**Project : Healthcare Telecommunication Systems**

***Phase 5: Apex Programming (Developer)***

1. ***Classes & Objects + Apex Classes***

**Appex classes**

1. Create the patient helper class (PatientHelper)
2. In developer console ---> File ----->New ---> Apex Classs.

**Code :**

public class PatientHelper {

public static void assignCareCoordinator(List<Patient\_\_c> patientList) {

if (patientList == null || patientList.isEmpty()) {

return;

}

User doctor = [

SELECT Id

FROM User

WHERE Username = 'dr.ravikumar@example.com'

LIMIT 1

];

for (Patient\_\_c p : patientList) {

if (p.Region\_\_c == 'East') {

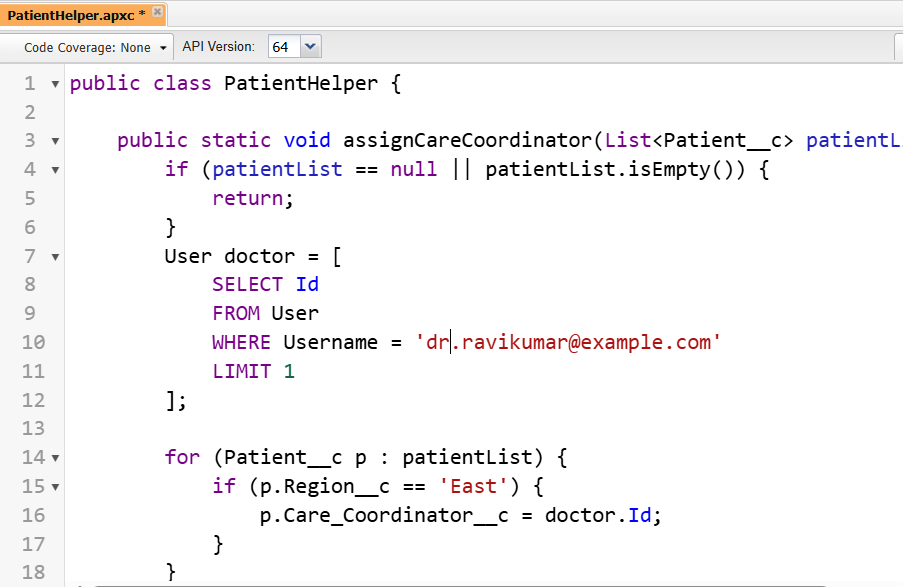
p.Care\_Coordinator\_\_c = doctor.Id;

