Project HealthConnect 360

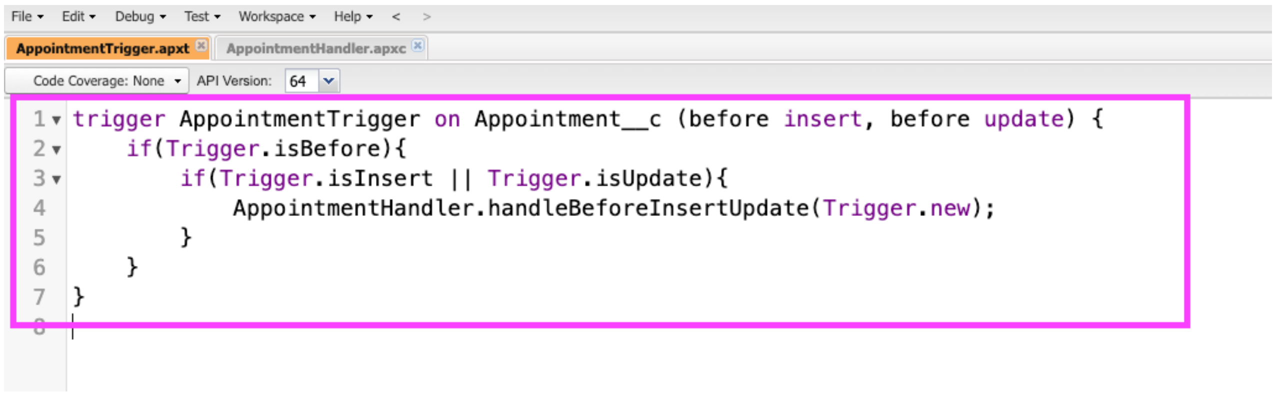
Phase 5: Apex Development & Testing

A Unified Patient Relationship & Clinic Management System

Prepared by: Harsh Raj Mishra

Apex Triggers

- Created a trigger on the **Appointment** object for **before insert** and **before update** events.
- The trigger is simple and calls the handler class method to process incoming appointment records.
- Role: Automate appointment status changes beyond what flows can do.

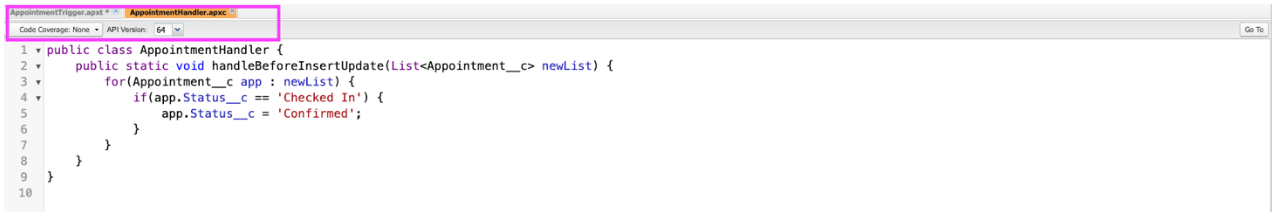


Code Coverage: None
API Version: 64

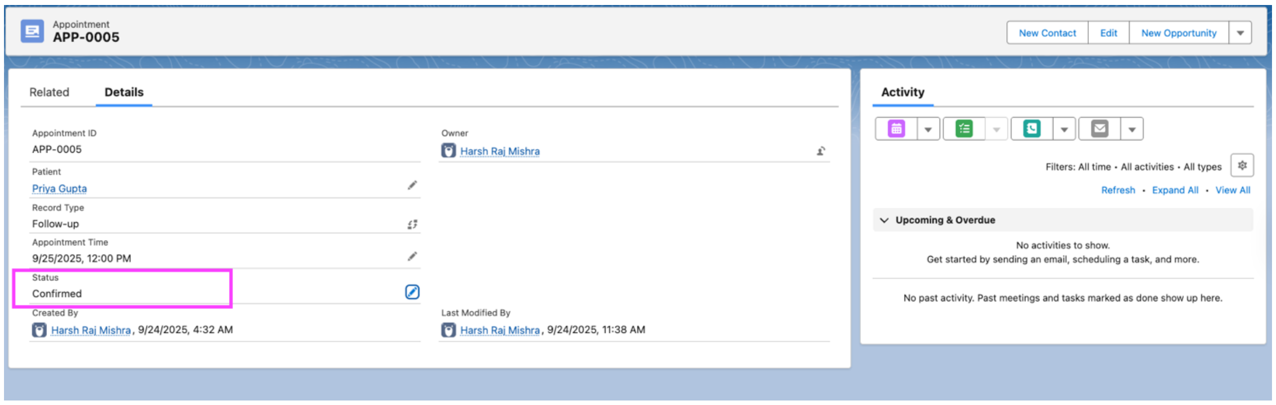
```
trigger AppointmentTrigger on Appointment__c (before insert, before update) {
    if(Trigger.isBefore){
        if(Trigger.isInsert || Trigger.isUpdate){
            AppointmentHandler.handleBeforeInsertUpdate(Trigger.new);
        }
    }
}
```

