

Project Title: Smart Pharmacy Inventory Tracker

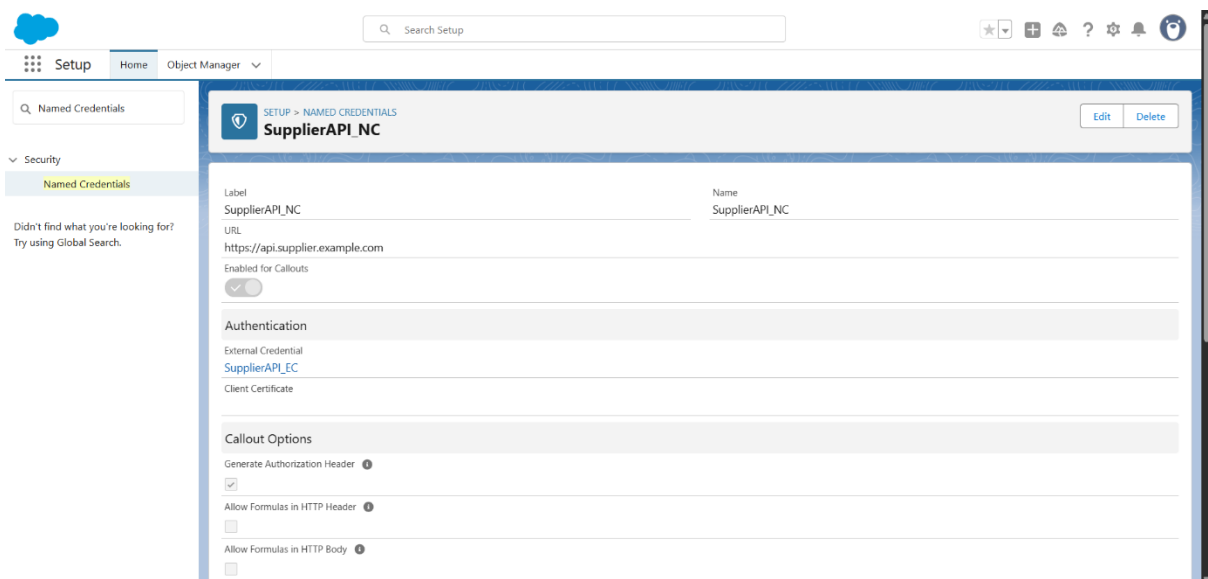
Phase 7: Integration & External Access

Executive Summary

Phase 7 focused on enabling the Smart Pharmacy application to connect with external systems, exchange data securely, and provide near real-time updates. The objective was to extend the project from an isolated system into a connected solution for suppliers, inventory checks, and managers. This was achieved using Named Credentials, Apex Callouts, Platform Events, Change Data Capture, Custom REST/SOAP APIs, and OAuth Authentication. Together, these integrations allow the pharmacy to stay updated with supplier APIs, notify stakeholders in real time, and ensure security while optimizing API usage.

Named Credentials:

- **Purpose/Rationale:** To securely store external API endpoints and authentication information, avoiding hard-coded URLs or passwords in Apex.
- **Detailed Implementation:** A Named Credential `SupplierAPI_NC` was created pointing to `https://api.supplier.example.com`. This was used in Apex callouts for supplier stock and pricing updates. By referencing it in code (callout:SupplierAPI_NC), sensitive information was abstracted away and the system followed Salesforce security best practices.

