

PHASE 7: INTEGRATION & EXTERNAL ACCESS – Step-by-Step Implementation

1. Configure Remote Site Settings

- Go to Setup → enter Remote Site Settings in Quick Find.
- Click New Remote Site.
- Set Remote Site Name: `ExternalAPI`.
- Set Remote Site URL: Use a mock API for testing, e.g., `https://jsonplaceholder.typicode.com`.
- Save the entry to allow Salesforce to securely send HTTP requests to this external endpoint.

The screenshot shows the 'Remote Site Settings' page in Salesforce. At the top, there's a 'SETUP' button and the title 'Remote Site Settings'. Below this, the 'Remote Site Details' section is visible. It contains a table with the following information:

Remote Site Detail		Edit	Delete	Clone
Remote Site Name	ExternalAPI			
Remote Site URL	https://jsonplaceholder.typicode.com	Modified By Kusuma Kumar , 04/10/2025, 3:44 pm		
Disable Protocol Security	<input type="checkbox"/>			
Description	Active			
Created By	Kusuma Kumar , 04/10/2025, 3:44 pm	Edit	Delete	Clone

2. Create Named Credentials

- In Setup, search for Named Credentials.
- Click New Named Credential.
- Enter a label, e.g., `TestAPI`.
- Set URL: `https://jsonplaceholder.typicode.com` (matches remote site).
- Set Authentication Protocol to `Anonymous` (for mock APIs).

- Save the Named Credential.
- This enables Apex callouts to reference the endpoint securely without hardcoding URLs or credentials.

The screenshot shows the 'New Named Credential' dialog box in Salesforce. The dialog is titled 'New Named Credential' and contains the following fields and sections:

- * Label:** TestAPI
- * Name:** TestAPI
- * URL:** https://jsonplaceholder.typicode.com
- Enabled for Callouts:** A toggle switch that is currently turned on (checked).
- Authentication:**
 - * External Credential:** A dropdown menu showing 'TestAPI'.
 - Client Certificate:** A search field with the placeholder text 'Search Certificates...' and a magnifying glass icon.
- Callout Options:**
 - Generate Authorization Header:** A checkbox that is checked.
 - Allow Formulas in HTTP Header:** A checkbox that is checked.

At the bottom right of the dialog are two buttons: 'Cancel' and 'Save'.

3. Develop Apex REST Callout Class

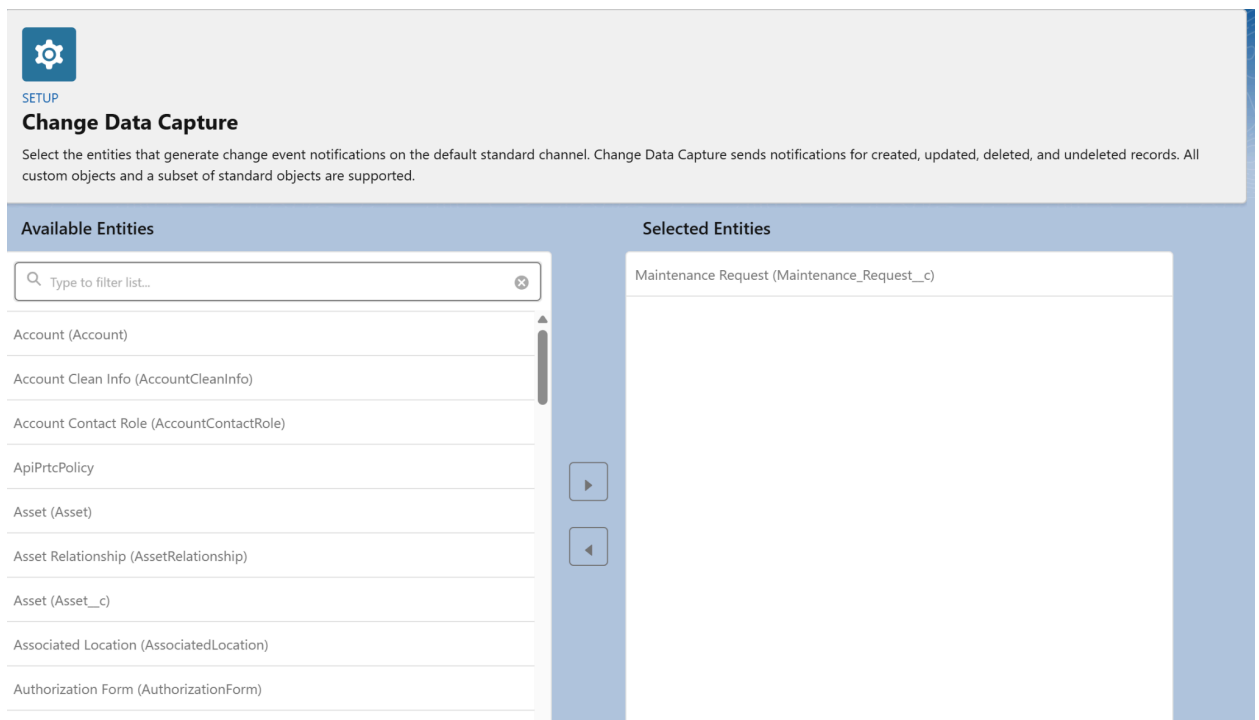
- In VS Code or Developer Console, create an Apex class called `CalloutDemo`.
- Implement a method annotated with `@future(callout=true)` that sends asynchronous HTTP POST requests.
- Example payload: JSON data representing maintenance request details (e.g., asset name, vendor, request description).
- Use Named Credential URL to build the endpoint for callout.
- Test callouts via debug logs to confirm receipt by the external system.

4. Define Platform Events

- Create a custom Platform Event, e.g., `MaintenanceRequestEvent__e`.
- Add relevant fields such as `Asset_Name__c`, `Vendor_Name__c`, and `Request_Details__c`.
- Publish these events from Apex or process builders when maintenance requests are created or updated.
- External systems can subscribe to these events for real-time integration and workflows.

5. Enable Change Data Capture (CDC)

- Enable CDC on the `Maintenance_Request__c` object.
- This automatically creates change events that external listeners can subscribe to.
- Ensure external integration systems are configured to consume and react to CDC events for near real-time data sync.



Achievements in Phase 7

- Successfully configured Remote Site Settings and Named Credentials for secure external callouts.
- Developed asynchronous Apex REST callouts to simulate sending maintenance data to third-party systems.
- Implemented custom Platform Events to publish maintenance business events instantly.
- Enabled Change Data Capture on key objects to keep external systems in sync with Salesforce records.
- Demonstrated robust and scalable Salesforce integration capabilities, enhancing interoperability for the Home Maintenance application.