Phase 7: Integration & External Access

1. Introduction

Explanation:
Integration is what makes Salesforce more than just a CRM — it becomes the center of an ecosystem.
This phase focuses on connecting Salesforce to external systems like APIs, web services, and other data sources.
These integrations enable real-time data sync, event-driven updates, and automated communication between Salesforce and third-party apps.

Objective:

- Enable Salesforce to communicate with external systems.
- Automate data sharing between Salesforce and external APIs.
- Demonstrate secure authentication and external callouts.

2. Named Credentials

Use Case:
Named Credentials store authentication and endpoint details for secure callouts to external systems — no hardcoding URLs or credentials inside Apex.

Scenario:
You want to connect Salesforce with a third-party API that provides supplier ratings (e.g., https://api.supplierdata.com).

Steps Implemented:

1. Go to **Setup → Named Credentials → New**.
2. Add:

- Label: Supplier_API
- URL: https://api.supplierdata.com
- Identity Type: Named Principal
- Authentication: Password Authentication / OAuth (depending on the API)

3. External Services

Use Case:
External Services allow you to import API specifications (Swagger/OpenAPI) into Salesforce so you can invoke them directly in Flow or Apex without writing complex code.

Scenario:
Integrate an external API that provides real-time medicine availability from distributors.

Steps Implemented:

1. Obtain API specification (Swagger/OpenAPI JSON).
2. In **Setup → External Services**, click **New External Service**.
3. Provide:
   - Name: Medicine Availability API
   - Named Credential: Supplier_API
   - Upload API schema file.
4. Automatically generates invocable actions for use in Flows.

4. Web Services (REST/SOAP)

Use Case 1: Exposing Salesforce Data as REST API

Scenario:
External systems should be able to fetch product data from Salesforce via REST.

Code Example:

```
@RestResource(urlMapping='/products/*')

global with sharing class ProductRestService {

    @HttpGet

    global static List<Product__c> getProducts() {

        return [SELECT Id, Name, Available_Stock__c, Unit_Price__c FROM
Product__c];

    }
```

5. Callouts

Use Case:
Callouts are HTTP requests from Salesforce to an external system — used to get or send data.

Scenario:
Whenever a new Supplier is added in Salesforce, make a callout to an external system to register that supplier.

Trigger Example:

```
trigger SupplierCalloutTrigger on Supplier__c (after insert) {

    for (Supplier__c s : Trigger.new) {

        SupplierCalloutService.sendSupplierData(s);

    }

}
```

Apex Class:

```
public class SupplierCalloutService {
    @future(callout=true)
    public static void sendSupplierData(Supplier__c supplier) {
        HttpRequest req = new HttpRequest();
        req.setEndpoint('callout:Supplier_API/suppliers');
        req.setMethod('POST');
        req.setHeader('Content-Type', 'application/json');
        req.setBody(JSON.serialize(supplier));
        Http http = new Http();
        HttpResponse res = http.send(req);
        System.debug('Response: ' + res.getBody());
    }
}
```

6. Platform Events

Use Case:
Platform Events enable asynchronous communication between systems — Salesforce can publish or subscribe to events.

Scenario:
When a new Purchase Order is approved, a “Purchase_Order_Approved__e” event is published for other systems (like an accounting app).

Steps Implemented:

1. Setup → Platform Events → New Platform Event.
2. Fields: Order_Id__c, Supplier__c, Total_Cost__c.

Apex Publisher Example:

```
Purchase_Order_Approved__e event = new Purchase_Order_Approved__e(
    Order_Id__c = po.Id,
    Supplier__c = po.Supplier__c,
    Total_Cost__c = po.Total_Order_Cost__c
);
EventBus.publish(event);
```

7. Change Data Capture (CDC)

Use Case:
CDC automatically sends real-time notifications when records are created or modified.

Scenario:
Track all changes to Product__c records so an external stock management system stays updated.

Steps Implemented:

1. Setup → Change Data Capture → Enable for Product__c.
2. Subscribed via CometD client or Platform Event listener.
3. Each update triggers a CDC event containing old and new values.

8. Salesforce Connect

Use Case:
Salesforce Connect allows you to access external data in real time — without storing it inside Salesforce.

Scenario:
Connect to an external ERP database (e.g., supplier ERP system) to view live stock data.

Steps Implemented:

1. Setup → External Data Sources → New.
2. Choose OData 4.0 as type.
3. URL: https://erp.suppliers.com/odata/service.svc.
4. Validate and sync external tables as External Objects.

9. API Limits

Use Case:
Every Salesforce org has API usage limits to ensure fair performance.

Steps Implemented:

1. Go to **Setup → System Overview** or **Company Information**.

2. View "API Usage" to monitor daily limits.

10. OAuth & Authentication

Use Case:
OAuth is used to authenticate Salesforce with third-party systems securely.

Scenario:
Allow an external app to access Salesforce data through a connected app.

Steps Implemented:

1. Setup → App Manager → New Connected App.
2. Enable OAuth settings and add callback URL.
3. Select OAuth scopes (e.g., Full Access, API).
4. Copy Consumer Key and Consumer Secret for external use.

11. Remote Site Settings

Use Case:
Before any HTTP callout can be made, Salesforce must whitelist the endpoint using Remote Site Settings.

Scenario:
Allow Salesforce to send callouts to https://api.supplierdata.com.

Steps Implemented:

1. Setup → Security → Remote Site Settings → New.
2. Enter:
    - Remote Site Name: SupplierAPI
    - URL: https://api.supplierdata.com
3. Save.