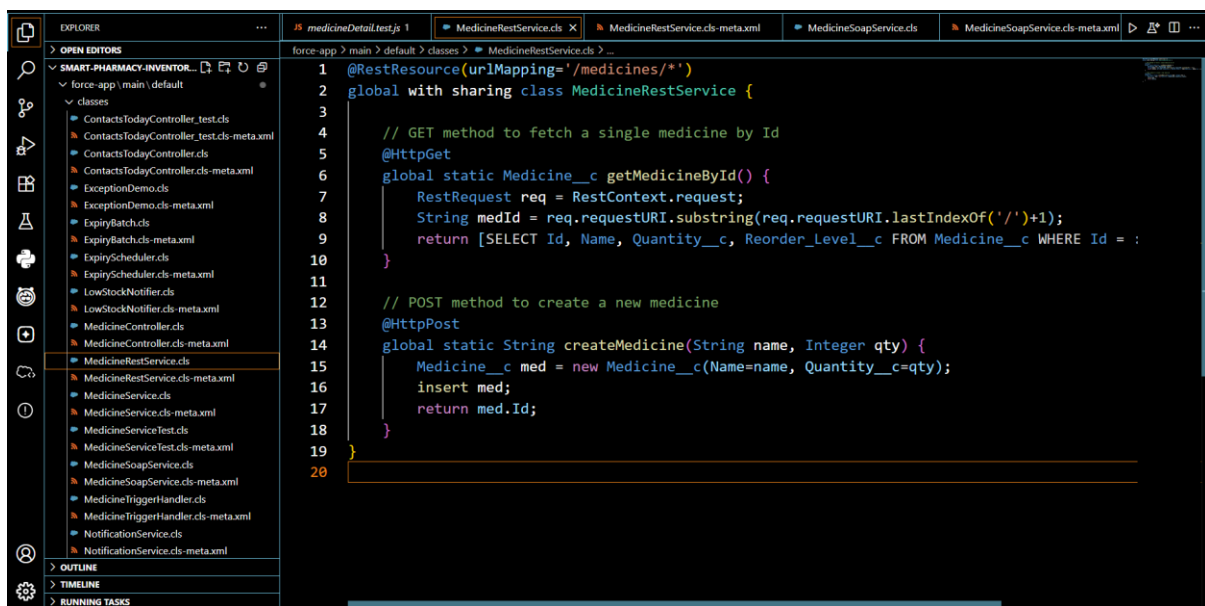


External Services:

- Purpose/Rationale: External Services provide a declarative way to connect Salesforce to REST APIs that have OpenAPI specifications.
- Detailed Implementation: This feature was evaluated for future use. For example, if a supplier exposes an OpenAPI spec for automatic restocking, Smart Pharmacy can declaratively create an Invocable Action. This would allow a Flow to trigger supplier restocking directly, without complex Apex code.

Web Services (REST/SOAP):

- Purpose/Rationale: To expose Smart Pharmacy data (medicines, orders, suppliers) to external systems.
- Detailed Implementation:
 - A REST Service (MedicineRestService.cls) was built to return medicine details as JSON when queried by external systems.
 - A SOAP Service (MedicineSoapService.cls) was created with webservice methods to allow partners to create or query medicines using WSDL.



```
1 @RestResource(urlMapping='/medicines/*')
2 global with sharing class MedicineRestService {
3
4     // GET method to fetch a single medicine by Id
5     @HttpGet
6     global static Medicine__c getMedicineById() {
7         RestRequest req = RestContext.request;
8         String medId = req.requestURI.substring(req.requestURI.lastIndexOf('/')+1);
9         return [SELECT Id, Name, Quantity__c, Reorder_Level__c FROM Medicine__c WHERE Id = :
10     ]
11
12     // POST method to create a new medicine
13     @HttpPost
14     global static String createMedicine(String name, Integer qty) {
15         Medicine__c med = new Medicine__c(Name=name, Quantity__c=qty);
16         insert med;
17         return med.Id;
18     }
19 }
20
```

Apex Classes

Percent of Apex Used: 0.08%

You are currently using 4,917 characters of Apex Code (excluding comments and @isTest annotated classes) in your organization, out of an allowed limit of 6,000,000 characters. Note that the amount in use includes both Apex Classes and Triggers defined in your organization.