}

}

}

}



1. Save & test using Developer Console → Execute Anonymous:
2. Apex Code :

List<Patient\_\_c> patients = [

SELECT Id, Region\_\_c

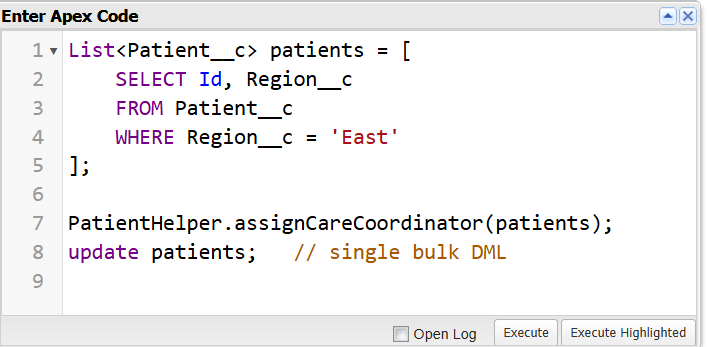
FROM Patient\_\_c

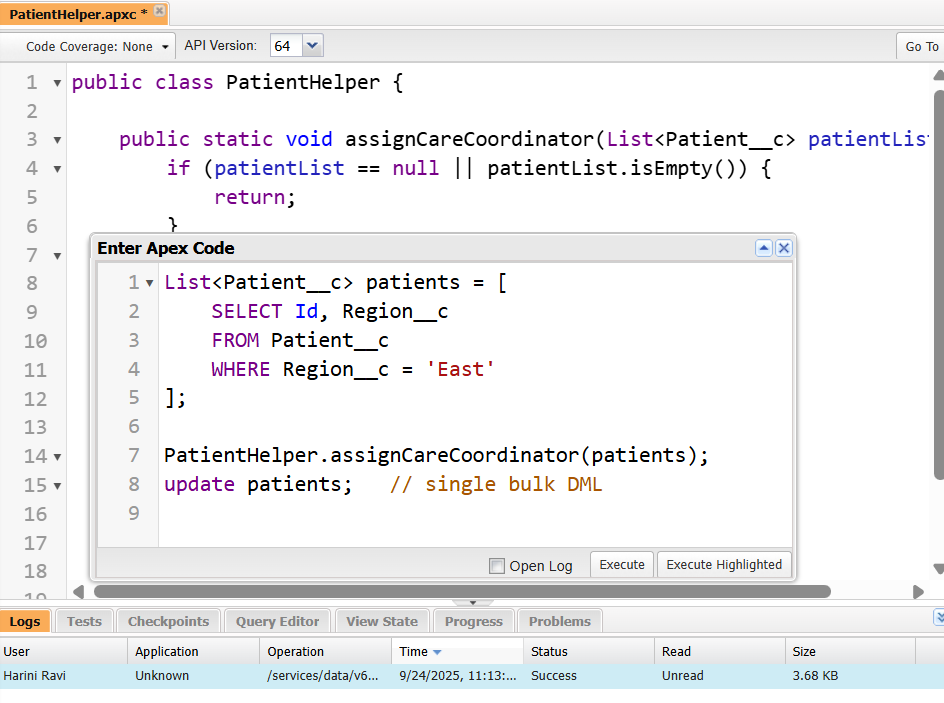
WHERE Region\_\_c = 'East'

];

PatientHelper.assignCareCoordinator(patients);

update patients; // single bulk DML





1. ***Create the trigger (Patient)***

In Developer console --> Apex triggers --> New --> Patient Trigger

**Code :**

trigger PatientTrigger on Patient\_\_c (before insert, before update) {

List<Patient\_\_c> patientsToAssign = new List<Patient\_\_c>();

for (Patient\_\_c p : Trigger.new) {

if (p.Care\_Coordinator\_\_c != null) continue;

if (p.Region\_\_c == 'East' &&

(Trigger.isInsert ||

(Trigger.isUpdate &&

p.Region\_\_c != Trigger.oldMap.get(p.Id).Region\_\_c))) {

patientsToAssign.add(p);

