MediCare Connect Phase 7 – Integration & External Access

1.Connected App / External Client App (OAuth)

I created a Connected App **"MediCare External Client App"** inside Salesforce to supply the OAuth Client ID & Secret and receive callback URLs. This allows Salesforce to securely obtain and refresh access tokens for external APIs.

Fig-1 ConnectedApp

2.External Auth Identity Provider

An **External Auth Identity Provider** "MediCare_Salesforce_IdP" was added so Salesforce can run an OAuth 2.0 Browser Flow against our Connected App. The provider generated a Callback URL, which i pasted back into the Connected App to resolve redirect URI mismatches.

Fig-2 OAuth_Config

3.External Credential + Named Credential

An **External Credential** "MediCare_EC" was created and linked to the Identity Provider.
A **Named Credential** "EHR_Service_NC" was then created pointing to https://httpbin.org (test endpoint). OAuth authentication was performed via the "Authenticate" button and tokens were stored securely. This allows Apex and Flows to call external APIs without hard-coding credentials.
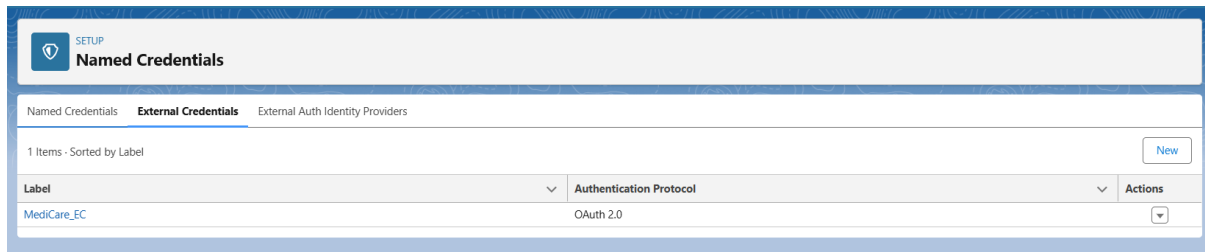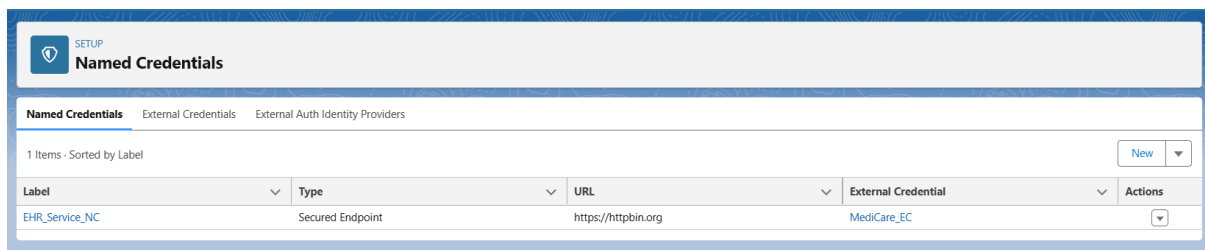

Fig-3 ExternalCredential

Fig-4 Named Credential

4.Apex REST Callout (Reminder Service)