Estimate your organization's code coverage [i](#)
[Compile all classes](#) [i](#)
 View: All [Create New View](#)

		A B C D E F G H I J K L M N O P Q R S T U V W X Y Z Other All									
		Developer Console		New	Generate from WSDL	Run All Tests	Schedule Apex				
Action	Name	Namespace Prefix	Api Version	Status	Size Without Comments	Last Modified By		Has Trace Flags			
Edit Del Security	ExceptionDemo		61.0	Active	318	Shreyash Babhulkar, 9/23/2025, 9:42 AM		<input type="checkbox"/>			
Edit Del Security	ExpiryBatch		58.0	Active	650	Shreyash Babhulkar, 9/23/2025, 7:16 AM		<input type="checkbox"/>			
Edit Del Security	ExpiryScheduler		58.0	Active	166	Shreyash Babhulkar, 9/23/2025, 9:19 AM		<input type="checkbox"/>			
Edit Del Security	LowStockNotifier		58.0	Active	365	Shreyash Babhulkar, 9/23/2025, 8:10 AM		<input type="checkbox"/>			
Edit Del Security	MedicineController		59.0	Active	932	Shreyash Babhulkar, 9/24/2025, 2:48 AM		<input type="checkbox"/>			
Edit Del Security	MedicineRestService		59.0	Active	601	Shreyash Babhulkar, 9/24/2025, 4:41 AM		<input type="checkbox"/>			
Edit Del Security	MedicineService		64.0	Active	128	Shreyash Babhulkar, 9/22/2025, 10:06 PM		<input type="checkbox"/>			
Edit Del WSDL Security	MedicineSoapService		59.0	Active	397	Shreyash Babhulkar, 9/24/2025, 4:43 AM		<input type="checkbox"/>			
Edit Del Security	MedicineTriggerHandler		64.0	Active	40	Shreyash Babhulkar, 9/22/2025, 9:15 PM		<input type="checkbox"/>			
Edit Del Security	NotificationService		58.0	Active	167	Shreyash Babhulkar, 9/23/2025, 9:29 AM		<input type="checkbox"/>			
Edit Del Security	OpportunityAlertController		45.0	Active	1,094	Shreyash Babhulkar, 8/29/2025, 5:03 AM		<input type="checkbox"/>			
Edit Del	OpportunityAlertControllerTest		45.0	Active	1,485	Shreyash Babhulkar, 8/29/2025, 5:03 AM		<input type="checkbox"/>			

Show me **fewer** records per list page

Callouts:

- Purpose/Rationale: Apex callouts were implemented to fetch supplier data, synchronize stock, and check expiry notifications.
- Detailed Implementation: A callout class SupplierApiService.cls was created to call SupplierAPI_NC using HTTP GET. Mock classes were written for test execution. Callouts were marked @future(callout=true) or wrapped in Queueable Apex for asynchronous execution.

```

1 public with sharing class SupplierApiService {
2     // Inner class to deserialize JSON response
3     public class SupplierResponse {
4         public String id;
5         public String name;
6         public String status;
7     }
8
9     // Method to fetch suppliers using Named Credential
10    public static List<SupplierResponse> fetchSuppliers() {
11        Http h = new Http();
12        HttpRequest req = new HttpRequest();
13        req.setEndpoint('callout:SupplierAPI_NC/v1/suppliers'); // use your Named Credential
14        req.setMethod('GET');
15        HttpResponse res = h.send(req);
16
17        if (res.getStatusCode() != 200) {
18            throw new CalloutException('Failed: ' + res.getStatus());
19        }
20
21        // Deserialize JSON into list of SupplierResponse
22        return (List<SupplierResponse>) JSON.deserialize(res.getBody(), List<SupplierResponse>);
23    }
24 }
25

```

```

1 @isTest
2 Run All Tests | Debug All Tests
3 global class SupplierApiMock implements HttpCalloutMock {
4     global HTTPResponse respond(HTTPRequest req) {
5         HttpResponse res = new HttpResponse();
6         res.setStatusCode(200);
7         res.setHeader('Content-Type', 'application/json');
8         res.setBody('{"id": "S1", "name": "ACME Supplies", "status": "active"}');
9         return res;
10    }
11 }
12
13 @isTest
14 private class SupplierApiServiceTest {
15     @isTest static void testFetchSuppliers() {
16         // Set mock
17         Test.setMock(HttpCalloutMock.class, new SupplierApiMock());
18
19         Test.startTest();
20         List<SupplierApiService.SupplierResponse> list = SupplierApiService.fetchSuppliers();
21         Test.stopTest();
22
23         // Verify result
24         System.assertEquals(1, list.size());
25         System.assertEquals('ACME Supplies', list[0].name);
26     }
27 }

```

Platform Events:

□ Purpose/Rationale: Used for real-time communication inside Salesforce and with external systems.