}

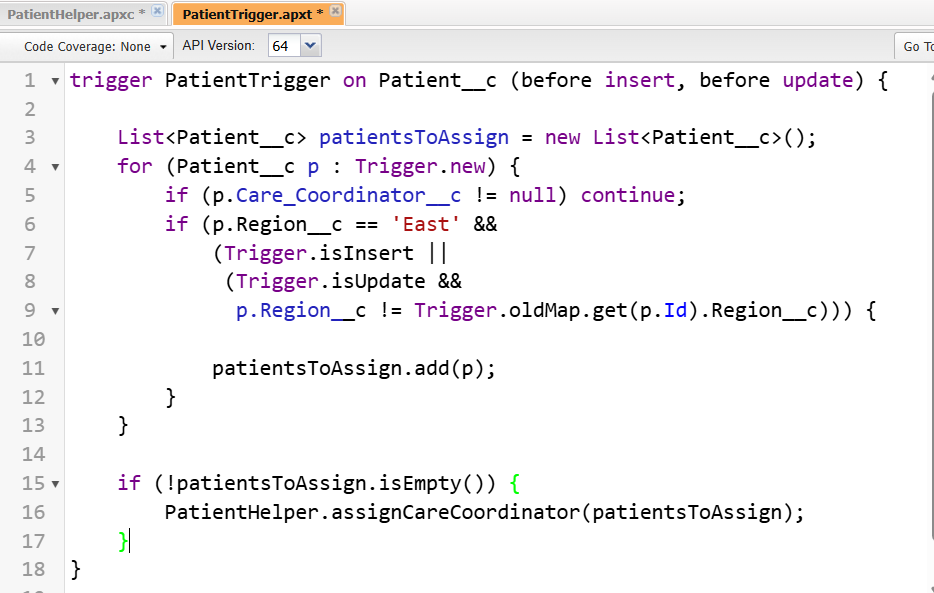
}

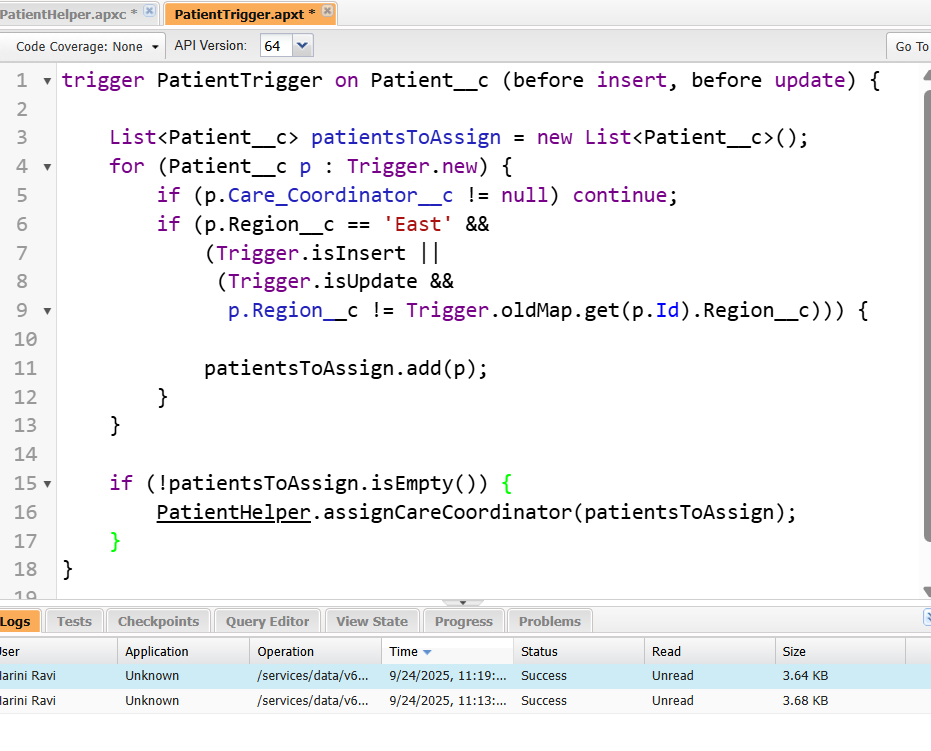
if (!patientsToAssign.isEmpty()) {

PatientHelper.assignCareCoordinator(patientsToAssign);

}

}





1. ***Test Classes***

In developer console---> File----->New--->Apex Classs.

**Name :** PatientTriggerTest

**Code :**

@isTest

private class PatientTriggerTest {

// Create a test user with FirstName='Dr.' and LastName='Ravi Kumar'

private static User createDoctor() {

// Find a profile to use (Standard User if available, otherwise first profile)

Profile prof;

List<Profile> profs = [SELECT Id FROM Profile WHERE Name = 'Standard User' LIMIT 1];

if (profs.isEmpty()) {

prof = [SELECT Id FROM Profile LIMIT 1];

} else {

prof = profs[0];

}

// Make username/email unique for test run

String uniq = String.valueOf(DateTime.now().getTime());

