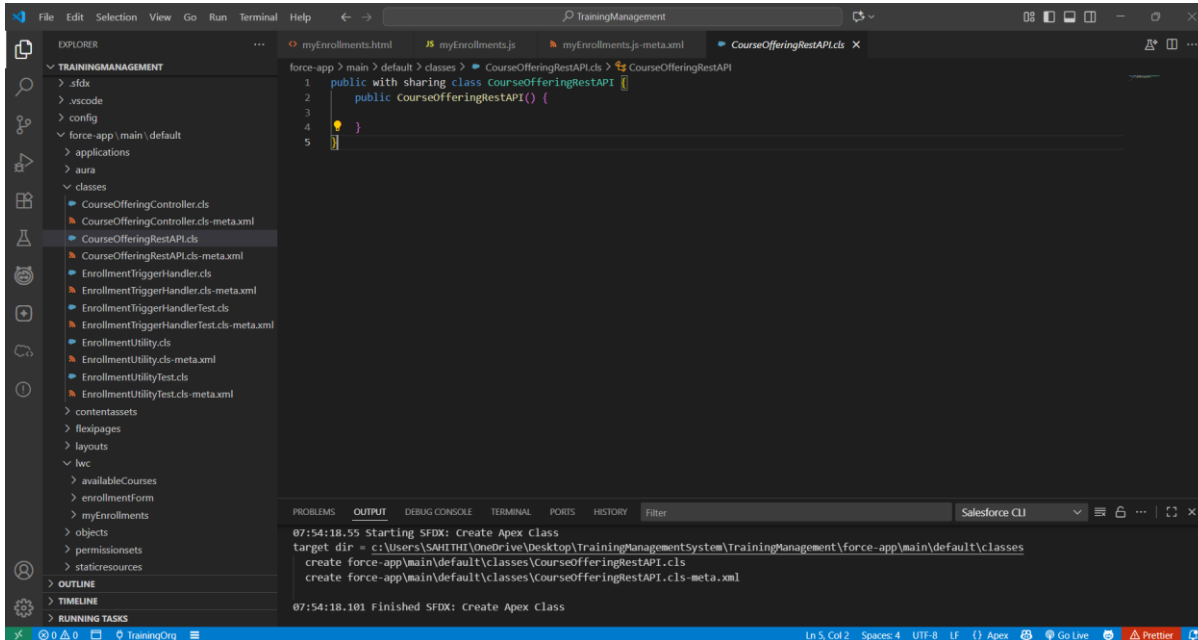


Phase 7: Integration & External Access

Section 1:

Create REST API Endpoint

Step 1: Create REST API Class



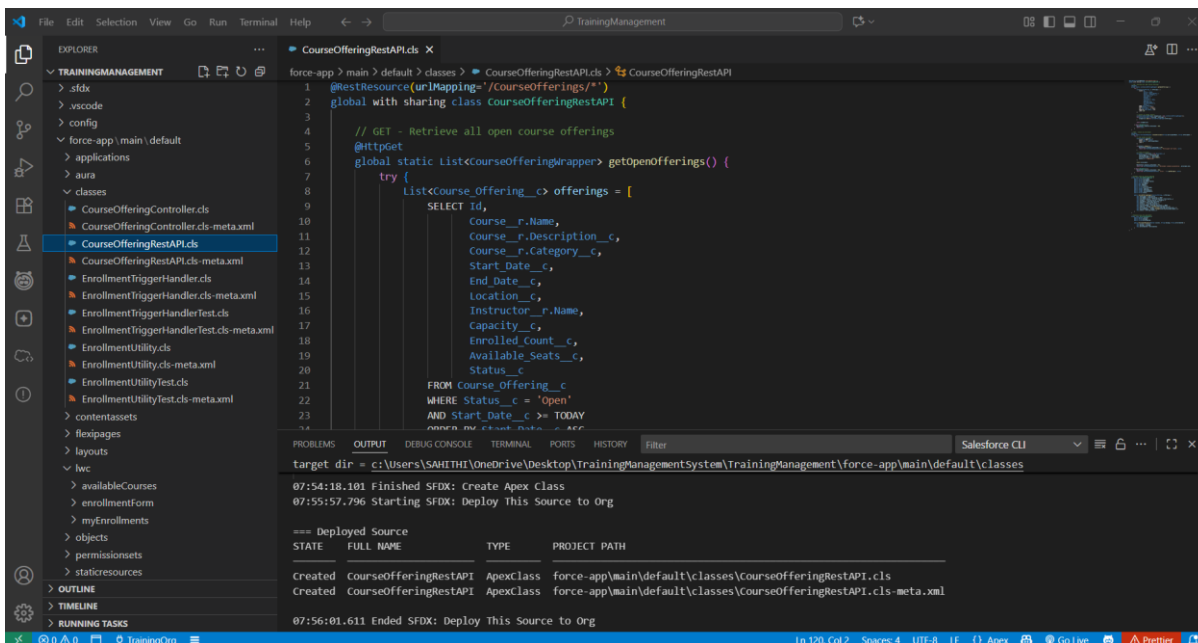
The screenshot shows the VS Code interface with the Salesforce CLI extension. The Explorer pane on the left shows the project structure for 'TRAININGMANAGEMENT'. The main editor displays the 'CourseOfferingRestAPI.cls' file with the following code:

```
1 public with sharing class CourseOfferingRestAPI {  
2     public CourseOfferingRestAPI() {  
3     }  
4 }  
5
```

