GARAGE MANAGEMENT SYSTEMS

BACHELOR OF TECHNOLOGY IN INFORMATION TECHNOLOGY

Team size: 4

A HARIHARAN (L).	**Team Leader**
S.SANJEEVI.	**Team Member**
M SACHIN TENDULKAR	**Team Member**
M.BALAMURUGAN.	**Team Member**

Under the Esteemed Guidance of
Mrs.Sweetha.
Assistant Professor of Computer Science Engineering

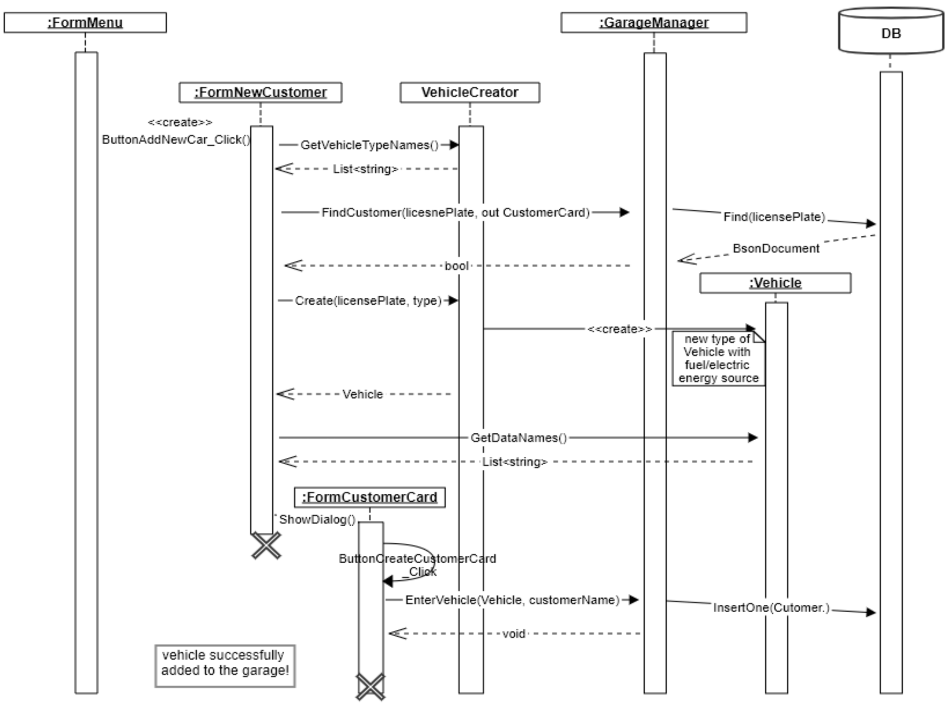
Department of Information Technology

Oxford Engineering College

1.INTRODUCTION

1.1 Project Overview:

The Garage Management System in Salesforce is a cloud-based solution to manage garage operations efficiently. It handles customer data, vehicle records, service bookings, inventory, and billing in one platform. The system automates workflows and improves service delivery. Using Salesforce ensures scalability, real-time tracking, and better customer engagement. It is designed for garages seeking streamlined, data-driven operations.



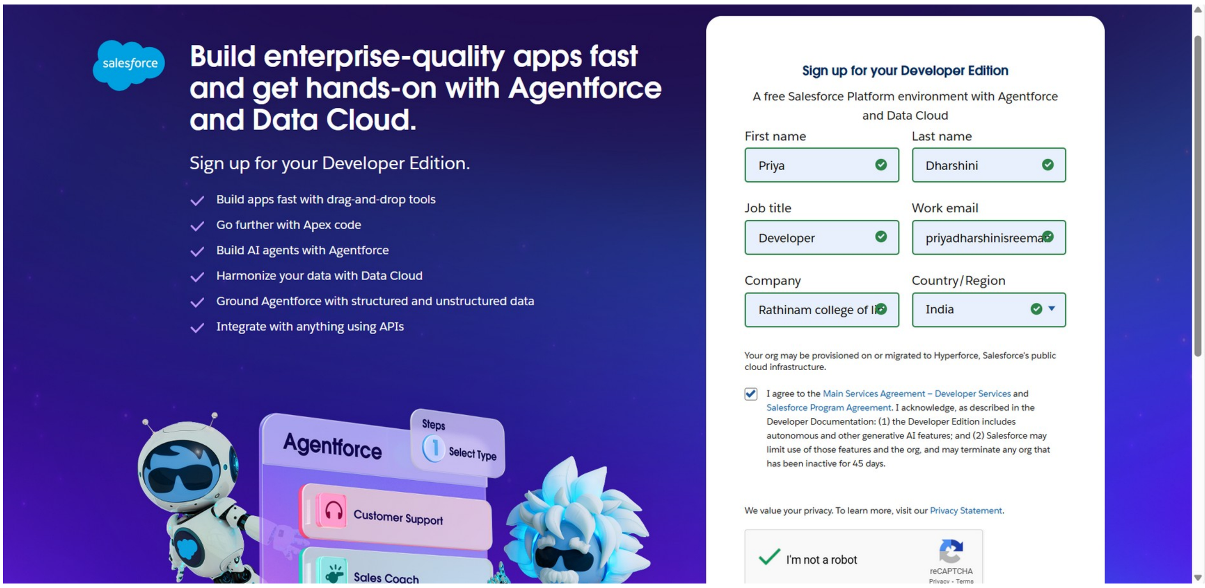
1.2 Purpose :

The **purpose of garage management** is to ensure the **efficient, safe, and cost-effective operation** of a garage—whether it's an automotive repair shop, service center, or fleet maintenance facility. It involves organizing and overseeing all aspects of the garage's operations to deliver high-quality service and maintain customer satisfaction.

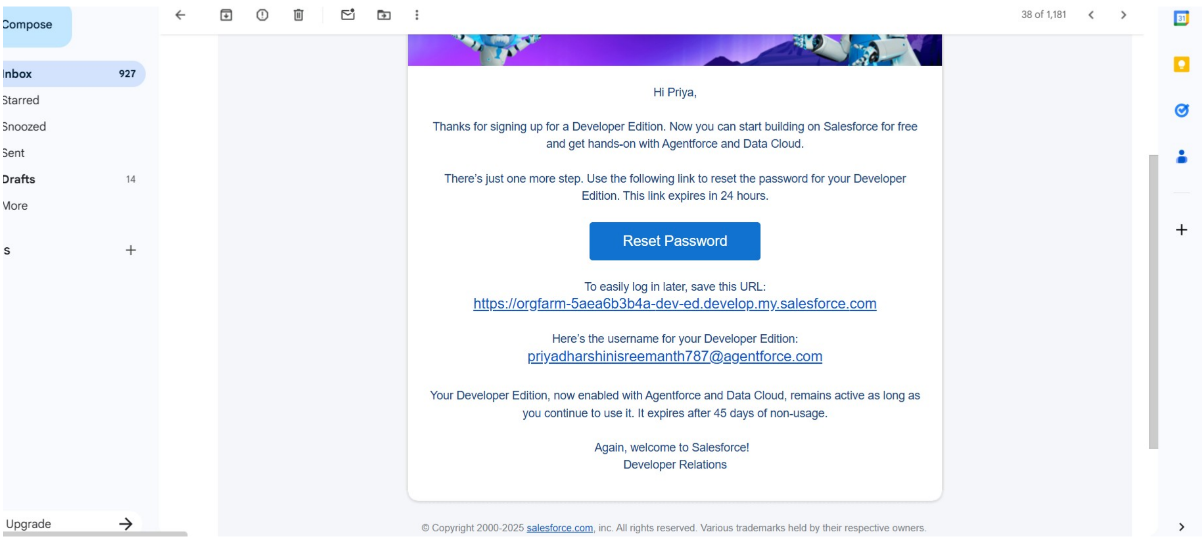
Milestone 1: Salesforce

i)Creating Developer Account:

Using this URL-
https://developer.salesforce.com/signup

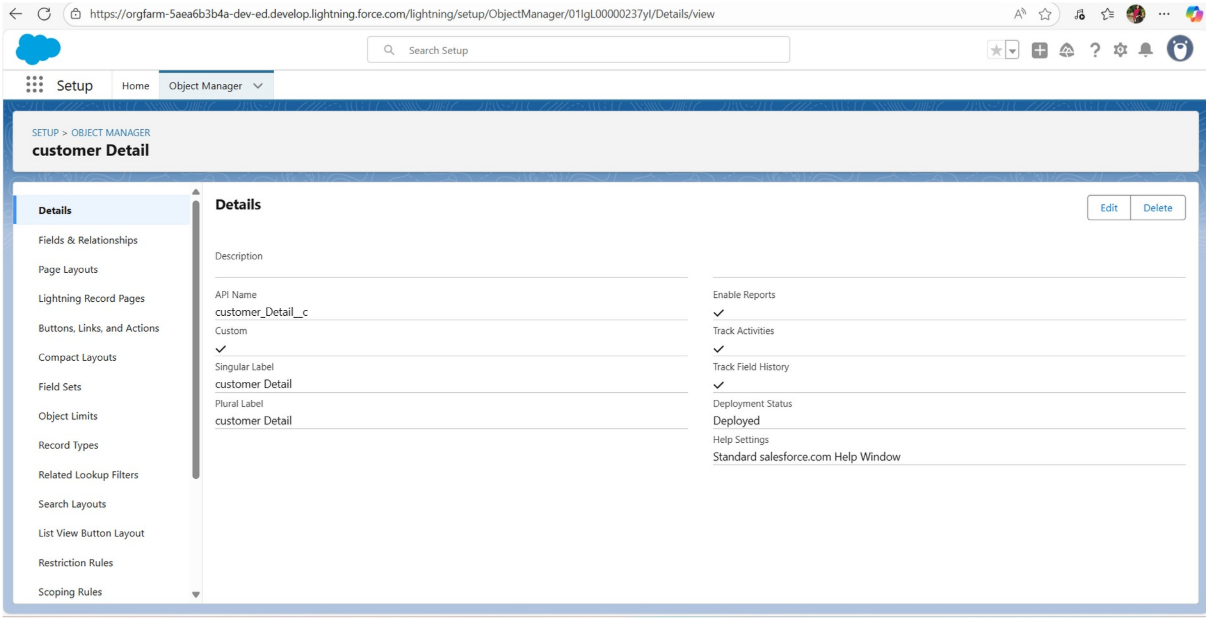


ii) Account Activation

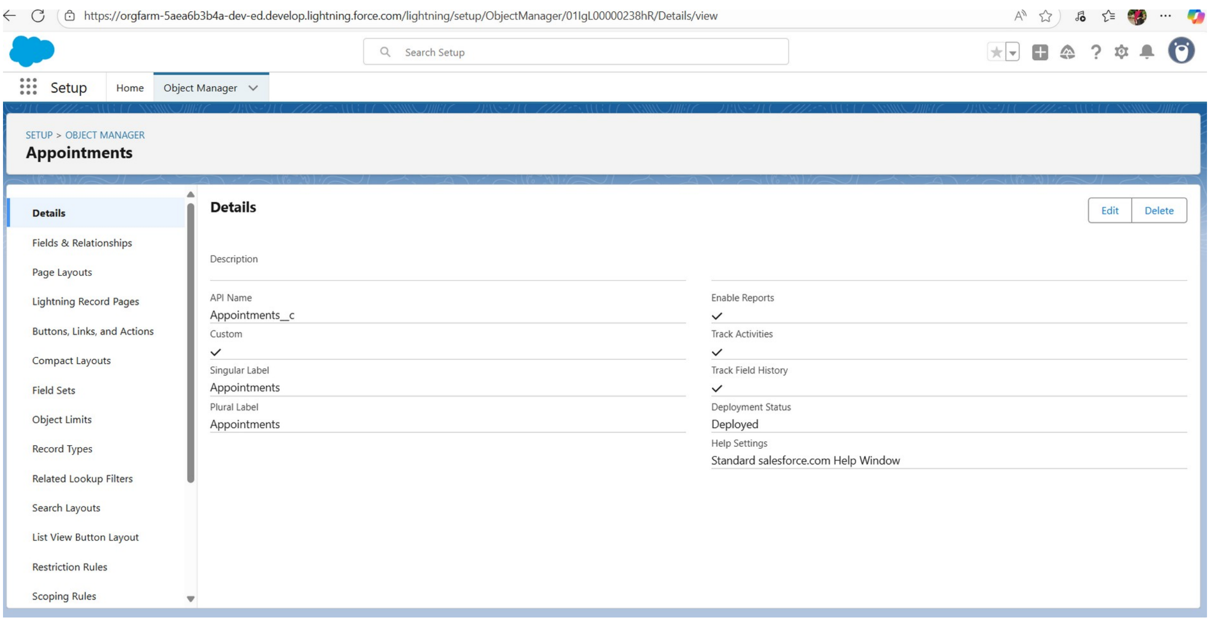


Milestone 2: Object

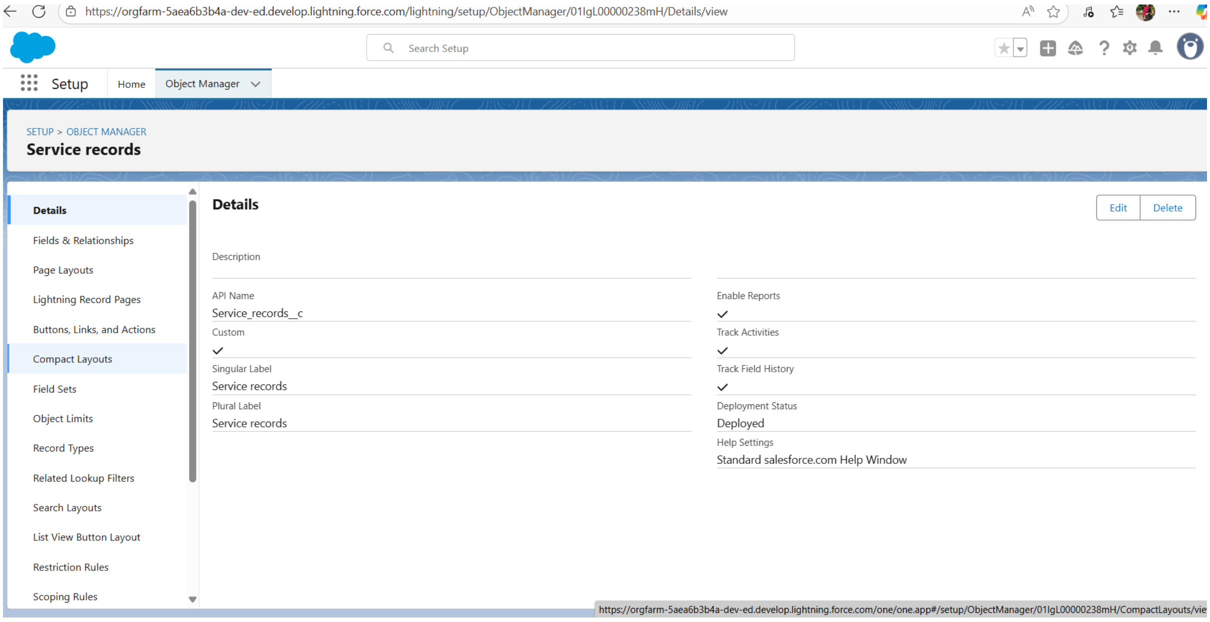
i)Creating customer detail:



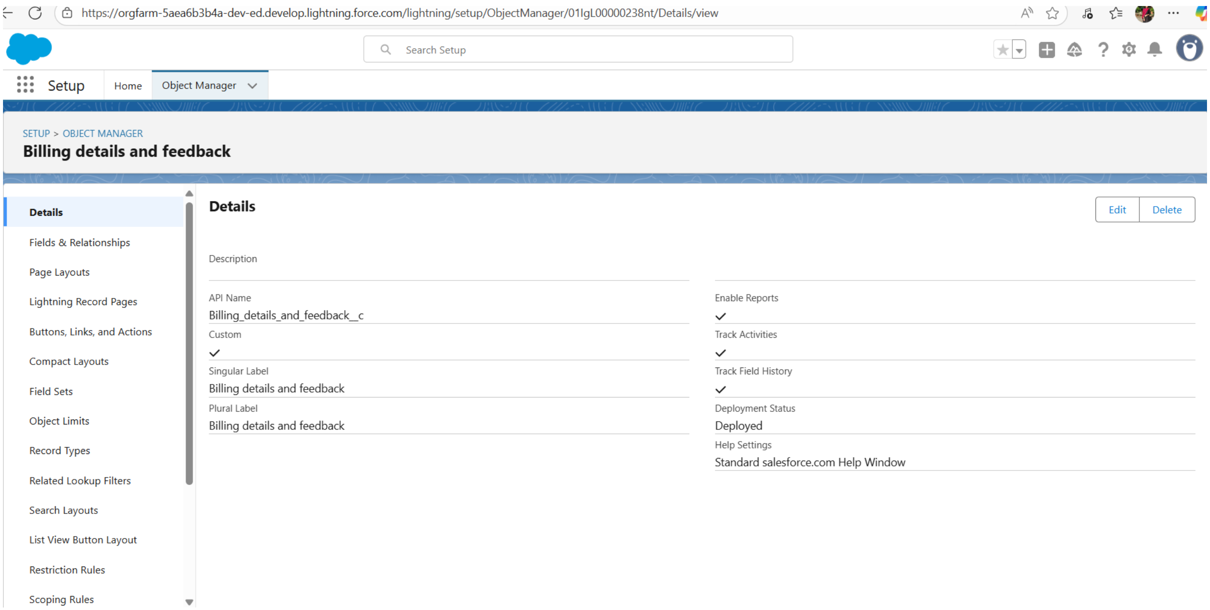
ii)Creating Appointments:



iii)Creating service records:

