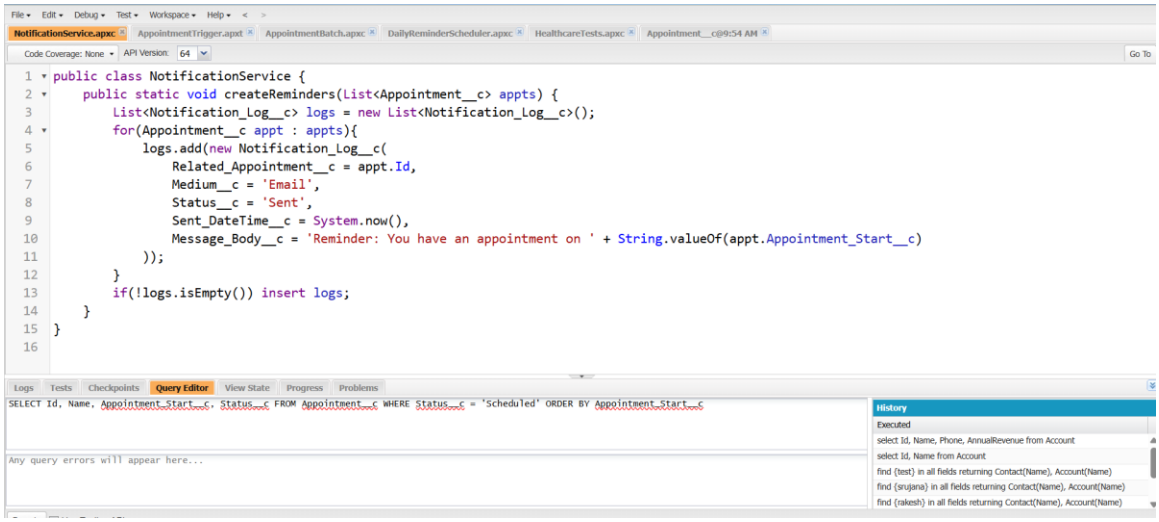
# Smart HealthCare Portal

## Phase 5: Apex Programming (Developer)

In this phase, Apex programming concepts were implemented to enhance the Smart Healthcare CRM project. This included writing Apex classes, triggers, batch processes, scheduled jobs, SOQL and SOSL queries, and test classes to ensure coverage and reliability.

### 1. Apex Classes

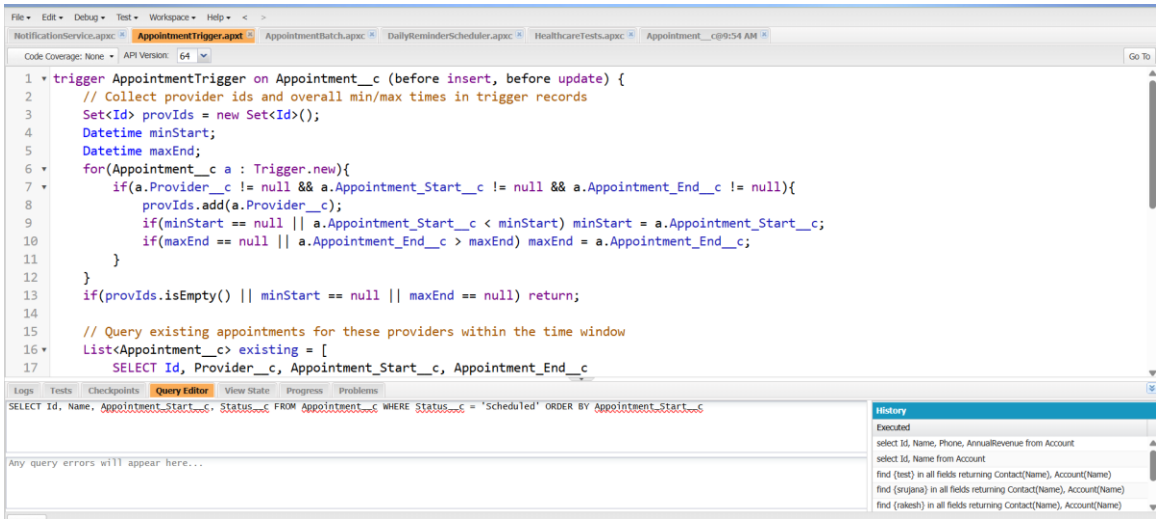
A utility class NotificationService was created to handle automatic reminder creation by inserting Notification_Log__c records when an appointment requires reminders.



```apex
public class NotificationService {
    public static void createReminders(List<Appointment__c> appts) {
        List<Notification_Log__c> logs = new List<Notification_Log__c>();
        for(Appointment__c appt : appts){
            logs.add(new Notification_Log__c(
                Related_Appointment__c = appt.Id,
                Medium__c = 'Email',
                Status__c = 'Sent',
                Sent_DateTime__c = System.now(),
                Message_Body__c = 'Reminder: You have an appointment on ' + String.valueOf(appt.Appointment_Start__c)
            ));
        }
        if(!logs.isEmpty()) insert logs;
    }
}
```