The Output pane at the bottom shows the following messages:

```
07:54:18.55 Starting SFDX: Create Apex Class  
target dir = c:\Users\SAHITHI\OneDrive\Desktop\TrainingManagementSystem\TrainingManagement\force-app\main\default\classes  
create force-app\main\default\classes\CourseOfferingRestAPI.cls  
create force-app\main\default\classes\CourseOfferingRestAPI.cls-meta.xml  
07:54:18.101 Finished SFDX: Create Apex Class
```

Step 2: Write REST API Code



The screenshot shows the VS Code interface with the Salesforce CLI extension. The main editor displays the 'CourseOfferingRestAPI.cls' file with the following code:

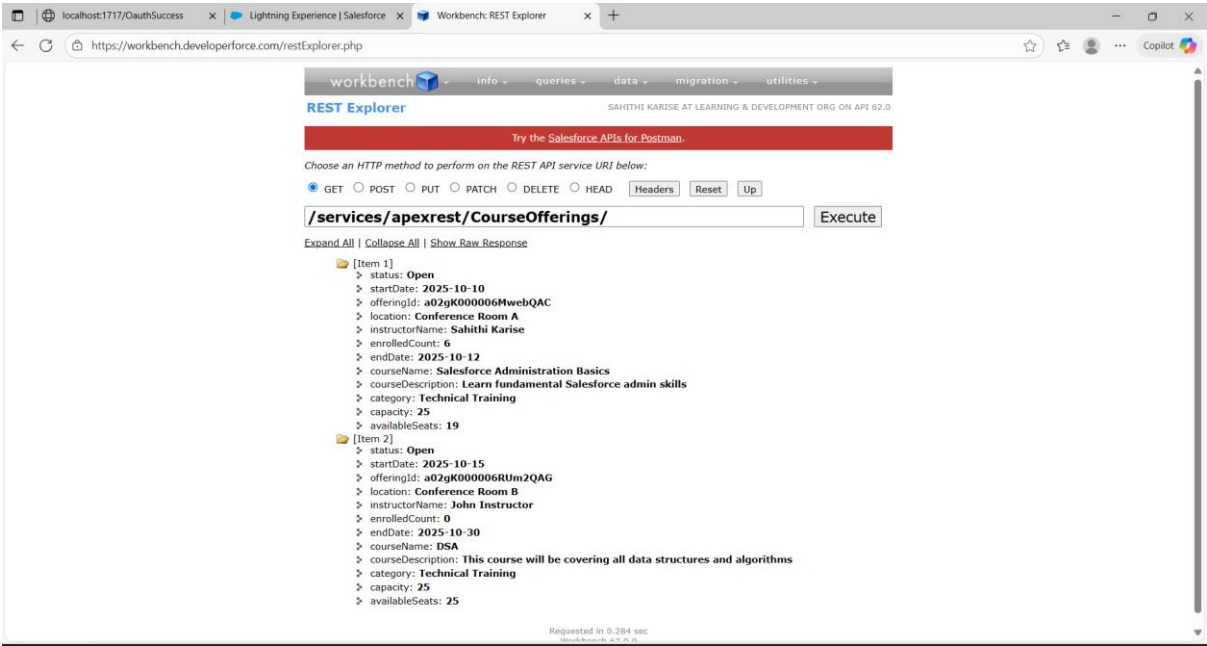
```
1 @RestResource(urlMapping='/CourseOfferings/*')  
2 global with sharing class CourseOfferingRestAPI {  
3  
4     // GET - Retrieve all open course offerings  
5     @HttpGet  
6     global static List<CourseOfferingWrapper> getOpenOfferings() {  
7         try {  
8             List<CourseOffering__c> offerings = [  
9                 SELECT Id,  
10                    Course__r.Name,  
11                    Course__r.Description__c,  
12                    Course__r.Category__c,  
13                    Start_Date__c,  
14                    End_Date__c,  
15                    Location__c,  
16                    Instructor__r.Name,  
17                    Capacity__c,  
18                    Enrolled_Count__c,  
19                    Available_Seats__c,  
20                    Status__c  
21                FROM CourseOffering__c  
22                WHERE Status__c = 'Open'  
23                AND Start_Date__c >= TODAY  
24            ]  
25            return offerings;  
26        } catch (Exception e) {  
27            return null;  
28        }  
29    }  
30 }
```