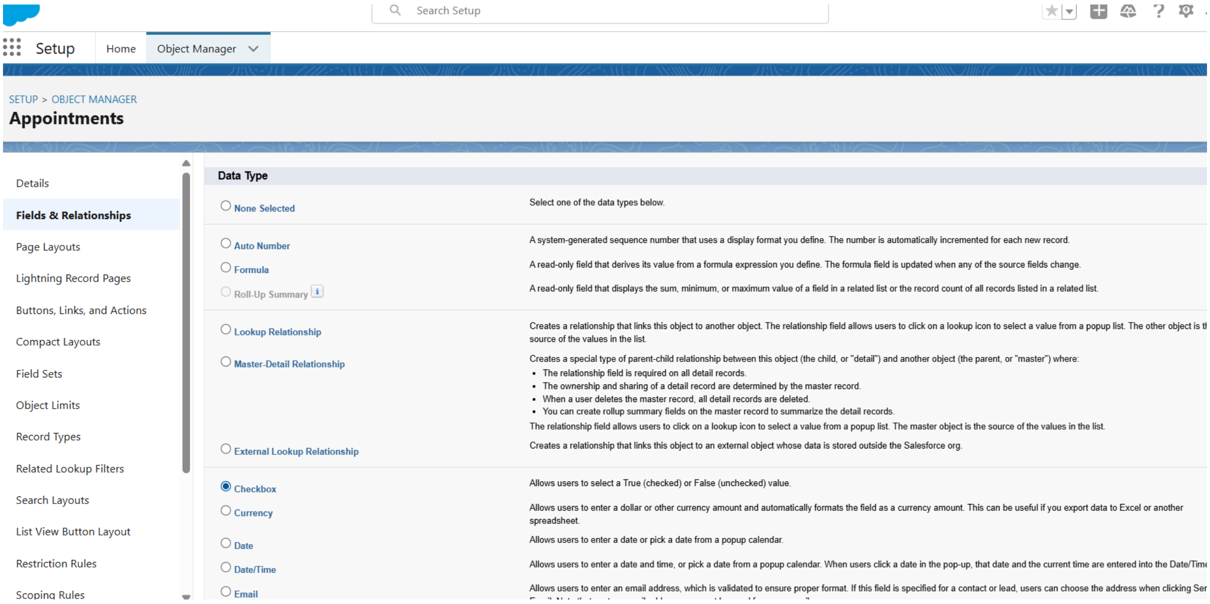


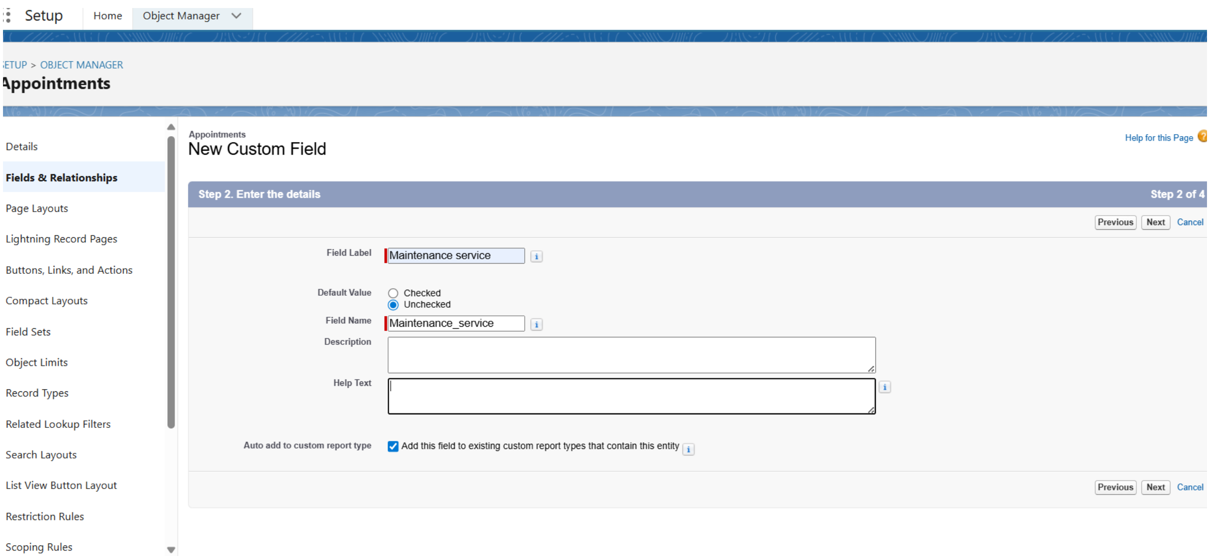
iv) Creating Billing details and feedback:



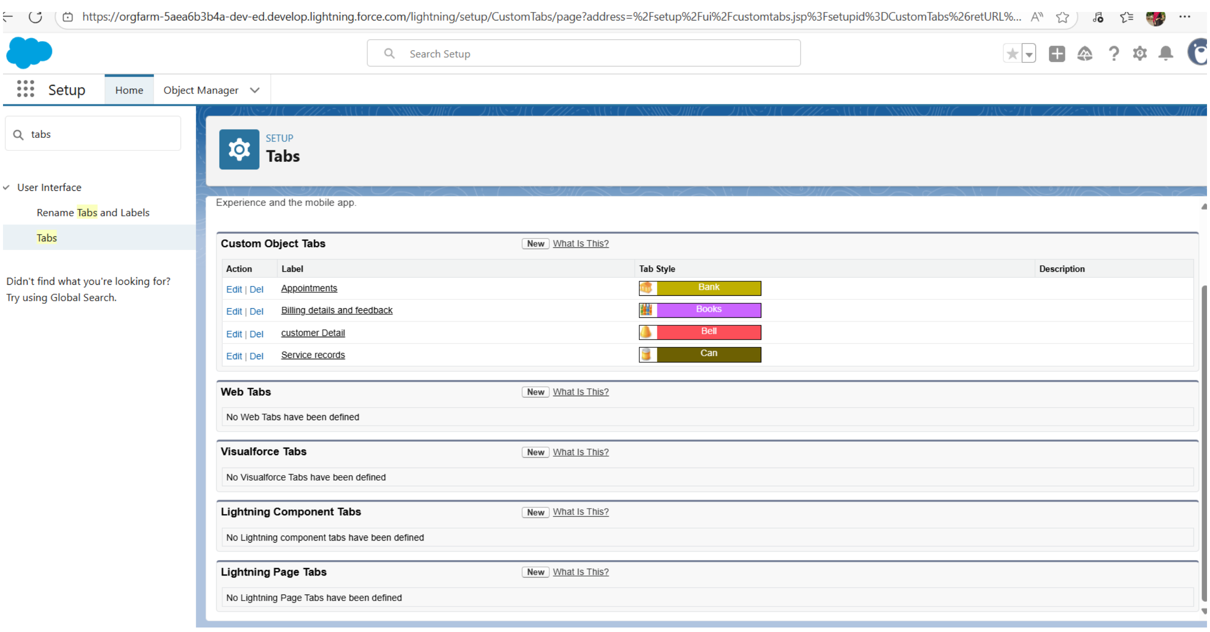
Milestone 3: Tabs

i)Creating custom tabs:



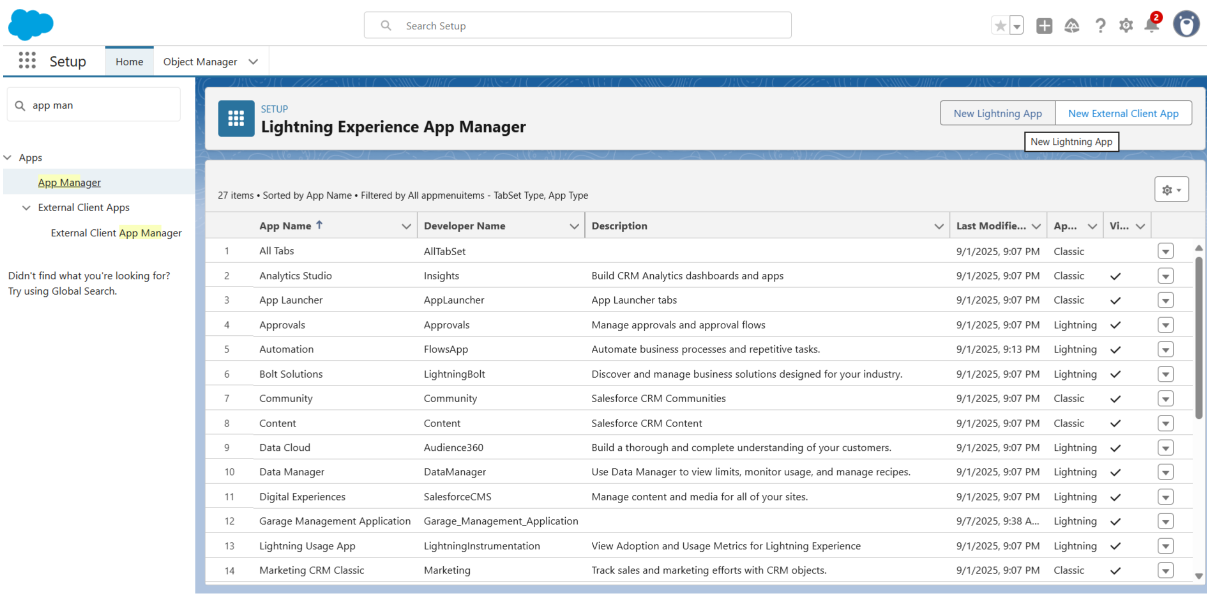


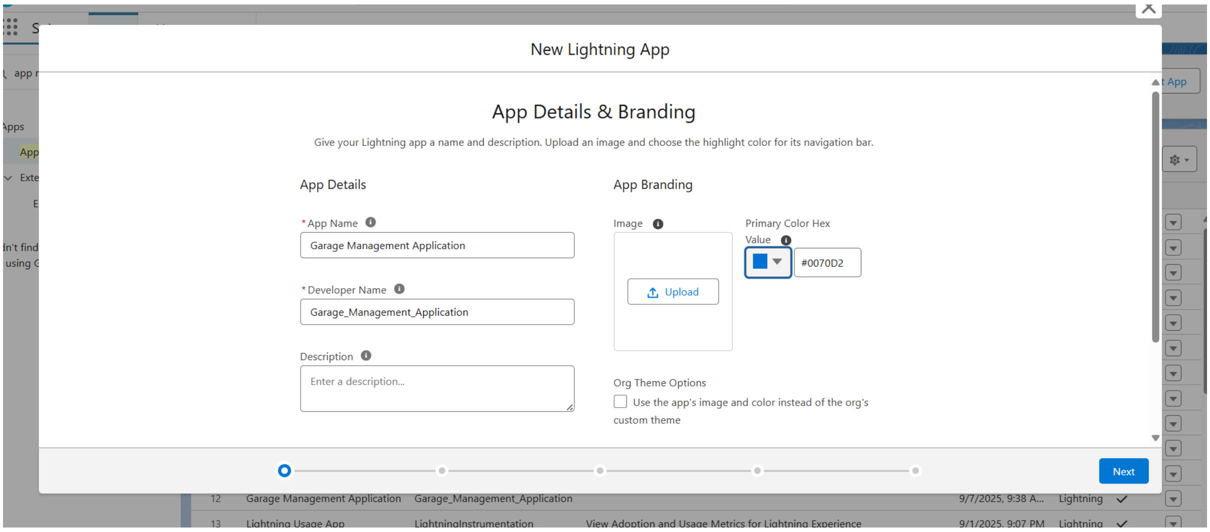
ii)Creating Remaining Tabs

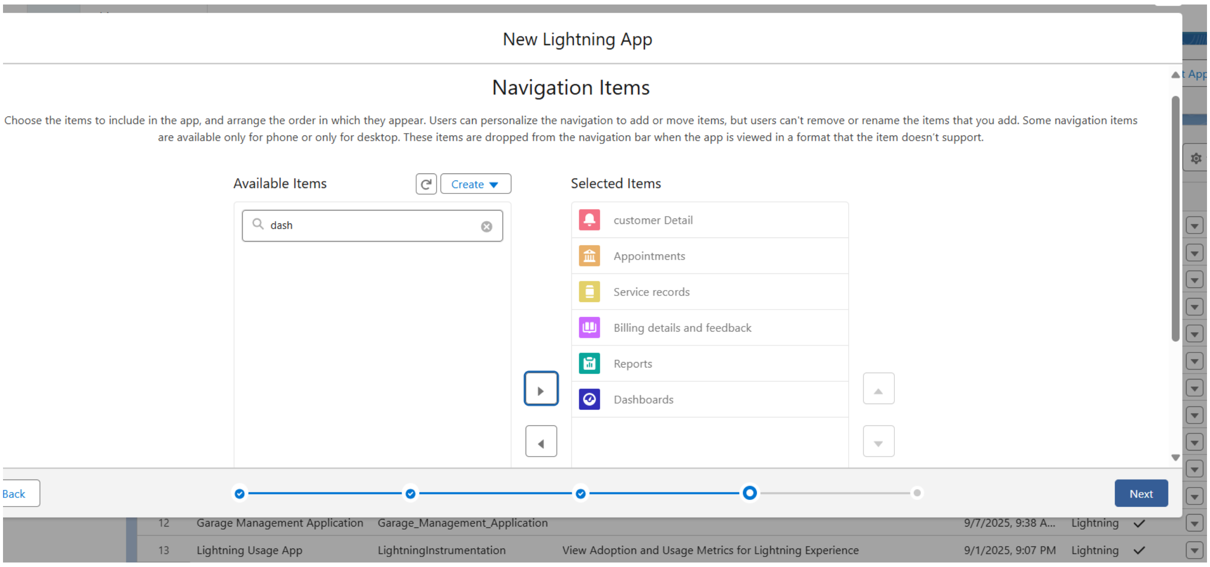


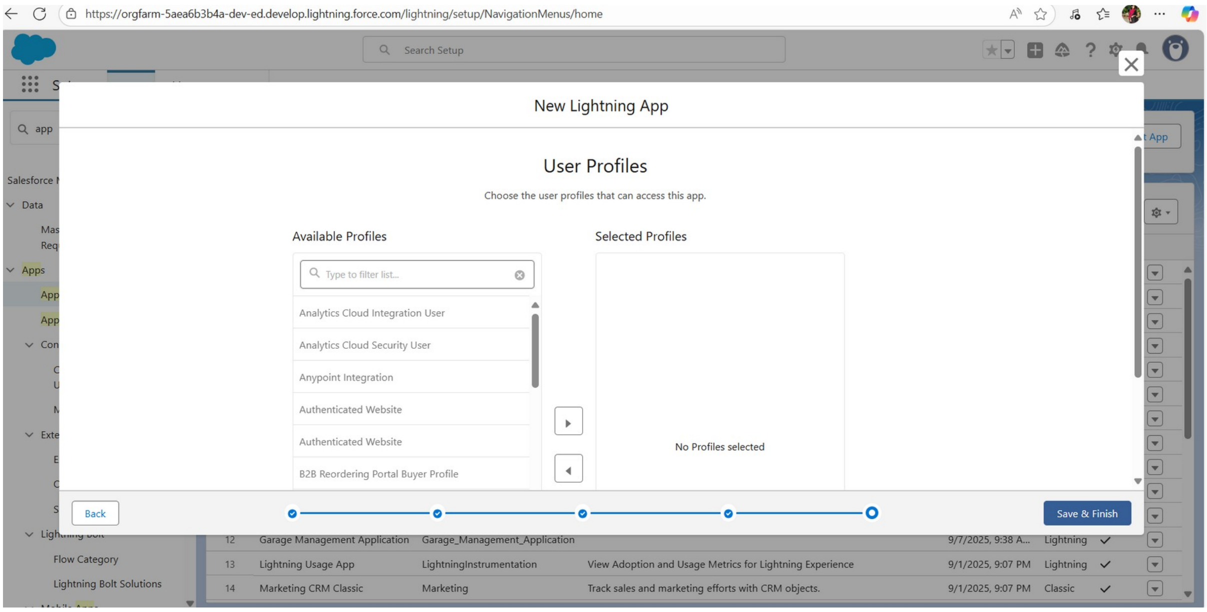
Milestone 4: The lightning app

i)Creating lightning app



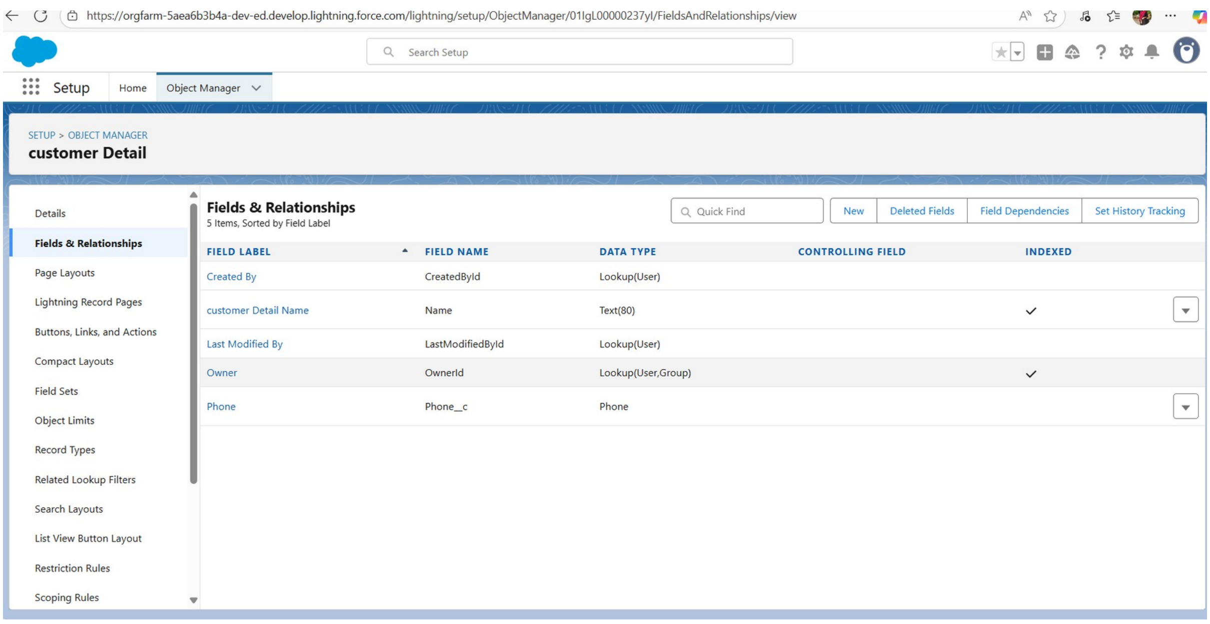

New Lightning App
App Details & Branding
Give your Lightning app a name and description. Upload an image and choose the highlight color for its navigation bar.
App Details
*App Name
Garage Management Application
*Developer Name
Garage_Management_Application
Description
Enter a description...
App Branding
Image
Upload
Primary Color Hex Value
#0070D2
Org Theme Options
Use the app's image and color instead of the org's custom theme
Next


New Lightning App
Navigation Items
Choose the items to include in the app, and arrange the order in which they appear. Users can personalize the navigation to add or move items, but users can't remove or rename the items that you add. Some navigation items are available only for phone or only for desktop. These items are dropped from the navigation bar when the app is viewed in a format that the item doesn't support.
Available Items
Create
dash
Selected Items
customer Detail
Appointments
Service records
Billing details and feedback
Reports
Dashboards
Back
Next

