Project Title: "Telecom Service Hub – Customer & SIM Request Management System"

Industry: Telecommunications

Project Type: B2C Salesforce CRM Implementation

Target Users: Customer Support Agents, Service Managers, and Telecom Customers

Problem Statement:

A telecom company is struggling to efficiently manage SIM activations, customer service requests, and cancellations. Customers face delays in activation, poor communication on ticket progress, and agents are overloaded due to manual assignment of service requests. Managers lack real-time visibility into churn rates, customer complaints, and service performance.

The company wants a Salesforce CRM to:

•Automate service request creation and assignment

•Track SIM activations, complaints, and cancellations

•Provide real-time SMS/email notifications to customers

•Offer self-service portals for customers to raise requests

•Deliver dashboards for churn analysis and agent productivity

Use cases:

Customer Management

•Maintain centralized customer profiles with contact details and SIM status.

•Link customers to their subscribed plans.

Service Request Management

•Customers raise activation, complaint, or cancellation requests.

•Requests automatically assigned to available agents.

•Agents update request progress (Open → In Progress → Closed).

SIM Activation & Cancellation

•On activation → CRM auto-sends SMS confirmation.

•On cancellation → SIM status is updated to Inactive automatically.

Plan Management

•Store plan details: Name, Data Limit, Price, Validity.

•Customers linked with their current active plan.

Reporting & Analytics

•Reports: Service Requests by Priority, Requests by Type.

•Dashboard: Active vs Cancelled Customers (Churn Analysis).

•Agent Performance: Requests resolved per agent.

Phase 2: Org Setup & Configuration

Goal of Phase 2

Prepare the Salesforce environment with company setup, users, roles, profiles, and security model so that development can begin smoothly.

---

1. Salesforce Edition

- **Developer Edition Org** (free) is used.
- This org acts as both development and testing environment.
- In real companies, Sandboxes are used, but for this project, Dev Org serves the purpose.

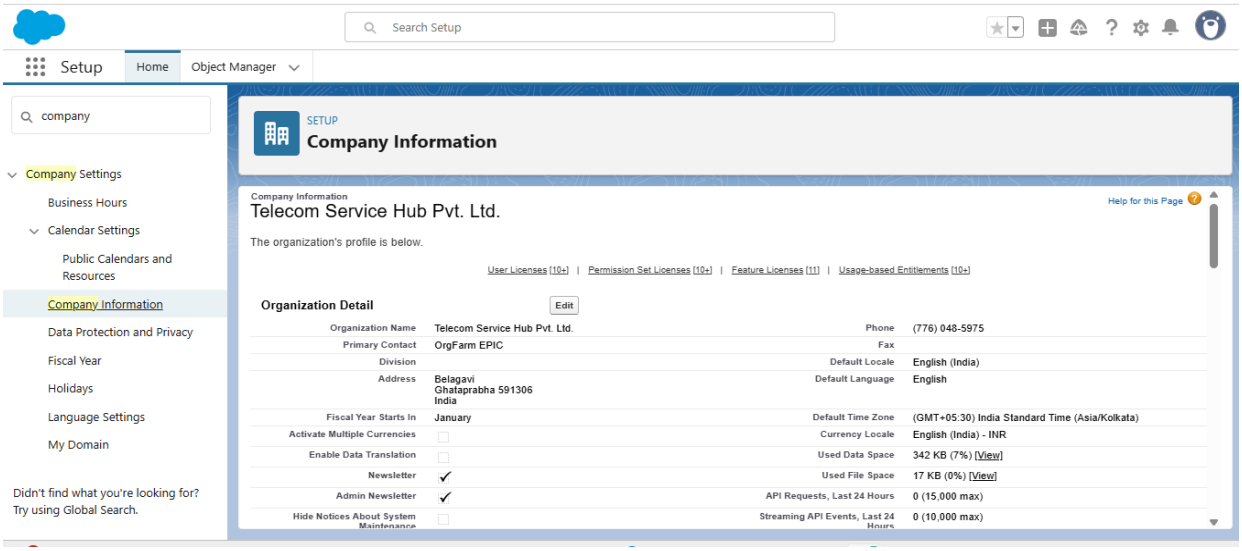
---

2. Company Profile Setup

- **Company Name**: Telecom Service Hub Pvt. Ltd.
- **Time Zone**: (GMT+05:30) Asia/Kolkata
- **Currency**: INR (₹)

- **Locale**: English (India)

“Company Profile setup with org information, timezone, and currency.”



---

3. Business Hours & Holidays

- **Standard Business Hours**: Monday–Friday, 9 AM – 6 PM.
- **Holiday Added**: Independence Day (Aug 15).
- Purpose: Ensures service requests and escalations only occur during working hours.

---

4. Fiscal Year Settings

- Chosen: **Standard Fiscal Year (Jan–Dec)**.
- Note: Can be changed to Apr–Mar if required for Indian telecom companies.

---

5. User Setup & Licenses

Created two sample users to simulate real-world roles:

1. **Manager User**
    - Username: manager1@telecomhub.com
    - Role: Service Manager
    - License: Salesforce
2. **Agent User**
    - Username: agent1@telecomhub.com

- Role: Customer Support Agent
- License: Salesforce

“Manager and Agent users created with Salesforce licenses.”

---

6. Profiles

Profiles were created to define object-level permissions:

- **Telecom Agent Profile**
  - Access: Create/Edit Service Requests, Read-only on Customers & Plans.
- **Telecom Manager Profile**
  - Full Access to all objects (Customers, Plans, Service Requests).

---

7. Roles

Role hierarchy created to manage record-level visibility:

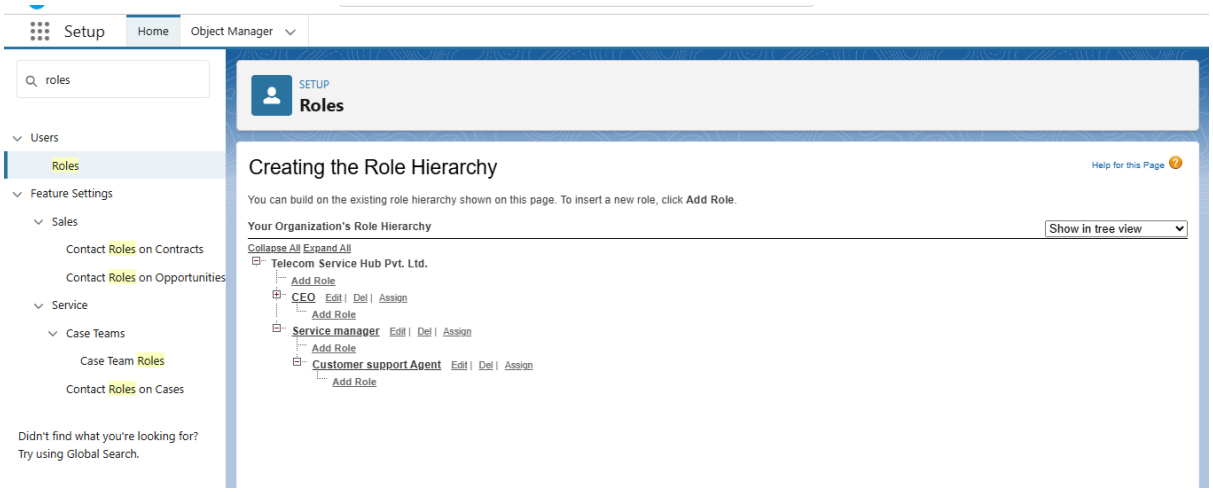
- **Service Manager** (Top role)
  - **Customer Support Agent** (Child role)

Assignments:

- Manager User → Service Manager
- Agent User → Customer Support Agent

Effect: Manager can see all records; Agent can only see their own.

“Role hierarchy created for Service Manager and Agents.”



---

8. Org-Wide Defaults (OWD)

Defined baseline data visibility:

- **Customer__c** → Public Read Only
- **Plan__c** → Public Read Only
- **ServiceRequest__c** → Private

This ensures sensitive service requests are only visible to the owner and their manager.

Phase 3: Data Modeling & Relationships

Goal of Phase 3

To design and configure the **data model** in Salesforce for managing **customers, telecom plans, and service requests**. This includes creating objects, fields, relationships, and layouts that support the real-time telecom use cases.

---

1. Standard & Custom Objects

Standard Objects

- **User** → For Agents and Managers who handle requests.

Custom Objects

1. **Customer__c** → Stores customer details and SIM information.

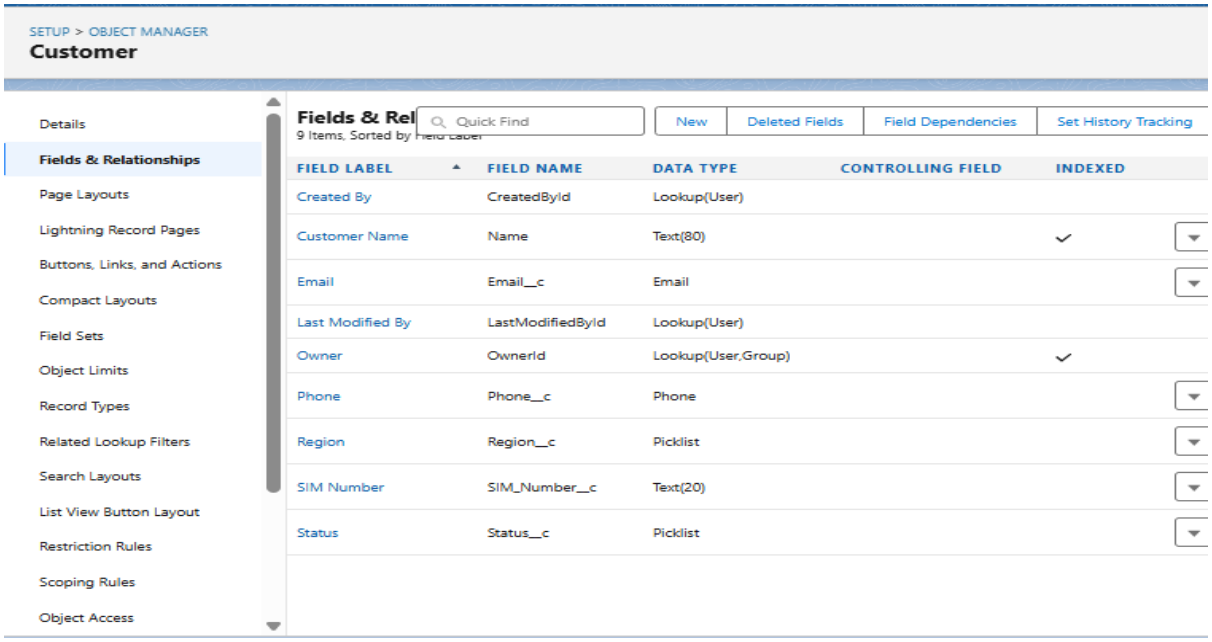
2. **Plan__c** → Stores telecom plan details (price, data, validity).
3. **ServiceRequest__c** → Tracks service requests such as activation, complaints, or cancellations.

---

2. Fields

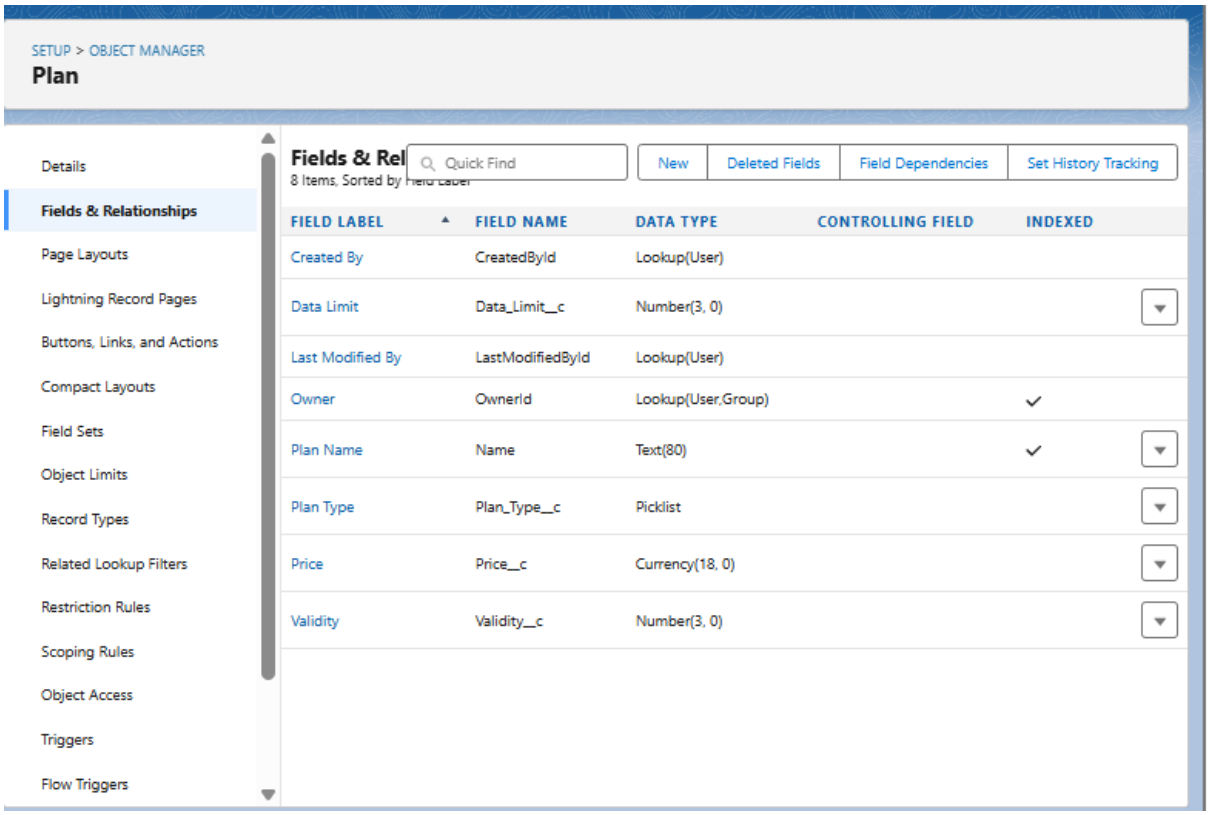
Customer__c

- Phone (Phone)
- Email (Email)
- SIM Number (Text, 20)
- Status (Picklist: Active, Inactive, Suspended)
- Region (Picklist: East, West, North, South)