The Output pane at the bottom shows the following messages:

```
target dir = c:\Users\SAHITHI\OneDrive\Desktop\TrainingManagementSystem\TrainingManagement\force-app\main\default\classes  
07:54:18.101 Finished SFDX: Create Apex Class  
07:55:57.796 Starting SFDX: Deploy This Source to Org  
=== Deployed Source  
STATE    FULL NAME    TYPE    PROJECT PATH  
Created  CourseOfferingRestAPI  ApexClass  force-app\main\default\classes\CourseOfferingRestAPI.cls  
Created  CourseOfferingRestAPI  ApexClass  force-app\main\default\classes\CourseOfferingRestAPI.cls-meta.xml  
07:56:01.611 Ended SFDX: Deploy This Source to Org
```

REST API class deployed

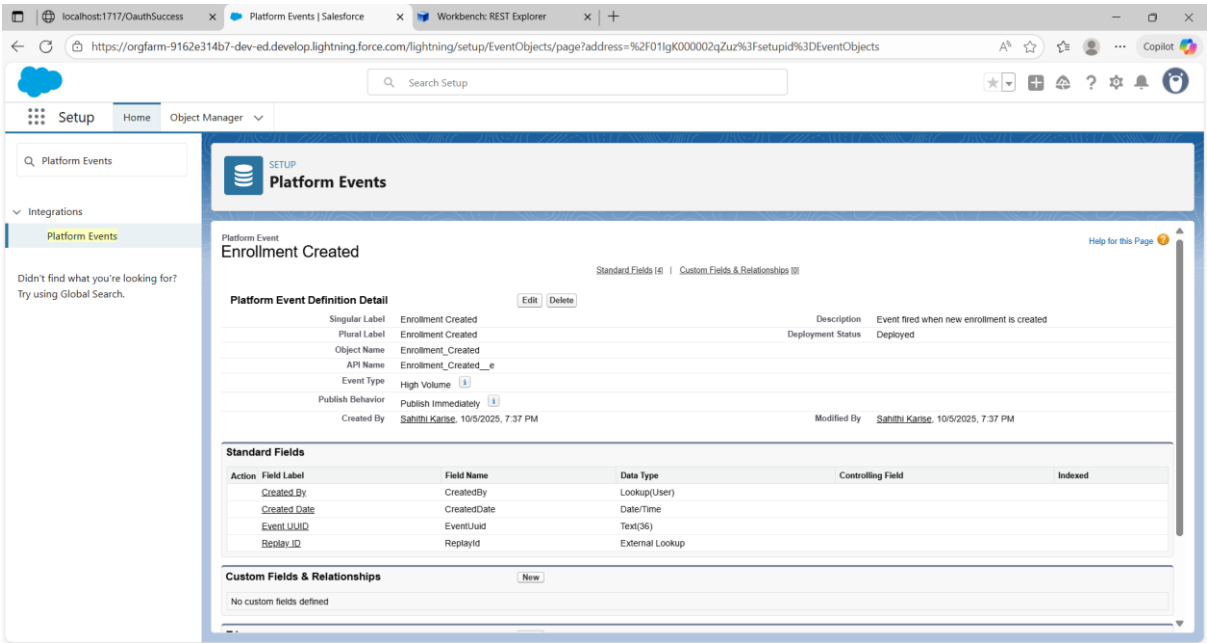
Step 3: Test REST API



REST API response in Workbench.