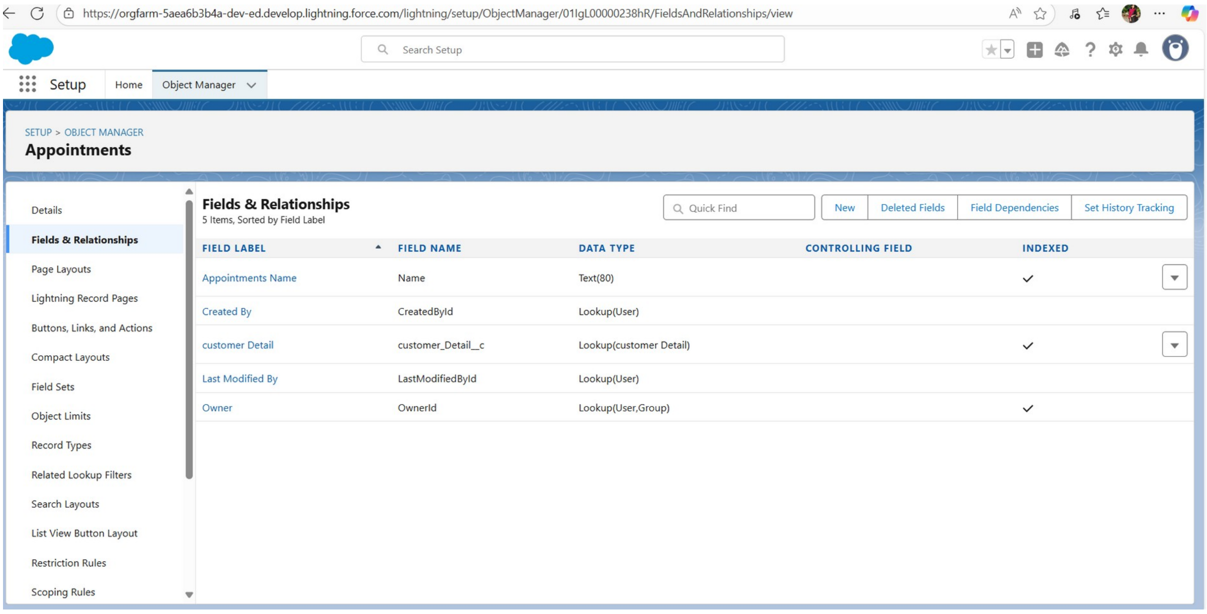


Milestone 5: Fields

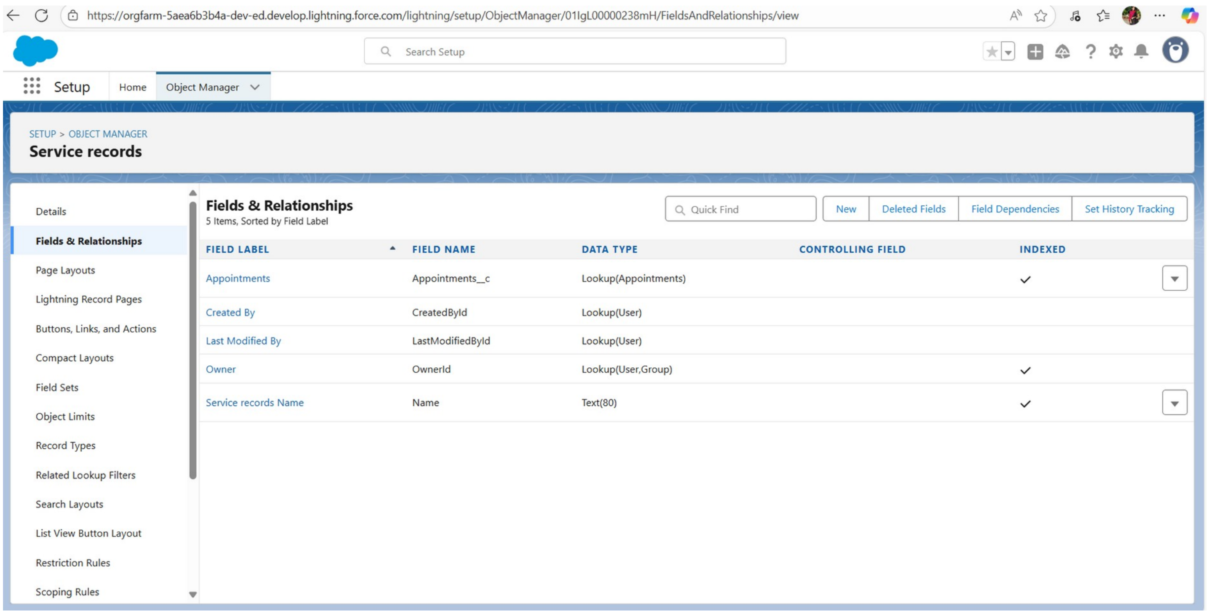
i)Creation of fields for the Customer Details object



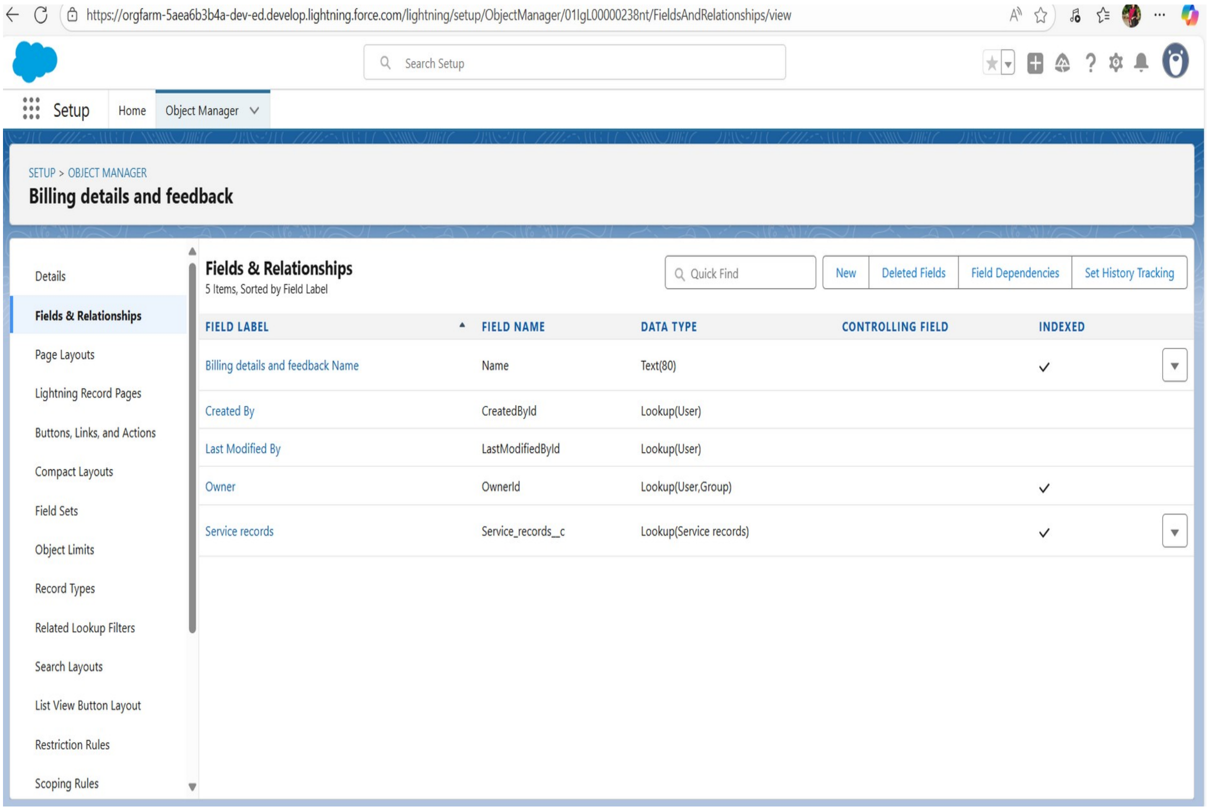
ii) Creation of Lookup Fields



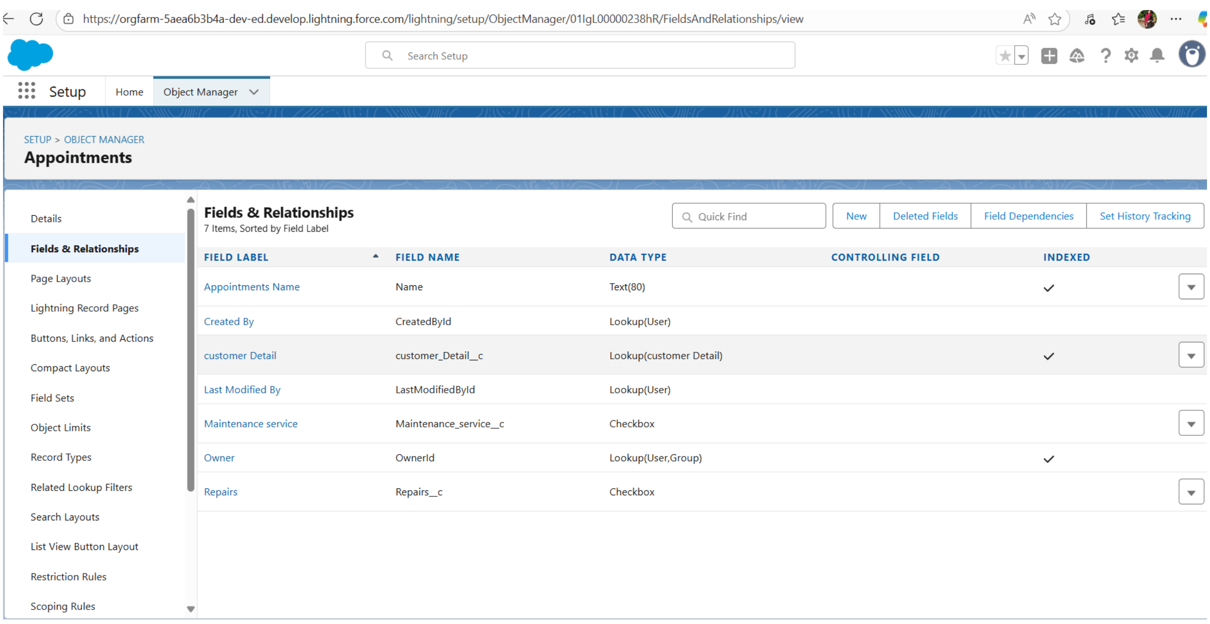
iii)Creation of Checkbox Fields



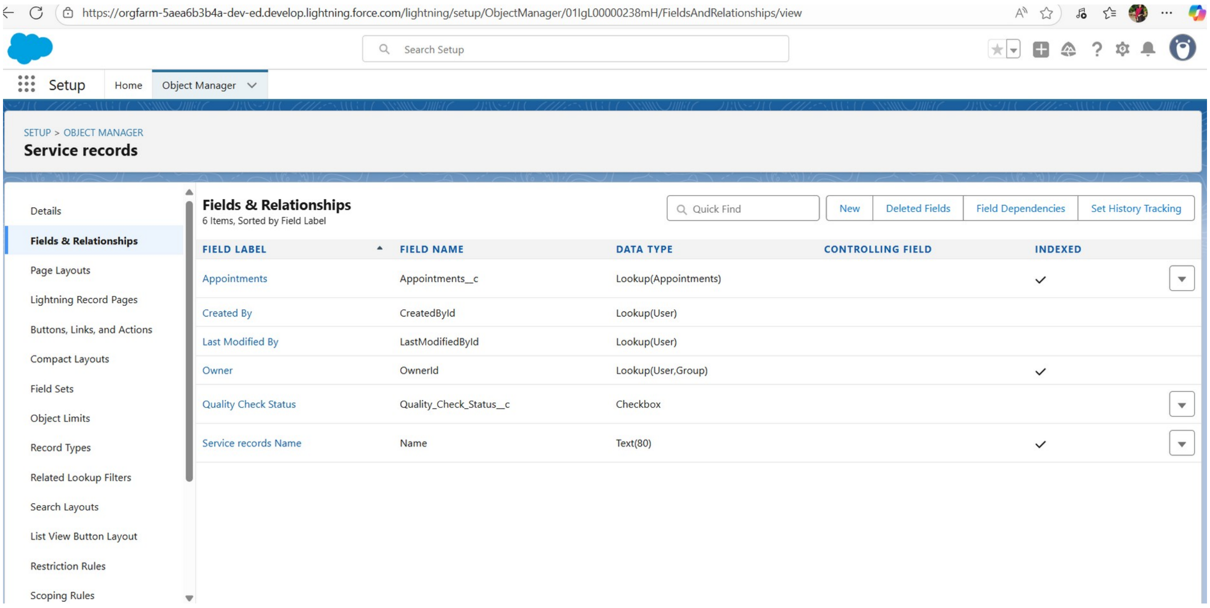
iv)Creation of date Fields



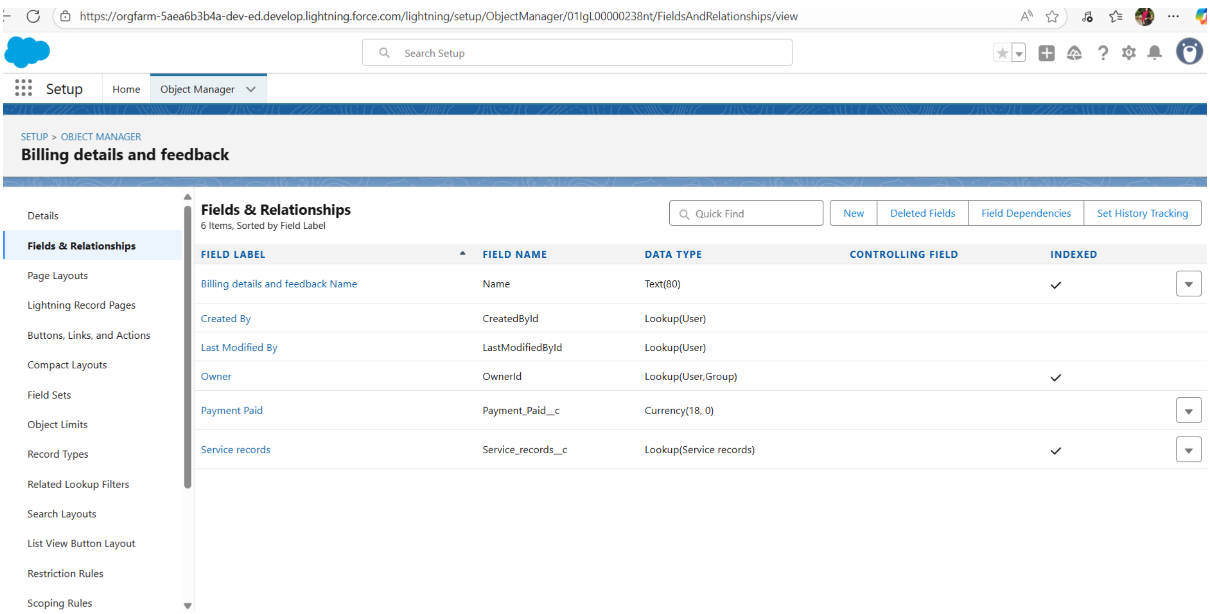
v)Creation of Currency Fields



vi)Creation of Text Fields

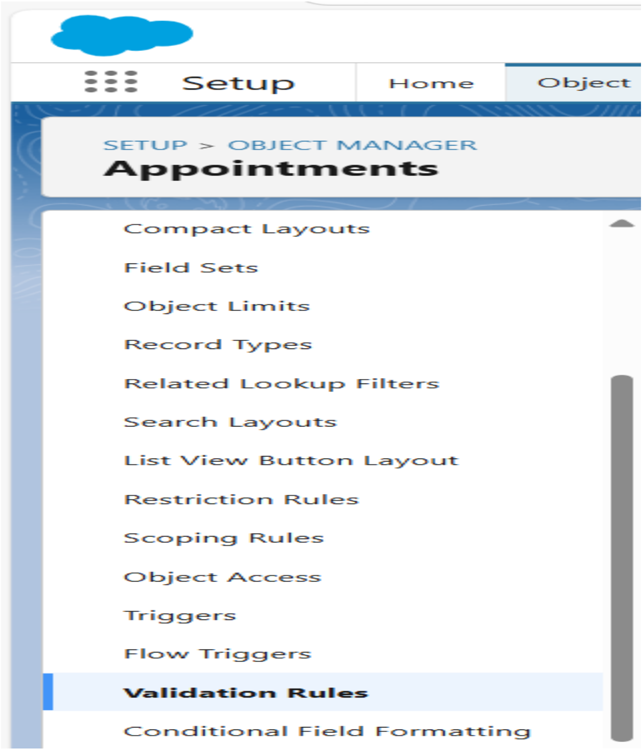


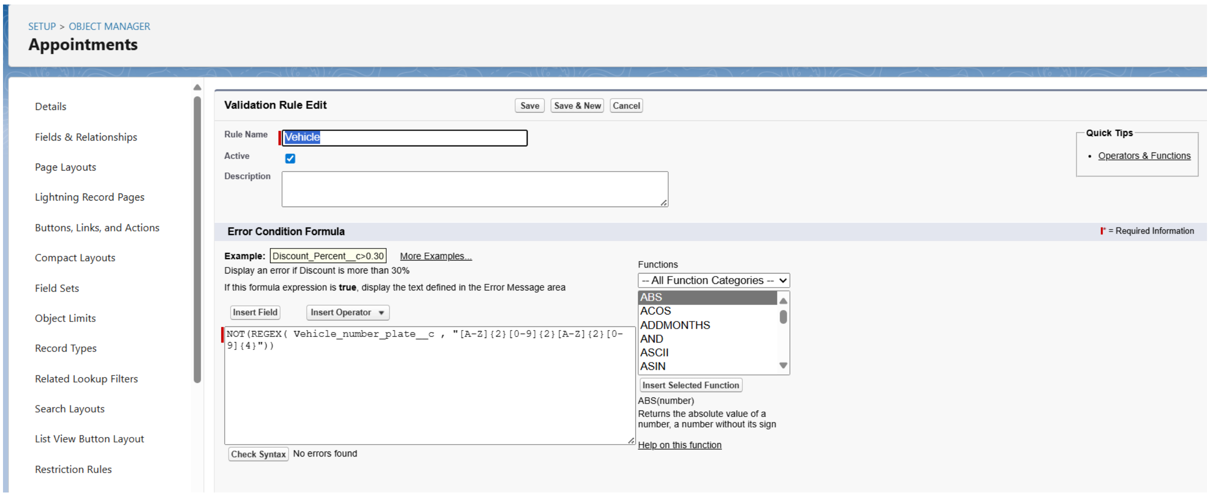
vii)Creating Formula Field in Service records Object