Apex Classes

- Developed a modular **AppointmentHandler** class containing all business logic.
- Logic maintained in methods called by the trigger, ensuring best practices and clean code.
- Business logic handles bulk appointments, updates status from "Checked In" to "Confirmed," and retrieves related patient data.



```
public class AppointmentHandler {
    public static void handleBeforeInsertUpdate(List<Appointment__c> newList) {
        for(Appointment__c app : newList) {
            if(app.Status__c == 'Checked In') {
                app.Status__c = 'Confirmed';
            }
        }
    }
}
```



Appointment
APP-0005

Appointment ID	APP-0005
Owner	Harsh Raj Mishra
Patient	Priya Gupta
Record Type	Follow-up
Appointment Time	9/25/2025, 12:00 PM
Status	Confirmed
Created By	Harsh Raj Mishra, 9/24/2025, 4:32 AM
Last Modified By	Harsh Raj Mishra, 9/24/2025, 11:38 AM

Filters: All time • All activities • All types

No activities to show.
Get started by sending an email, scheduling a task, and more.

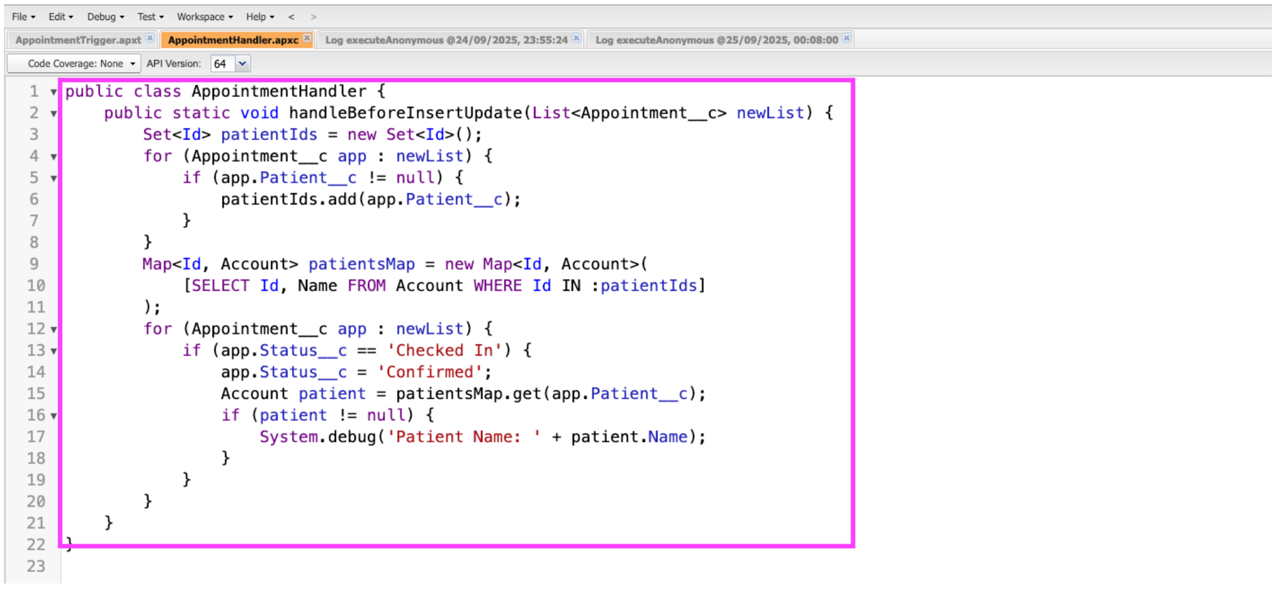
No past activity. Past meetings and tasks marked as done show up here.

SOQL & SOSL Queries

- Used SOQL in the handler class to query related **Account** records (patients) based on unique patient IDs collected from appointments.
- Queries executed once per bulk trigger run to optimize performance.
- Data stored in a Map for quick lookup and further processing.