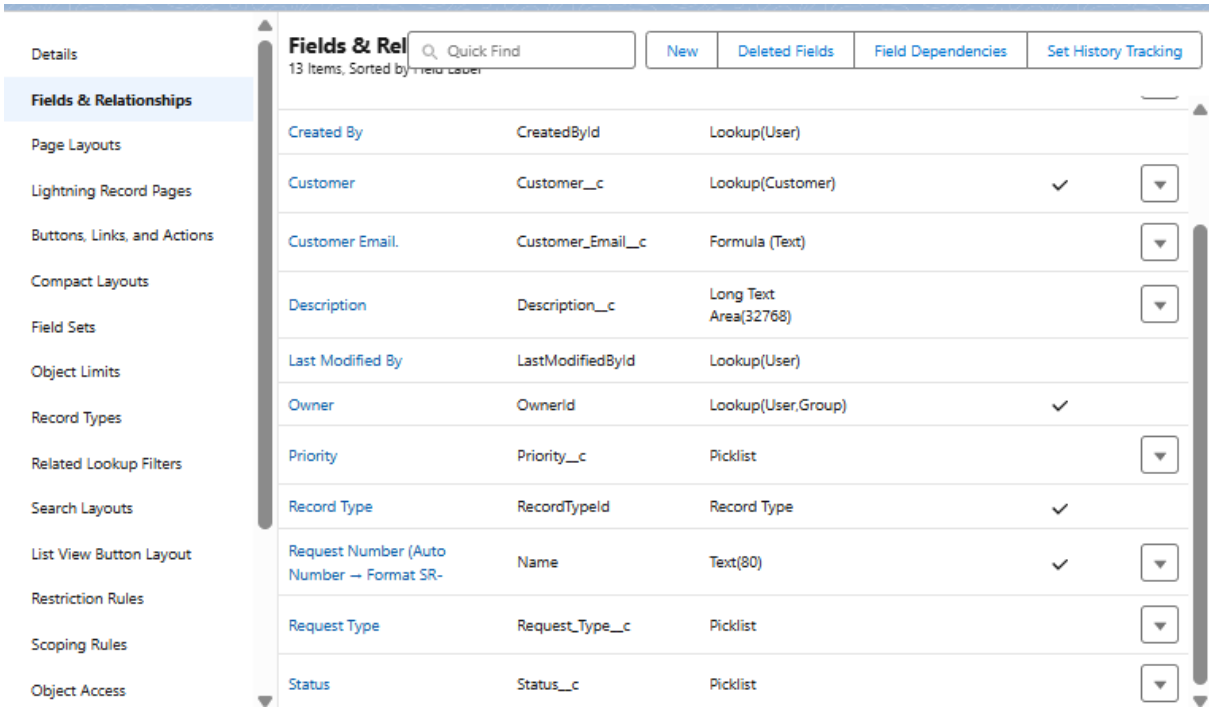
Plan__c

- Price (Currency)
- Validity (Number, 3 digits)
- Data Limit (Number, 3 digits, GB)
- Plan Type (Picklist: Prepaid, Postpaid, Broadband)



ServiceRequest__c

- Request Type (Picklist: Activation, Cancellation, Network Issue, Billing)
- Status (Picklist: New, In Progress, Resolved, Cancelled)
- Priority (Picklist: High, Medium, Low)
- Description (Long Text Area)



3. Relationships

- **Customer → Plan →** *Lookup* (a customer subscribes to a plan).
- **ServiceRequest → Customer →** *Lookup* (each request belongs to a customer).

---

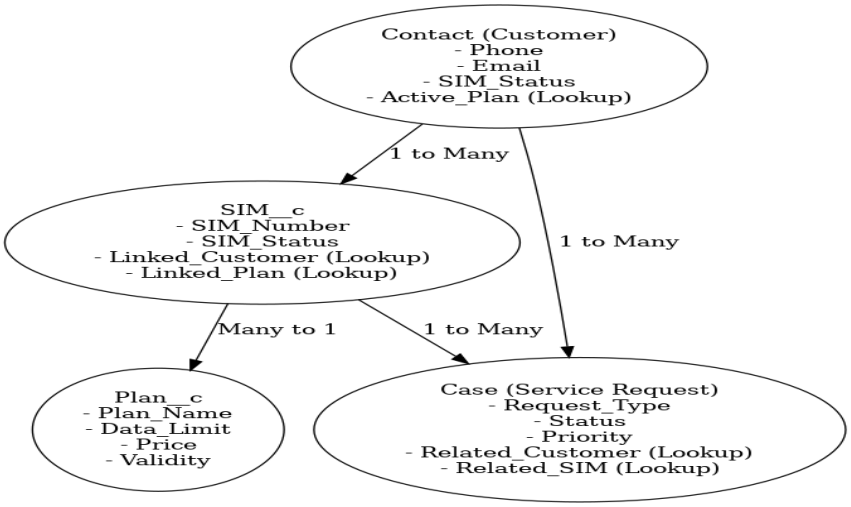
4. Page Layouts

- **Customer Layout →** Phone, Email, SIM Number, Status, Region.
- **Plan Layout →** Price, Validity, Data Limit, Plan Type.
- **Service Request Layout →** Request Type, Status, Priority, Description + related Customer.

---

5. Schema Builder

Schema Builder used to visualize objects and their relationships:

- Customer__c linked to Plan__c (Lookup).
- ServiceRequest__c linked to Customer__c (Lookup).
- 

---

6. Record Types

Record Types (ServiceRequest__c)

Created **different record types** to simplify request handling:

- **Activation Request →** For SIM activations.
- **Complaint Request →** For issues like network/billing.

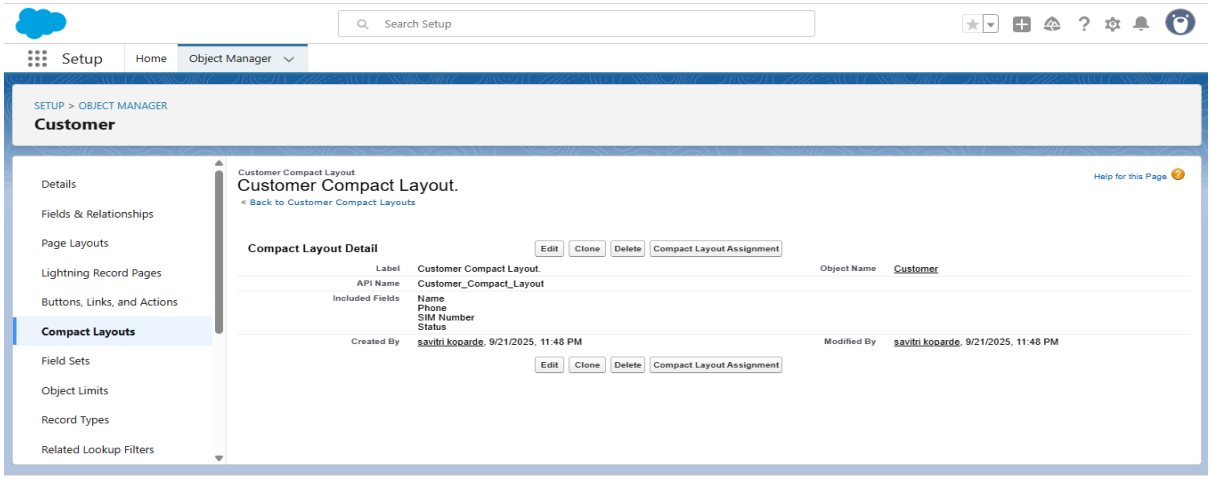
- **Cancellation Request** → For SIM termination.

This helps agents capture the right type of request quickly.

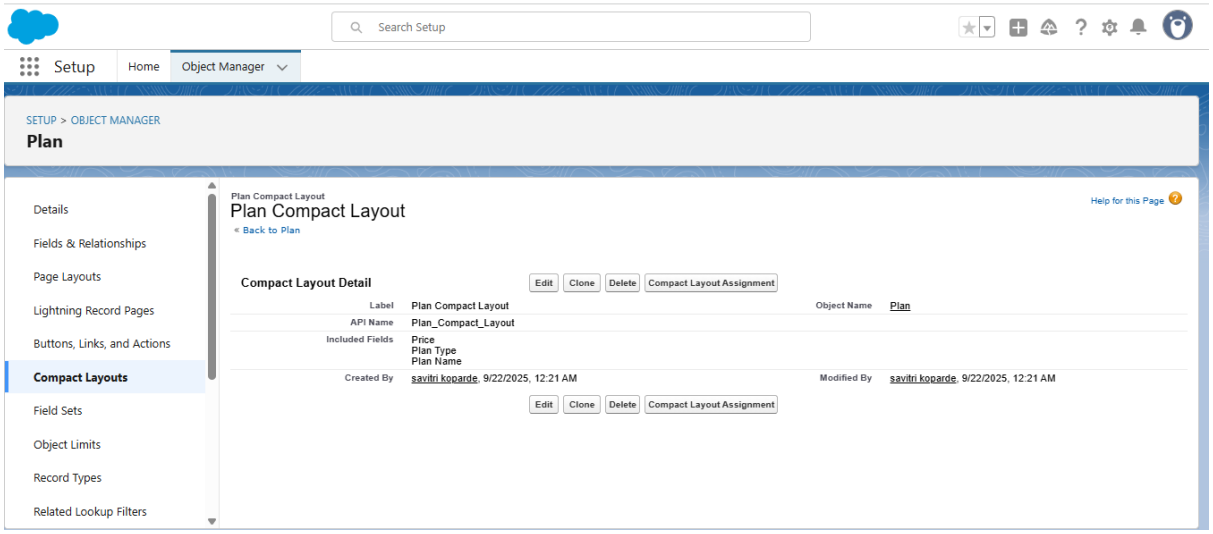
7.Compact Layouts

Optimized for Salesforce Mobile:

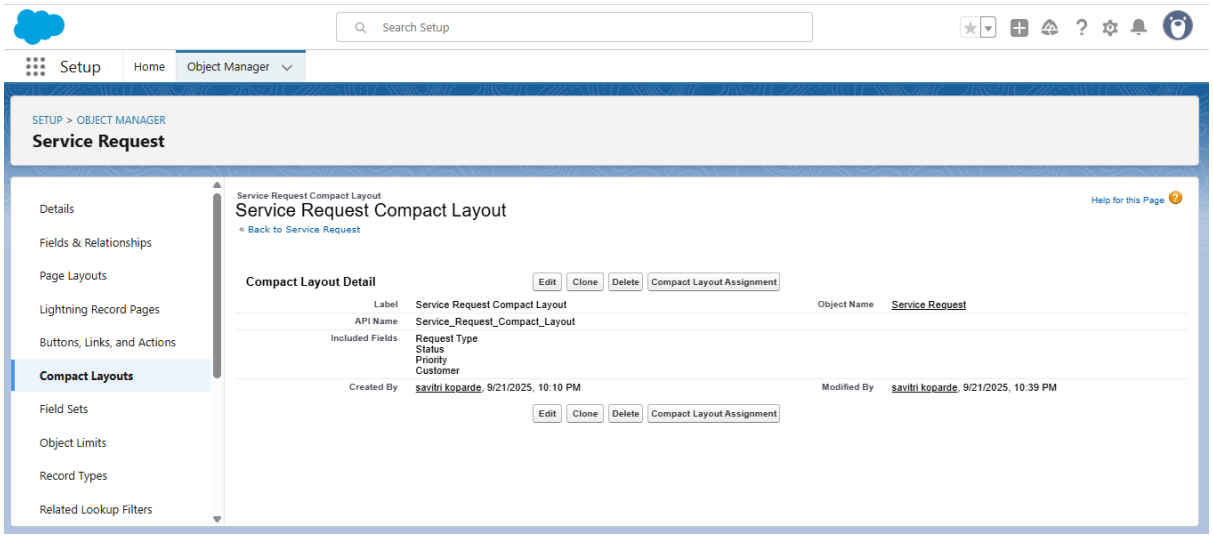
- **Customer__c Compact Layout** → Shows Name, Phone, SIM Number, Status.



- **Plan__c Compact Layout** → Shows Name, Price, Plan Type.



- **ServiceRequest__c Compact Layout** → Shows Request Type, Status, Priority.



Phase 4 Report: Must-Have Process Automation (Admin)

Goal

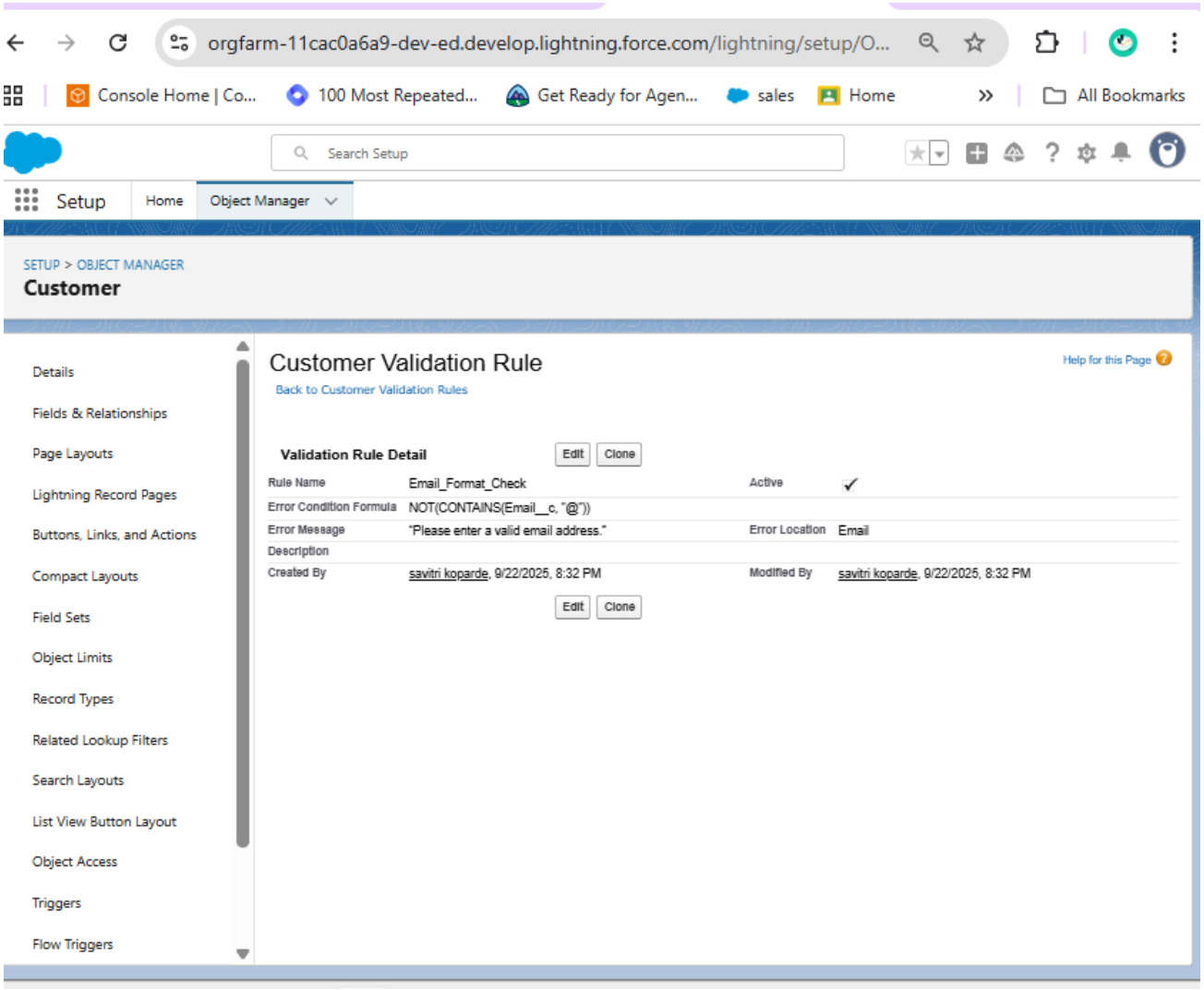
Automate critical telecom workflows in Salesforce to reduce manual work, improve service turnaround, and keep customers informed in real time.