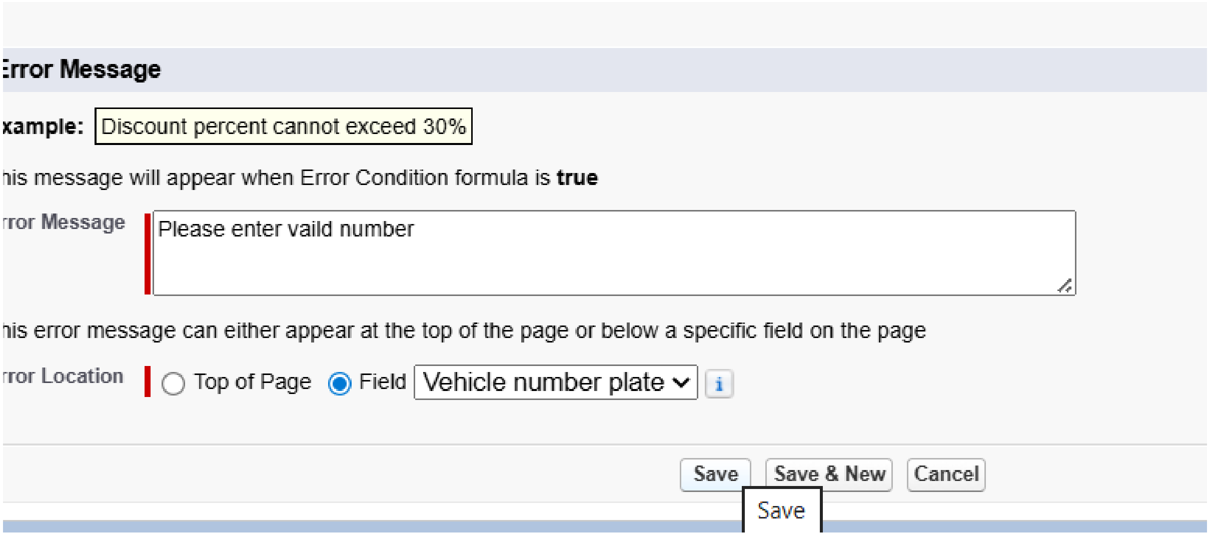


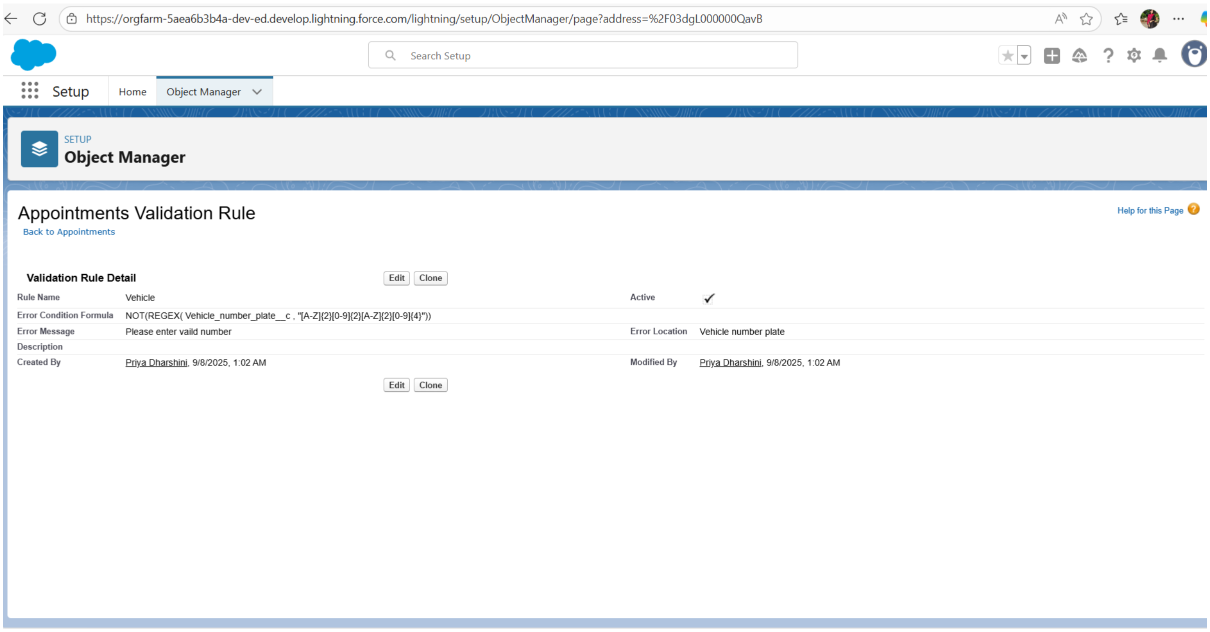
Milestone 6: Validation rule

i)To create a validation rule to an Appointment Object

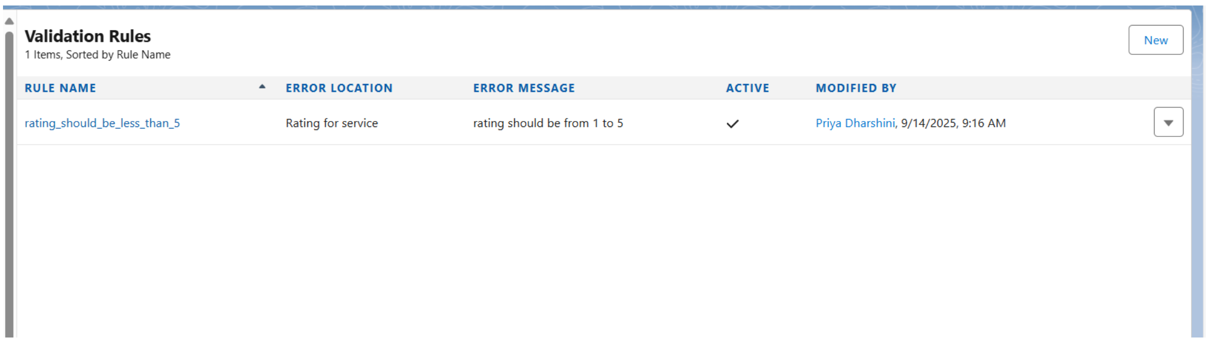






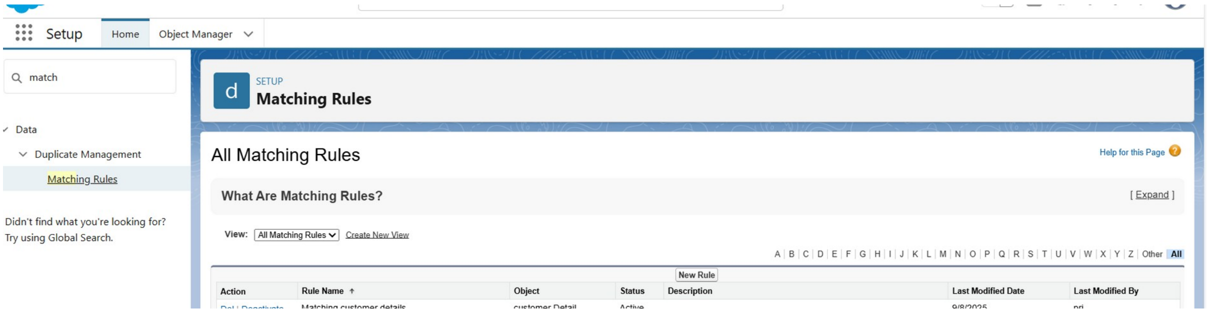


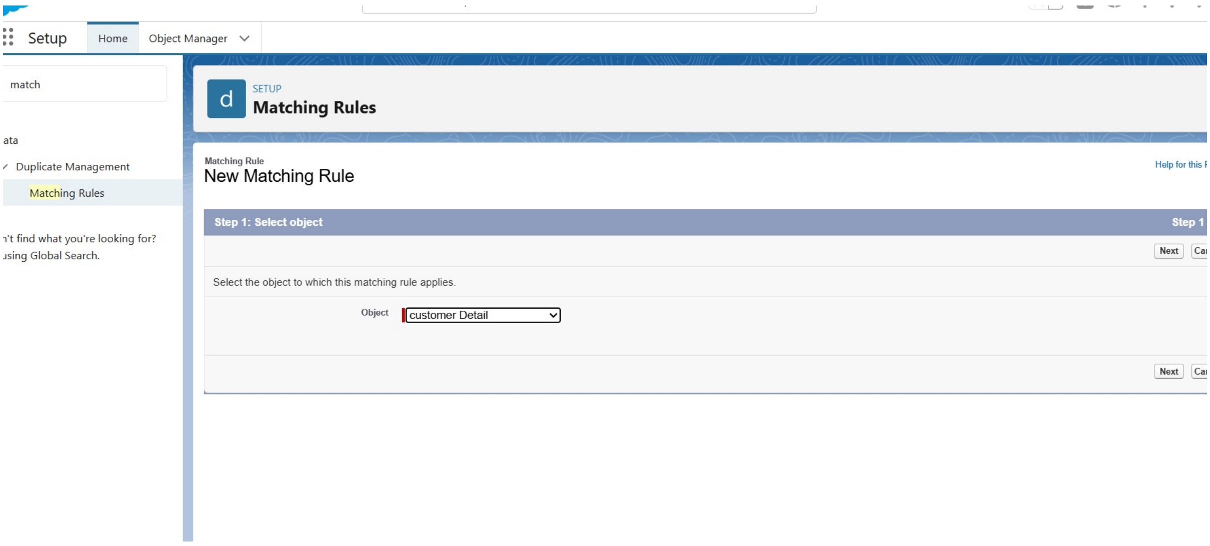
ii)To create a validation rule to an Billing details and feedback Object

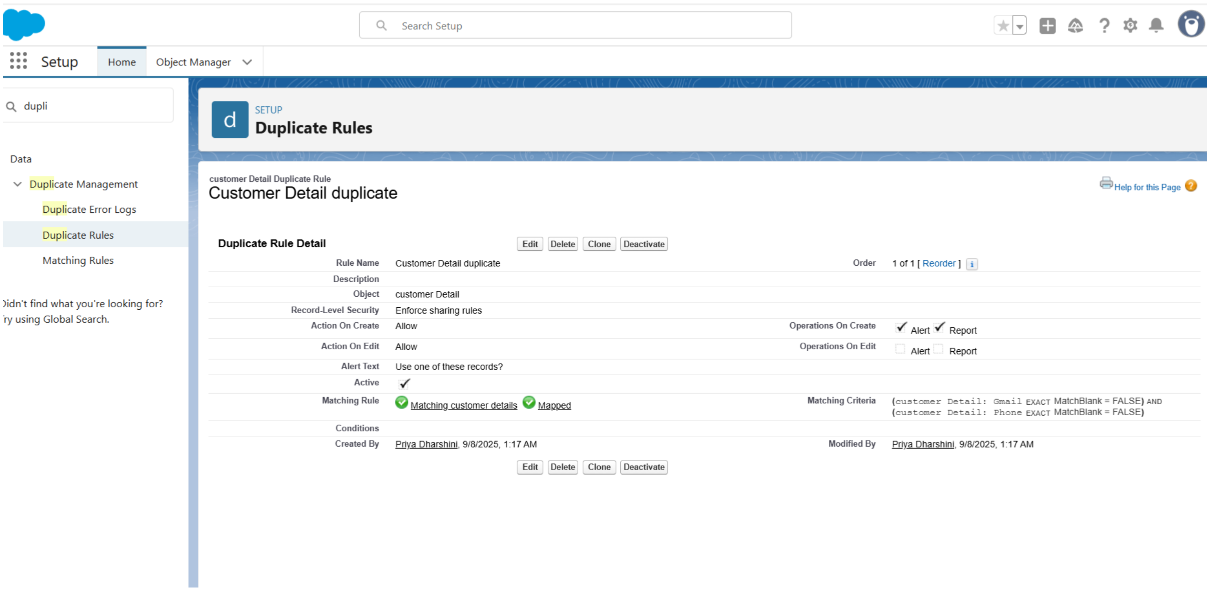


Milestone 7: Duplicate rule

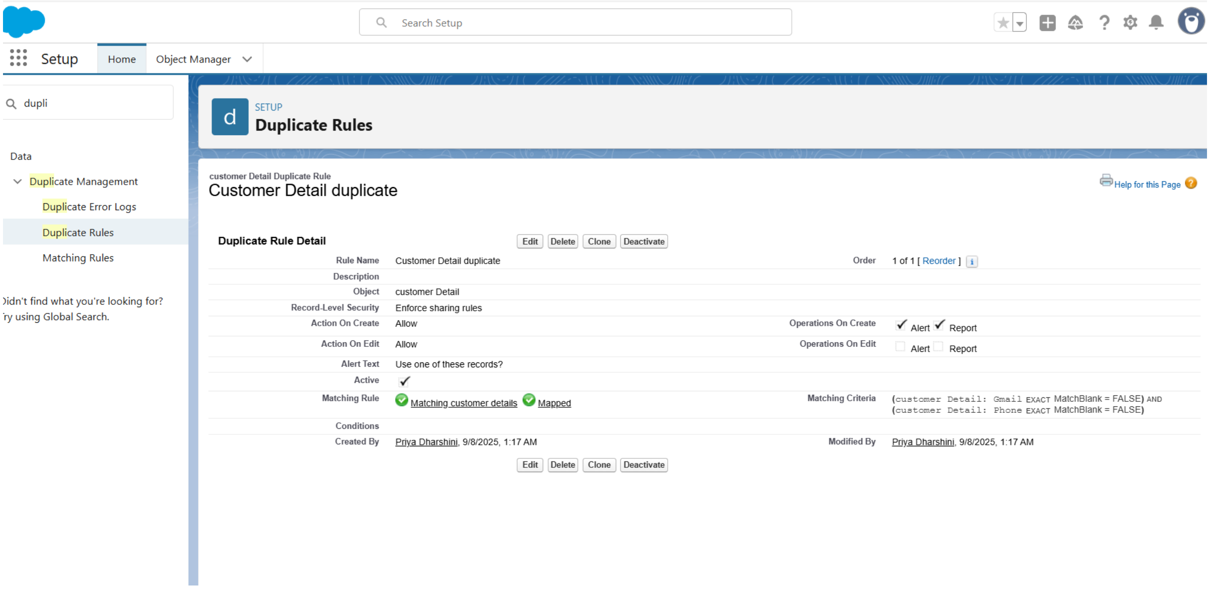
i)To create a matching rule to an Customer details Object

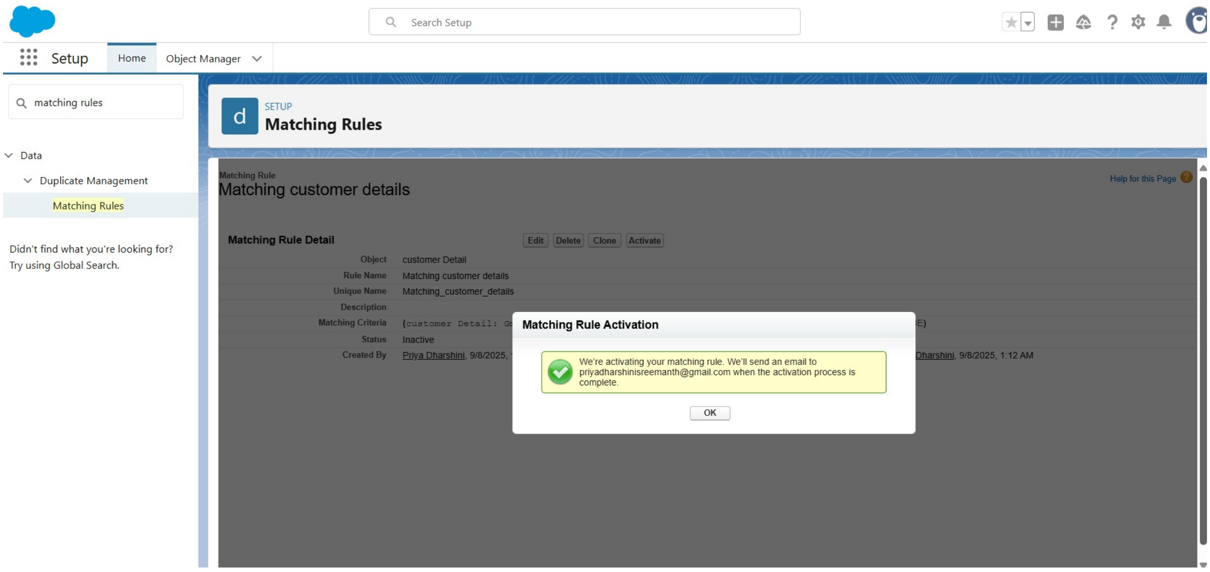






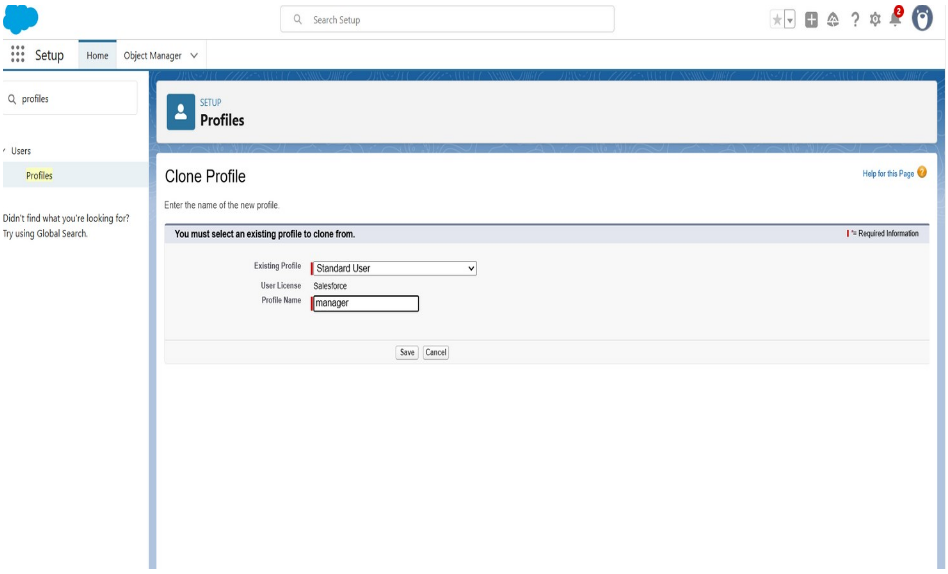
ii)To create a Duplicate rule to an Customer details Object