An Apex class ReminderService was written to call the external endpoint via the Named Credential. We tested it with httpbin.org to confirm connectivity.
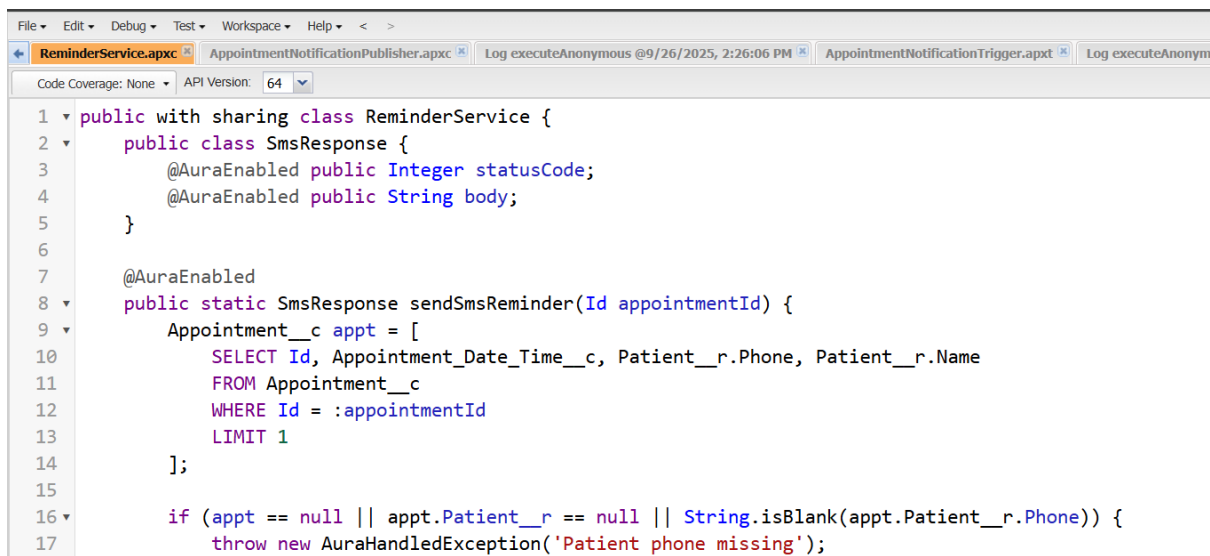
```
public with sharing class ReminderService {
    public class SmsResponse {
        @AuraEnabled public Integer statusCode;
        @AuraEnabled public String body;
    }

    @AuraEnabled
    public static SmsResponse sendSmsReminder(Id appointmentId) {
        Appointment__c appt = [
            SELECT Id, Appointment_Date_Time__c, Patient__r.Phone, Patient__r.Name
            FROM Appointment__c
            WHERE Id = :appointmentId
            LIMIT 1
        ];

        if (appt == null || appt.Patient__r == null || String.isBlank(appt.Patient__r.Phone)) {
            throw new AuraHandledException('Patient phone missing');
```

Fig-5 ApexCallout

Fig-6 CalloutLog

5.Platform Events – Publish & Subscribe

A Platform Event Appointment_Notification__e was created with fields AppointmentId__c (Text), Type__c (Picklist), and Message__c (Text Area).
Apex class AppointmentNotificationPublisher publishes events when needed.
A trigger on the Platform Event inserts a Task for each event received.
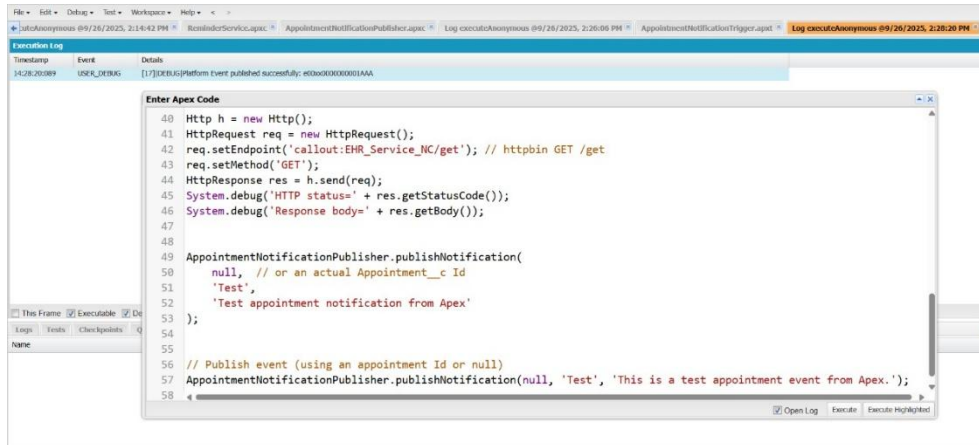

Fig-7 PlatformEvent

Fig-8 Published Successfully

6.Change Data Capture (CDC)