---

1. Validation Rules

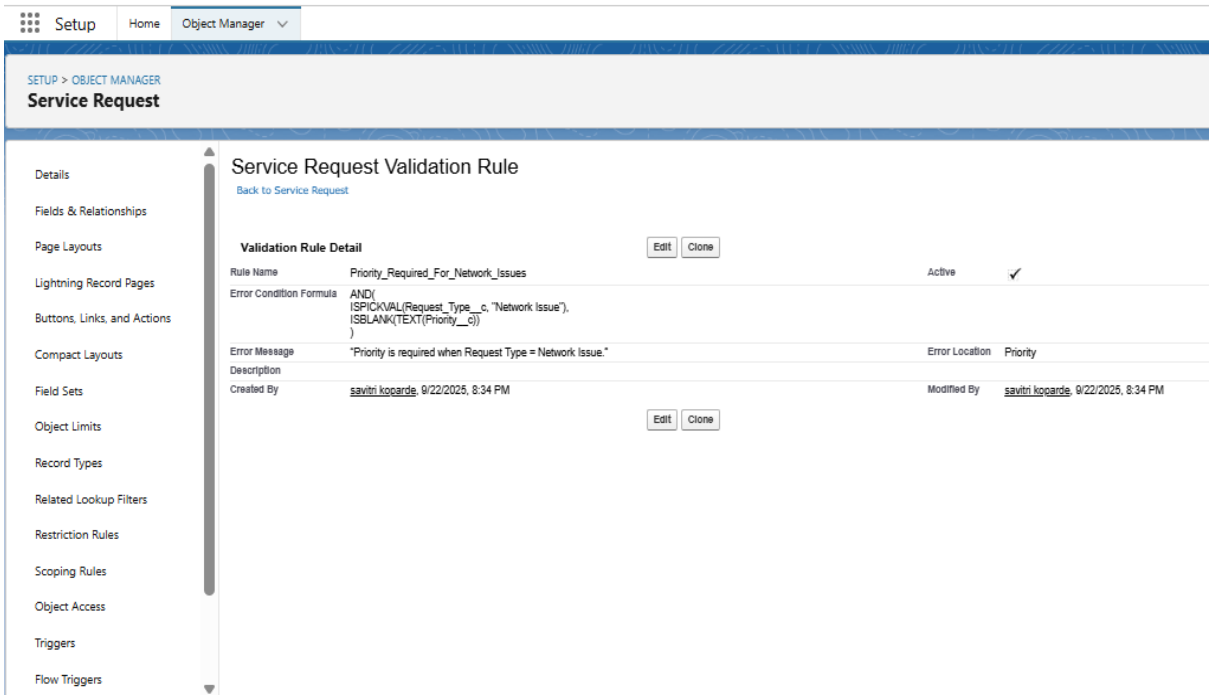
Purpose: Ensure data integrity and prevent incomplete/incorrect records.

- **Customer Email Validation**
    - Formula: NOT(CONTAINS(Email, "@"))
    - Error: “Enter a valid Email Address.”



- **Service Request Priority Validation**
  - Formula: AND(ISPICKVAL(Request_Type__c, "Network Issue"), ISBLANK(TEXT(Priority__c)))
  - Error: “Priority is required for Network Issues.”

Impact: Prevents bad customer data and enforces business rules.

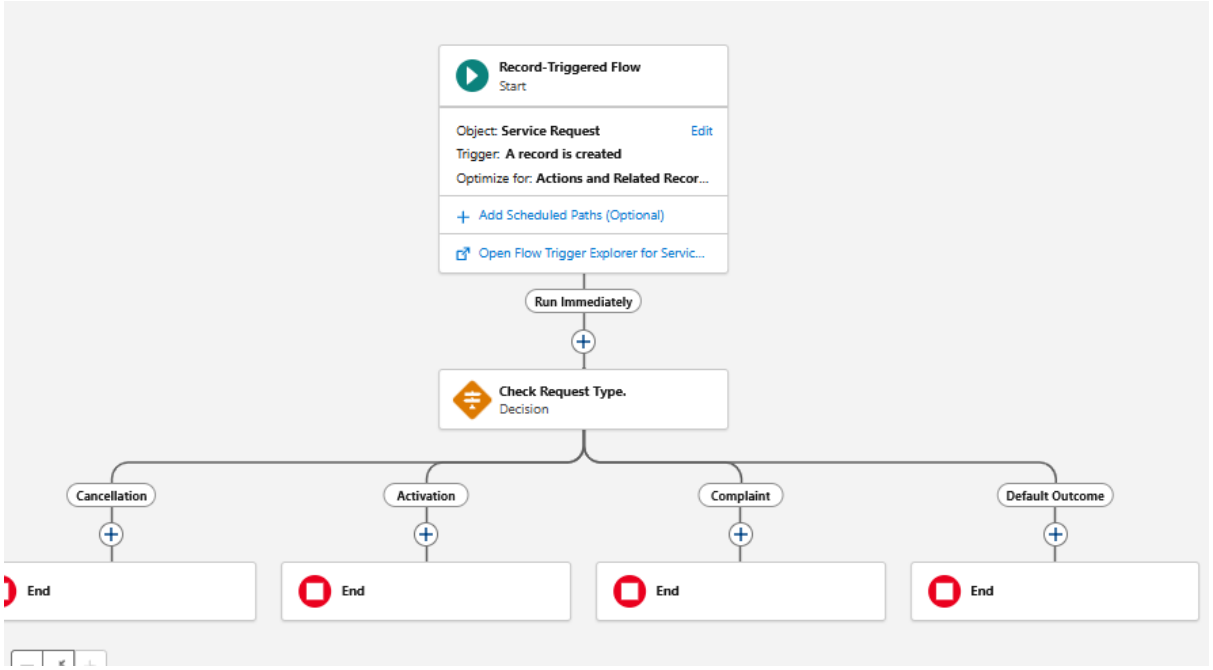


---

2. Flow Builder (Record-Triggered)

Purpose: Automate request handling and SIM status updates.

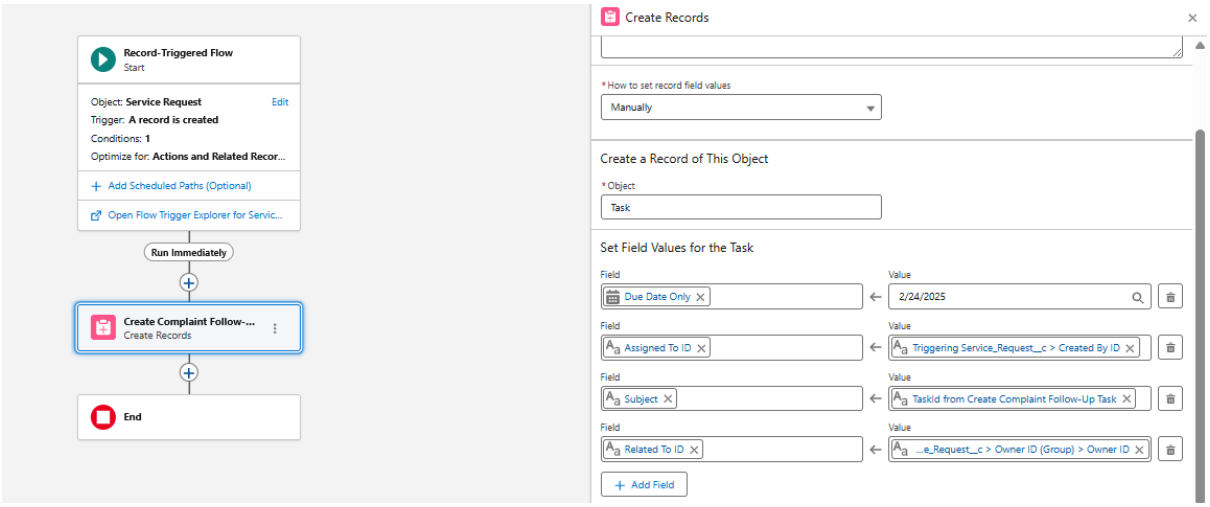
- **Round-Robin Assignment Flow**
    - Trigger: On ServiceRequest__c creation.
    - Logic: Distributes requests evenly among active Agents.
    - Outcome: No manual assignment needed, balanced workload.



- **SIM Cancellation Flow**

- Trigger: When Request_Type = Cancellation.
- Logic: Auto-updates Customer__c.Status = Inprogress.
- Outcome: SIM instantly deactivated, no agent action required.

Impact: Fast response, reduced manual errors, higher customer satisfaction.



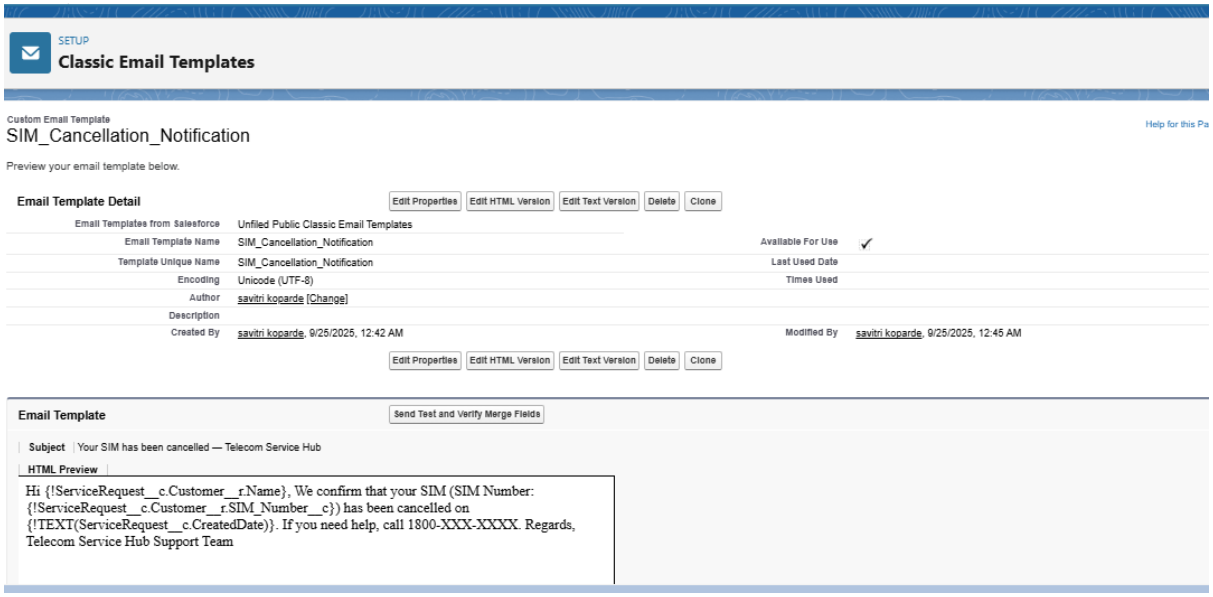
---

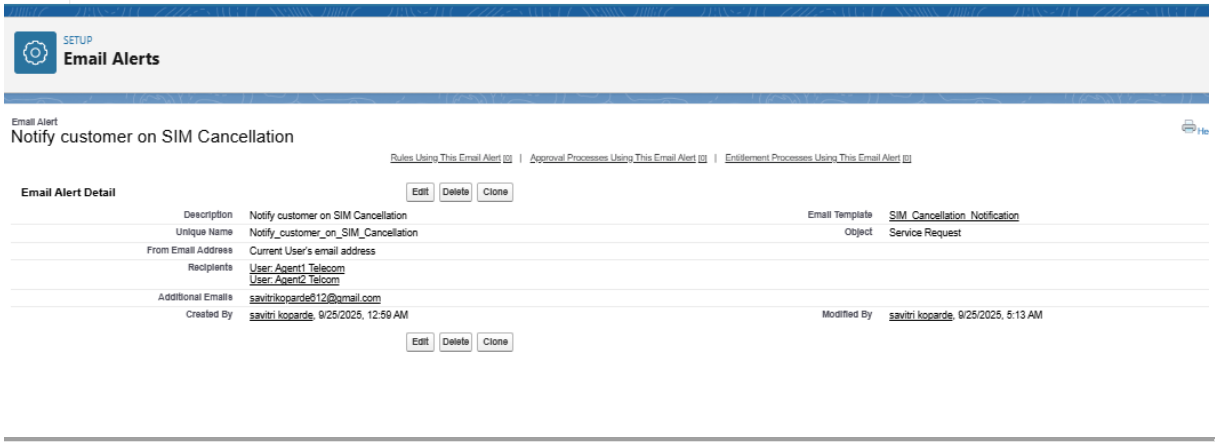
3. Email Alerts

Purpose: Keep customers informed automatically.

- **Activation Confirmation** → "Your SIM {SIM Number} has been activated successfully."
- **Cancellation Confirmation** → "Your SIM has been deactivated."

Impact: Customers receive real-time updates, boosting trust.





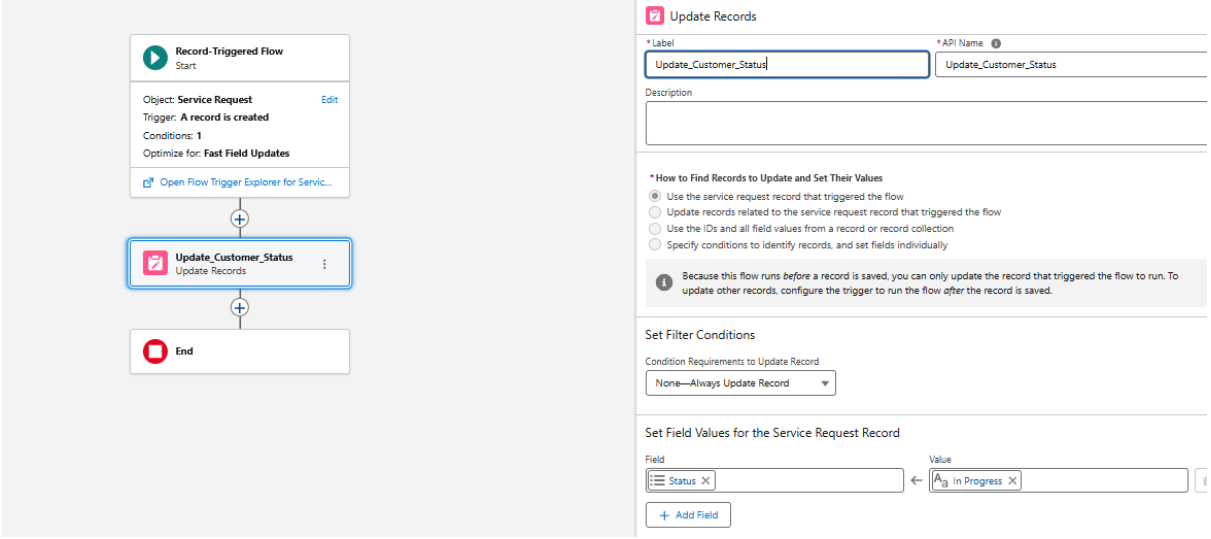
---

4. Field Updates

Purpose: Maintain accurate record states automatically.

- **Resolved Request Auto-Close**
  - Condition: When ServiceRequest__c.Status = Resolved.
  - Action: Update Status = Closed.

Impact: Ensures consistent lifecycle tracking of service requests.



---

5. Tasks

Purpose: Drive timely follow-ups for customer complaints.

- **Complaint Follow-Up Task**
    - Trigger: When Request_Type = Complaint.
    - Auto-created Task:
        - Subject: “Follow up with Customer”
        - Due Date: 2 hours from request creation.