### 2. Apex Triggers

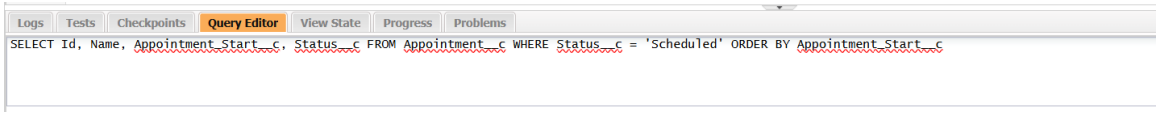
An Apex Trigger on Appointment__c was developed to prevent double booking of providers. It ensures that no two appointments overlap for the same provider.




## 3. SOQL and SOSL Queries

SOQL Example: Fetch scheduled appointments for a provider.

SOSL Example: Search for a patient by name.

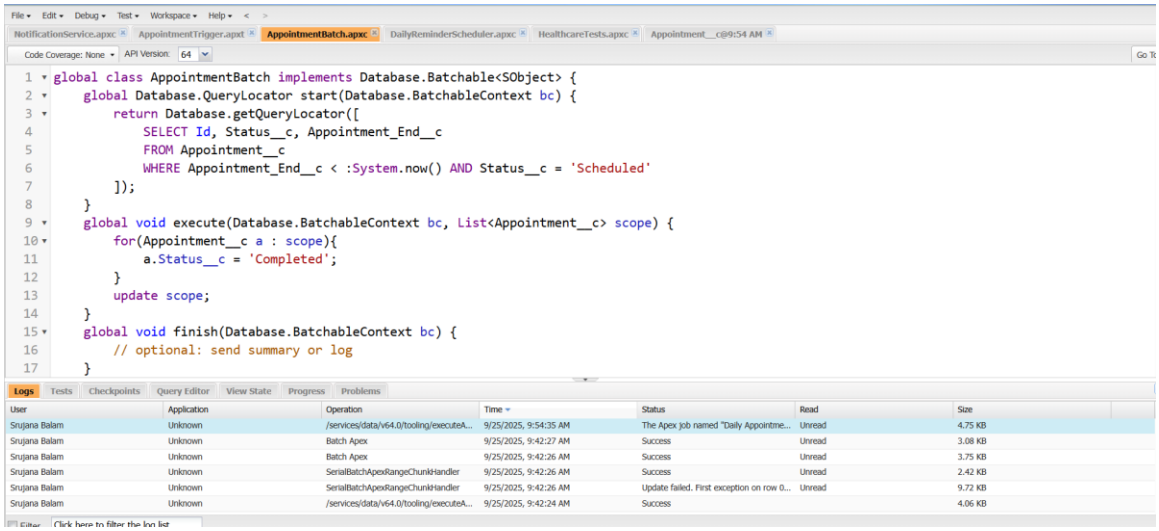



**List<List<Contact>> results = [FIND 'Anita*' IN ALL FIELDS RETURNING Contact(Id, Name, Phone, Email)];**

**System.debug(results);**

## 4. Batch Apex

A batch class AppointmentBatch was created to mark past appointments as Completed automatically.

