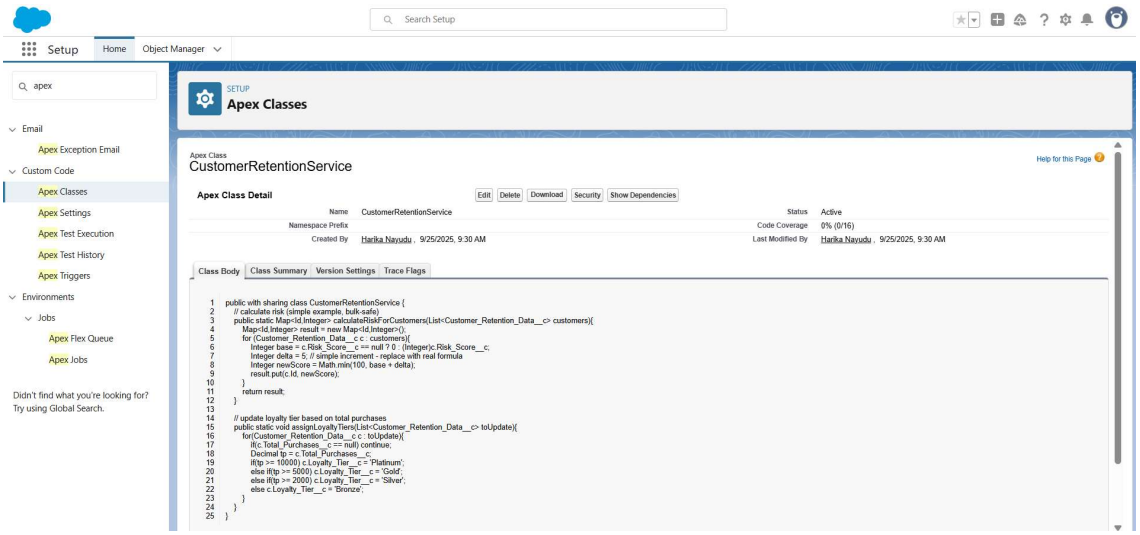
Project Title: Smart Customer Retention & Loyalty Management System

Phase 5: Apex Programming (Developer)

1. Classes & Objects (Utility Class)

Steps

1. Setup → Quick Find → **Apex Classes** → **New**.
2. Paste code → Save.



2. Apex Trigger (before/after insert/update/delete)

Steps

1. Setup → Quick Find → **Object Manager** → choose **Engagement Activity** object.
2. Scroll down → **Triggers** → **New**.
3. Paste code → Save.

Code

```
trigger EngagementActivityTrigger on Engagement_Activity__c (
    before insert, before update, before delete,
    after insert, after update, after delete
) {
    if (Trigger.isBefore) {
        if (Trigger.isInsert)
EngagementActivityTriggerHandler.beforeInsert(Trigger.new);
```

```
        if (Trigger.isUpdate)
EngagementActivityTriggerHandler.beforeUpdate(Trigger.new,
Trigger.oldMap);

        if (Trigger.isDelete)
EngagementActivityTriggerHandler.beforeDelete(Trigger.old);

    }

    if (Trigger.isAfter) {

        if (Trigger.isInsert)
EngagementActivityTriggerHandler.afterInsert(Trigger.newMap);

        if (Trigger.isUpdate)
EngagementActivityTriggerHandler.afterUpdate(Trigger.newMap,
Trigger.oldMap);

        if (Trigger.isDelete)
EngagementActivityTriggerHandler.afterDelete(Trigger.oldMap);

    }

}
```

3. Trigger Design Pattern (Handler Class)

Steps

1. Setup → Apex Classes → **New**.
2. Paste code → Save.

Code

```
public with sharing class EngagementActivityTriggerHandler {

    public static void beforeInsert(List<Engagement_Activity__c>
newList){

        for(Engagement_Activity__c ea : newList){

            if(ea.Activity_Type__c == null) ea.Activity_Type__c =
'Email';

        }

    }


    public static void afterInsert(Map<Id, Engagement_Activity__c>
newMap){

        Set<Id> custIds = new Set<Id>();

        for(Engagement_Activity__c ea : newMap.values()){

            if(ea.Customer_Retention_Data__c != null)
custIds.add(ea.Customer_Retention_Data__c);
```