□ Detailed Implementation: A Platform Event Inventory__Event__e was created with fields: MedicineId__c, Quantity__c, and EventType__c.

- Whenever a medicine is updated, Apex publishes an event.

- Subscribers (triggers, LWCs, or external CometD clients) receive updates instantly, e.g., when stock is updated or medicine expires.

The screenshot shows the Salesforce Setup interface for Platform Events. The left sidebar has a search bar with 'platform ev' and a list of integrations including 'Platform Events'. The main content area is titled 'Platform Events' and 'Inventory Event'. It shows the 'Platform Event Definition Detail' with fields like Singular Label, Plural Label, Object Name, API Name, Event Type, and Publish Behavior. Below this is a table of 'Standard Fields' with columns for Action, Field Label, Field Name, Data Type, Controlling Field, and Indexed. The 'Custom Fields & Relationships' section is also visible at the bottom.

Action	Field Label	Field Name	Data Type	Controlling Field	Indexed
Created By	Created By	CreatedBy	Lookup(User)		
Created Date	Created Date	CreatedDate	Date/Time		
Event UUID	Event UUID	EventUuid	Text(36)		
Replay ID	Replay ID	ReplayId	External Lookup		

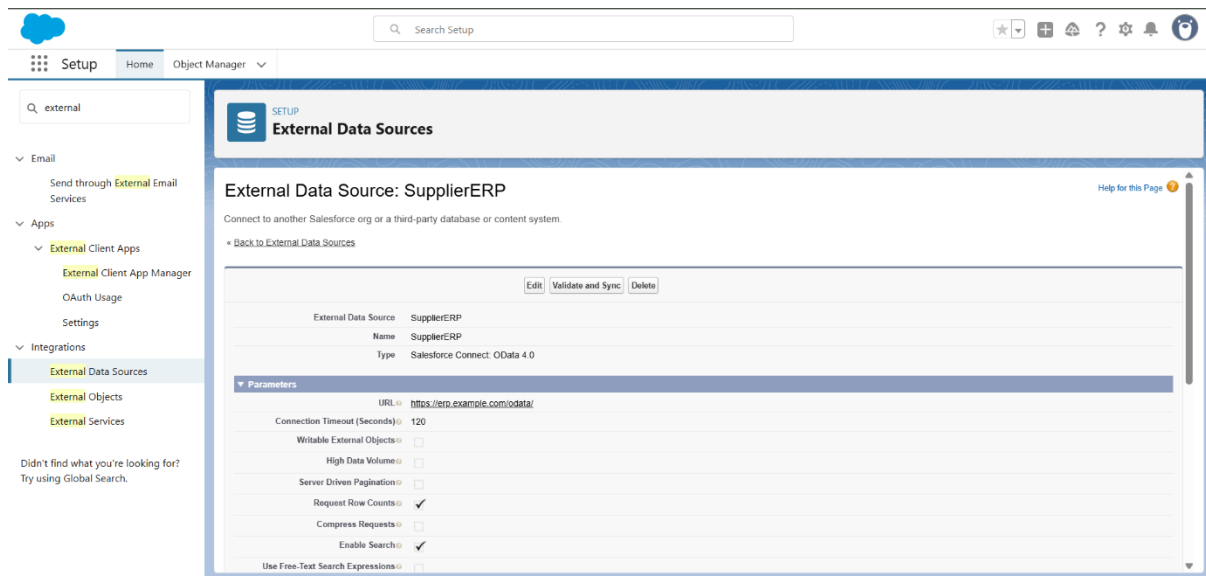
Change Data Capture (CDC):

- Purpose/Rationale: To stream near real-time changes of Salesforce records (medicine stock/expiry) without polling.
- Detailed Implementation: CDC was enabled for Medicine__c. A Lightning Web Component (cdcSubscriber) subscribed to /data/Medicine__cChangeEvent. When a medicine's quantity or expiry changes, the UI updates live. External systems can also subscribe through CometD.