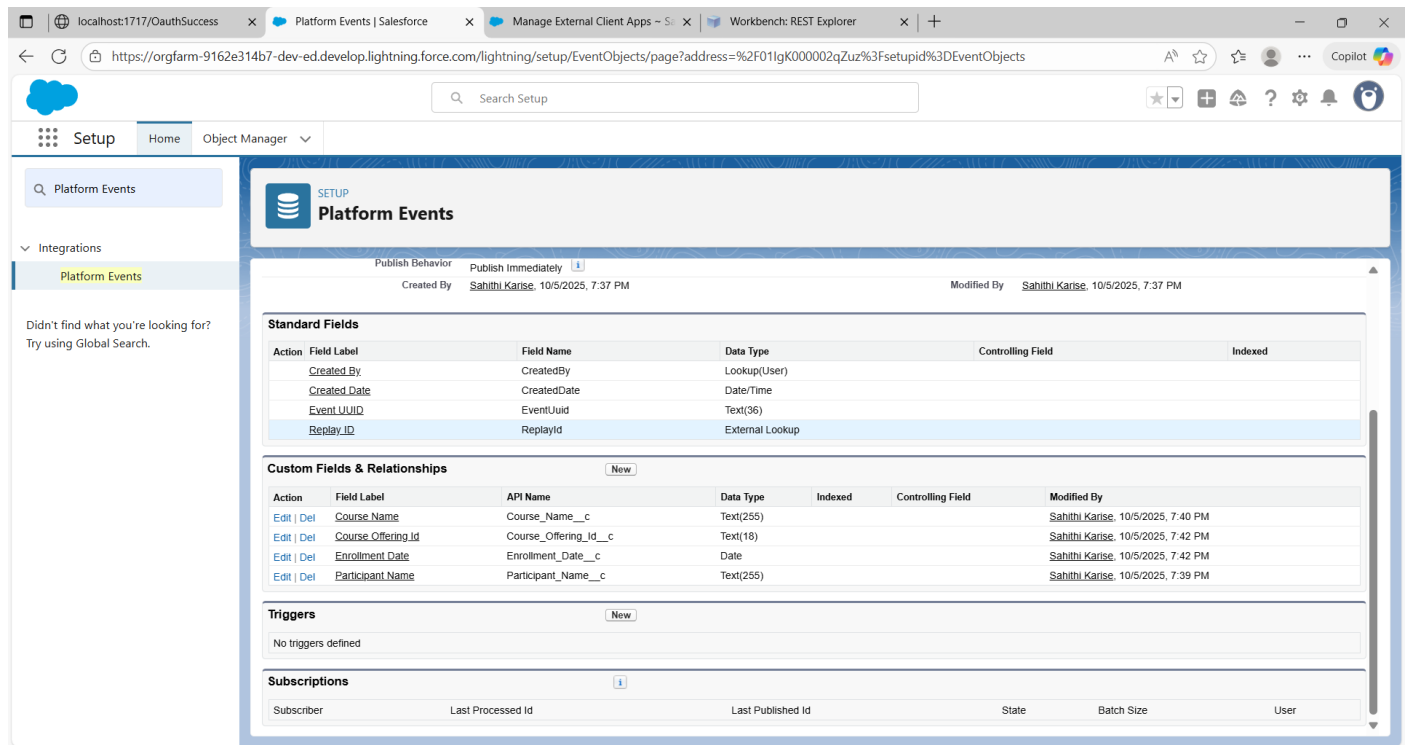
Section 2: Platform Events

Step 1: Create Platform Event



Step 2: Add Fields to Platform Event

Added Participant Name, Course Name, Course Offering Id and Enrollment Date fields.



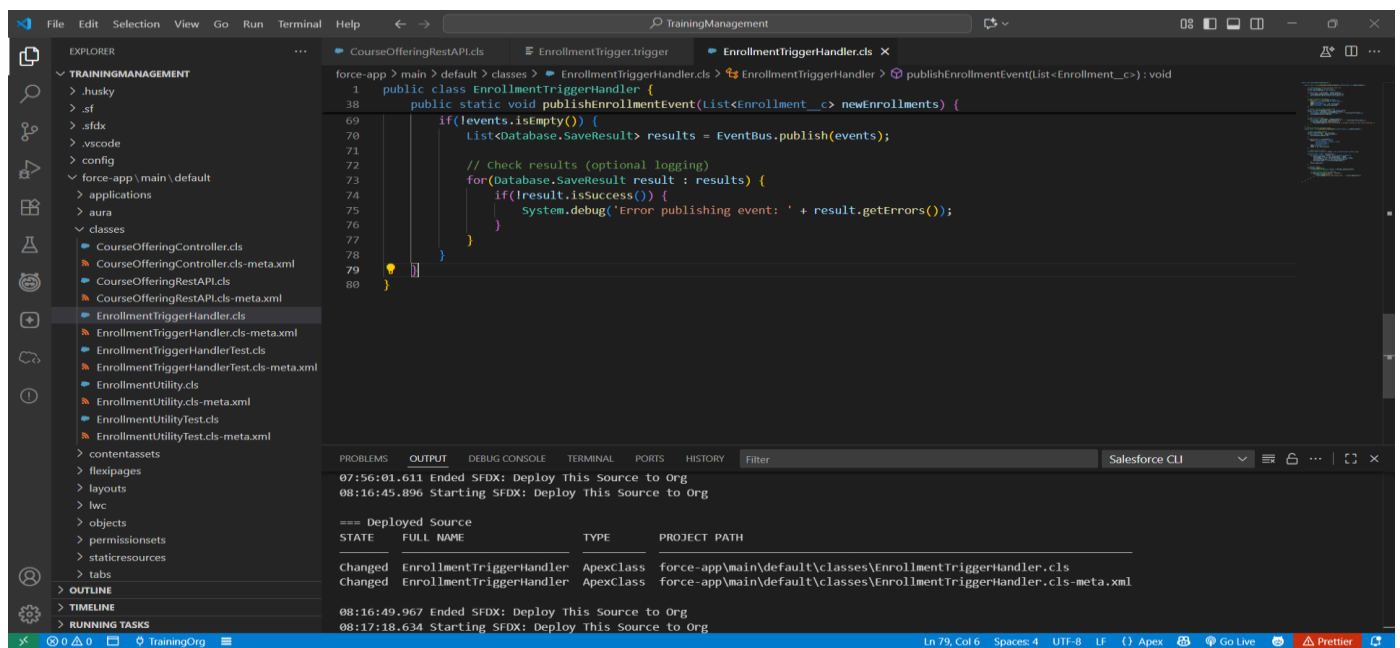
The screenshot shows the Salesforce Setup interface for Platform Events. The left sidebar contains navigation links for Setup, Home, and Object Manager. The main content area displays the 'Platform Events' configuration page. At the top, it shows 'Publish Behavior' set to 'Publish Immediately', 'Created By' as 'Sahithi Karise', and 'Modified By' as 'Sahithi Karise'. Below this, there are three sections: 'Standard Fields', 'Custom Fields & Relationships', and 'Triggers'. The 'Standard Fields' section lists fields like 'Created By', 'Created Date', 'Event UUID', and 'Replay ID'. The 'Custom Fields & Relationships' section lists newly added fields: 'Course Name', 'Course Offering Id', 'Enrollment Date', and 'Participant Name'. The 'Triggers' section shows 'No triggers defined'. The 'Subscriptions' section is partially visible at the bottom.