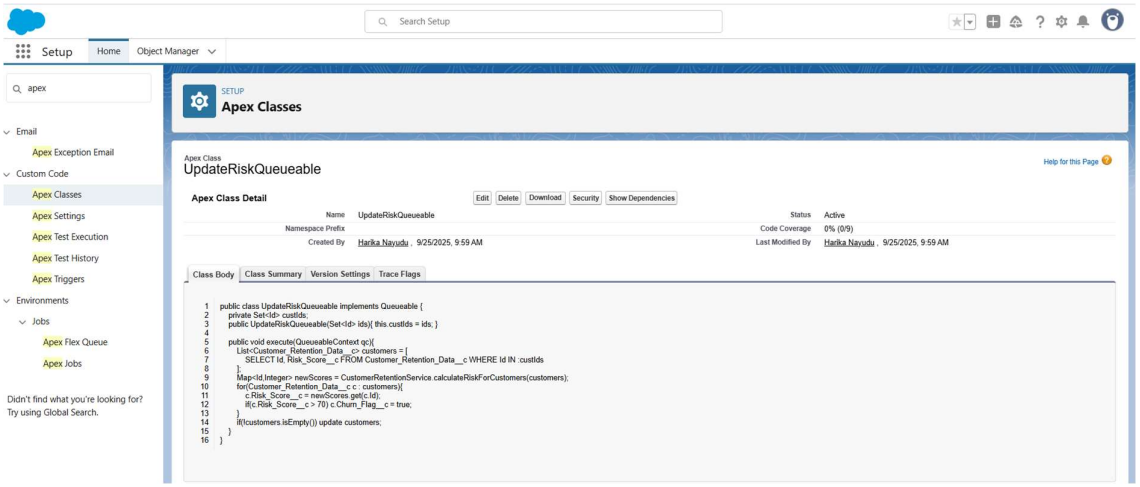
```
        }
        if(!custIds.isEmpty()){
            System.enqueueJob(new UpdateRiskQueueable(custIds));
        }
    }

    // stubs for other events
    public static void beforeUpdate(List<Engagement_Activity__c> newList, Map<Id, Engagement_Activity__c> oldMap){}
    public static void beforeDelete(List<Engagement_Activity__c> oldList){}
    public static void afterUpdate(Map<Id, Engagement_Activity__c> newMap, Map<Id, Engagement_Activity__c> oldMap){}
    public static void afterDelete(Map<Id, Engagement_Activity__c> oldMap){}
}
```

4. Queueable Apex

Steps

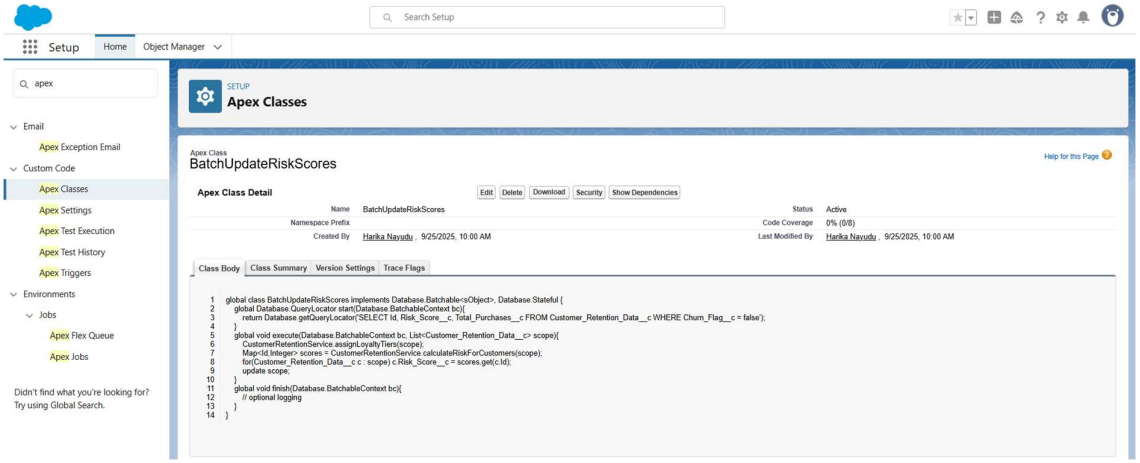
1. Setup → Apex Classes → **New**.
2. Paste code → Save.