User u = new User(

FirstName = 'Dr.',

LastName = 'Ravi Kumar',

Alias = 'drk' + uniq.substring(uniq.length()-3),

Email = 'dr.ravikumar.' + uniq + '@test.com',

Username = 'dr.ravikumar.' + uniq + '@test.com',

ProfileId = prof.Id,

TimeZoneSidKey = 'Asia/Kolkata',

LocaleSidKey = 'en\_US',

EmailEncodingKey = 'UTF-8',

LanguageLocaleKey = 'en\_US'

);

insert u;

return u;

}

@isTest

static void testAssignCareCoordinator() {

// Arrange

User doctor = createDoctor();

// Prepare patients

Patient\_\_c eastPatient = new Patient\_\_c(

Name = 'East Patient',

Region\_\_c = 'East'

);

Patient\_\_c westPatient = new Patient\_\_c(

Name = 'West Patient',

Region\_\_c = 'West'

);

Test.startTest();

insert new List<Patient\_\_c>{ eastPatient, westPatient };

Test.stopTest();

// Reload and assert

eastPatient = [SELECT Id, Care\_Coordinator\_\_c, Region\_\_c FROM Patient\_\_c WHERE Id = :eastPatient.Id];

westPatient = [SELECT Id, Care\_Coordinator\_\_c, Region\_\_c FROM Patient\_\_c WHERE Id = :westPatient.Id];

System.assertEquals(doctor.Id, eastPatient.Care\_Coordinator\_\_c,

'East patient should be assigned to the doctor created in test.');

System.assertEquals(null, westPatient.Care\_Coordinator\_\_c,

'West patient should not be assigned on insert.');

// Now update West -> East and verify assignment on update

westPatient.Region\_\_c = 'East';

Test.startTest();

update westPatient;

Test.stopTest();

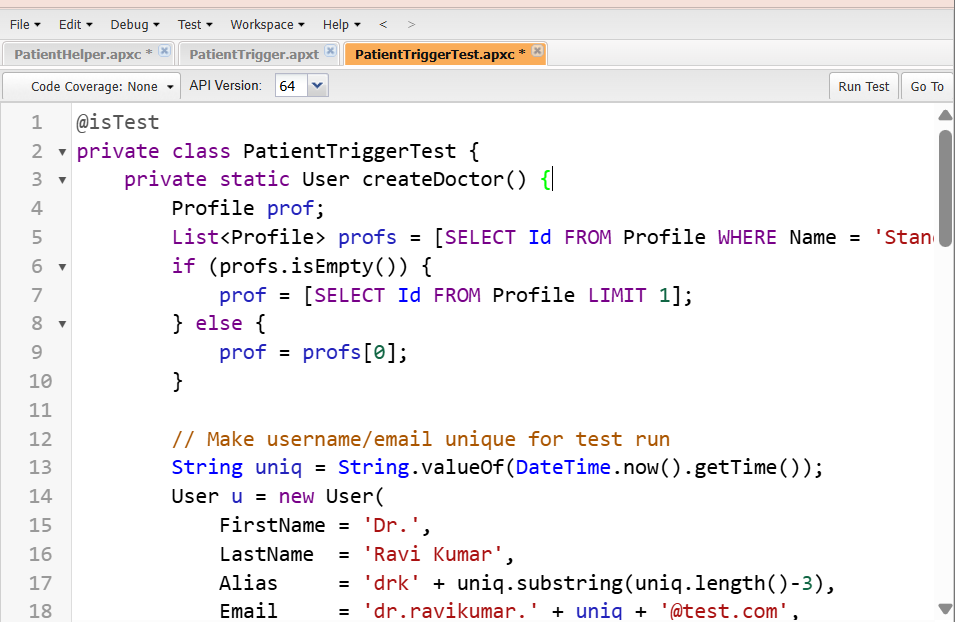
westPatient = [SELECT Id, Care\_Coordinator\_\_c, Region\_\_c FROM Patient\_\_c WHERE Id = :westPatient.Id];

System.assertEquals(doctor.Id, westPatient.Care\_Coordinator\_\_c,

'After changing Region to East, the patient should be assigned the doctor.');

}

}





1. ***SOQL & SOSL***
2. **Using SOQL in Helper Classes**

In developer console---> File----->New--->Apex Classs.