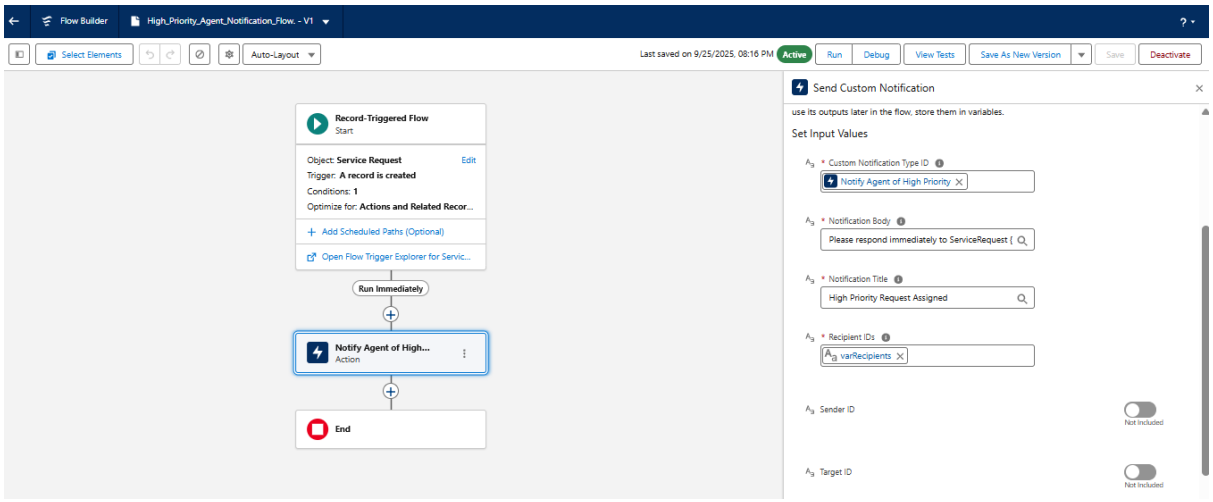
Impact: Agents act quickly on complaints, reducing churn risk.

---

6. Custom Notifications

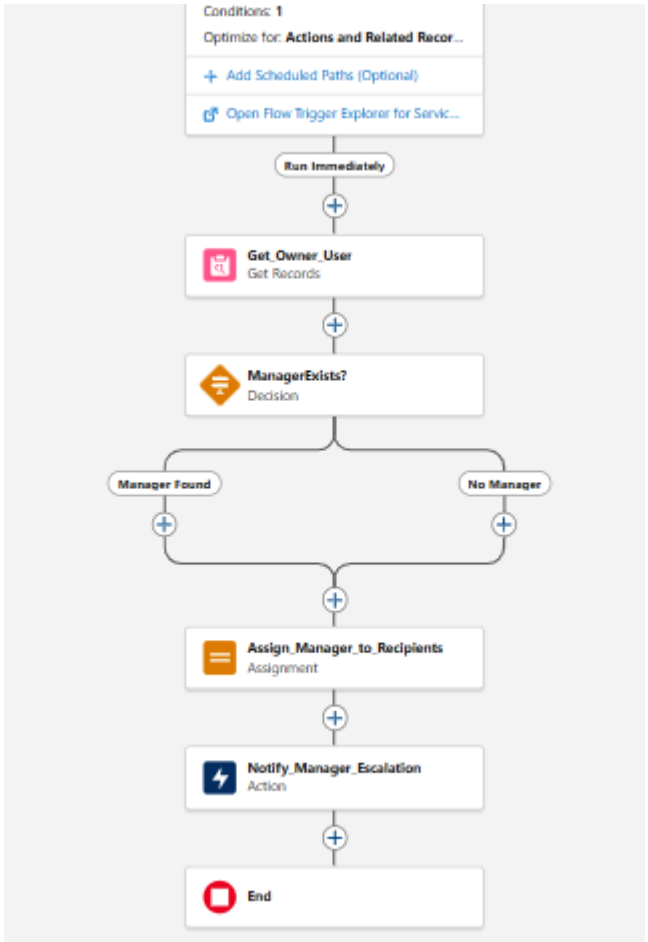
Purpose: Escalate critical service requests to Agents/Managers.

- **Agent Notification:**
    - Trigger: When High Priority Service Request assigned.
    - Message: “⚠️ High Priority Request Assigned: Please respond immediately.”



- **Manager Notification:**
  - Trigger: When approval or escalation required.
  - Message: “Approval required for ServiceRequest #{!Id}.”

Impact: Ensures immediate attention to high-risk cases.



Phase 5: Apex Programming (Developer)

Goal

Enable advanced customization in Salesforce using **Apex programming** to handle scenarios not possible with declarative automation. This includes writing triggers, classes, queries, and test code for maintaining data integrity and automating complex telecom workflows.

---

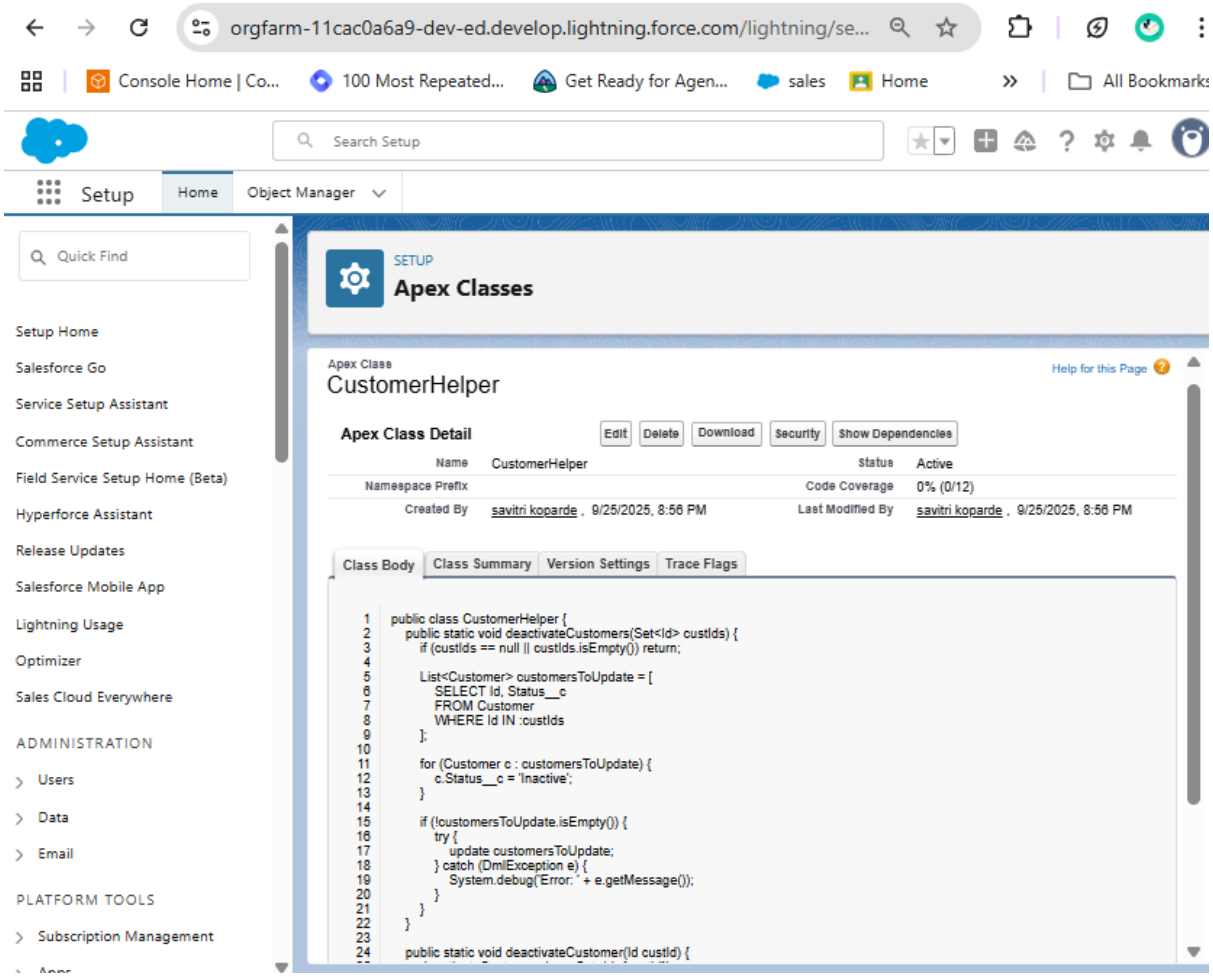
1. Apex Classes & Objects

Implementation:

- Created CustomerHelper.cls with reusable logic to update Customer status when a Service Request is processed.

```
public class CustomerHelper {
   public static void deactivateCustomer(Id custId) {
      try {
         Customer__c cust = [SELECT Id, Status__c FROM Customer__c WHERE Id = :custId LIMIT 1];
         cust.Status__c = 'Inactive';
         update cust;
      } catch (Exception e) {
         System.debug('Error while deactivating customer: ' + e.getMessage());
      }
   }
}
```

Purpose: Centralized business logic for reusability.



---

2. Apex Trigger with Design Pattern

Implementation:

- Created trigger ServiceRequestTrigger on ServiceRequest__c.
- Delegated logic to ServiceRequestTriggerHandler.cls.

Trigger:

```
trigger ServiceRequestTrigger on ServiceRequest__c (after insert, after update) {
    if(Trigger.isAfter && Trigger.isInsert) {
        ServiceRequestTriggerHandler.handleAfterInsert(Trigger.new);
    }
}
```

Handler Class:

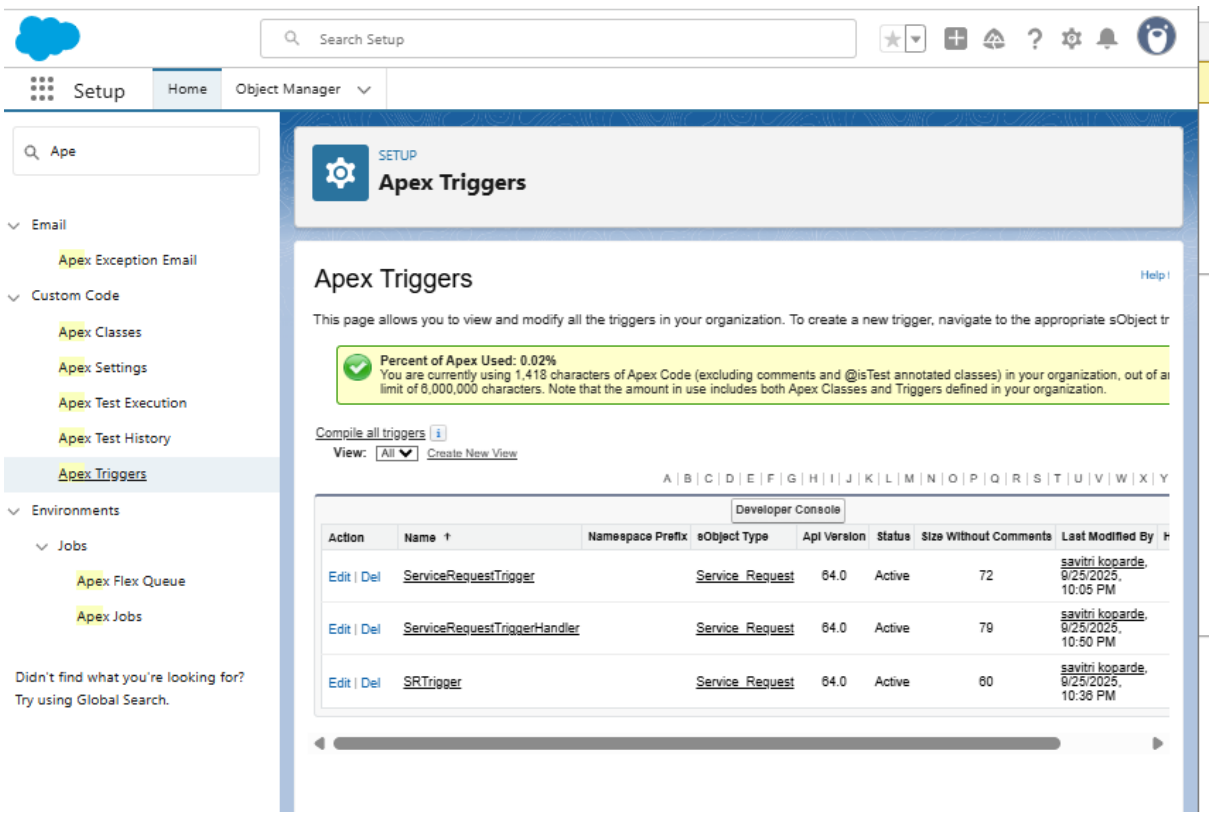
```
public class ServiceRequestTriggerHandler {
    public static void handleAfterInsert(List<ServiceRequest__c> newReqs) {
```

```
        Set<Id> custIds = new Set<Id>();


        for(ServiceRequest__c req : newReqs) {
            if(req.Request_Type__c == 'Cancellation' && req.Customer__c != null) {
                custIds.add(req.Customer__c);
            }
        }


        if(!custIds.isEmpty()) {
            for(Id cId : custIds) {
                CustomerHelper.deactivateCustomer(cId);
            }
        }
    }
}
```

Purpose: Updates related Customer to “Inactive” when a cancellation request is created.



---

3. SOQL Query

- Used in CustomerHelper.cls to fetch Customer details:

```
Customer__c cust = [SELECT Id, Status__c FROM Customer__c WHERE Id = :custId LIMIT 1];
```

Purpose: Retrieve customer data from Salesforce for processing.

---

4. Collections (List, Set, Map)

- **Set**: custIds ensures only unique Customers are updated.
- **List**: Trigger passes list of new ServiceRequests.
- **Map**: Could be used to map Customer Id → Customer record for bulk operations.

Example:

```
Map<Id, Customer__c> custMap = new Map<Id, Customer__c>(
   [SELECT Id, Status__c FROM Customer__c WHERE Id IN :custIds]
);
```

---

5. Control Statements

- **If condition** checks request type.
- **For loop** iterates over trigger records.

Example:

```
for(ServiceRequest__c req : newReqs) {
   if(req.Request_Type__c == 'Cancellation') {
      // logic
   }
}
```

---

6. Exception Handling

- Used try-catch in CustomerHelper.cls:

```
try {
```

```
    update cust;
} catch (Exception e) {
    System.debug('Error: ' + e.getMessage());
}
```

Purpose: Ensures errors don’t break transaction flow.

---

7. Test Class

Implementation:

```
@isTest
public class TestServiceRequestTrigger {
    @isTest
    static void testCancellationFlow() {
        // Create test customer
        Customer__c cust = new Customer__c(
            Name = 'Test Customer',
            Email__c = 'test@test.com',
            SIM_Number__c = '12345',
            Status__c = 'Active'
        );
        insert cust;

        // Create cancellation request
        ServiceRequest__c req = new ServiceRequest__c(
            Request_Type__c = 'Cancellation',
            Customer__c = cust.Id,
            Status__c = 'New'
        );
        insert req;
```

```
        // Fetch updated customer
        Customer__c updatedCust = [SELECT Status__c FROM Customer__c WHERE Id = :cust.Id];
        System.assertEquals('Inactive', updatedCust.Status__c);
    }
}
```

Purpose: Validates that trigger and helper class logic works correctly. Achieves code coverage required for deployment.