5. Batch Apex

Steps

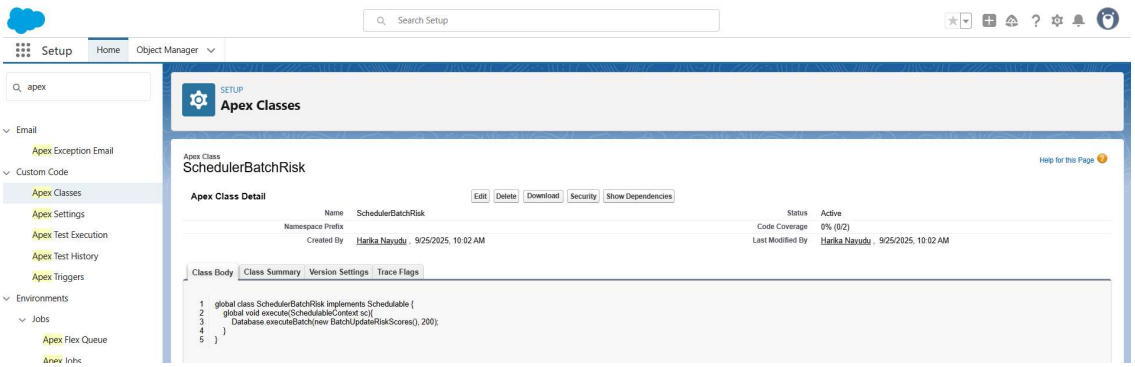
1. Setup → Apex Classes → **New**.
2. Paste code → Save.



6. Scheduled Apex

Steps

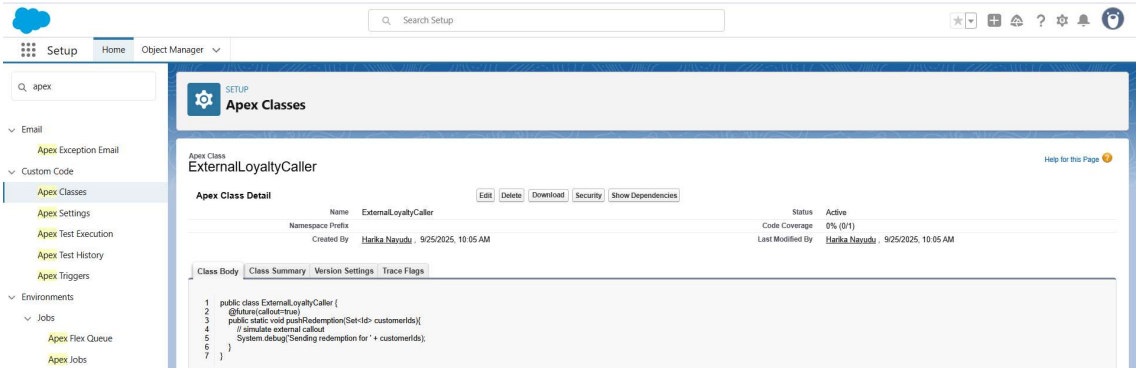
1. Setup → Apex Classes → **New**.
2. Paste code → Save.
3. Setup → Quick Find → **Apex Classes** → **Schedule Apex** → choose this class.



7. Future Method (Example)

Steps

1. Setup → Apex Classes → **New**.
2. Paste code → Save.



8. Test Class

Steps

1. Setup → Apex Classes → **New**.
2. Paste code → Save.

Code

```
@isTest
private class CustomerRetentionTests {
   @isTest static void testRiskUpdate(){
      Customer_Retention_Data__c c = new Customer_Retention_Data__c(Name='T1',
Total_Purchases__c=500);
      insert c;

      Engagement_Activity__c ea = new Engagement_Activity__c(Name='E1',
Customer_Retention_Data__c=c.Id, Activity_Type__c='Call');
      Test.startTest();
        insert ea;
      Test.stopTest();

      Customer_Retention_Data__c c2 = [SELECT Risk_Score__c FROM
Customer_Retention_Data__c WHERE Id=:c.Id];
      System.assertNotEquals(null, c2.Risk_Score__c);
   }
}
```