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

Action	Field Label	API Name	Data Type	Indexed	Controlling Field	Modified By
Edit Del	Course Name	Course_Name__c	Text(255)			Sahithi Karise, 10/5/2025, 7:40 PM
Edit Del	Course Offering Id	Course_Offering_Id__c	Text(18)			Sahithi Karise, 10/5/2025, 7:42 PM
Edit Del	Enrollment Date	Enrollment_Date__c	Date			Sahithi Karise, 10/5/2025, 7:42 PM
Edit Del	Participant Name	Participant_Name__c	Text(255)			Sahithi Karise, 10/5/2025, 7:39 PM

Step 3: Publish Platform Event from Trigger

Updated EnrollmentTrigger



The screenshot shows a Visual Studio Code editor with an Apex class named 'EnrollmentTriggerHandler'. The code defines a 'publishEnrollmentEvent' method that publishes a platform event. Below the editor, the 'OUTPUT' pane shows the results of SFDX commands, indicating that the source was successfully deployed.

```
force-app > main > default > classes > EnrollmentTriggerHandler > EnrollmentTriggerHandler > publishEnrollmentEvent(List<Enrollment__c>) : void
1 public class EnrollmentTriggerHandler {
2     public static void publishEnrollmentEvent(List<Enrollment__c> newEnrollments) {
3         if(!events.isEmpty()) {
4             List<Database.SaveResult> results = EventBus.publish(events);
5             // Check results (optional logging)
6             for(Database.SaveResult result : results) {
7                 if(!result.isSuccess()) {
8                     System.debug('Error publishing event: ' + result.getErrors());
9                 }
10            }
11        }
12    }
13 }
```

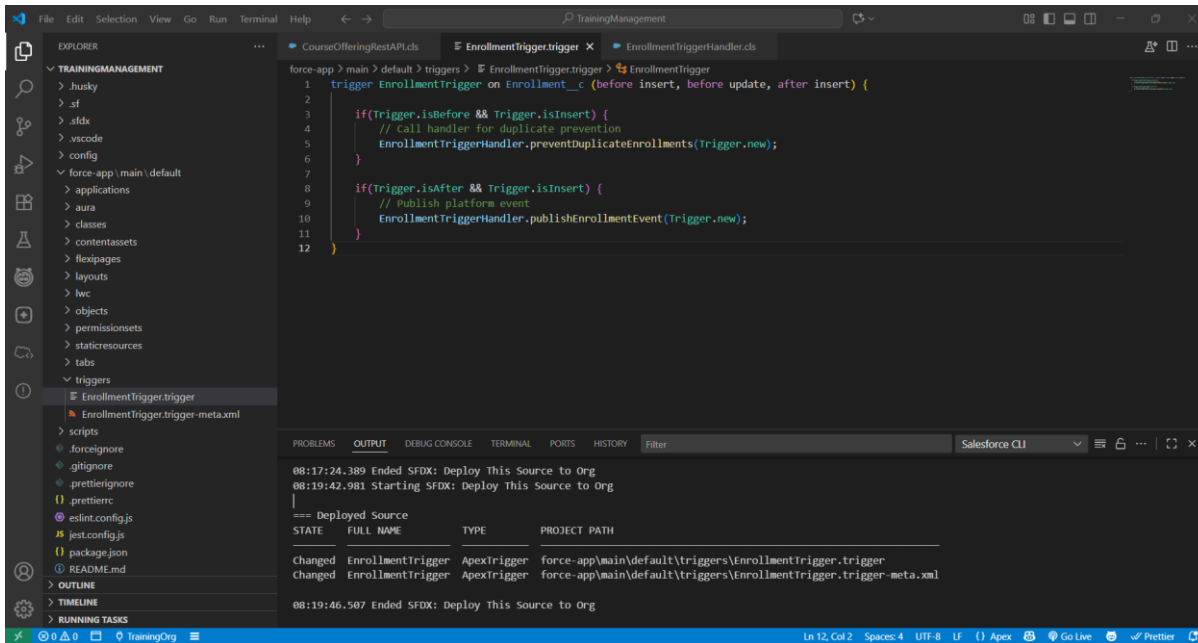
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS HISTORY Filter Salesforce CLI