CDC was enabled for Appointment__c and Contact.
A custom object Integration_Log__c with field Message__c was created.
An Apex trigger AppointmentChangeLogger on Appointment__ChangeEvent writes log entries whenever an Appointment record changes.
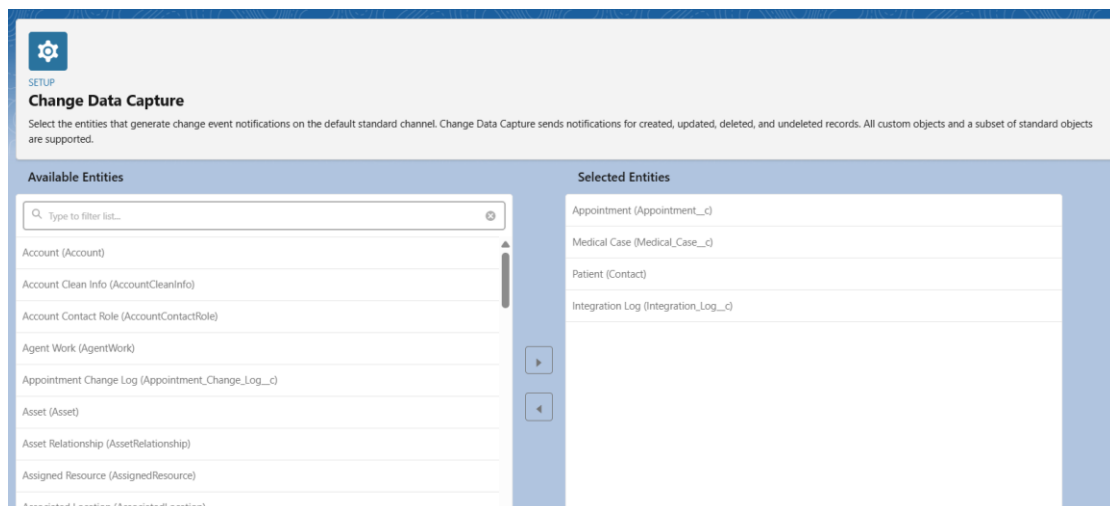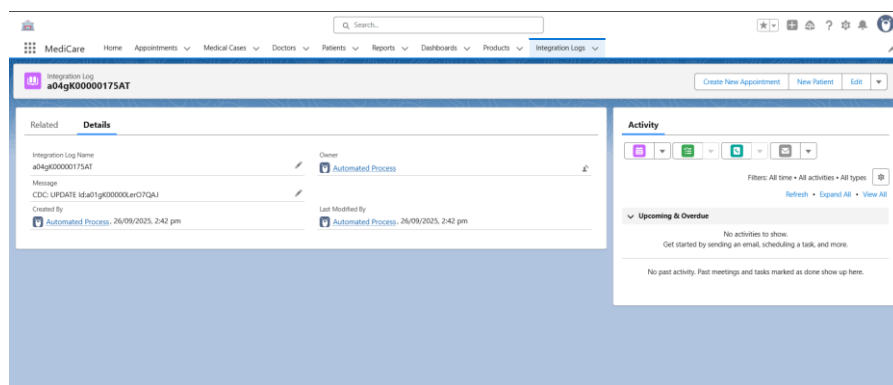

Fig-9 CDC_Config

Fig-10 log entries after an Appointment update

7.Salesforce Connect (optional demo)