Collections (List, Set, Map)

- Utilized collections for efficient and scalable processing:
    - **Set<Id>** to gather unique patient IDs from the appointments.
    - **Map<Id, Account>** to hold queried patient records.
    - Handled appointments in a loop using these collections.

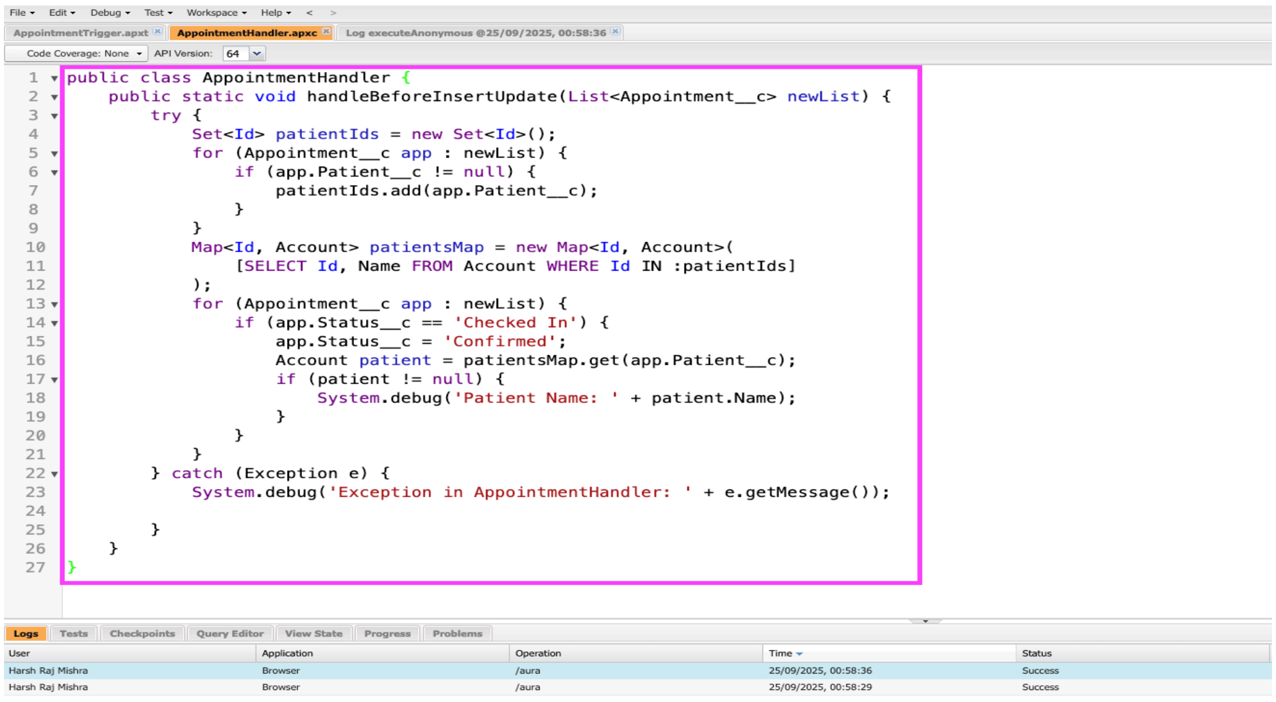

File
Edit
Debug
Test
Workspace
Help
AppointmentTrigger.apxt
AppointmentHandler.apxc
Log executeAnonymous @24/09/2025, 23:55:24
Log executeAnonymous @25/09/2025, 00:08:00

Code Coverage: None   API Version: 64

```apex
public class AppointmentHandler {
    public static void handleBeforeInsertUpdate(List<Appointment__c> newList) {
        Set<Id> patientIds = new Set<Id>();
        for (Appointment__c app : newList) {
            if (app.Patient__c != null) {
                patientIds.add(app.Patient__c);
            }
        }
        Map<Id, Account> patientsMap = new Map<Id, Account>(
            [SELECT Id, Name FROM Account WHERE Id IN :patientIds]
        );
        for (Appointment__c app : newList) {
            if (app.Status__c == 'Checked In') {
                app.Status__c = 'Confirmed';
                Account patient = patientsMap.get(app.Patient__c);
                if (patient != null) {
                    System.debug('Patient Name: ' + patient.Name);
                }
            }
        }
    }
}
```

Exception Handling

- Wrapped Apex logic in try-catch blocks to gracefully handle any runtime exceptions.
- Logged exceptions to aid debugging and ensure smooth execution.