07:56:01.611 Ended SFDX: Deploy This Source to Org
08:16:45.896 Starting SFDX: Deploy This Source to Org

=== Deployed Source		
STATE	FULL NAME	PROJECT PATH
Changed	EnrollmentTriggerHandler	force-app\main\default\classes\EnrollmentTriggerHandler.cls
Changed	EnrollmentTriggerHandler	force-app\main\default\classes\EnrollmentTriggerHandler.cls-meta.xml

08:16:49.967 Ended SFDX: Deploy This Source to Org
08:17:18.634 Starting SFDX: Deploy This Source to Org

Updated EnrollmentTriggerHandler



```
1 force-app > main > default > triggers > EnrollmentTrigger.trigger > EnrollmentTrigger
2 trigger EnrollmentTrigger on Enrollment__c (before insert, before update, after insert) {
3
4     if (Trigger.isBefore && Trigger.isInsert) {
5         // Call handler for duplicate prevention
6         EnrollmentTriggerHandler.preventDuplicateEnrollments(Trigger.new);
7     }
8
9     if (Trigger.isAfter && Trigger.isInsert) {
10        // Publish platform event
11        EnrollmentTriggerHandler.publishEnrollmentEvent(Trigger.new);
12    }
13 }
```

08:17:24.389 Ended SFDX: Deploy This Source to Org
08:19:42.981 Starting SFDX: Deploy This Source to Org

STATE	FULL NAME	TYPE	PROJECT PATH
Changed	EnrollmentTrigger	ApexTrigger	force-app\main\default\triggers\EnrollmentTrigger.trigger
Changed	EnrollmentTrigger	ApexTrigger	force-app\main\default\triggers\EnrollmentTrigger.trigger-meta.xml

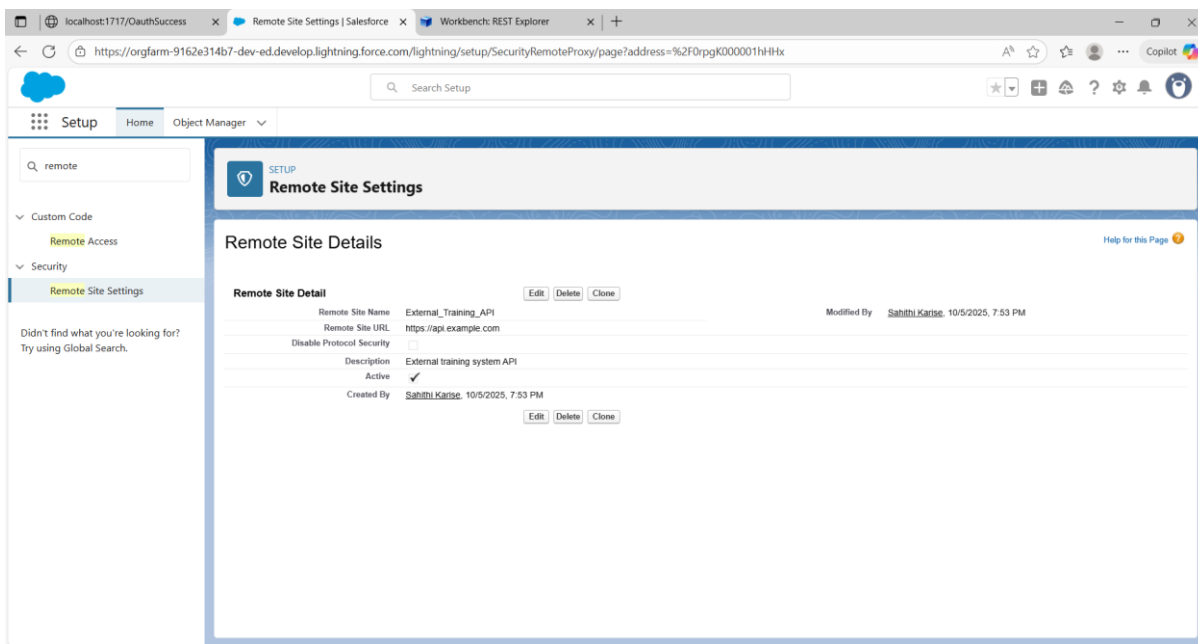
08:19:46.507 Ended SFDX: Deploy This Source to Org