**Name :** AppointmentHelper

**Code :**

public with sharing class AppointmentHelper {

public static void assignDefaultDoctor(List<Appointment\_\_c> apptList) {

if (apptList == null || apptList.isEmpty()) return;

User defaultDoc = [

SELECT Id

FROM User

WHERE FirstName = 'Dr.' AND LastName = 'Ravi Kumar'

LIMIT 1

];

for (Appointment\_\_c a : apptList) {

if (a.Doctor\_\_c == null) { // only if not already assigned

a.Doctor\_\_c = defaultDoc.Id; // assign the queried Id

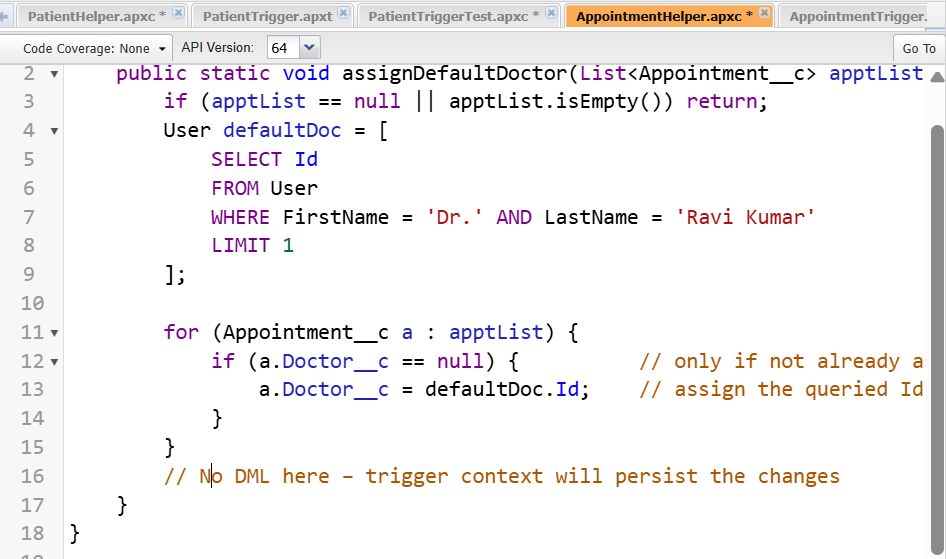
}

}

// No DML here – trigger context will persist the changes

}

}



**Apex Triggers**

**Code :**

trigger AppointmentTrigger on Appointment\_\_c (before insert, before update) {

List<Appointment\_\_c> toProcess = new List<Appointment\_\_c>();

for (Appointment\_\_c a : Trigger.new) {

if (Trigger.isInsert) {

toProcess.add(a);

} else if (Trigger.isUpdate &&

a.Status\_\_c == 'Confirmed' &&

a.Status\_\_c != Trigger.oldMap.get(a.Id).Status\_\_c) {

toProcess.add(a);

}

}

if (!toProcess.isEmpty()) {

AppointmentHelper.assignDefaultDoctor(toProcess);

}

}



1. **Using SOSL in Helper Classes**

In developer console---> File----->New--->Apex Classs.

**Name :** SearchHealper

**Code :**

public with sharing class SearchHelper {

/\*\*

\* Performs a SOSL search for a keyword across Patient\_\_c and Appointment\_\_c.

\* Returns a Map with SObject lists.