Code Coverage: None   API Version: 64

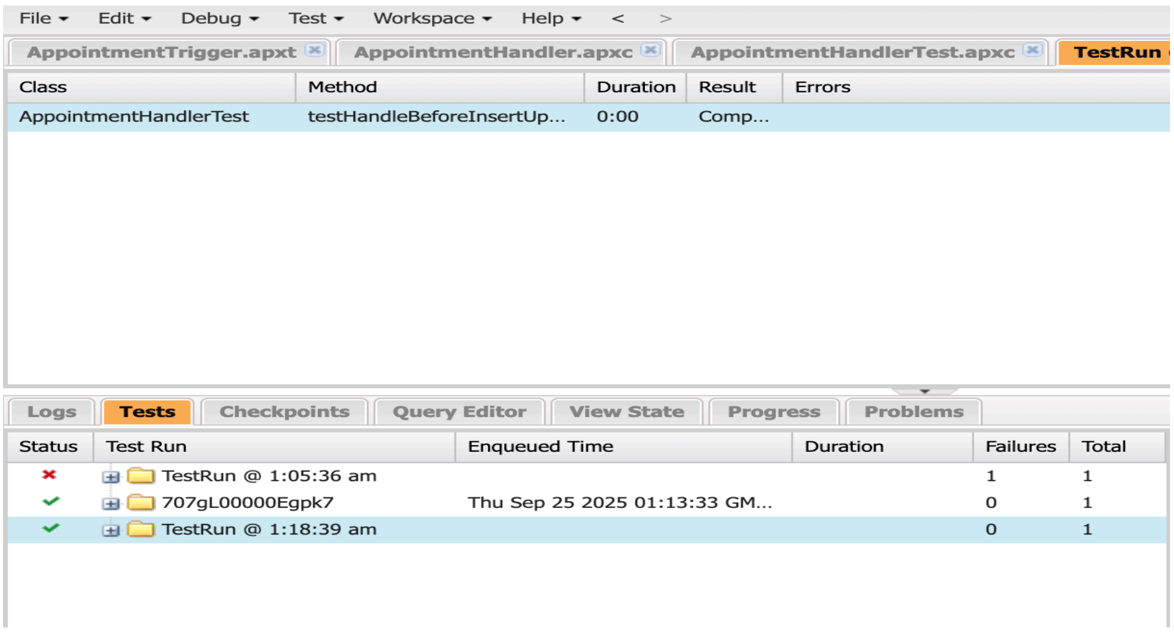
```apex
public class AppointmentHandler {
    public static void handleBeforeInsertUpdate(List<Appointment__c> newList) {
        try {
            Set<Id> patientIds = new Set<Id>();
            for (Appointment__c app : newList) {
                if (app.Patient__c != null) {
                    patientIds.add(app.Patient__c);
                }
            }
            Map<Id, Account> patientsMap = new Map<Id, Account>(
                [SELECT Id, Name FROM Account WHERE Id IN :patientIds]
            );
            for (Appointment__c app : newList) {
                if (app.Status__c == 'Checked In') {
                    app.Status__c = 'Confirmed';
                    Account patient = patientsMap.get(app.Patient__c);
                    if (patient != null) {
                        System.debug('Patient Name: ' + patient.Name);
                    }
                }
            }
        } catch (Exception e) {
            System.debug('Exception in AppointmentHandler: ' + e.getMessage());

        }
    }
}
```

User	Application	Operation	Time	Status
Harsh Raj Mishra	Browser	/aura	25/09/2025, 00:58:36	Success
Harsh Raj Mishra	Browser	/aura	25/09/2025, 00:58:29	Success

Test Classes

- Created an Apex test class **AppointmentHandlerTest** for automated testing.
- Tests insert appointments with all required fields including Record Type and Appointment Time.
- Validates that appointment status updates correctly to “Confirmed.”
- Run tests successfully with passing status.



Class	Method	Duration	Result	Errors
AppointmentHandlerTest	testHandleBeforeInsertUp...	0:00	Comp...	

Status	Test Run	Enqueued Time	Duration	Failures	Total
✗	TestRun @ 1:05:36 am			1	1
✓	707gL00000Egpk7	Thu Sep 25 2025 01:13:33 GM...		0	1
✓	TestRun @ 1:18:39 am			0	1

Asynchronous Processing (Future Development)

- Identified future needs for asynchronous operations such as sending notifications or integrating external systems.
- Implementation of Queueable Apex or Future methods is planned for long-running tasks beyond current scope.

Summary:

Phase 5 successfully automated appointment status updates and included performance and error handling improvements using Apex triggers and classes. The solution supports bulk data processing with clean modular design and comprehensive testing meeting Salesforce deployment standards.