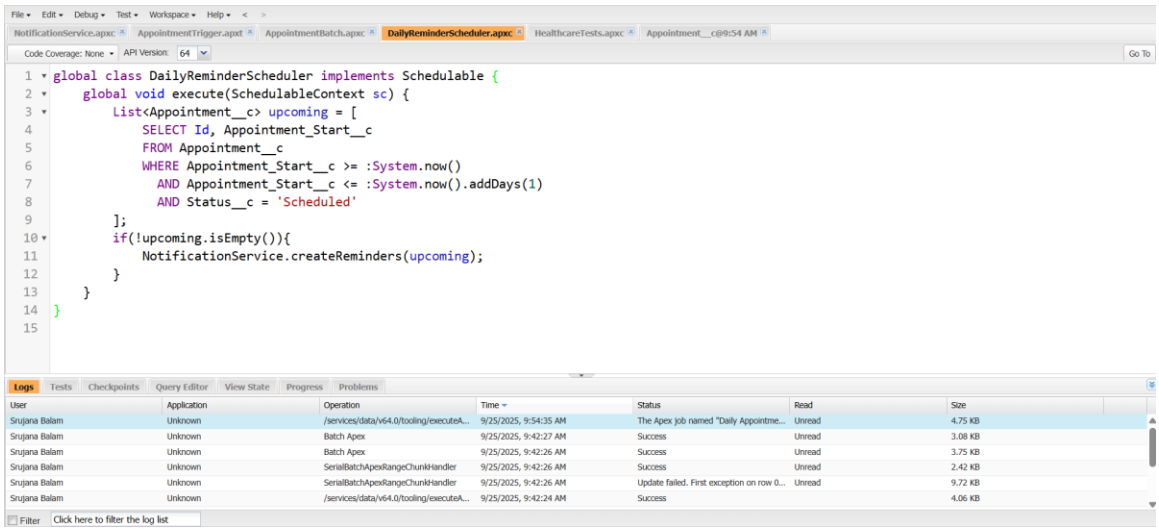


```
global class AppointmentBatch implements Database.Batchable<SObject> {
    global Database.QueryLocator start(Database.BatchableContext bc) {
        return Database.getQueryLocator([
            SELECT Id, Status__c, Appointment_End__c
            FROM Appointment__c
            WHERE Appointment_End__c < :System.now() AND Status__c = 'Scheduled'
        ]);
    }
    global void execute(Database.BatchableContext bc, List<Appointment__c> scope) {
        for(Appointment__c a : scope){
            a.Status__c = 'Completed';
        }
        update scope;
    }
    global void finish(Database.BatchableContext bc) {
        // optional: send summary or log
    }
```

| User | Application | Operation | Time | Status | Read | Size |
|---|---|---|---|---|---|---|
| Srujana Balam | Unknown | /services/data/v64.0/tooling/executeA... | 9/25/2025, 9:54:35 AM | The Apex job named "Daily Appointme... | Unread | 4.75 KB |
| Srujana Balam | Unknown | Batch Apex | 9/25/2025, 9:42:27 AM | Success | Unread | 3.08 KB |
| Srujana Balam | Unknown | Batch Apex | 9/25/2025, 9:42:26 AM | Success | Unread | 3.75 KB |
| Srujana Balam | Unknown | SerialBatchApexRangeChunkHandler | 9/25/2025, 9:42:26 AM | Success | Unread | 2.42 KB |
| Srujana Balam | Unknown | SerialBatchApexRangeChunkHandler | 9/25/2025, 9:42:26 AM | Update failed. First exception on row 0... | Unread | 9.72 KB |
| Srujana Balam | Unknown | /services/data/v64.0/tooling/executeA... | 9/25/2025, 9:42:24 AM | Success | | 4.06 KB |

## 5. Scheduled Apex

A scheduled class DailyReminderScheduler was created to send reminders every morning at 8 AM for appointments scheduled within the next day.