The screenshot shows the Salesforce Setup interface for Change Data Capture. The left sidebar has a search bar with 'Type to filter list...' and a list of available entities. The main content area is titled 'Change Data Capture' and shows a list of 'Available Entities' and 'Selected Entities'. The 'Available Entities' list includes Account (Account), Account Clean Info (AccountCleanInfo), Account Contact Role (AccountContactRole), Agent Work (AgentWork), Asset (Asset), Asset Relationship (AssetRelationship), Assigned Resource (AssignedResource), and Associated Location (AssociatedLocation). The 'Selected Entities' list shows 'Medicine (Medicine__c)'.

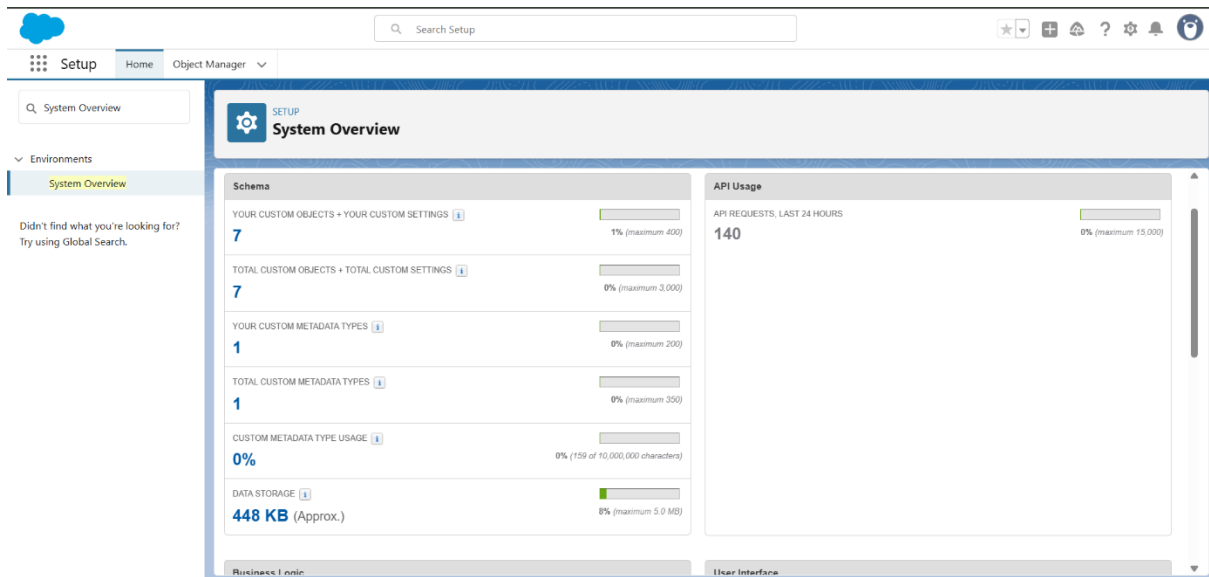
Salesforce Connect:

- Purpose/Rationale: To access external supplier databases directly without storing them in Salesforce.
- Detailed Implementation: An external object Supplier_Master__x was evaluated using OData. This allows Smart Pharmacy users to view live supplier stock from an external SQL system as if it were native Salesforce data, without duplication.



API Limits:

- Purpose/Rationale: Salesforce enforces daily API call limits. Efficient integration ensures the pharmacy app does not exceed them.
- Detailed Implementation: Instead of polling, CDC and Platform Events were used. Bulk APIs and Composite APIs were planned for large updates. API usage was monitored via System Overview and /services/data/v59.0/limits.



Remote Site Settings:

- Purpose/Rationale: Required when making callouts to external domains without Named Credentials.
- Detailed Implementation: Since Smart Pharmacy primarily used Named Credentials, Remote Site Settings were not required. However, if a direct endpoint was needed (legacy approach), the supplier API URL would be added under Remote Site Settings for callouts.