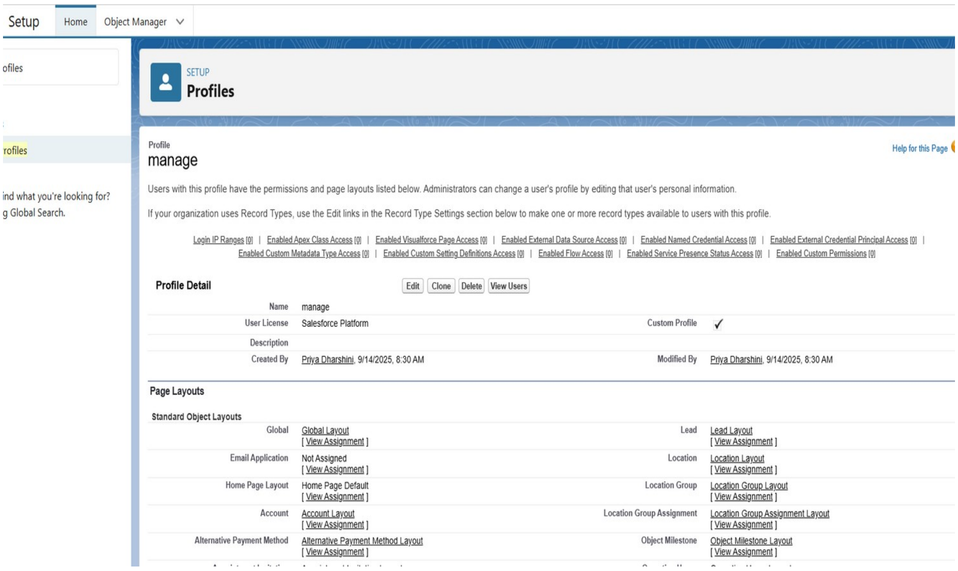


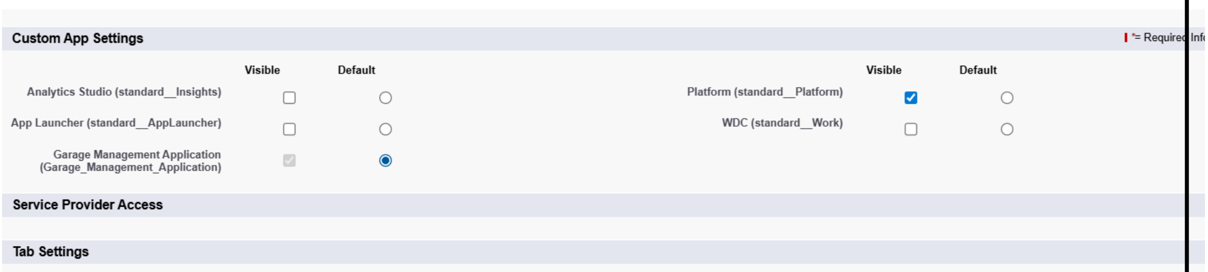


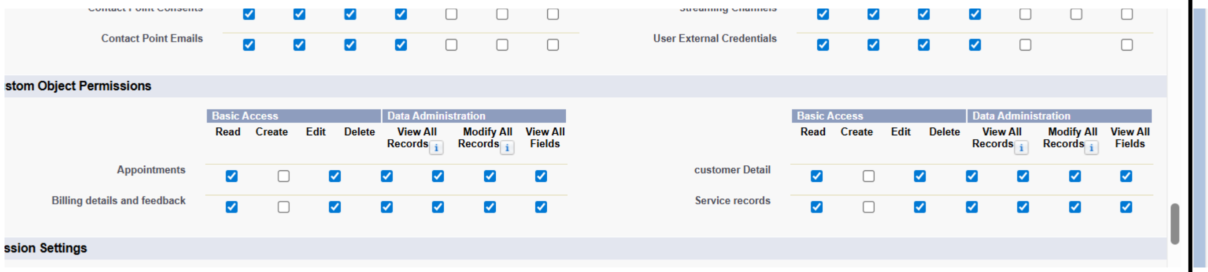
Milestone 8: Profiles

i) Manager Profile

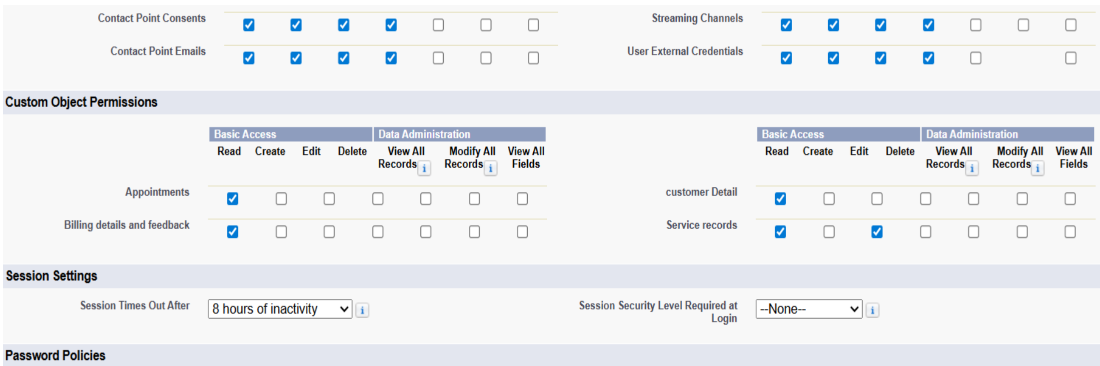


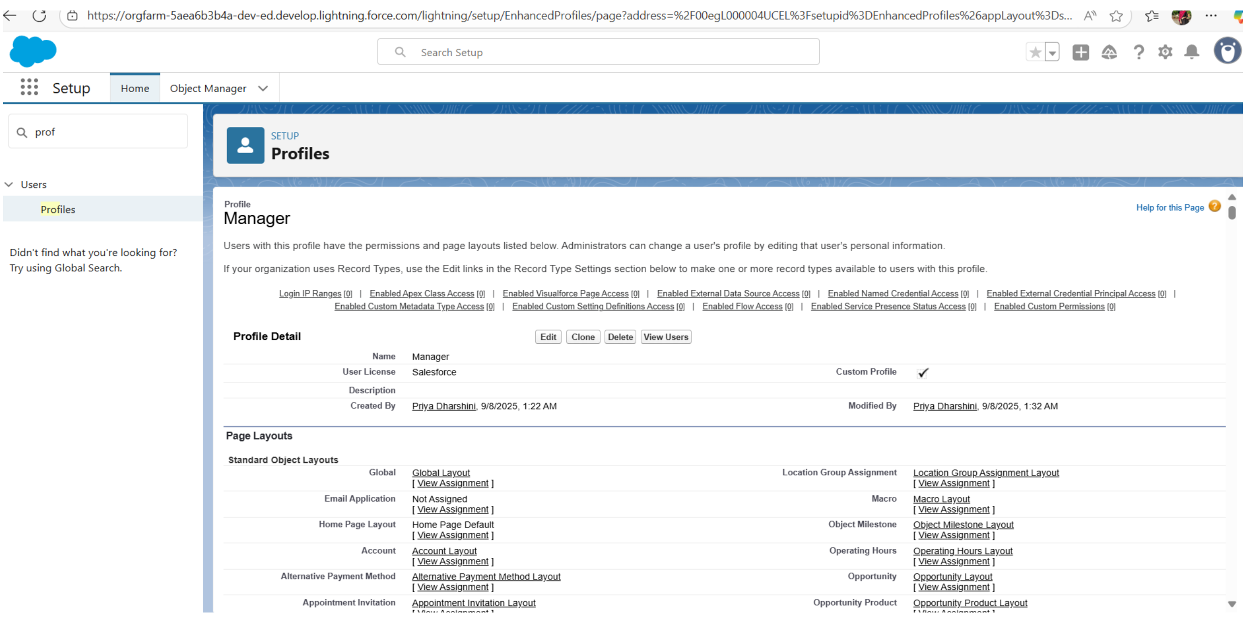






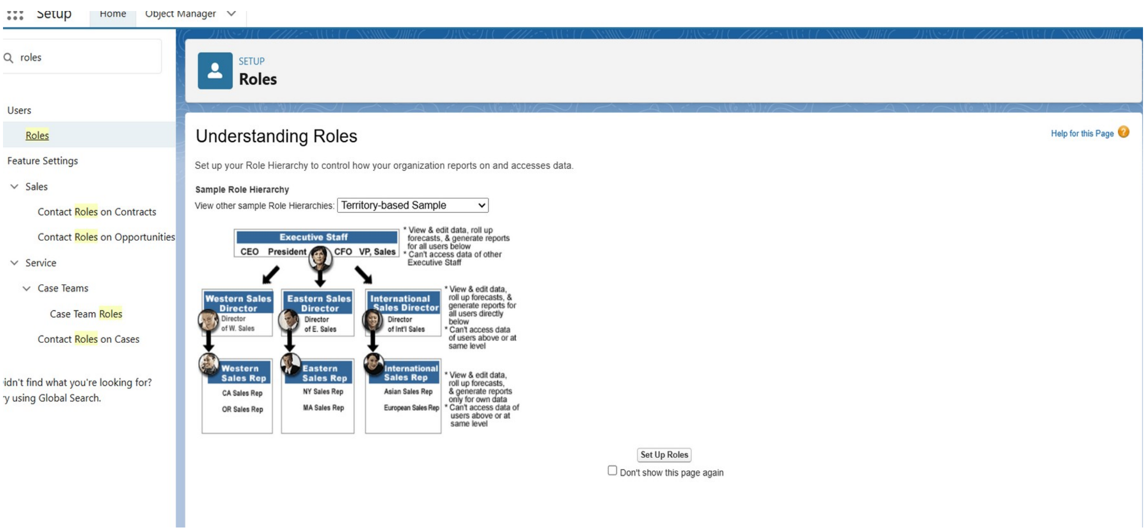
ii) sales person Profile

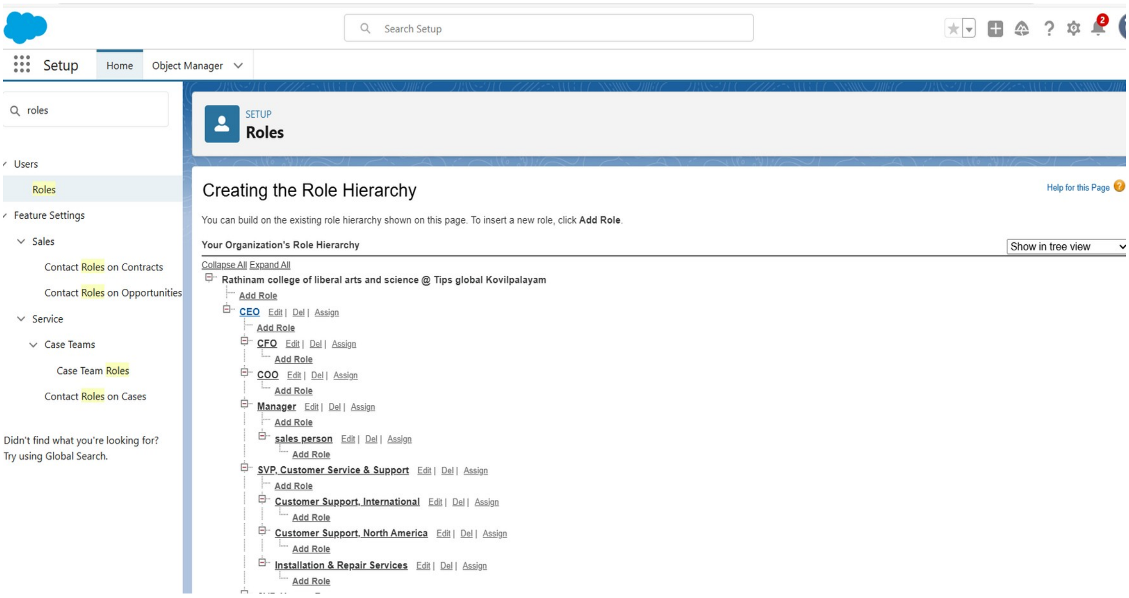


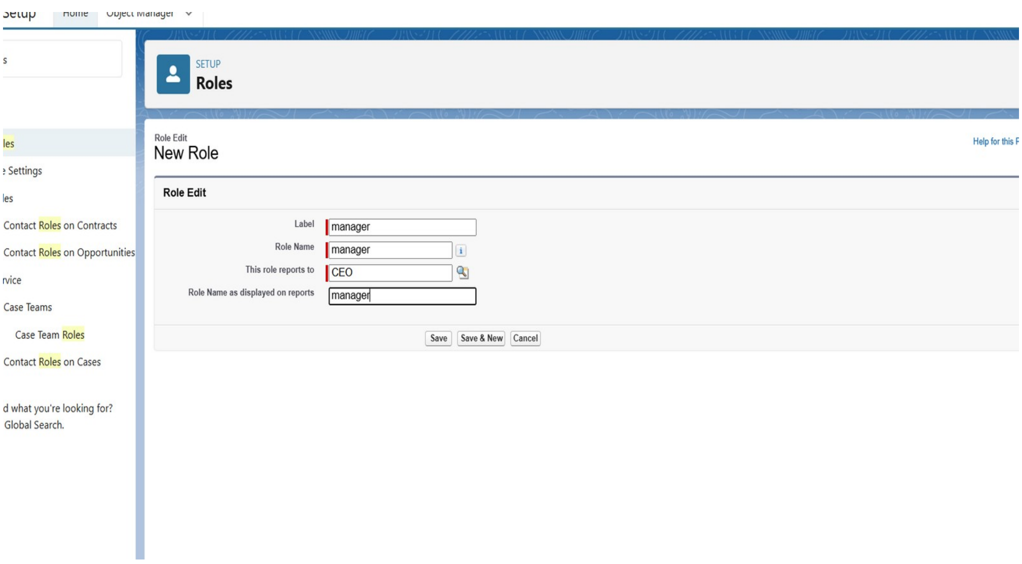


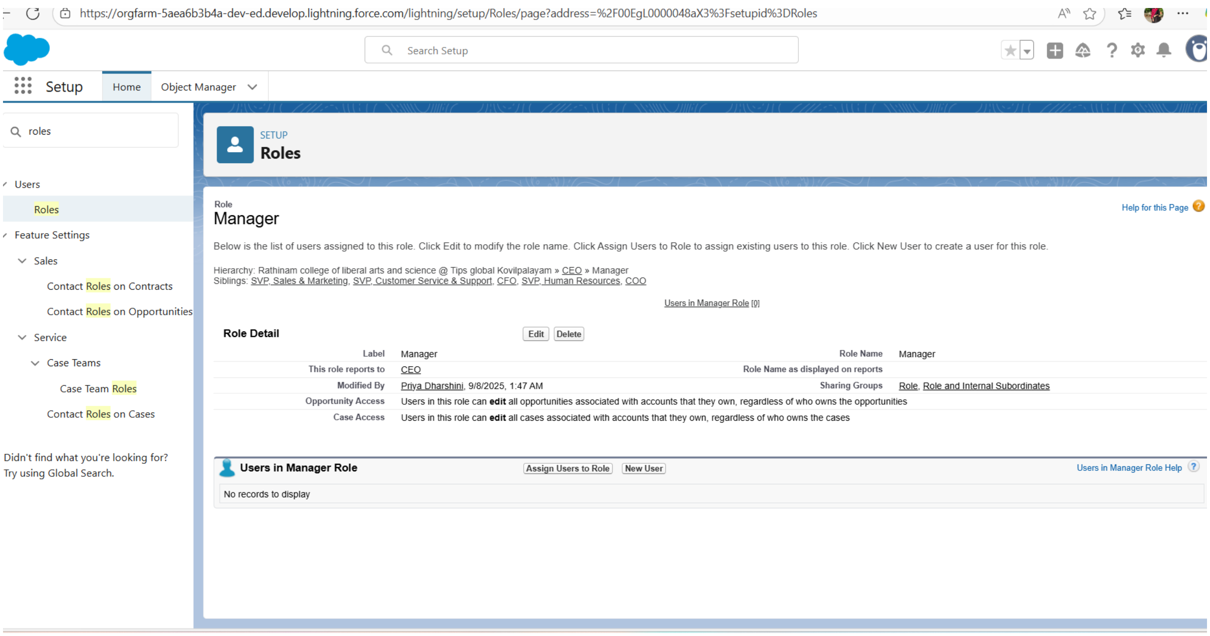
Milestone 9: Role & Role Hierarchy

i) Creating Manager Role

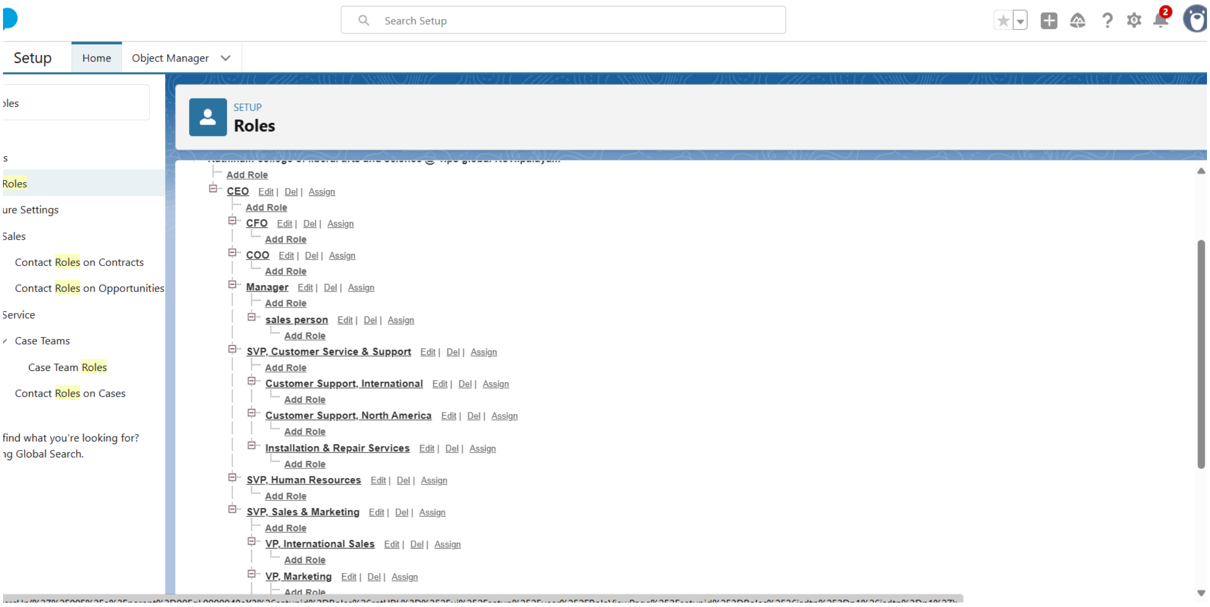


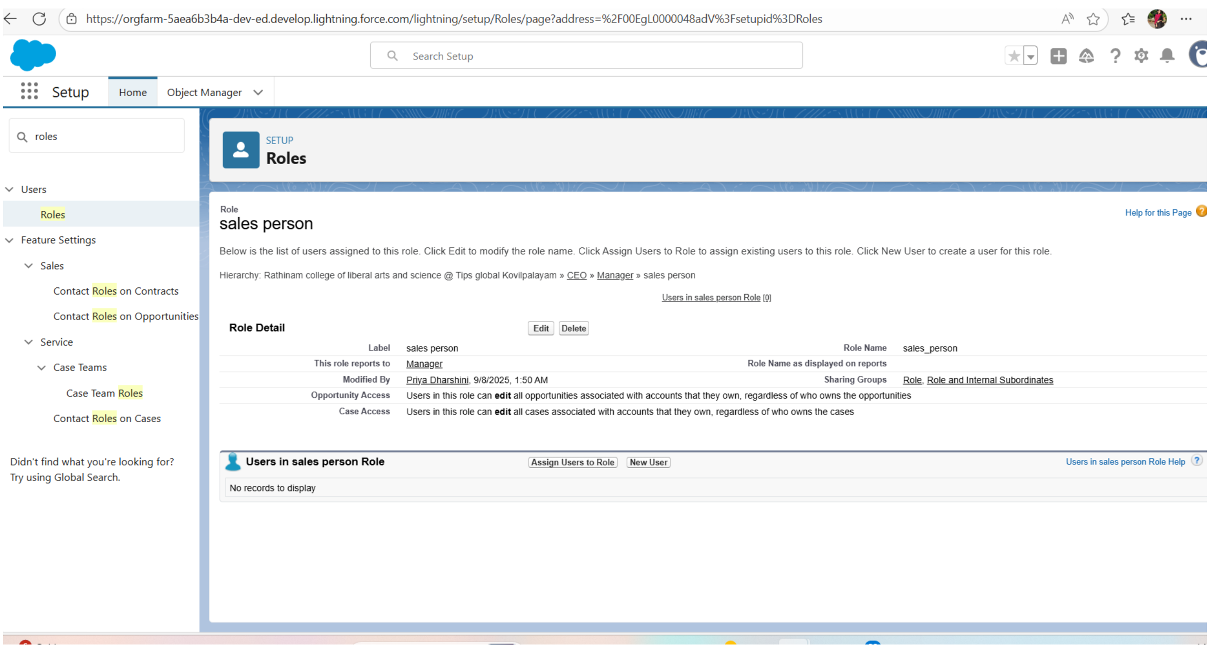






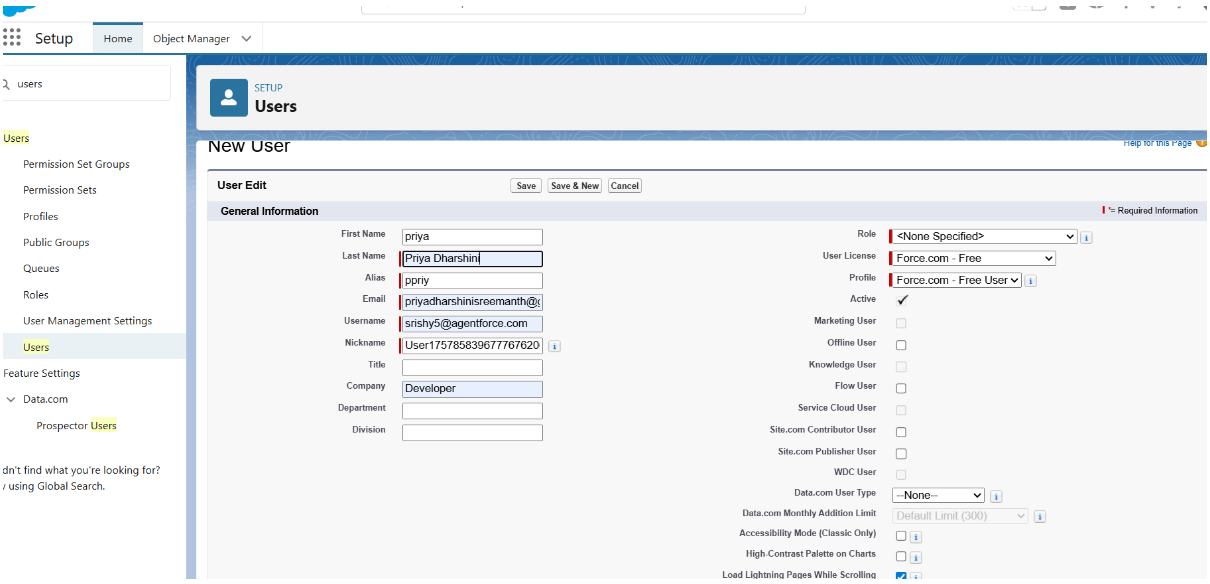
ii)Creating another roles



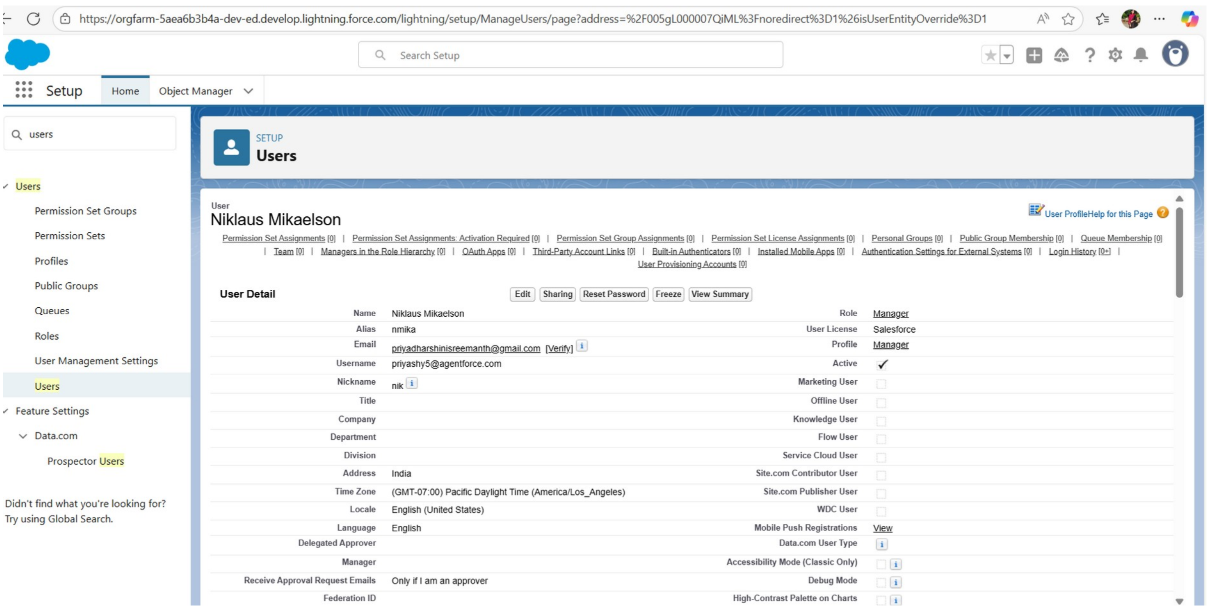


Milestone 10: Users

i)Create User

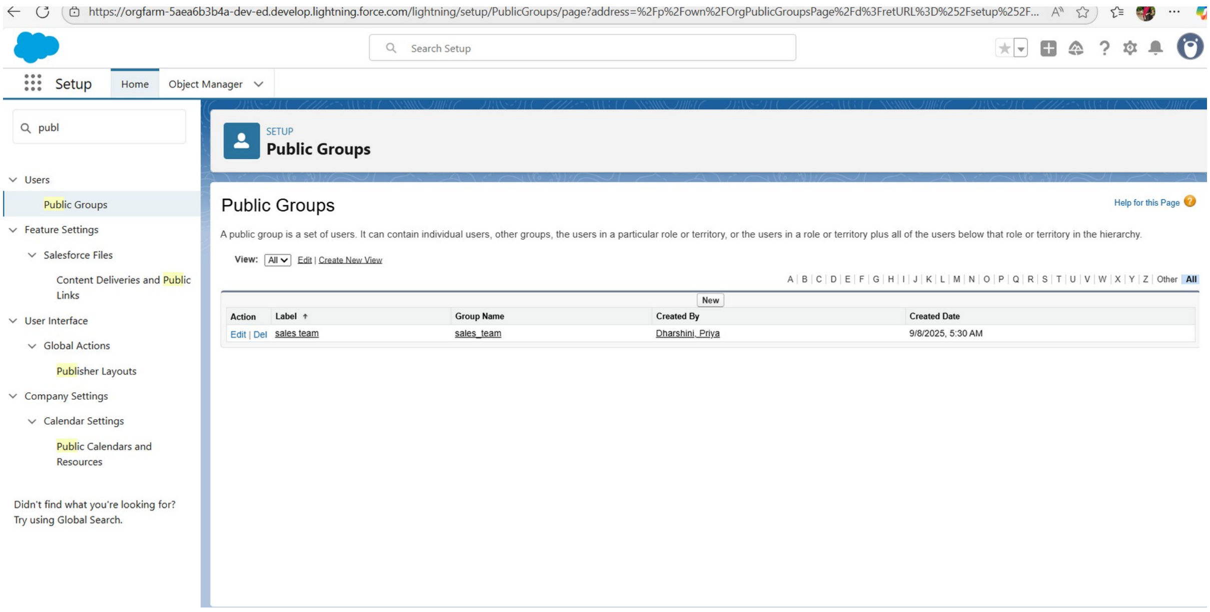


ii) creating another users



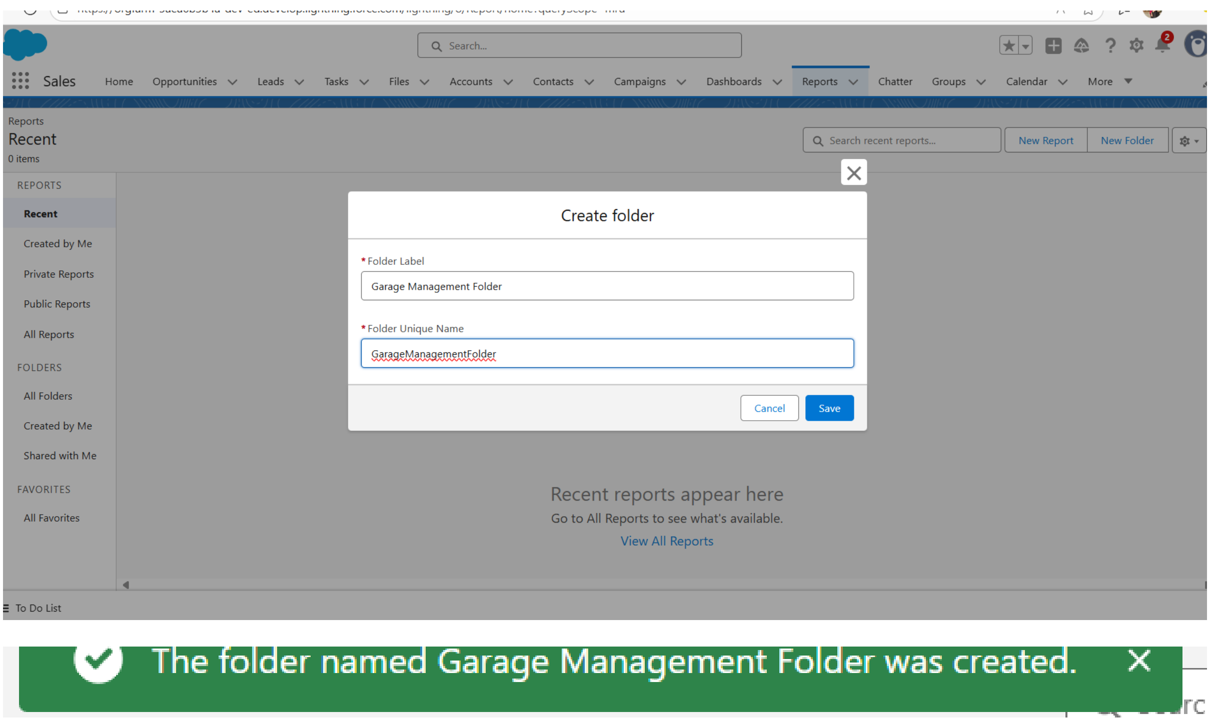
Milestone 11: Public groups

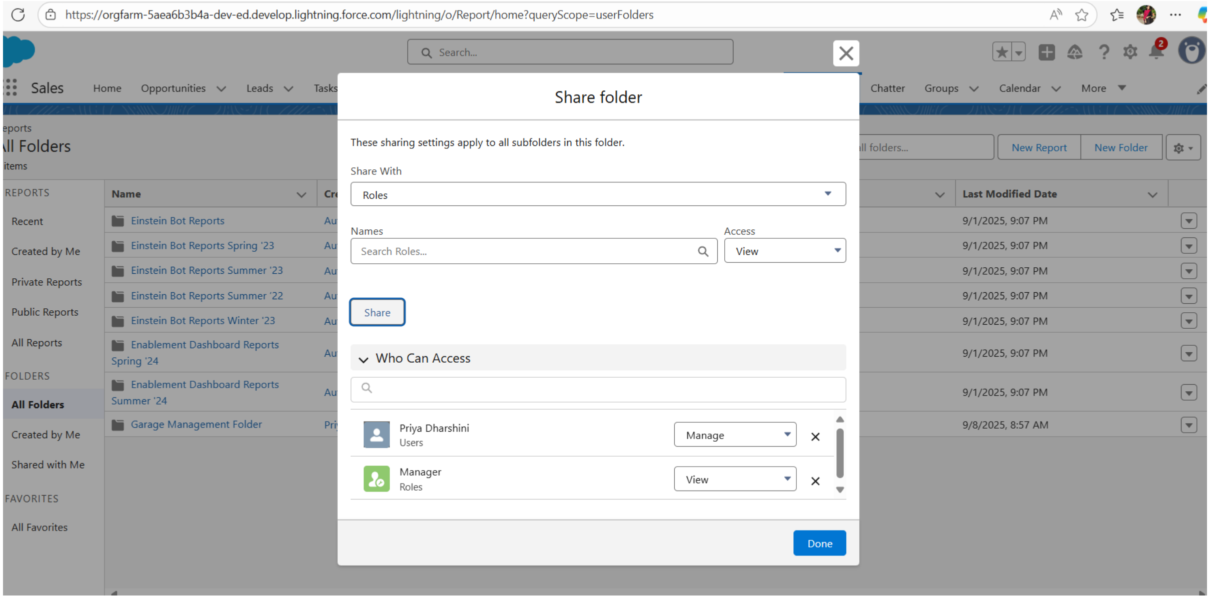
i)Creating New Public Group



Milestone 12: Share setting

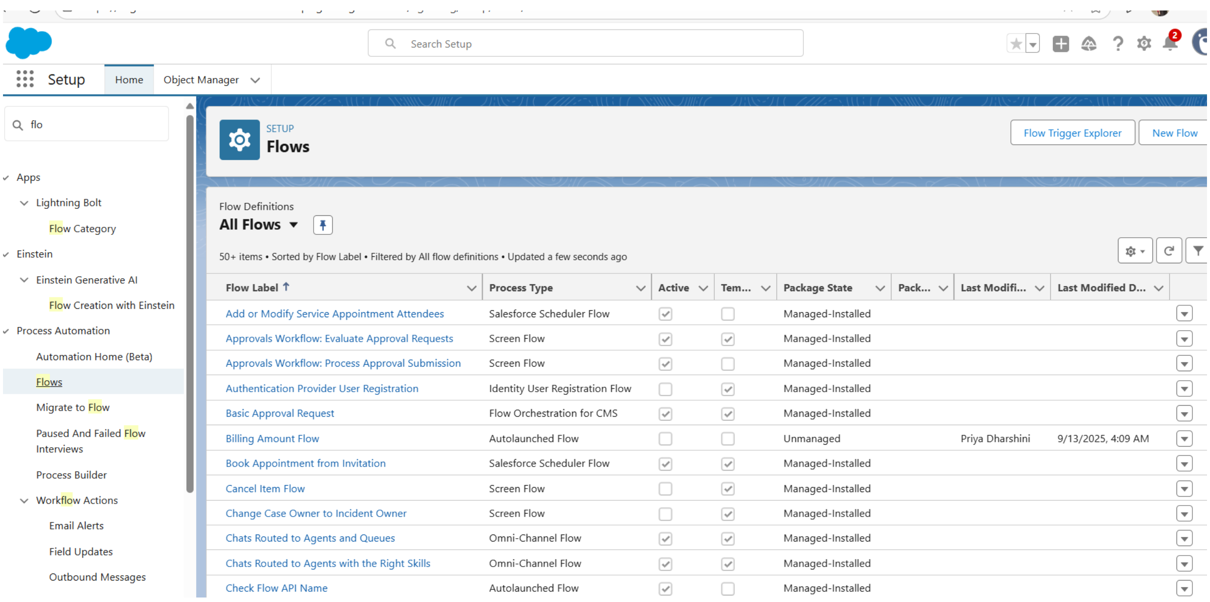
i)Creating Sharing settings

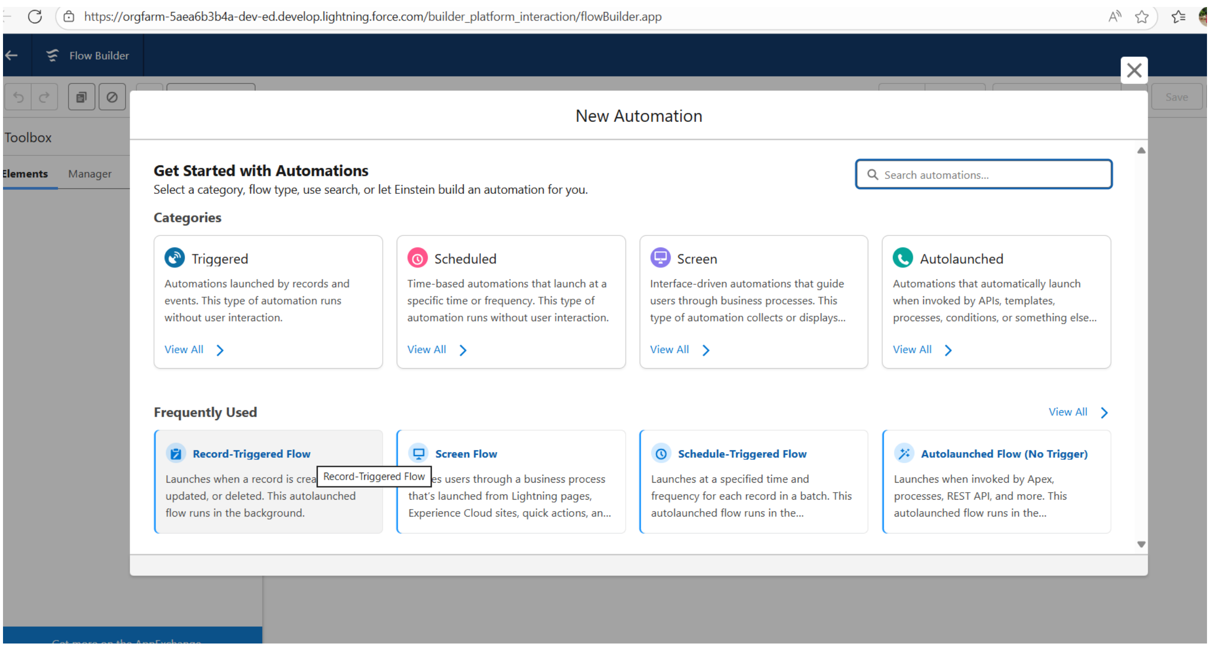


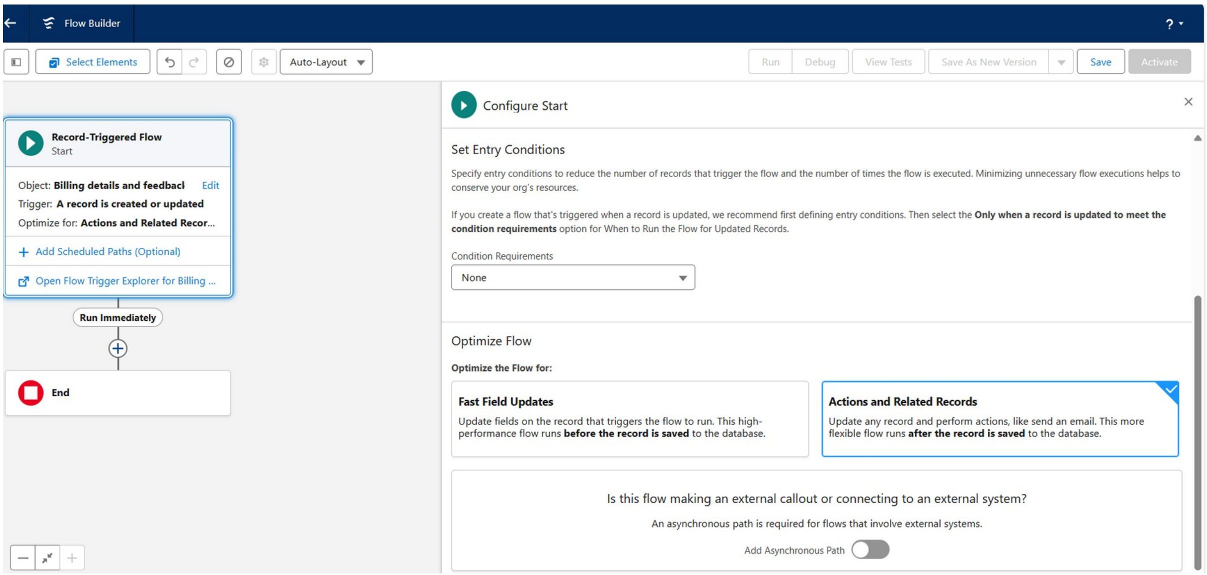


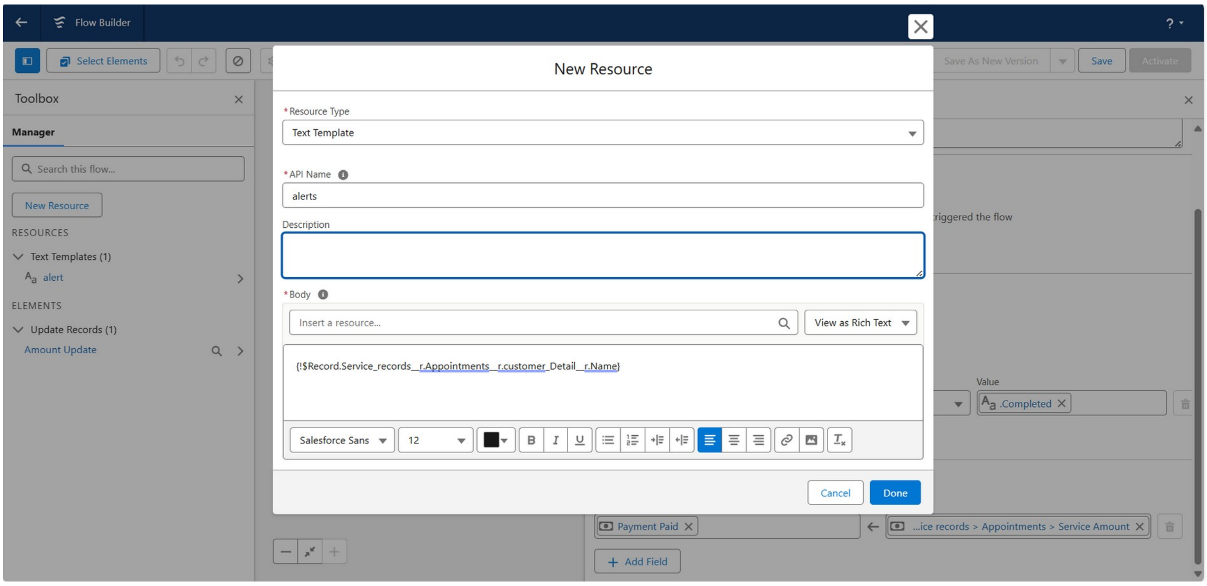
Milestone 13: Flows

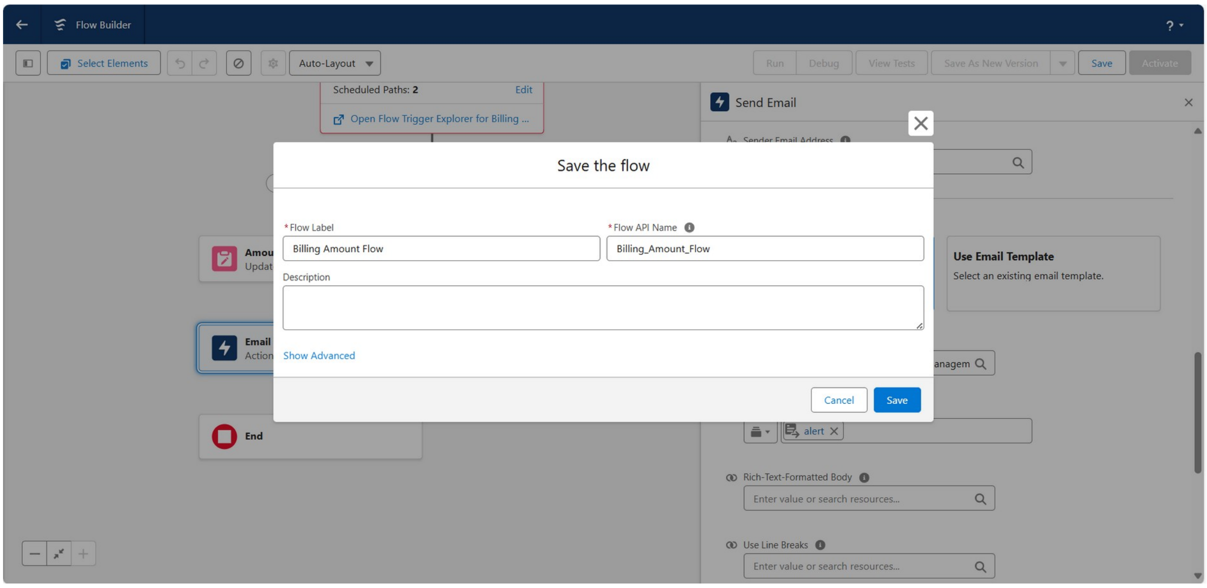
i)Create a Flow

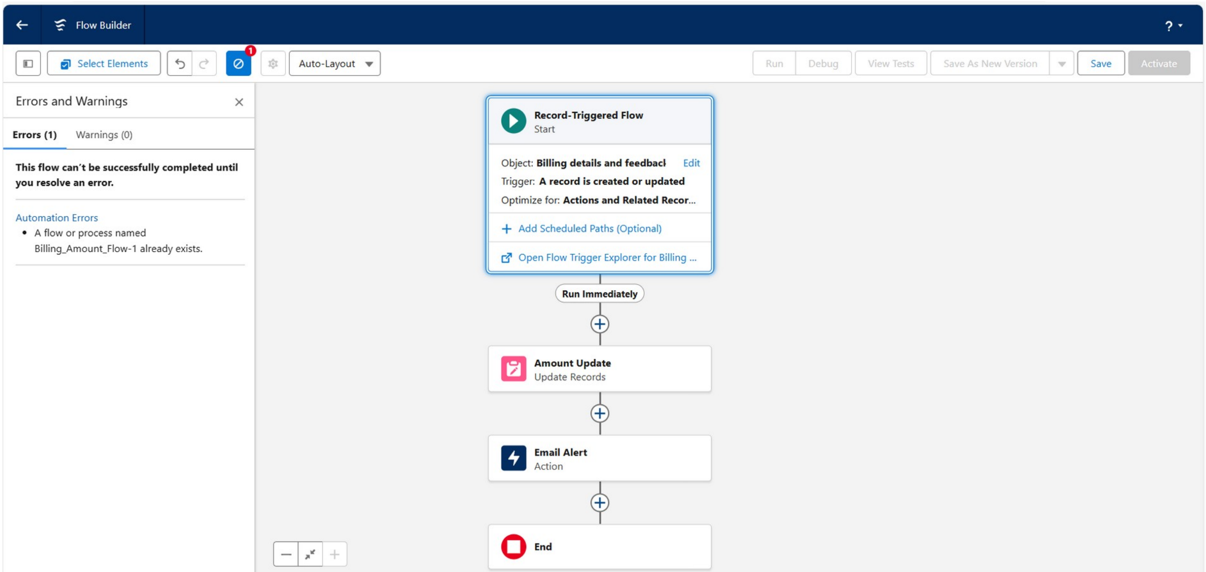



Flow Builder
Toolbox
Elements
Manager
Save
New Automation
Get Started with Automations
Select a category, flow type, use search, or let Einstein build an automation for you.