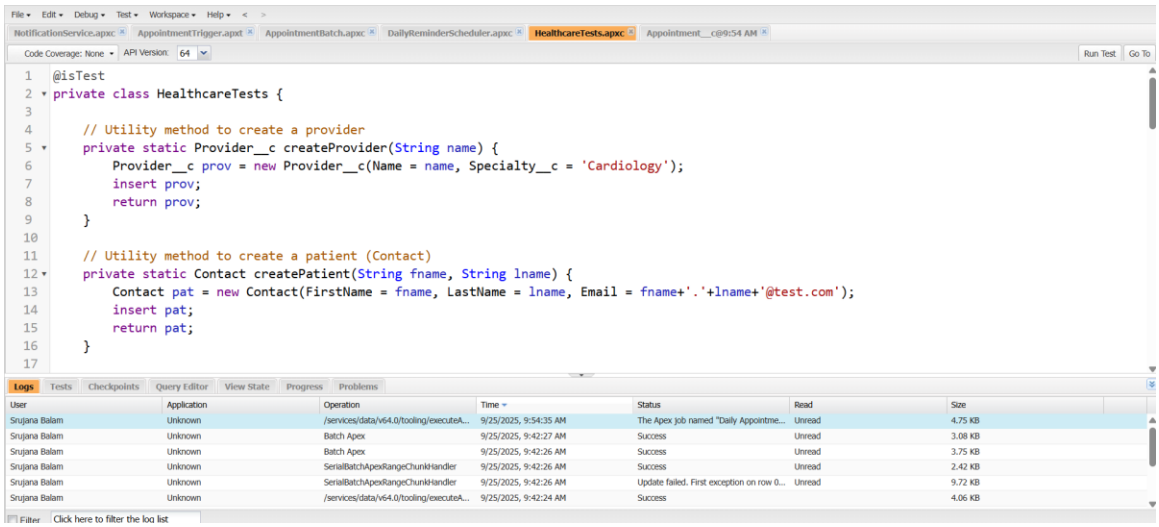


```
global class DailyReminderScheduler implements Schedulable {
    global void execute(SchedulableContext sc) {
        List<Appointment__c> upcoming = [
            SELECT Id, Appointment_Start__c
            FROM Appointment__c
            WHERE Appointment_Start__c >= :System.now()
              AND Appointment_Start__c <= :System.now().addDays(1)
              AND Status__c = 'Scheduled'
        ];
        if(!upcoming.isEmpty()){
            NotificationService.createReminders(upcoming);
        }
    }
}
```

| User | Application | Operation | Time | Status | Read | Size |
|---|---|---|---|---|---|---|
| Srujana Balam | Unknown | /services/data/v64.0/tooling/executeA... | 9/25/2025, 9:54:35 AM | The Apex job named "Daily Appointme... | Unread | 4.75 KB |
| Srujana Balam | Unknown | Batch Apex | 9/25/2025, 9:42:27 AM | Success | Unread | 3.08 KB |
| Srujana Balam | Unknown | Batch Apex | 9/25/2025, 9:42:26 AM | Success | Unread | 3.75 KB |
| Srujana Balam | Unknown | SerialBatchApexRangeChunkHandler | 9/25/2025, 9:42:26 AM | Success | Unread | 2.42 KB |
| Srujana Balam | Unknown | SerialBatchApexRangeChunkHandler | 9/25/2025, 9:42:26 AM | Update failed. First exception on row 0... | Unread | 9.72 KB |
| Srujana Balam | Unknown | /services/data/v64.0/tooling/executeA... | 9/25/2025, 9:42:24 AM | Success | | 4.06 KB |

## 6. Test Classes

A test class HealthcareTests was developed to cover triggers, batch apex, and the notification service. It creates test data including providers and patients (Contacts), verifies that double booking is prevented, ensures batch jobs mark past appointments completed, and confirms notification logs are created.




## 7. Summary

In Phase 5, Apex programming was applied to automate and enforce complex business logic that could not be handled by declarative tools alone. This included preventing provider double-booking, automatically marking past appointments as completed, sending daily reminders, and ensuring robust testing with Apex test classes. Together, these features strengthened the Smart Healthcare CRM and made it more reliable for end users.