An External Data Source EHR_OData_DS was configured against a public OData endpoint (https://services.odata.org/V4/OData/OData.svc/) and synced to create External Objects inside Salesforce, demonstrating real-time access without copying data.
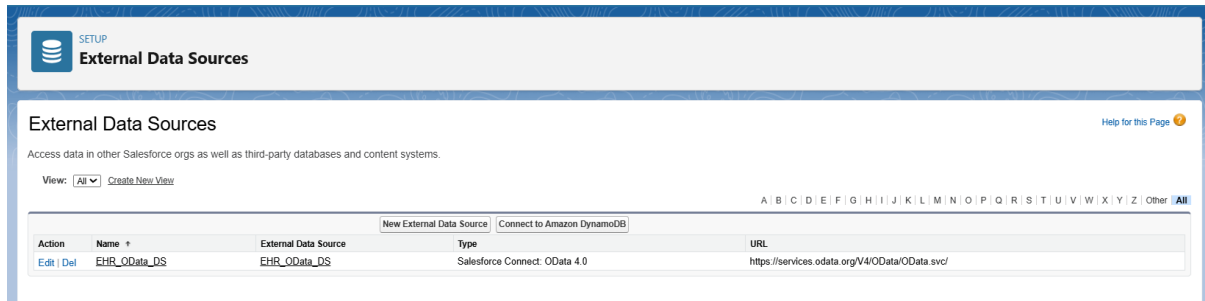

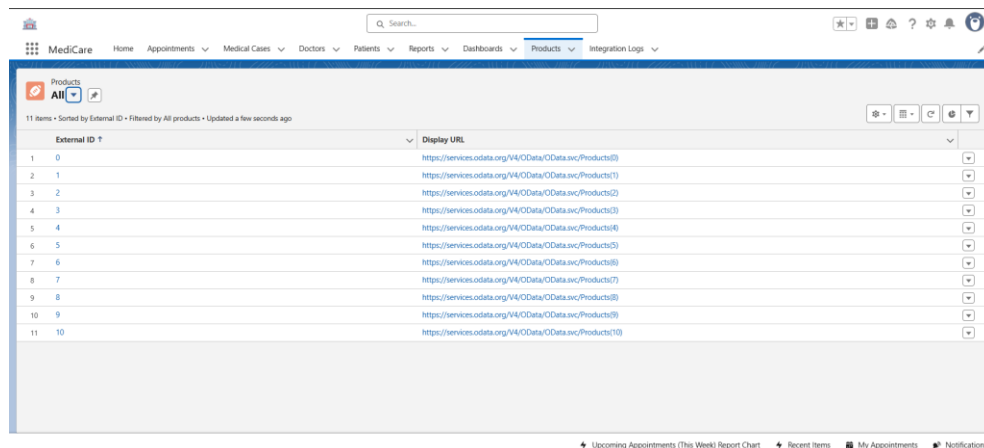**Fig-11 ExternalDataSource**

Fig-12 ExternalObject

8.API Limits & Monitoring

System Overview and the REST /limits endpoint in Workbench were used to view current API consumption and configure alerts.
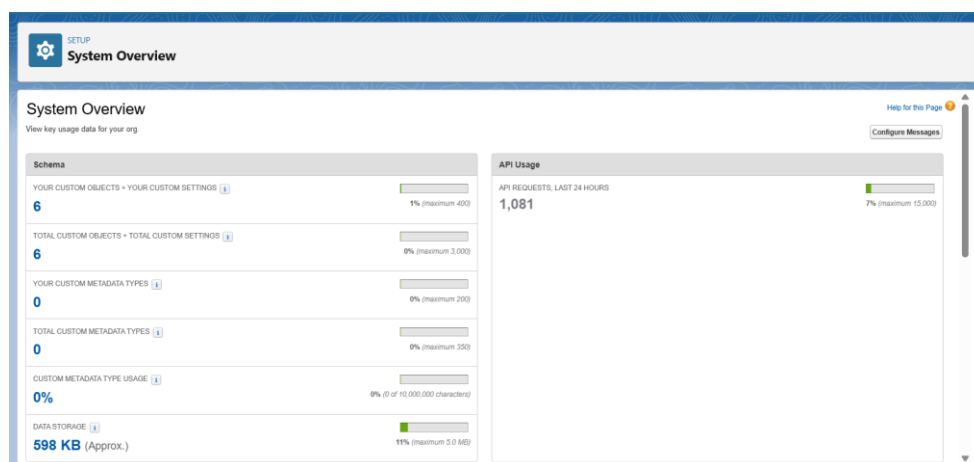

Fig-13 API_Limits

9.Acceptance Tests

Fig-14 Apex Test Execution Successfull