based on skills (future scope).
MonthlyReminderBatch	Batch Apex	Sends bulk monthly reminders without Flow limits.
DuplicateDonationValidator	Apex Trigger	Prevents duplicate donations from being recorded.
Test Classes	Apex Tests	Ensure 75%+ coverage for deployment readiness.

Benefits of Apex Programming

- Handles **bulk processing** (Batch Apex) for large donor/volunteer data.
- Allows **complex logic** not possible in Flows (e.g., duplicate donation prevention).
- Prepares for **future scalability** (AI donor insights, smart volunteer matching).
- Ensures system meets **deployment requirements** (test class coverage ≥ 75%).