Phase 6: User Interface Development

Goal:
Provide a functional Salesforce Lightning UI for Agents and Managers to manage Customers, SIM activations/cancellations, and Service Requests efficiently, with only essential features.

---

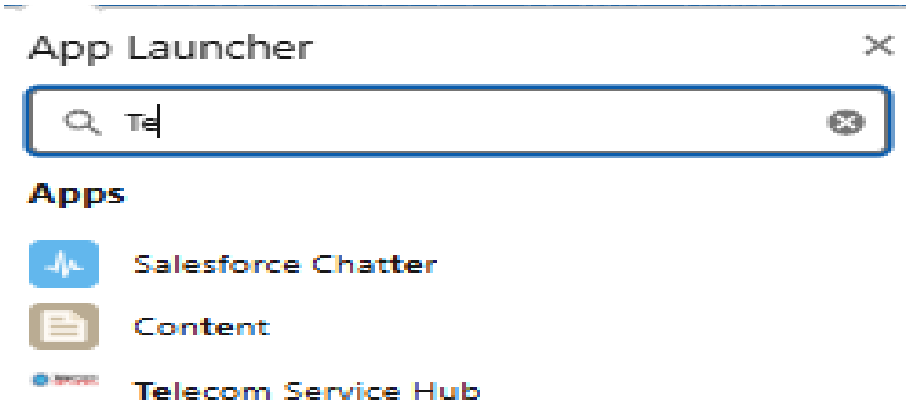
1. Lightning App

Purpose: Central workspace for Agents and Managers.

Steps:

1. Setup → **App Builder → Apps → New Lightning App**
2. Enter Details:
    - Name: **Telecom Service Hub**
    - Description: Telecom CRM for Customer & SIM Request Management
    - Logo & Theme (optional)
3. Navigation: **Standard Navigation**
4. Add Tabs:
    - **Customers (Customer__c)**
    - **Service Requests (Service_Request__c)**
    - **Plans (Plan__c)**
5. Assign App to **Agent** and **Manager** profiles
6. Click **Finish → Save**

Impact: Users have a single workspace with quick access to main objects.



---

2. Tabs

Purpose: Quick access to main objects.

Steps:

- Ensure **Custom Object Tabs** exist for Customer__c and Service_Request__c
- Use existing tab for Plan__c

Impact: Objects are accessible from the app navigation.

---

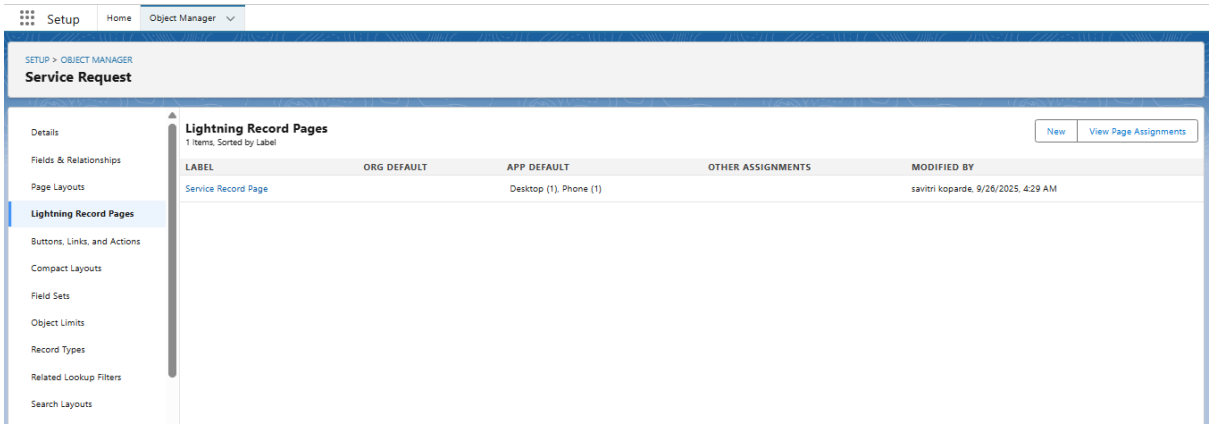
3. Record Pages

Purpose: Customize layout for clarity and efficiency.

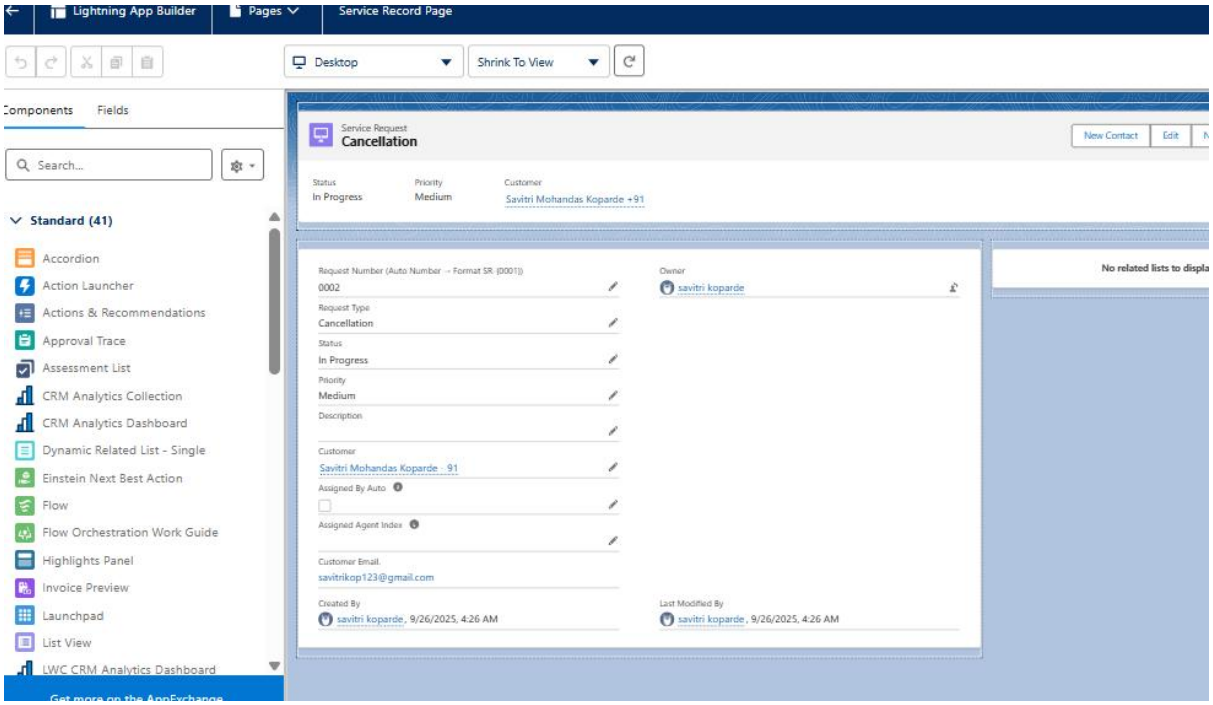
Steps:

1. Setup → **Object Manager → Customer / Service_Request__c → Lightning Record Page**
2. Add Components:
    - **Highlights Panel**: Key fields like Status, Plan, Priority
    - **Related Lists**: Service Requests per Customer
3. Assign Lightning Page to **Agent** and **Manager** profiles
4. Save & Activate

Impact: Shows important information clearly for workflow efficiency.



Structure:



Phase 7: Integration & External Access – Must-Have Documentation

Goal: Enable Salesforce to securely communicate with external systems, exchange data, and automate processes for the Telecom Service Hub project.

1. Named Credentials

Purpose: Store external API authentication securely for callouts. **Steps:**

1. Setup → **Named Credentials → New Named Credential**

Fill in: Label: TelecomAPI

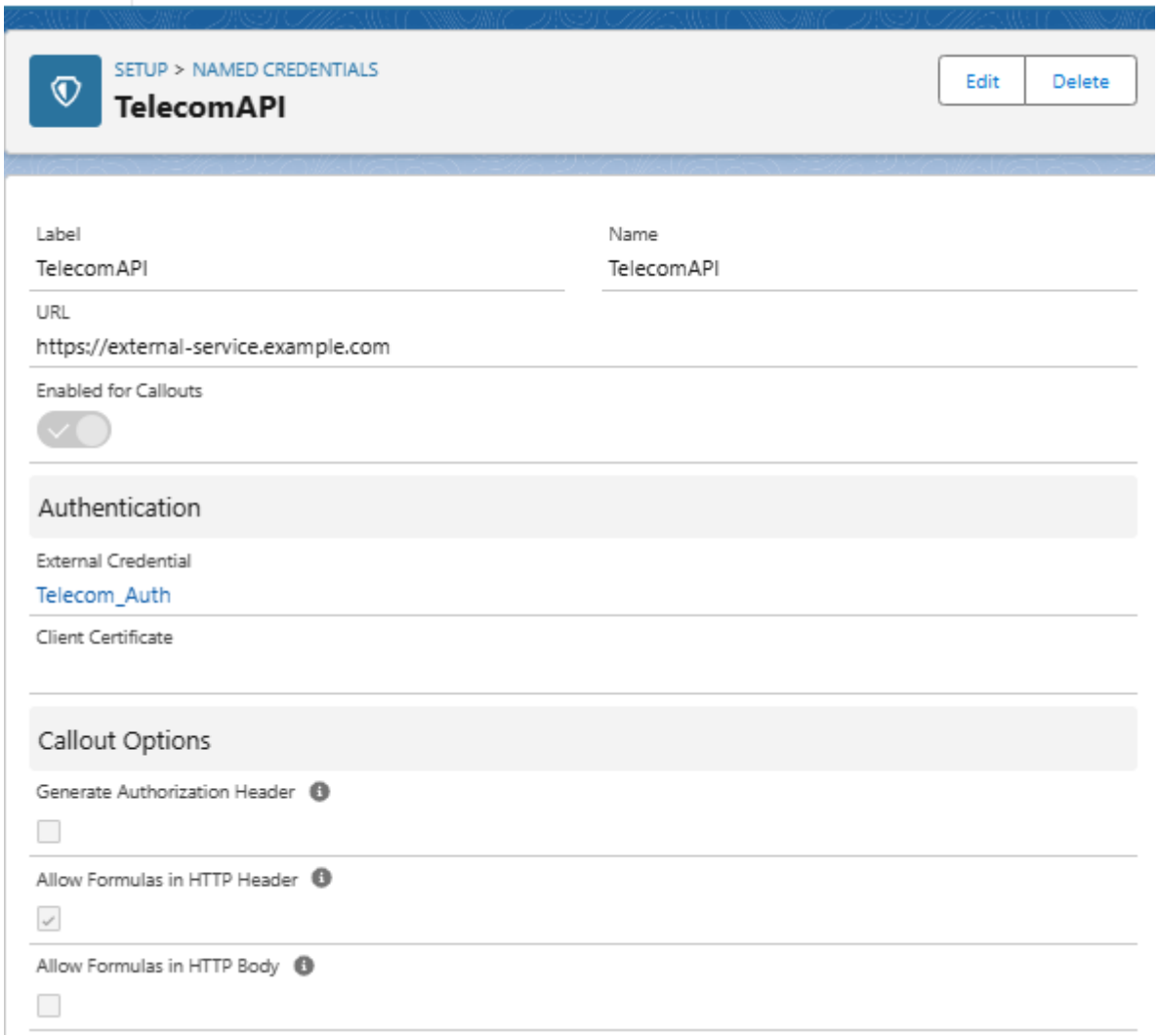
URL: https://external-service.example.com