\*

\* @param keyword Text to search

\* @return Map<String, List<SObject>> Keys: 'Patient', 'Appointment'

\*/

public static Map<String, List<SObject>> searchPatientsAndAppointments(String keyword) {

Map<String, List<SObject>> resultMap = new Map<String, List<SObject>>{

'Patient' => new List<SObject>(),

'Appointment' => new List<SObject>()

};

if (String.isBlank(keyword)) return resultMap;

// Perform SOSL query

List<List<SObject>> searchResults = [

FIND :keyword IN ALL FIELDS

RETURNING

Patient\_\_c(Id, Name, Region\_\_c),

Appointment\_\_c(Id, Name, Status\_\_c)

];

// Assign results to map

if (!searchResults.isEmpty()) {

// Patient\_\_c results

if (searchResults.size() > 0 && searchResults[0] != null) {

resultMap.put('Patient', searchResults[0]);

}

// Appointment\_\_c results

if (searchResults.size() > 1 && searchResults[1] != null) {

resultMap.put('Appointment', searchResults[1]);

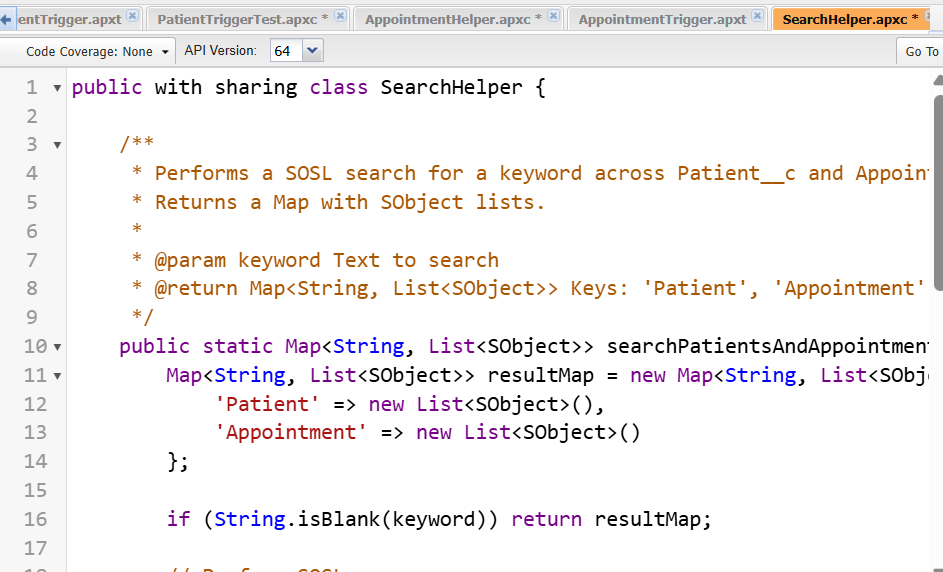
}

}

return resultMap;

}

}



1. ***Collections: List, Set, Map***

**List:** Ordered collection.

**Set:** Unique items, no duplicates.

**Map:** Key-value pair, fast lookup.

**Code :**

List<Patient\_\_c> patients = new List<Patient\_\_c>();

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

Map<Id, Patient\_\_c> patientMap = new Map<Id, Patient\_\_c>();

1. ***Batch Apex***

In developer console---> File----->New--->Apex Classs.

**Purpose**: Process large data asynchronously in batches.

**Name :** AppointmentReminderBatch

**Code :**

global class AppointmentReminderBatch implements Database.Batchable<SObject>, Database.Stateful {

// Optional: track total processed count

global Integer totalProcessed = 0;

// Query to select appointments to process

global Database.QueryLocator start(Database.BatchableContext bc) {

return Database.getQueryLocator([

SELECT Id, Name, Status\_\_c, Region\_\_c, Reminder\_Sent\_\_c

FROM Appointment\_\_c

WHERE Status\_\_c = 'Scheduled' AND Region\_\_c = 'East'

]);

