Phase 5:

Apex Programming (Lightweight Implementation)

1) Why Use Apex Here?

- Flows handle most automation, but Apex shows we can code when needed.
- Lightweight Apex = **simple trigger + utility class + test class**.
- This satisfies mentors and keeps the project efficient.

2) Apex Trigger – Auto Calculate Impact Score

📍 Purpose: When an **Eco Activity** is inserted, calculate **CO_2 saved** and update related **Impact Score**.

Trigger:

```
trigger EcoActivityTrigger on Eco_Activity__c (after insert,
after update) {
    if(Trigger.isAfter && (Trigger.isInsert ||
Trigger.isUpdate)) {
        EcoActivityHandler.updateImpactScores(Trigger.new);
    }
}
```

3) Apex Handler Class (Best Practice)

📍 Keeps logic out of the trigger for clarity.

```
public class EcoActivityHandler {
    public static void updateImpactScores(List<Eco_Activity__c>
ecoList) {
        Map<Id, Decimal> userImpact = new Map<Id, Decimal>();

        for(Eco_Activity__c eco : ecoList) {
            Decimal co2Saved = (eco.Trees_Planted__c != null ?
eco.Trees_Planted__c * 21 : 0) +
                               (eco.Waste_Recycled__c != null ?
eco.Waste_Recycled__c * 1.5 : 0);

            if(eco.CreatedById != null) {
                if(userImpact.containsKey(eco.CreatedById)) {
                    userImpact.put(eco.CreatedById,
userImpact.get(eco.CreatedById) + co2Saved);
                } else {
                    userImpact.put(eco.CreatedById, co2Saved);
                }
            }
        }

        List<Impact_Score__c> updates = new
List<Impact_Score__c>();
        for(Id userId : userImpact.keySet()) {
            updates.add(new Impact_Score__c(
                User__c = userId,
                Total_CO2_Saved__c = userImpact.get(userId)
            ));
        }

        if(!updates.isEmpty()) {
            upsert updates User__c; // Upsert ensures new or
existing score gets updated
        }
    }
}
```

---

4) Apex Test Class (Mandatory for Deployments)

📍 Proves our code works.

```
@isTest
public class EcoActivityHandlerTest {
    @isTest
    static void testUpdateImpactScores() {
        // Create a test user
        User u = [SELECT Id FROM User WHERE Profile.Name =
'Standard User' LIMIT 1];

        // Insert Eco Activity
        Eco_Activity__c eco = new Eco_Activity__c(
            Name = 'Tree Plantation Drive',
            Activity_Type__c = 'Tree Plantation',
            Trees_Planted__c = 10,
            CreatedById = u.Id
        );
        insert eco;

        // Verify Impact Score created
        Impact_Score__c score = [SELECT Total_CO2_Saved__c FROM
Impact_Score__c WHERE User__c = :u.Id LIMIT 1];
        System.assertEquals(210, score.Total_CO2_Saved__c);
    }
}
```

---

5) Optional Small Enhancements

- **Future Method / Queueable Apex** → Use if we want to simulate async behavior (e.g., sending bulk notifications).

- **Scheduled Apex** → Run monthly summary to recalc all impact scores (optional, but not necessary for capstone).

---

6) Deliverables for Phase 5

1. **Trigger**: `EcoActivityTrigger`

2. **Handler Class**: `EcoActivityHandler`

3. **Test Class**: `EcoActivityHandlerTest`

4. **GitHub Commit**: `phase5: apex trigger, handler, test class for impact score`

---

7) Testing Checklist

- Insert Eco Activity with Trees_Planted = 10 → Check Impact Score = 210.

- Update Eco Activity → Verify CO_2 saved updates correctly.

- Run Test Class → Should pass 100% coverage.