Search automations...
Categories
Triggered
Automations launched by records and events. This type of automation runs without user interaction.
View All
Scheduled
Time-based automations that launch at a specific time or frequency. This type of automation runs without user interaction.
View All
Screen
Interface-driven automations that guide users through business processes. This type of automation collects or displays...
View All
Autolaunched
Automations that automatically launch when invoked by APIs, templates, processes, conditions, or something else...
View All
Frequently Used
View All
Record-Triggered Flow
Record-Triggered Flow
updated, or deleted. This autolaunched flow runs in the background.
Screen Flow
that's launched from Lightning pages, Experience Cloud sites, quick actions, an...
Schedule-Triggered Flow
Launches at a specified time and frequency for each record in a batch. This autolaunched flow runs in the...
Autolaunched Flow (No Trigger)
Launches when invoked by Apex, processes, REST API, and more. This autolaunched flow runs in the...

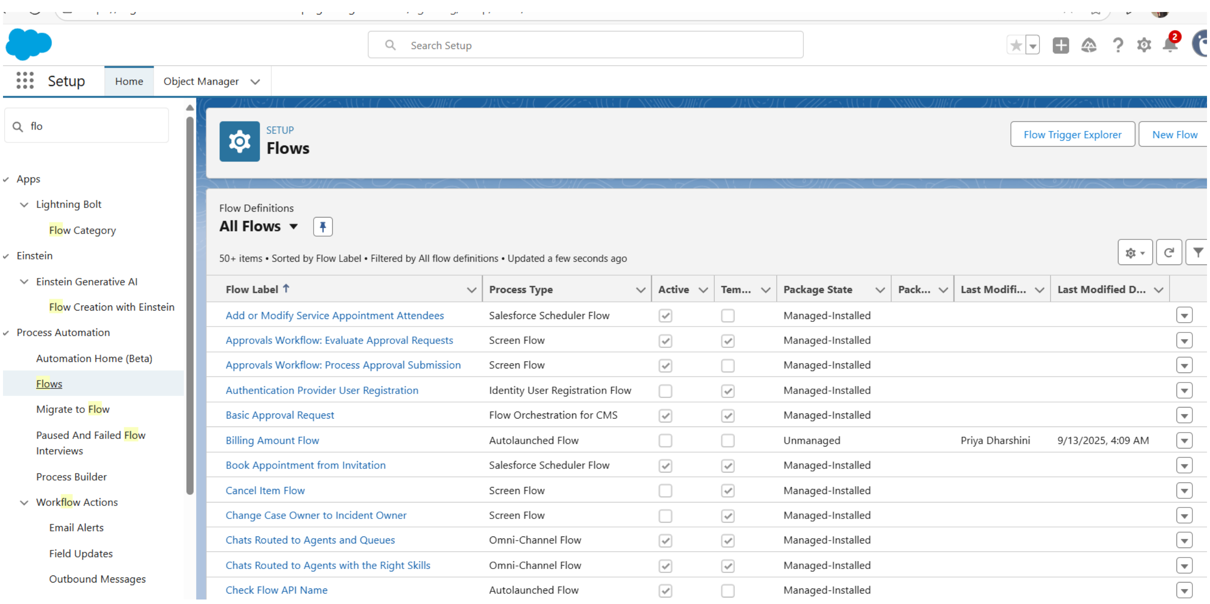

Flow Builder
Select Elements
Auto-Layout
Run
Debug
View Tests
Save As New Version
Save
Activate
Record-Triggered Flow
Start
Object: Billing details and feedbacl
Edit
Trigger: A record is created or updated
Optimize for: Actions and Related Recor...
Add Scheduled Paths (Optional)
Open Flow Trigger Explorer for Billing ...