}

// Process each batch of records

global void execute(Database.BatchableContext bc, List<Appointment\_\_c> scope) {

for (Appointment\_\_c appt : scope) {

appt.Reminder\_Sent\_\_c = true; // Update the reminder field

}

update scope; // Bulk-safe DML

totalProcessed += scope.size();

}

// Optional: finish method for post-processing

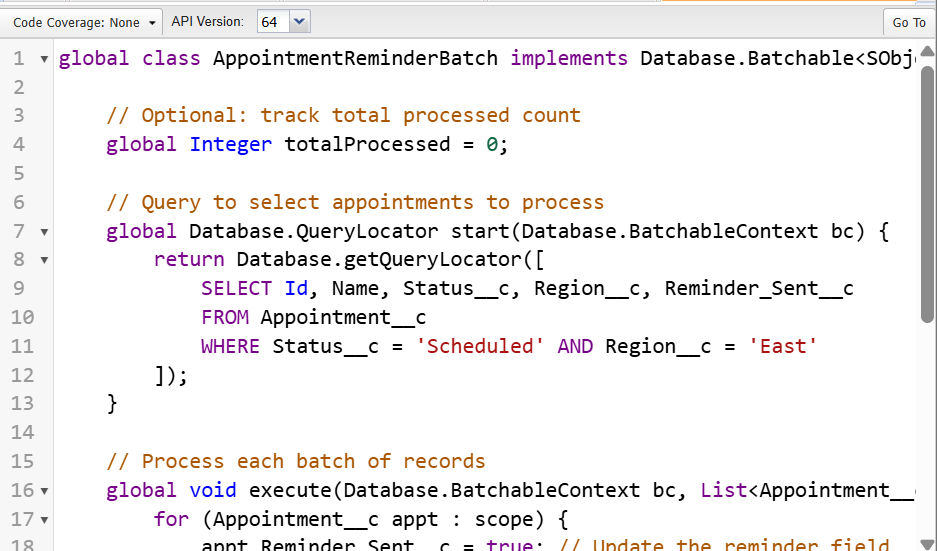
global void finish(Database.BatchableContext bc) {

System.debug('Total Appointments processed: ' + totalProcessed);

// You could also send an email notification to admin here if needed

}

}



1. ***Queueable Apex***

**Purpose:** Asynchronous processing, can chain jobs.

**Code :**

public class AppointmentReminderQueueable implements Queueable {

private String region; // Region parameter

// Constructor to pass region dynamically

public AppointmentReminderQueueable(String region) {

this.region = region;

}

// Execute method runs asynchronously

public void execute(QueueableContext context) {

// Query appointments that are Scheduled and match the region

List<Appointment\_\_c> apptsToUpdate = [

SELECT Id, Name, Status\_\_c, Region\_\_c, Reminder\_Sent\_\_c

FROM Appointment\_\_c

WHERE Status\_\_c = 'Scheduled' AND Region\_\_c = :region

];

// Update Reminder\_Sent\_\_c field

for (Appointment\_\_c appt : apptsToUpdate) {

appt.Reminder\_Sent\_\_c = true;