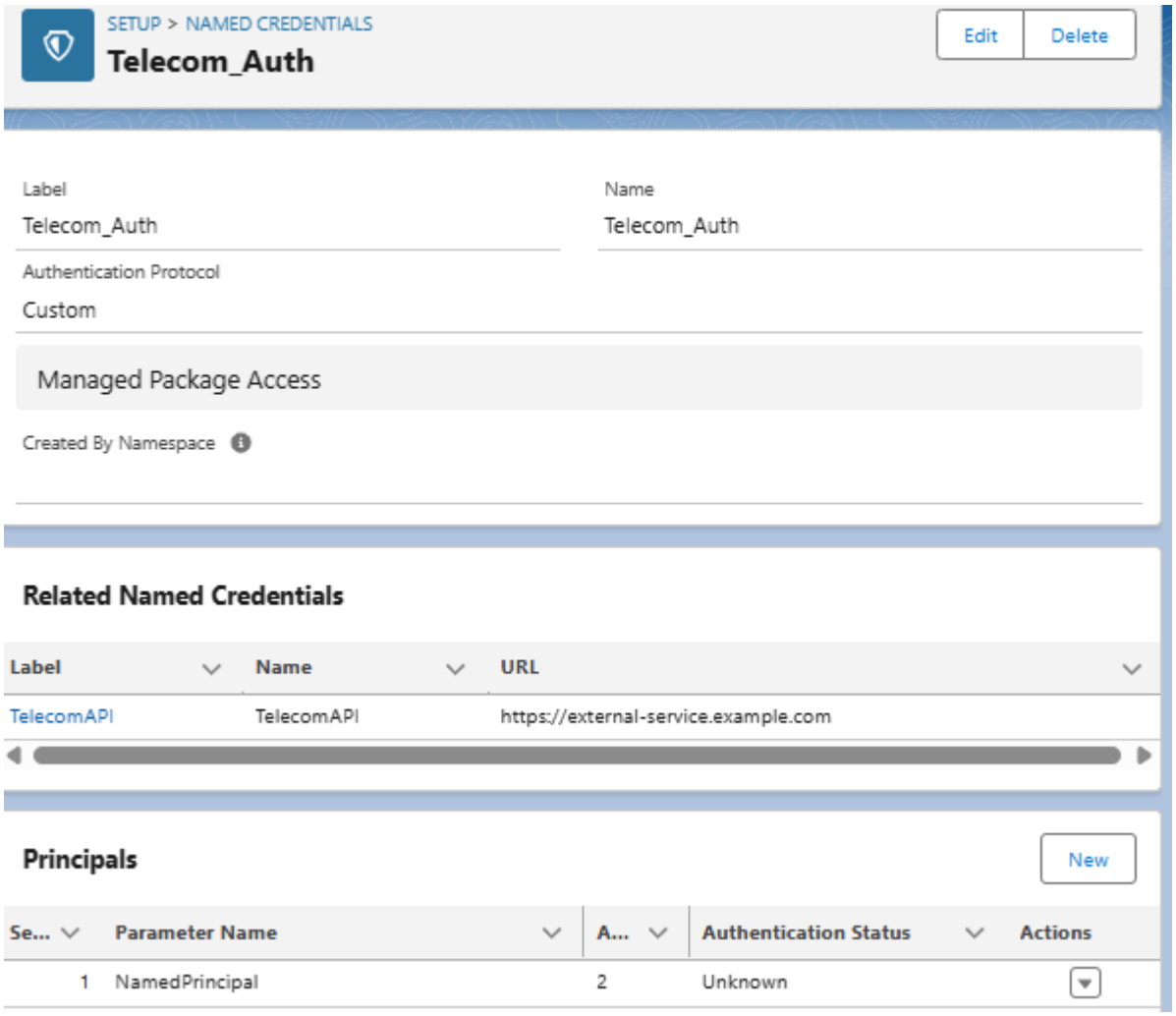
Identity Type: Named Principal

Authentication: Password or OAuth

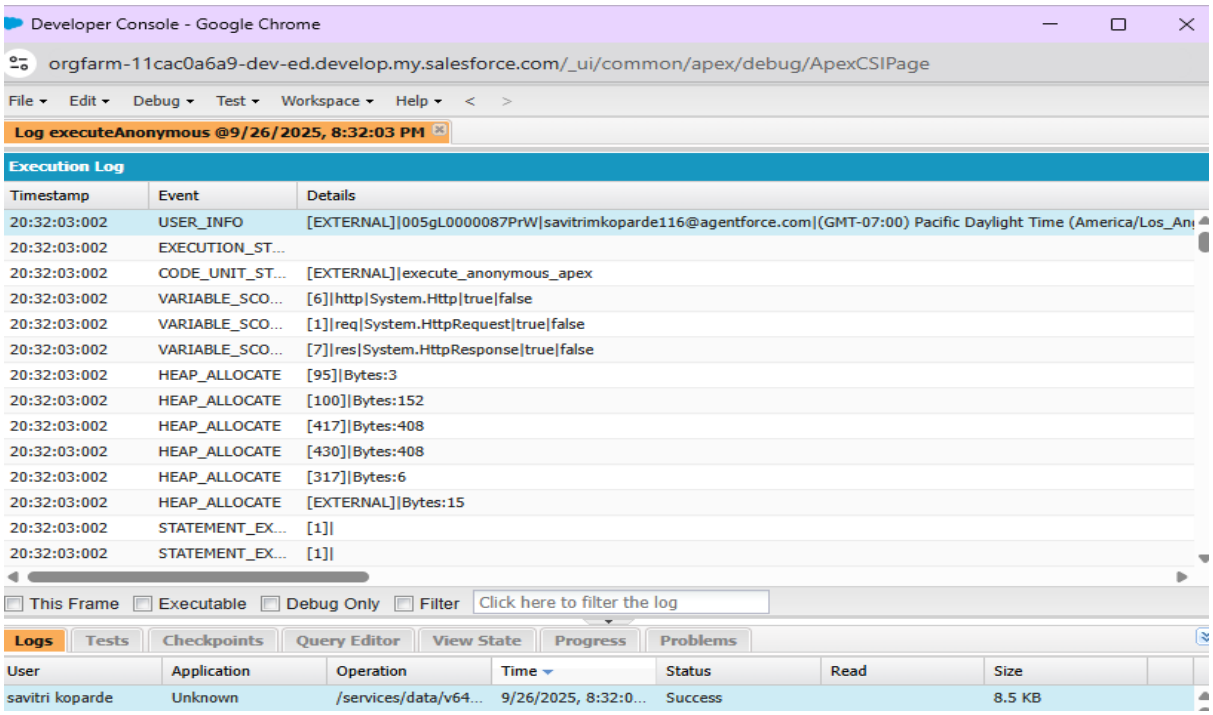
Impact: Makes external API calls secure and easy to manage.

Named Credentials:





Developer Console (Apex code)



2. Remote Site Settings

Purpose

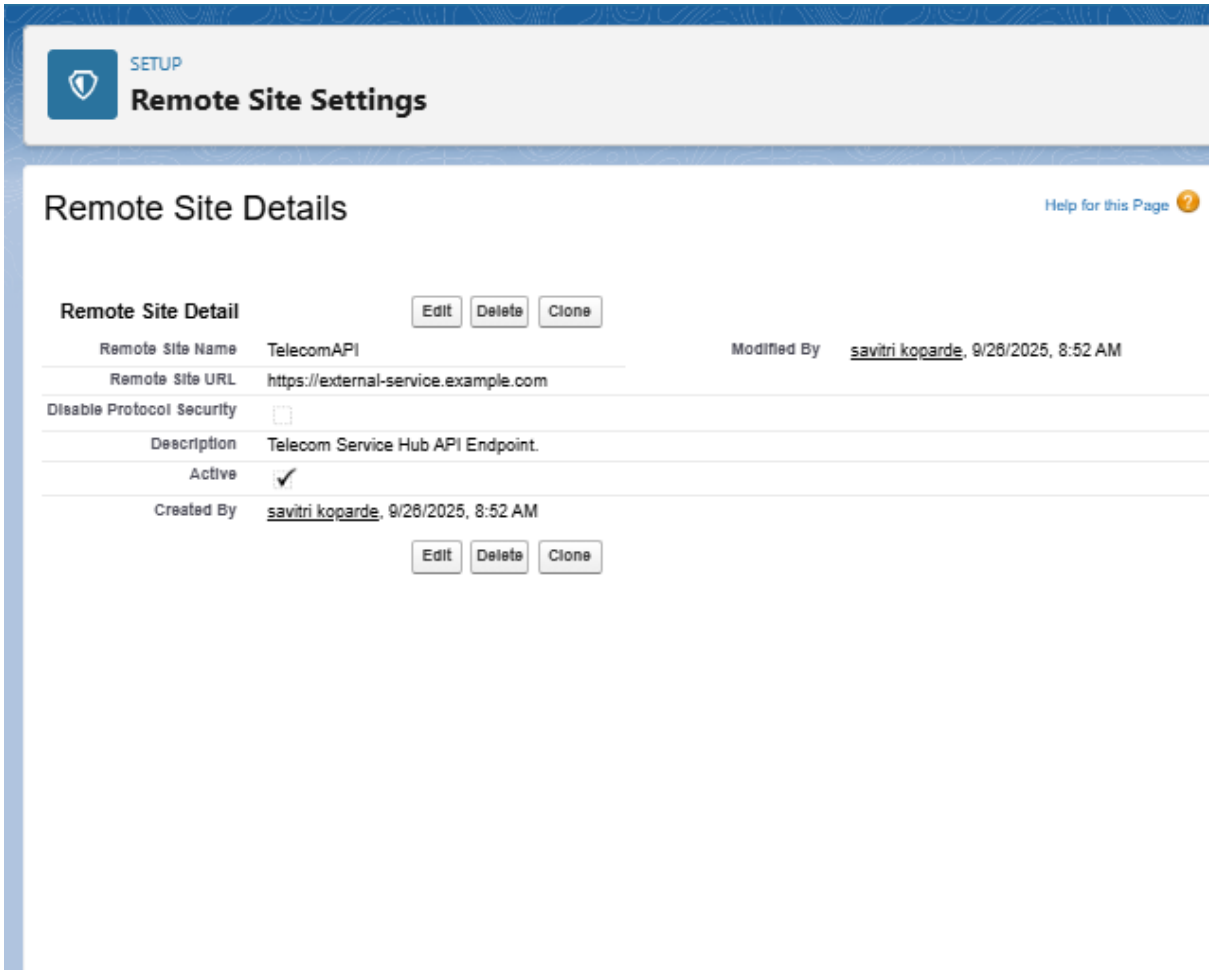
If using legacy authentication (without Named Credentials), Salesforce requires whitelisting the external API domain.

Steps

1. Go to **Setup → Remote Site Settings → New Remote Site**.
2. Enter:
    - **Remote Site Name:** TelecomAPI
    - **Remote Site URL:** https://external-service.example.com
3. Save

Impact

Ensures Salesforce allows outbound calls to the external API.



---

3. Sample REST Callout (Apex Integration)

Purpose

Demonstrates how Salesforce can fetch data from an external system.

🛠 Code Example

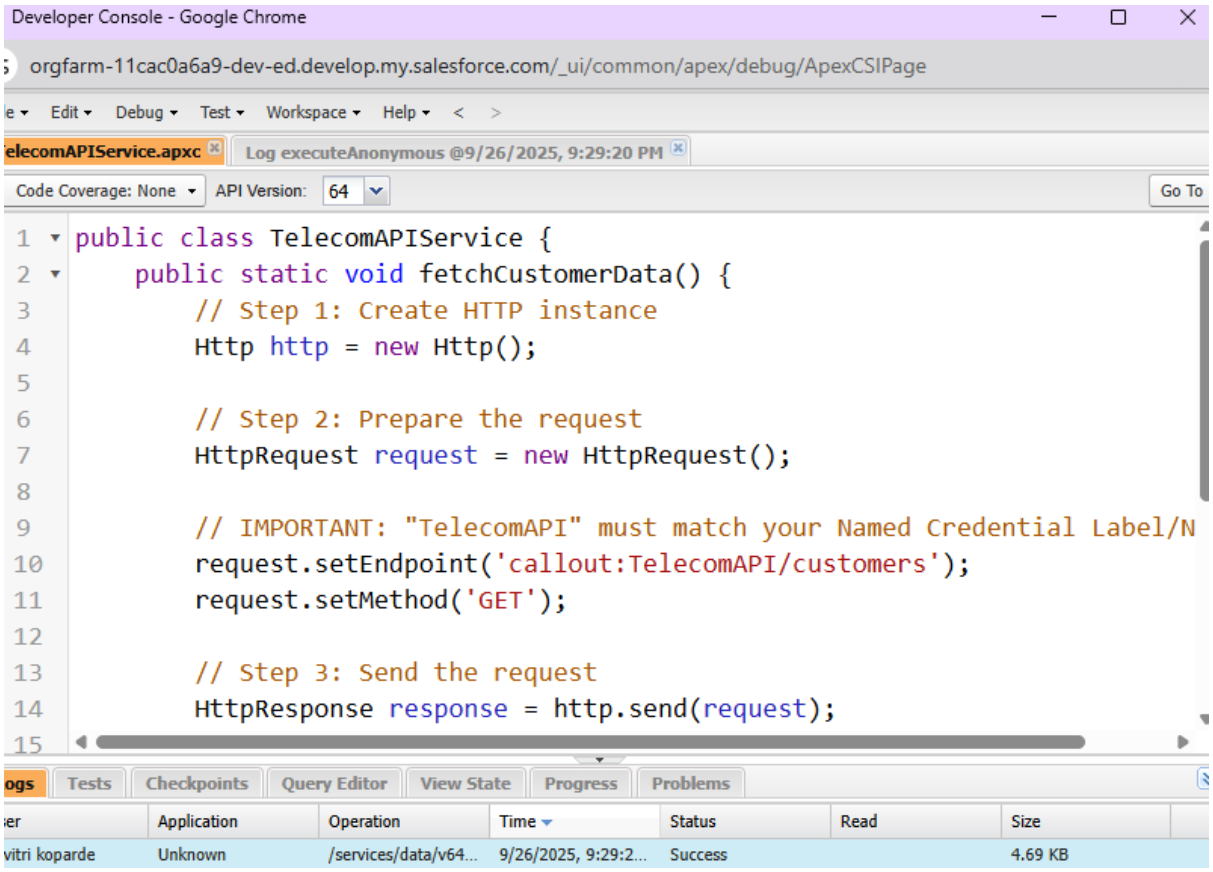
```
public class TelecomAPIService {
    public static void fetchCustomerData() {
        Http http = new Http();
        HttpRequest request = new HttpRequest();
        // Use Named Credential
        request.setEndpoint('callout:TelecomAPI/customers');
        request.setMethod('GET');

        HttpResponse response = http.send(request);

        if (response.getStatusCode() == 200) {
            System.debug('Response: ' + response.getBody());
        } else {
            System.debug('Error: ' + response.getStatus());
        }
    }
}
```

Impact

- Shows Salesforce making a secure outbound call.
- This can later be extended to fetch Service Requests, Plans, or SIM status.



By completing this phase, our **Telecom Service Hub** can:

- Securely connect to external APIs.
- Demonstrate real-time data exchange.
- Keep credentials secure with **Named Credentials**.

Phase 8: Data Management & Deployment –

Goal: Manage data and deploy project changes effectively with only the essential steps.

---

1. Data Import Wizard

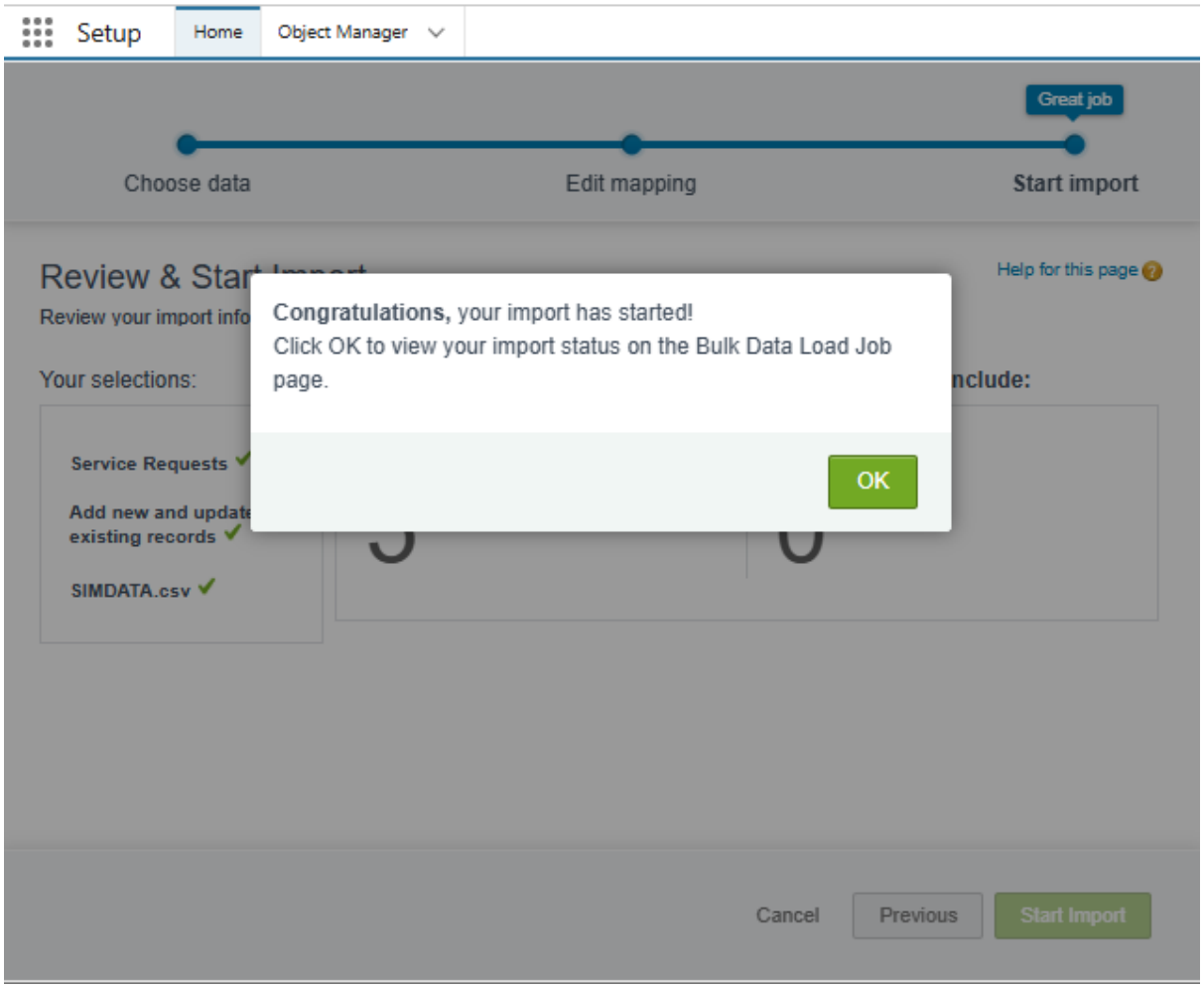
Purpose: Quickly import records into Salesforce.