Run Immediately
End
Configure Start
Set Entry Conditions
Specify entry conditions to reduce the number of records that trigger the flow and the number of times the flow is executed. Minimizing unnecessary flow executions helps to conserve your org's resources.
If you create a flow that's triggered when a record is updated, we recommend first defining entry conditions. Then select the Only when a record is updated to meet the condition requirements option for When to Run the Flow for Updated Records.
Condition Requirements
None
Optimize Flow
Optimize the Flow for:
Fast Field Updates
Update fields on the record that triggers the flow to run. This high-performance flow runs before the record is saved to the database.
Actions and Related Records
Update any record and perform actions, like send an email. This more flexible flow runs after the record is saved to the database.
Is this flow making an external callout or connecting to an external system?
An asynchronous path is required for flows that involve external systems.
Add Asynchronous Path

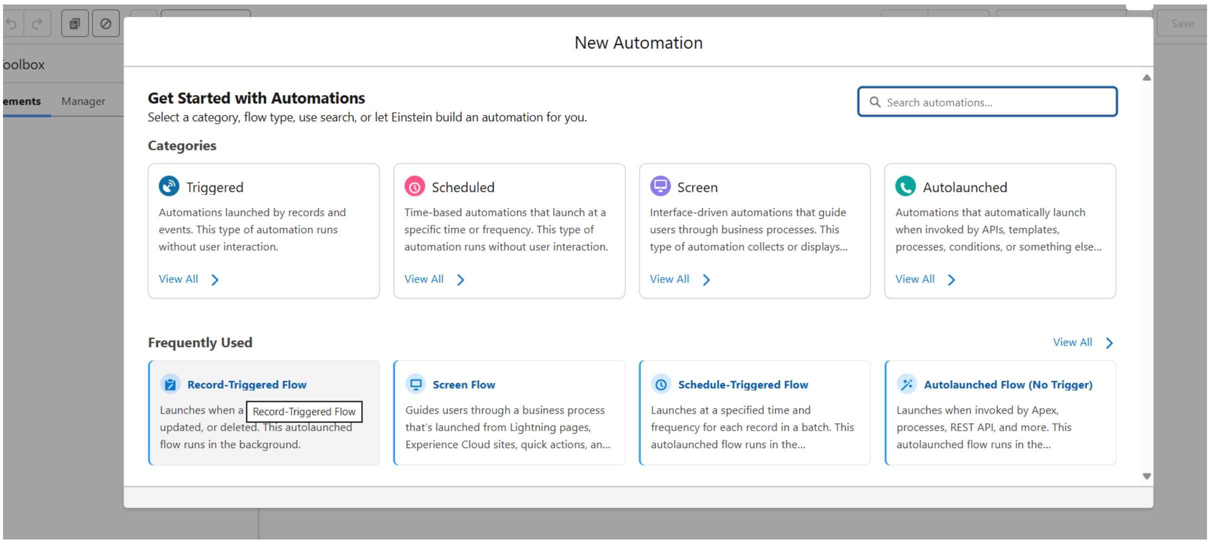

New Resource
Resource Type
Text Template
API Name
alerts
Description
Body
Insert a resource...
View as Rich Text
{!$Record.Service_records__r.Appointments__r.customer_Detail__r.Name}
Salesforce Sans
12
Cancel
Done


Save the flow
Flow Label
Billing Amount Flow
Flow API Name
Billing_Amount_Flow
Description
Show Advanced
Cancel
Save