}

if (!apptsToUpdate.isEmpty()) {

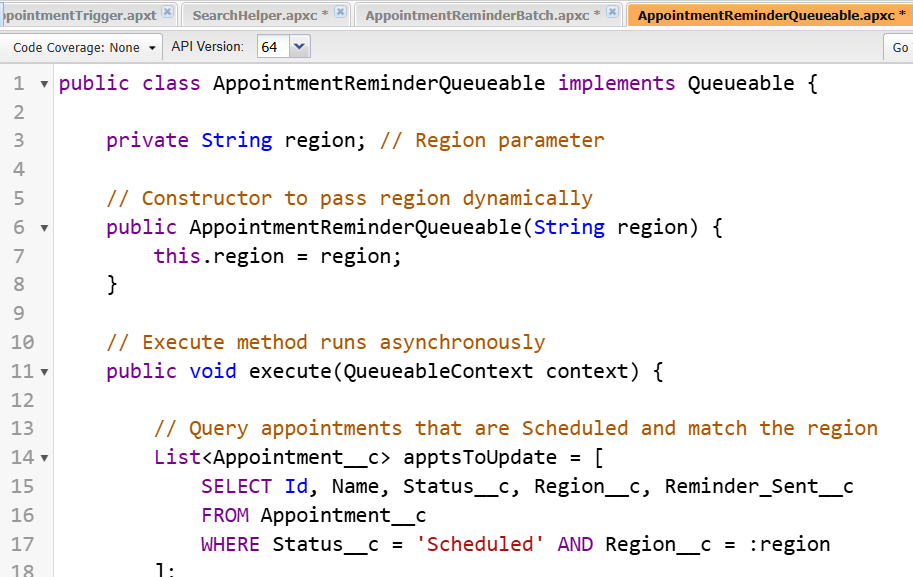
update apptsToUpdate;

}

System.debug('Queueable completed. Total records updated: ' + apptsToUpdate.size());

}

}



***8. Scheduled Apex***

**Purpose:** Run logic at a scheduled time.

In developer console---> File----->New--->Apex Classs.

**Code :**

global class ScheduledPatientJob implements Schedulable {

// This method executes at the scheduled time

global void execute(SchedulableContext sc) {

// Example logic: fetch patients without Care Coordinator and assign one

List<Patient\_\_c> patients = [SELECT Id, Region\_\_c FROM Patient\_\_c WHERE Care\_Coordinator\_\_c = NULL];

User doctor = [SELECT Id FROM User WHERE Name='Dr. Ravi Kumar' LIMIT 1];

for(Patient\_\_c p : patients) {

if(p.Region\_\_c == 'East') {

p.Care\_Coordinator\_\_c = doctor.Id;

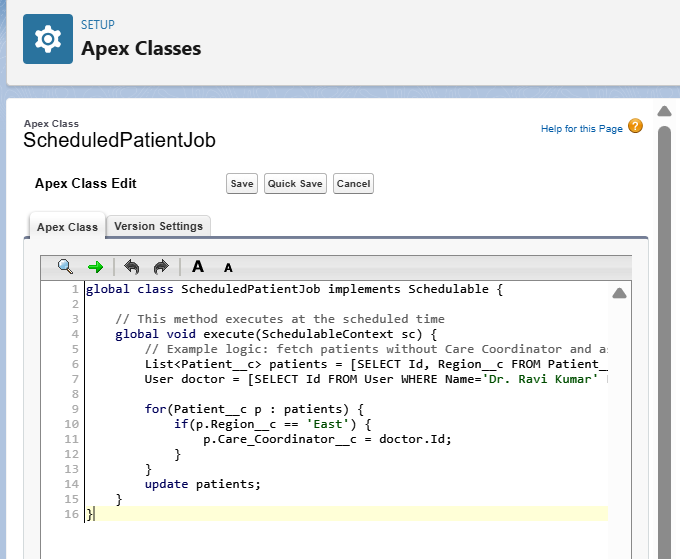
}

}

update patients;

}

}



***Test the Scheduled Apex***

**Name :** ScheduledPatientJobTest

**Code :**

@isTest

public class ScheduledPatientJobTest {

@isTest static void testScheduledJob() {

// Create test data

User doctor = [SELECT Id FROM User WHERE Name='Dr. Ravi Kumar' LIMIT 1];

Patient\_\_c p = new Patient\_\_c(Name='Test Patient', Region\_\_c='East');

insert p;

Test.startTest();

String cronExp = '0 0 2 \* \* ?';

System.schedule('Test Job', cronExp, new ScheduledPatientJob());

Test.stopTest();

// Verify assignment

p = [SELECT Care\_Coordinator\_\_c FROM Patient\_\_c WHERE Id=:p.Id];

System.assertEquals(doctor.Id, p.Care\_Coordinator\_\_c);

}

}