Saved and Deployed both trigger and handler

Section 3:

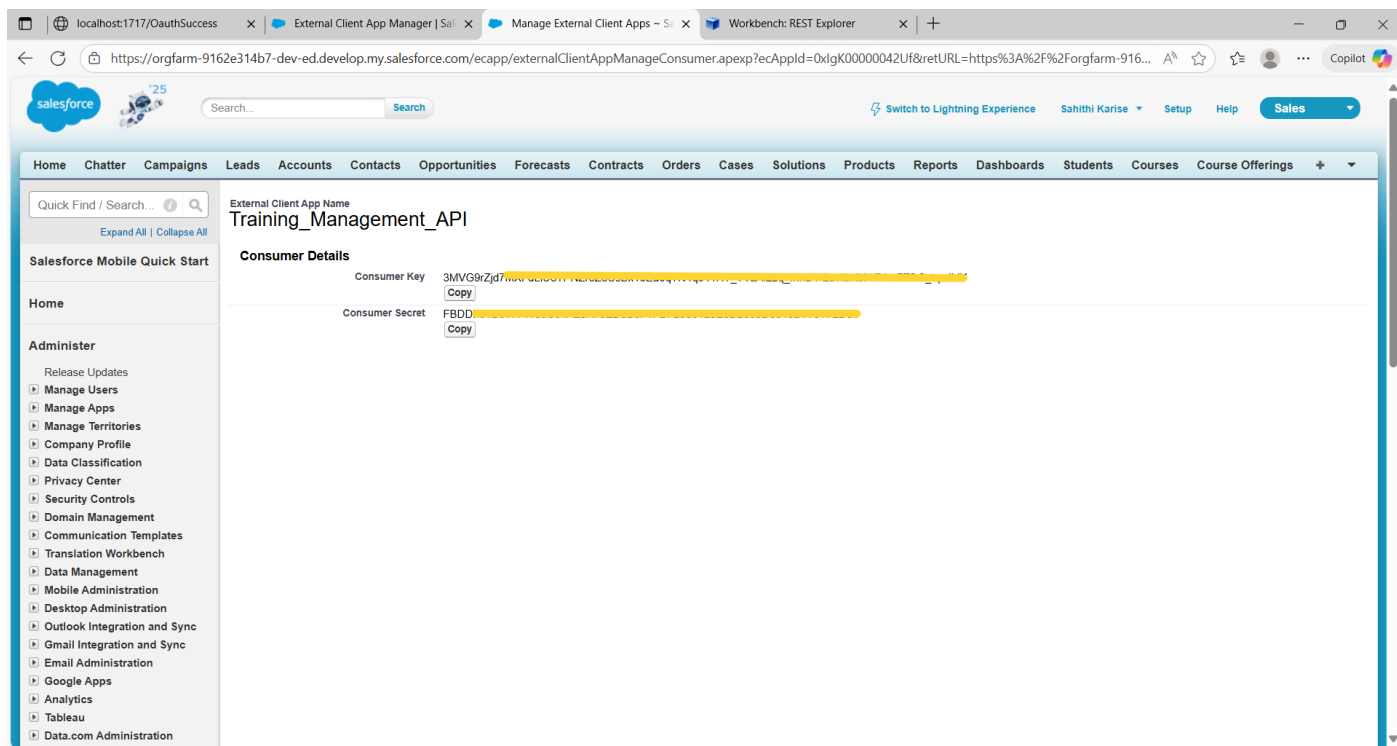
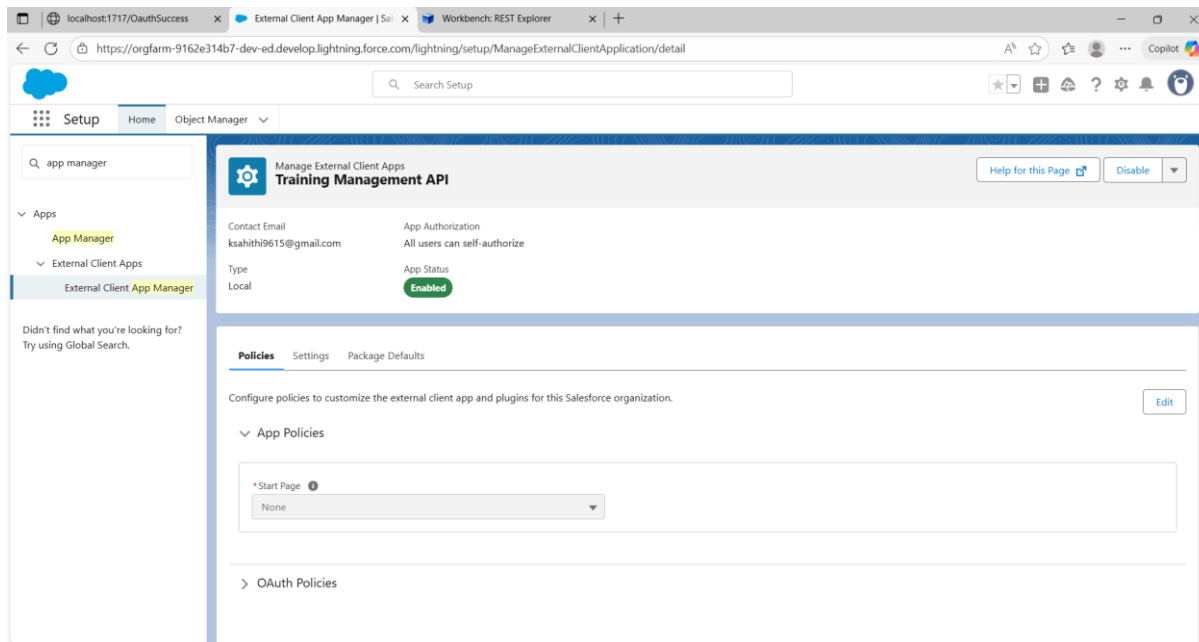
Remote Site Settings