Steps:

1. **Setup → Data → Data Import Wizard**
2. Choose object: **Customer__c, Service_Request__c, Plan__c**
3. Upload CSV → Map fields → Start Import

4. Review import results

Impact: Populate initial data for testing or demonstration.



---

2. Duplicate Rules

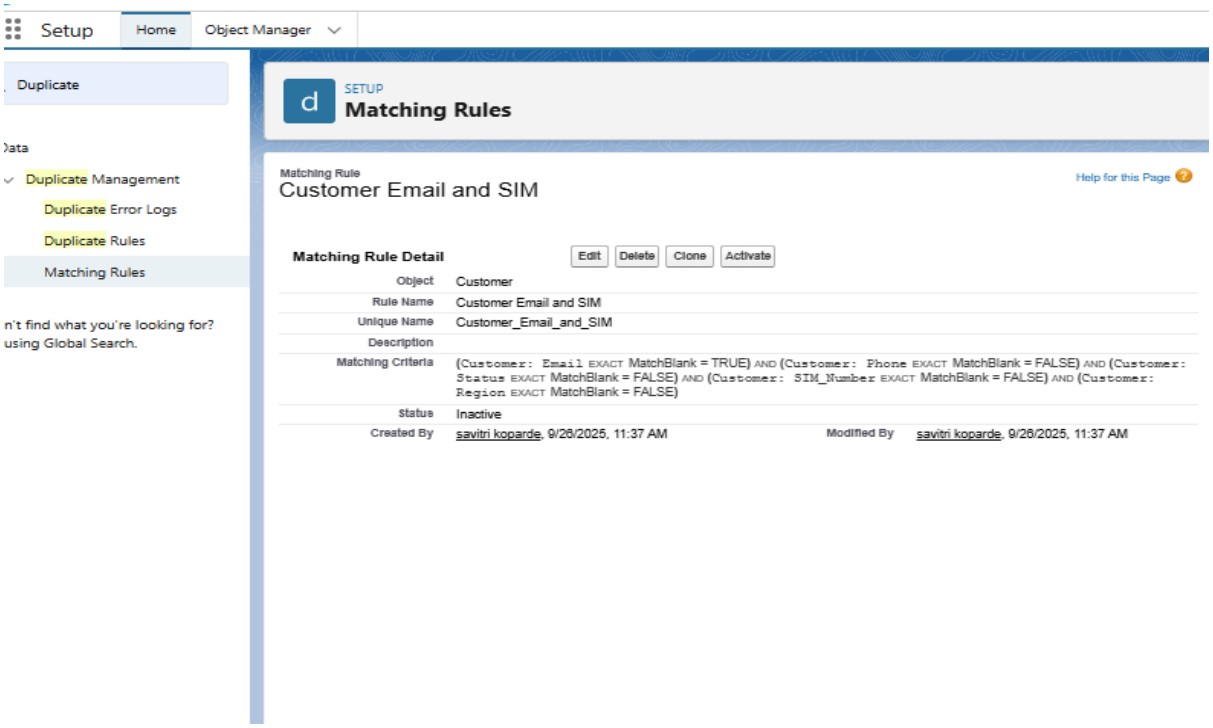
Purpose: Prevent duplicate records (important for Customer data).

Steps:

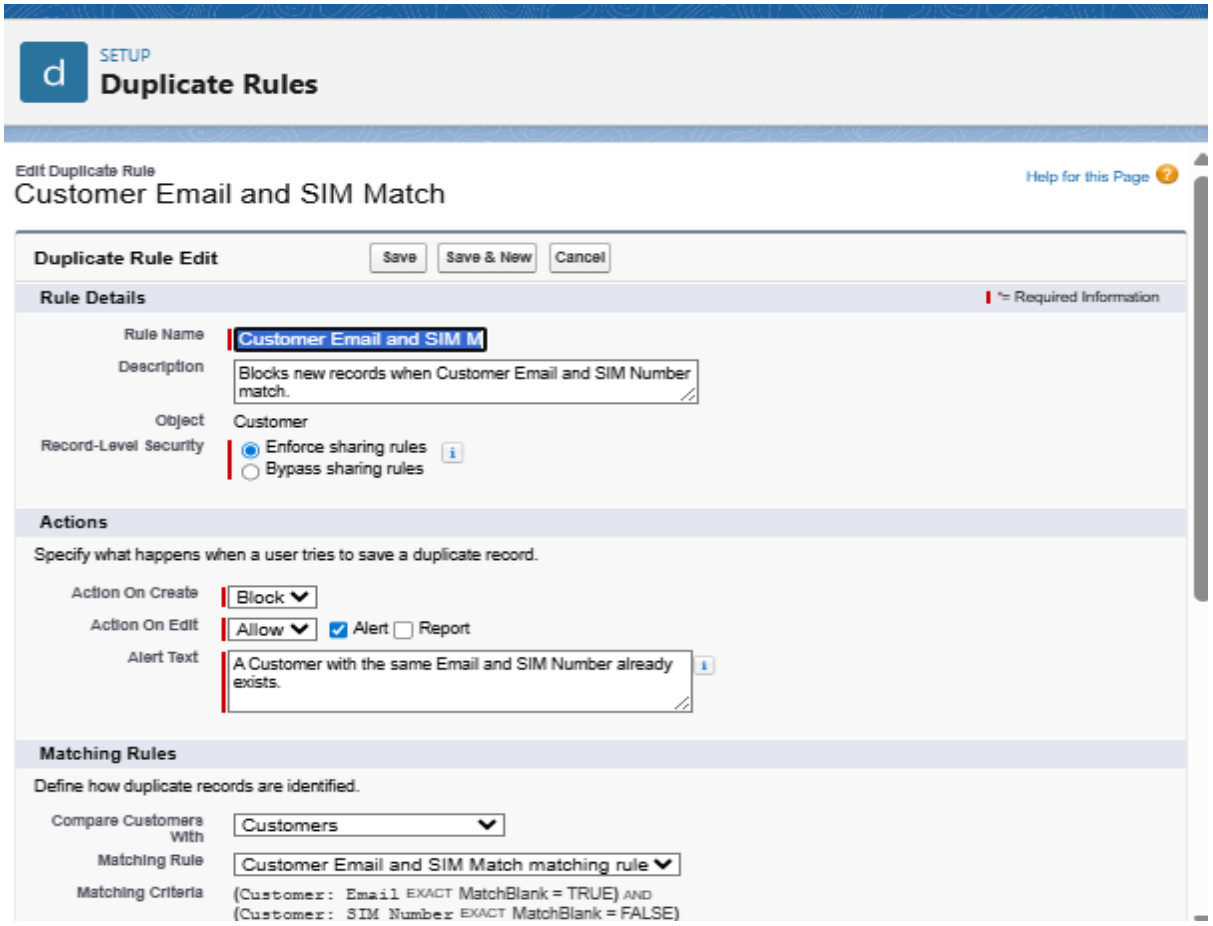
1. **Setup → Duplicate Management → Duplicate Rules → New Rule**
2. Select object: **Customer__c**
3. Matching Rule: **Email__c + SIM_Number__c**
4. Action: **Alert / Block duplicates** → Activate

Impact: Ensures data integrity and avoids duplicate entries.

Matching rules:



Duplicate rules:



---

3. Change Sets

Purpose: Deploy metadata (objects, flows, triggers) from Sandbox to Production.

Steps:

1. **Setup → Outbound Change Sets → New Change Set**
2. Add components: Custom Objects, Flows, Apex Classes
3. Upload to target org → Validate & Deploy

Impact: Moves Salesforce configurations safely between environments.

---

4. VS Code & Salesforce CLI (SFDX)

Purpose: Optional but helpful for deployment and version control.

Steps:

1. Install VS Code + Salesforce Extension Pack
2. Authenticate org:

sfdx force:auth:web:login -a DevOrg

3. Deploy metadata:

sfdx force:source:deploy -p force-app

Impact: Provides developer-friendly deployment workflow.

Phase 9: Reporting, Dashboards & Security Review – Must-Have Documentation

Goal: Enable Managers and Agents to track performance, monitor customer requests, and maintain data security in Salesforce.

---

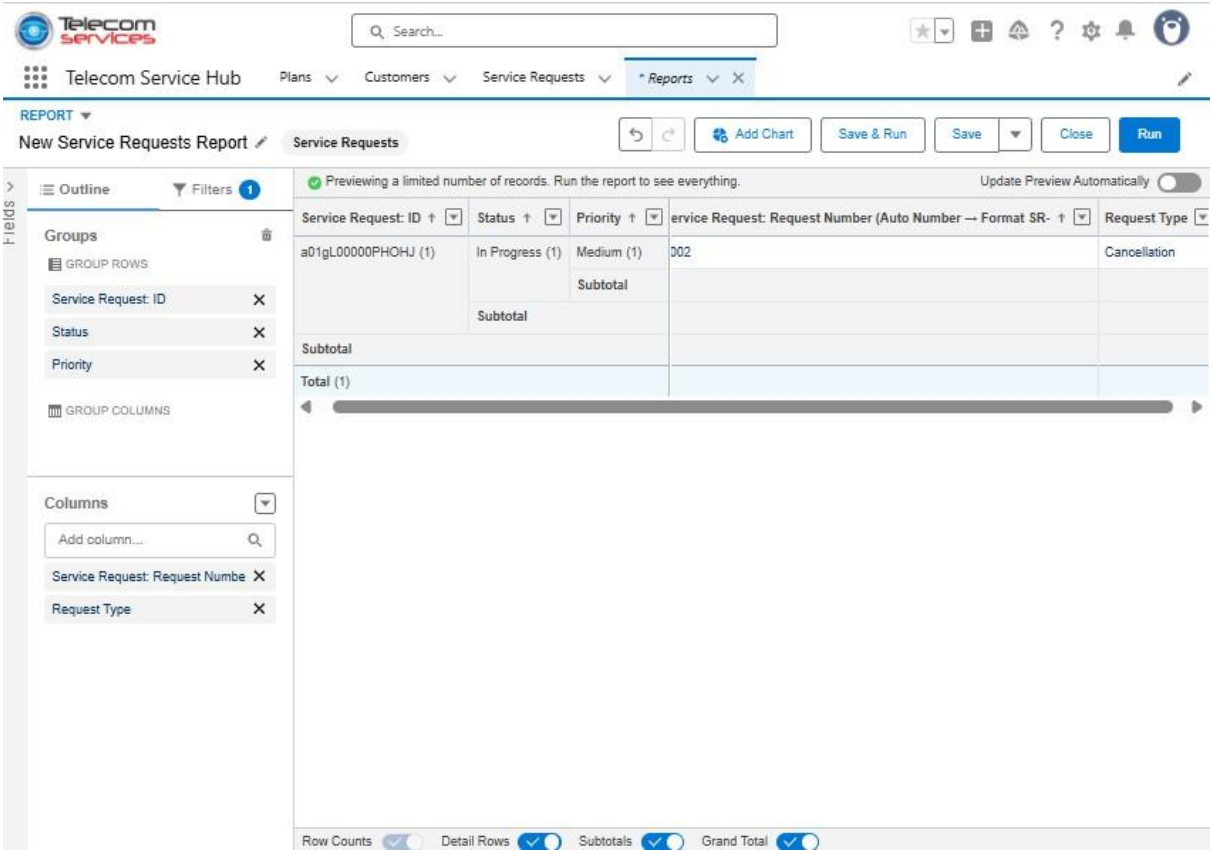
1. Reports

Purpose: Summarize and analyze data for decision-making.

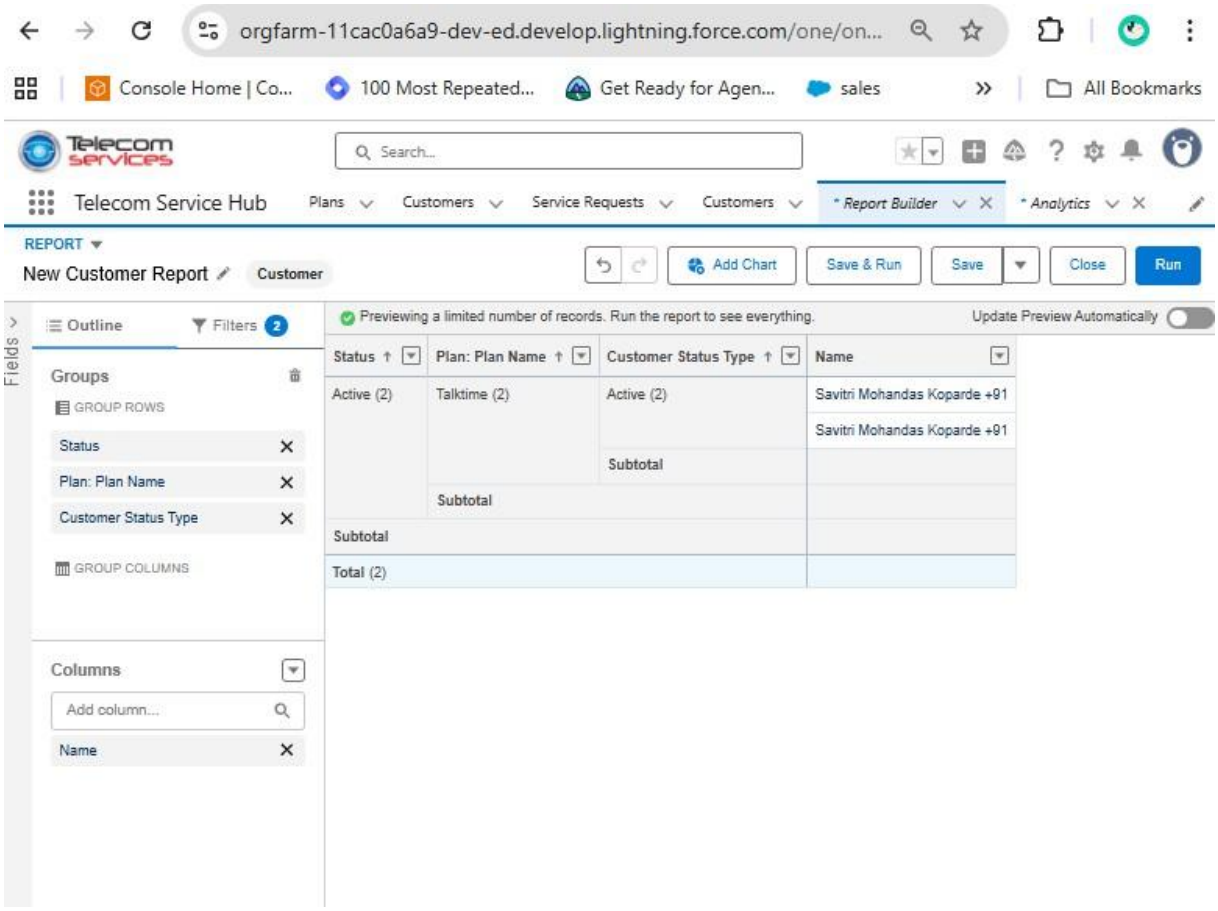
Steps (must-have reports):