Step 1: Add Remote Site



OAuth & Connected Apps

Step 1: Create Connected App

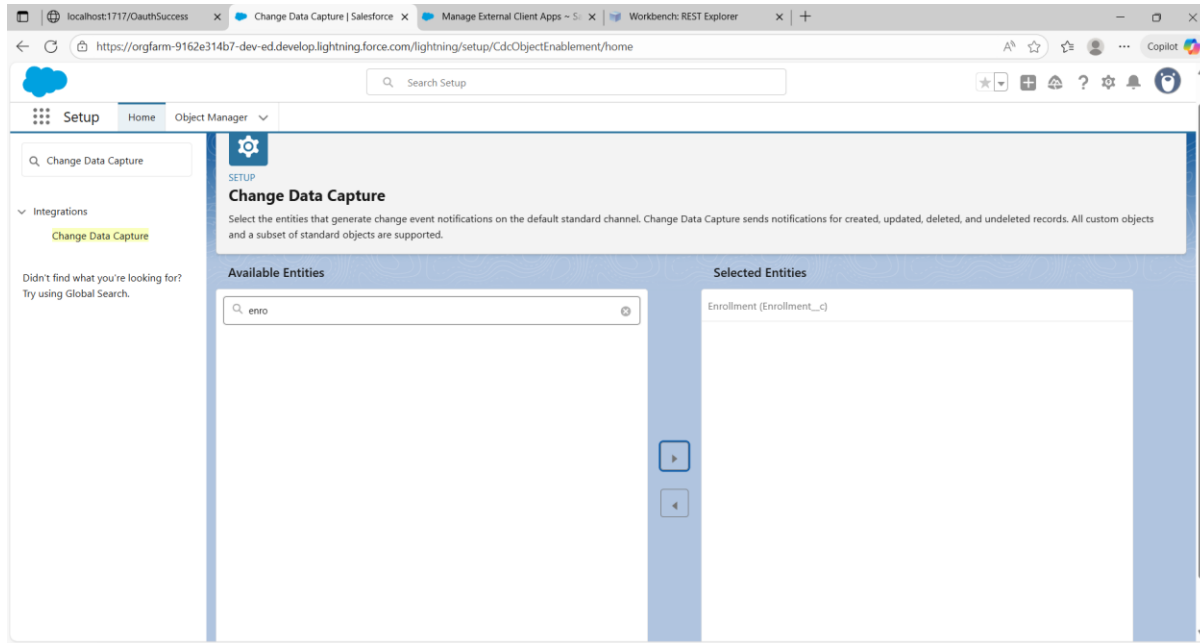


Consumer Key and Consumer Secret

Section 5:

Change Data Capture

Step 1: Enable Change Data Capture



Step 2: Document CDC Usage

Change Data Capture Use Case:

- When enrollment record is created/updated/deleted
- CDC publishes event to external systems
- External systems subscribe to these changes
- Real-time data synchronization
- No polling required

Benefits:

- Near real-time updates
- Reduced API calls
- Efficient data sync
- Event-driven architecture

Summary

What We Built:

REST API:

- GET endpoint for course offerings
- POST endpoint for enrollment creation
- Wrapper classes for JSON serialization
- Error handling and status codes

Platform Events:

- Enrollment_Created__e event
- 4 custom fields
- Publishing from trigger
- Event-driven architecture

Integration Components:

- Remote Site Settings
- Connected App for OAuth
- Change Data Capture enabled
- API documentation

Security:

- OAuth authentication
- API access control
- Secure external callouts