1. **Service Requests by Type**

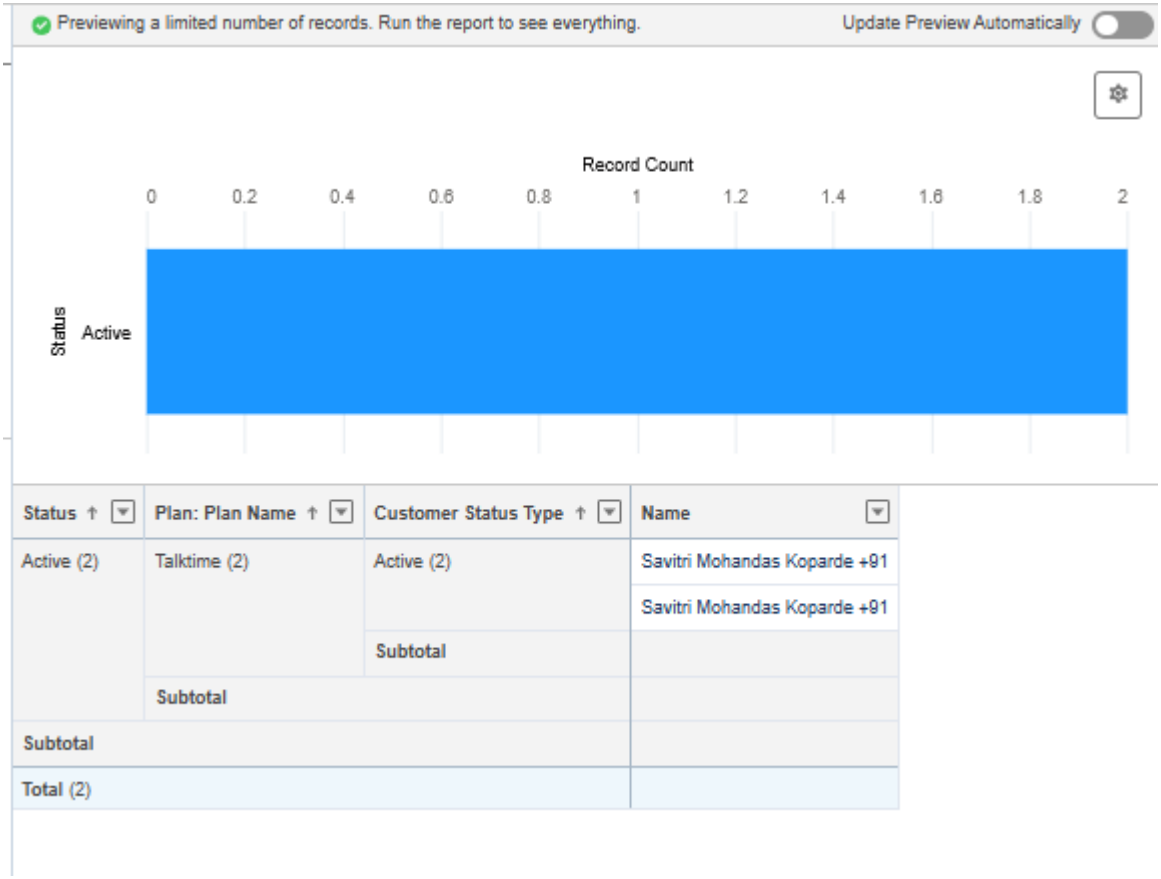
- Object: Service_Request_c
- Type: Summary Report
- Group by: Request_Type_c
- Show: Count of records, Status, Priority



2. **Active vs Cancelled Customers (Churn Analysis)**
   - Object: Customer_c
   - Type: Summary Report
   - Group by: Status_c
   - Metrics: Count



Anlysis of report using bar chart:



Status	Plan: Plan Name	Customer Status Type	Name
Active (2)	Talktime (2)	Active (2)	Savitri Mohandas Koparde +91
			Savitri Mohandas Koparde +91
		Subtotal	
	Subtotal		
Subtotal			
Total (2)			

3. **Agent Performance Report**
    - Object: Service_Request_c
    - Type: Summary Report
    - Group by: Owner (Agent)
    - Show: Count of resolved requests

Impact: Real-time insights into requests, customer churn, and agent efficiency.

---

2. Report Types

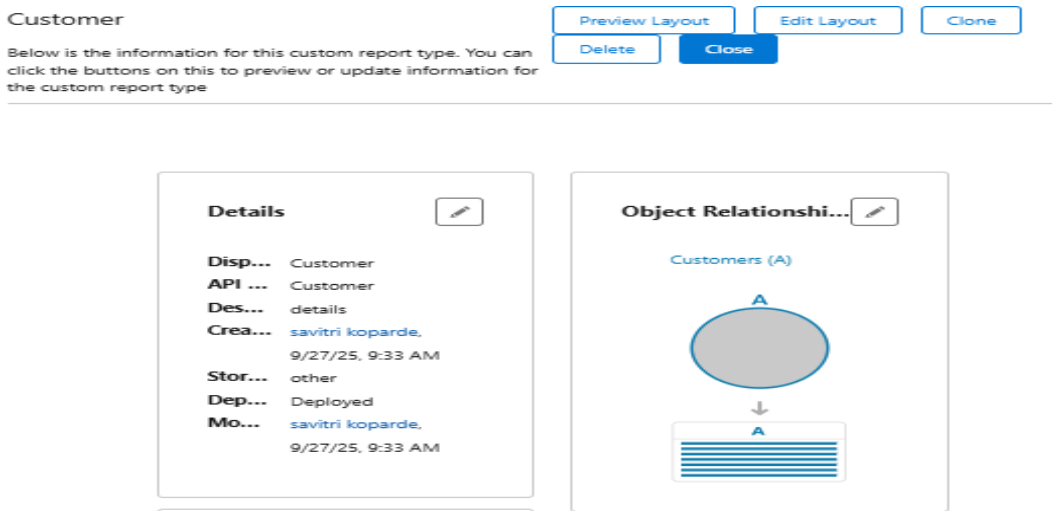
Purpose: Define which objects and fields are available for reporting.

Steps:

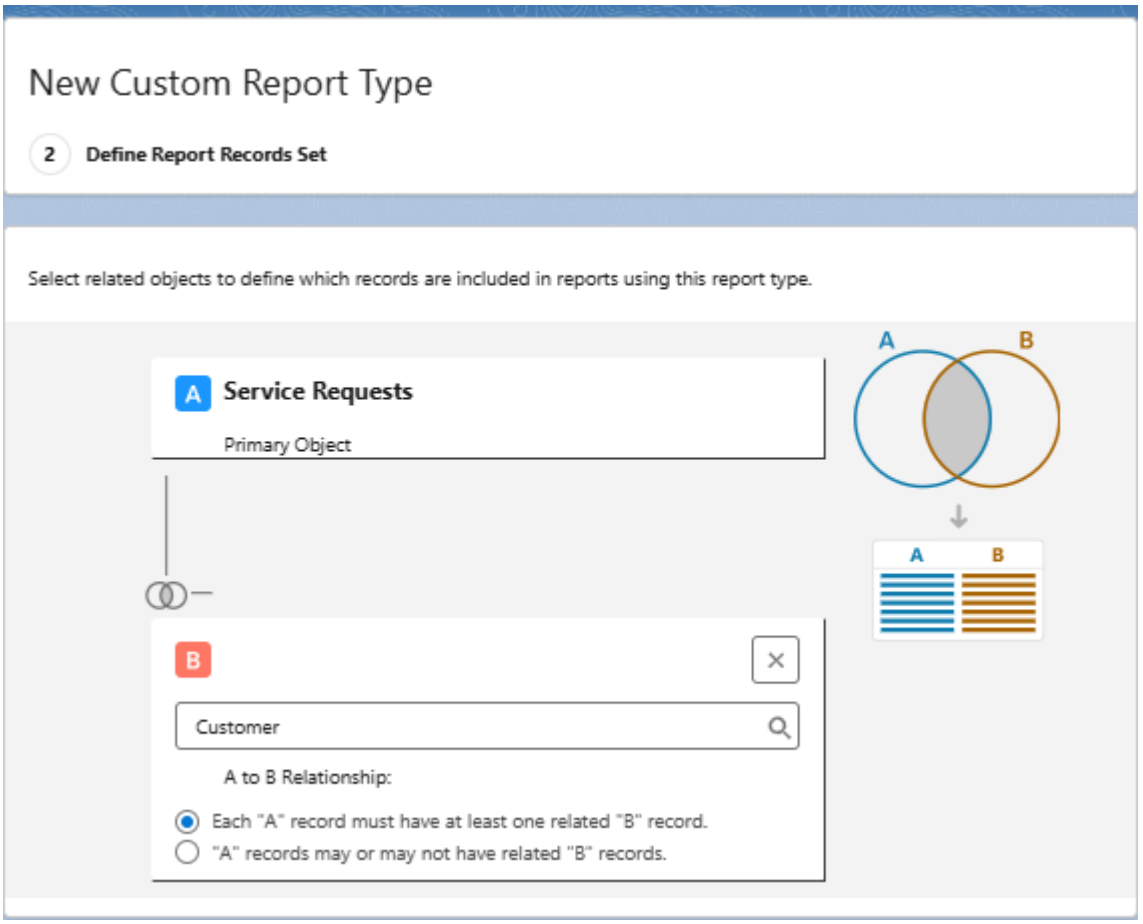
1. Setup → Report Types → New Custom Report Type

2. Primary Object: Service_Request_c or Customer_c
3. Related Object: Optional (Customer_c for Service Requests)
4. Deploy → Make it available for reporting

Impact: Ensures all necessary fields and relationships are reportable.



Service Request Custom Report:



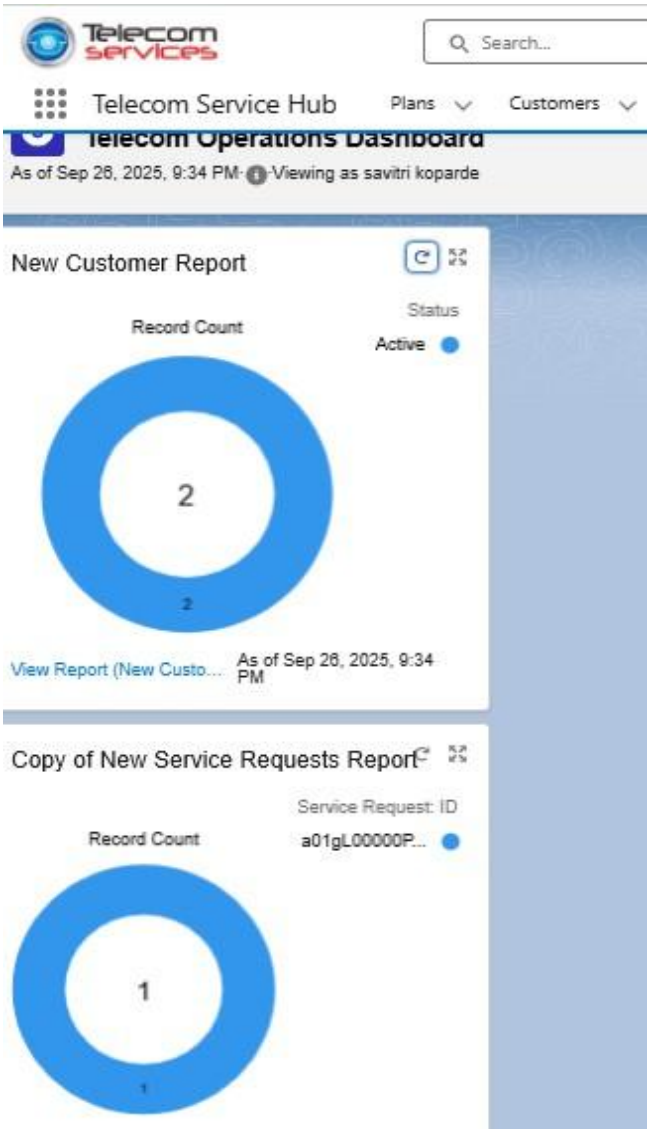
3. Dashboards

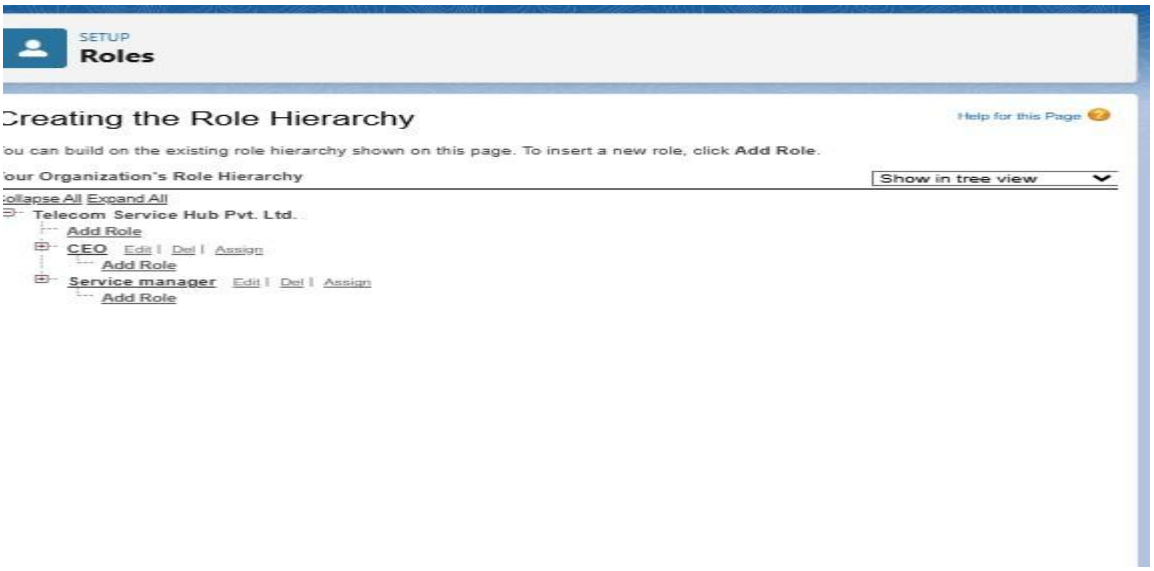
Purpose: Visualize report data for quick insights.

Steps:

1. Setup → Dashboards → New Dashboard
2. Add Components:
    - Pie chart → Service Requests by Type
    - Bar chart → Active vs Cancelled Customers
    - Table → Top Agents by Resolved Requests
3. Assign running user: Manager (so dashboard shows data according to manager access)
4. Save & Activate

Impact: Provides Managers and Agents a quick snapshot of operations.





---

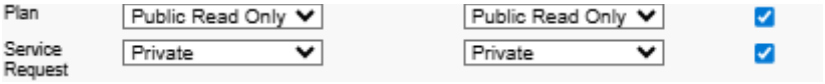
4. Sharing Settings

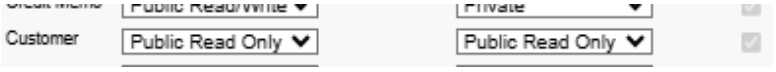
Purpose: Control who sees which records in Salesforce.

Steps (must-have):

1. Setup → Sharing Settings
2. Ensure Customer_c and Plan_c → Public Read Only
3. Service_Request_c → Private
4. Role hierarchy ensures Managers see all requests, Agents see only their own

Impact: Protects sensitive customer data while enabling operational visibility. Sharing setting:





Role hierarchy:

5. Field Level Security (FLS)

Purpose: Restrict access to specific fields on objects.

Steps (must-have):

1. Setup → Object Manager → Object → Fields & Relationships → Set Field-Level Security
2. Example: Hide Email from Agents if required
3. Ensure Managers have full visibility

Impact: Maintains confidentiality and enforces data access policies.