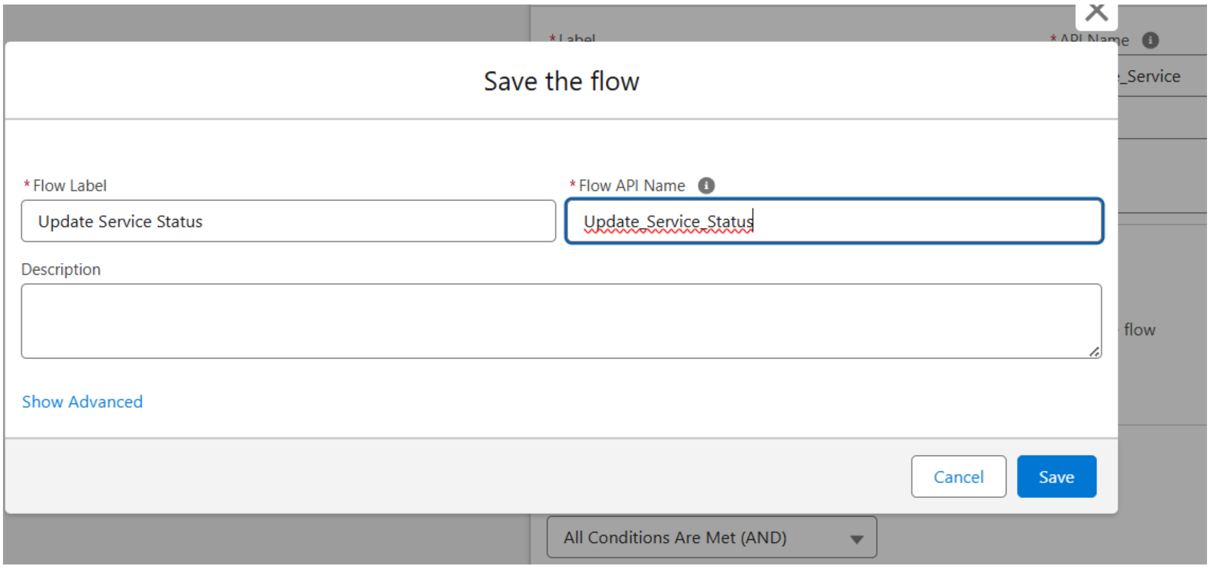


ii)Create another Flow

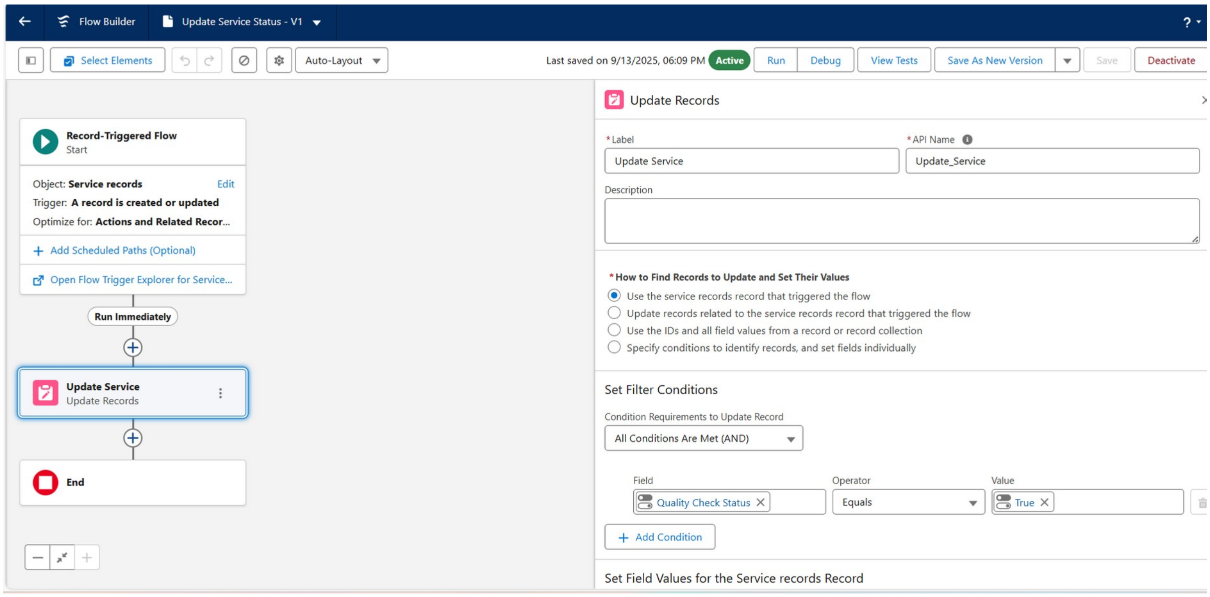




New Automation
Get Started with Automations
Select a category, flow type, use search, or let Einstein build an automation for you.
Search automations...
Categories
Triggered
Automations launched by records and events. This type of automation runs without user interaction.
View All
Scheduled
Time-based automations that launch at a specific time or frequency. This type of automation runs without user interaction.
View All
Screen
Interface-driven automations that guide users through business processes. This type of automation collects or displays...
View All
Autolaunched
Automations that automatically launch when invoked by APIs, templates, processes, conditions, or something else...
View All
Frequently Used
View All
Record-Triggered Flow
Launches when a record is created, updated, or deleted. This autolaunched flow runs in the background.
Screen Flow
Guides users through a business process that's launched from Lightning pages, Experience Cloud sites, quick actions, an...
Schedule-Triggered Flow
Launches at a specified time and frequency for each record in a batch. This autolaunched flow runs in the...
Autolaunched Flow (No Trigger)
Launches when invoked by Apex, processes, REST API, and more. This autolaunched flow runs in the...

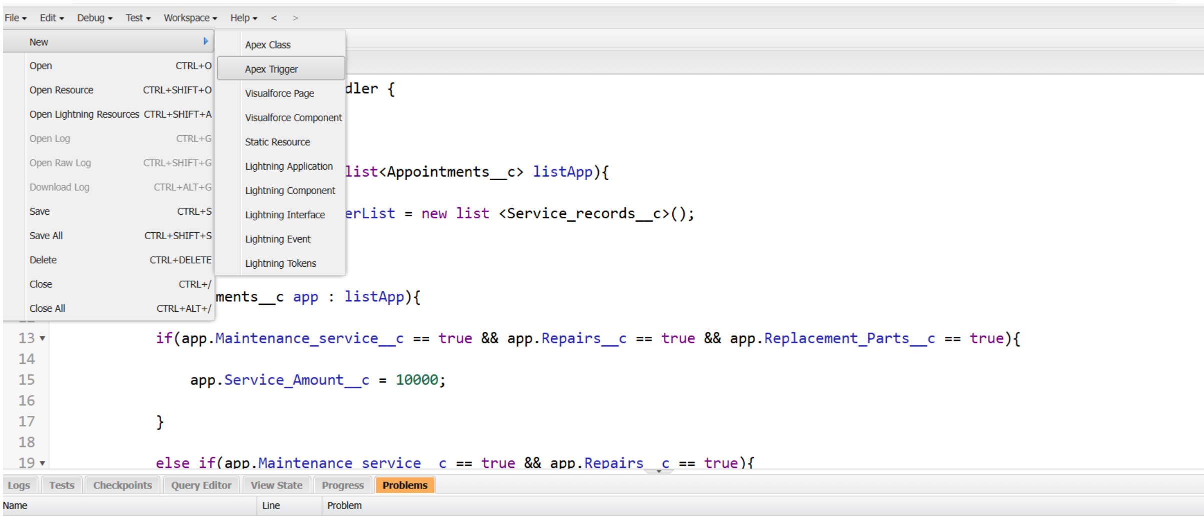


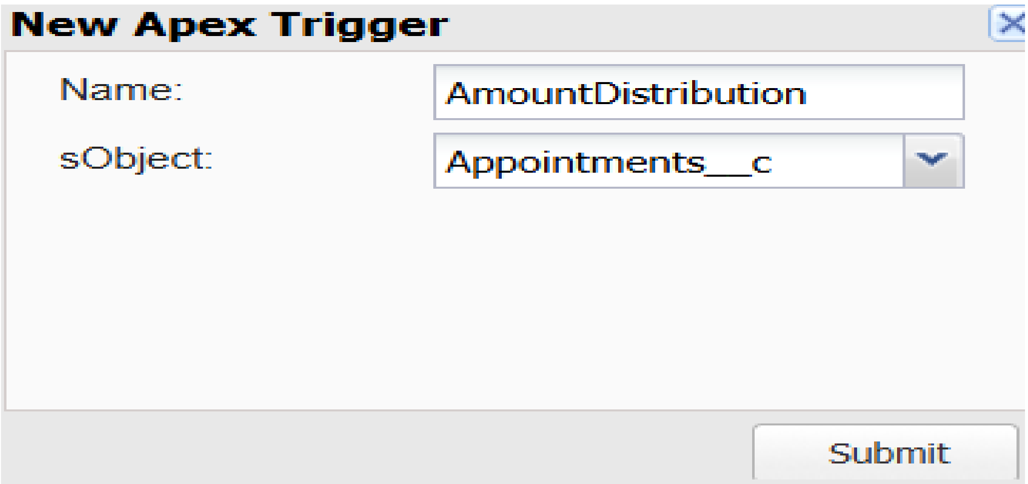
Save the flow
* Flow Label
Update Service Status
* Flow API Name
Update_Service_Status
Description
Show Advanced
Cancel
Save
All Conditions Are Met (AND)

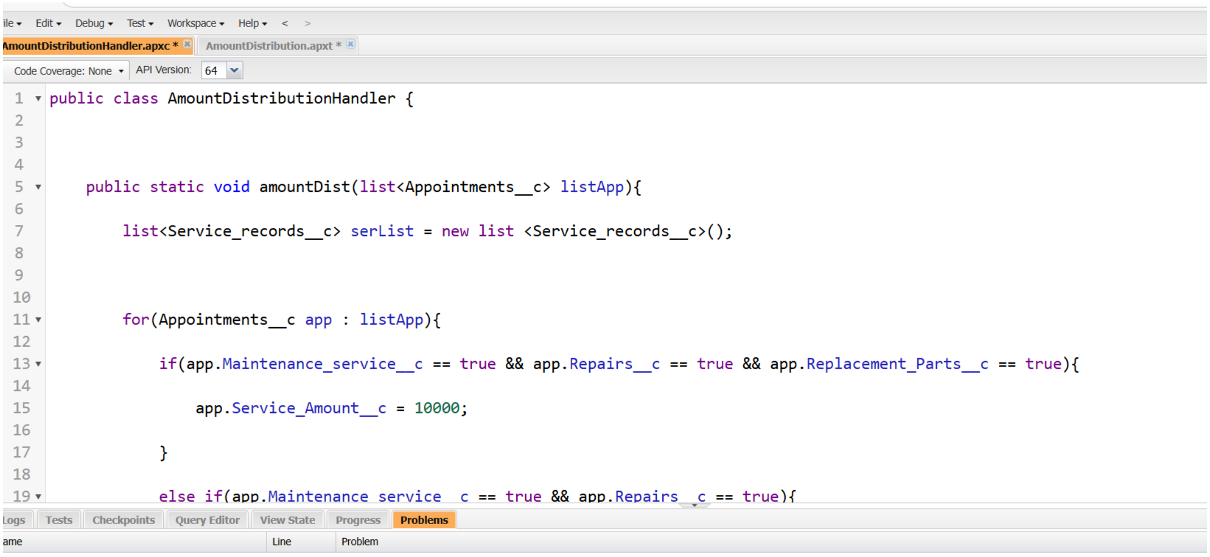


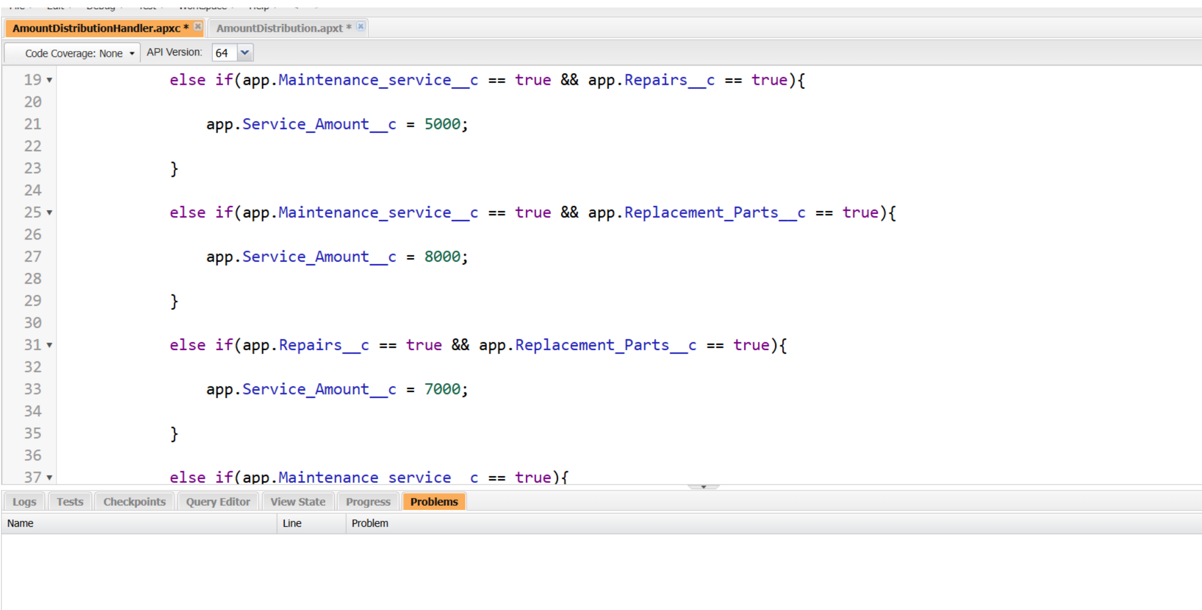
Milestone 14: Apex trigger

i) Apex handler




New Apex Trigger
Name:
AmountDistribution
sObject:
Appointments__c
Submit


File Edit Debug Test Workspace Help
AmountDistributionHandler.apxc * AmountDistribution.apxt *
Code Coverage: None API Version: 64
public class AmountDistributionHandler {
public static void amountDist(list<Appointments__c> listApp){
list<Service_records__c> serList = new list <Service_records__c>();
for(Appointments__c app : listApp){
if(app.Maintenance_service__c == true && app.Repairs__c == true && app.Replacement_Parts__c == true){
app.Service_Amount__c = 10000;
}
else if(app.Maintenance_service__c == true && app.Repairs__c == true){
Logs Tests Checkpoints Query Editor View State Progress Problems
Name Line Problem


AmountDistributionHandler.apxc * AmountDistribution.apxt *
Code Coverage: None API Version: 64
else if(app.Maintenance_service__c == true && app.Repairs__c == true){
app.Service_Amount__c = 5000;
}
else if(app.Maintenance_service__c == true && app.Replacement_Parts__c == true){
app.Service_Amount__c = 8000;
}
else if(app.Repairs__c == true && app.Replacement_Parts__c == true){
app.Service_Amount__c = 7000;
}
else if(app.Maintenance_service__c == true){
Logs Tests Checkpoints Query Editor View State Progress Problems
Name Line Problem

```
            else if(app.Maintenance_service__c == true){

                app.Service_Amount__c = 2000;

            }

            else if(app.Repairs__c == true){

                app.Service_Amount__c = 3000;

            }

            else if(app.Replacement_Parts__c == true){

                app.Service_Amount__c = 5000;

            }
```

```
trigger AmountDistribution on Appointments__c (before insert, before update) {
    if (Trigger.isBefore && (Trigger.isInsert || Trigger.isUpdate)) {
        AmountDistributionHandler.amountDist(Trigger.new);
    }
}
```

CODE FOR APEX CLASS:

```
public class AmountDistributionHandler {
public static void amountDist(list<Appointments__c> listApp){
 list<Service_records__c> serList = new list <Service_records__c>();
 for(Appointments__c app : listApp){

        if(app.Maintenance_service__c == true && app.Repairs__c == true &&
app.Replacement_Parts__c == true){
 app.Service_Amount__c = 10000;
}
else if(app.Maintenance_service__c == true && app.Repairs__c == true){
 app.Service_Amount__c = 5000;

        }
  else if(app.Maintenance_service__c == true && app.Replacement_Parts__c == true){
app.Service_Amount__c = 8000;

        }
 else if(app.Repairs__c == true && app.Replacement_Parts__c == true){
 app.Service_Amount__c = 7000;
 }
else if(app.Maintenance_service__c == true){
app.Service_Amount__c = 2000;
 }
 else if(app.Repairs__c == true){

            app.Service_Amount__c = 3000;
```

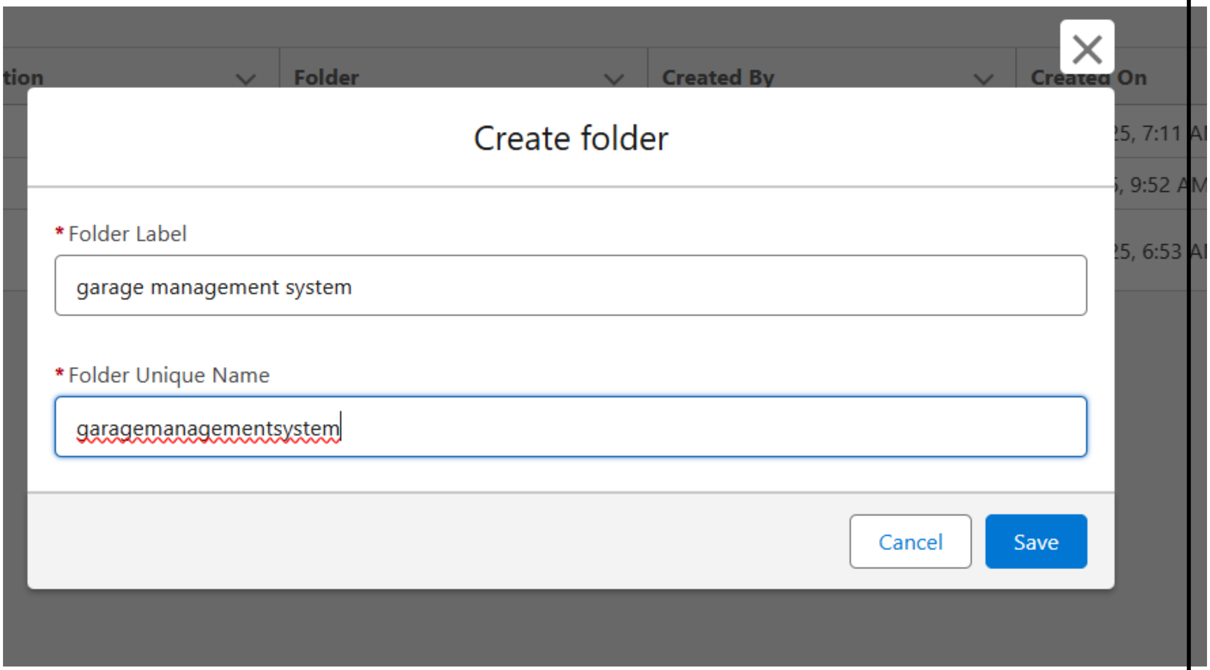
```
        }

        else if(app.Replacement_Parts__c == true){
 app.Service_Amount__c = 5000;

        }
}
}
```

CODE FOR APEX TRIGGER:

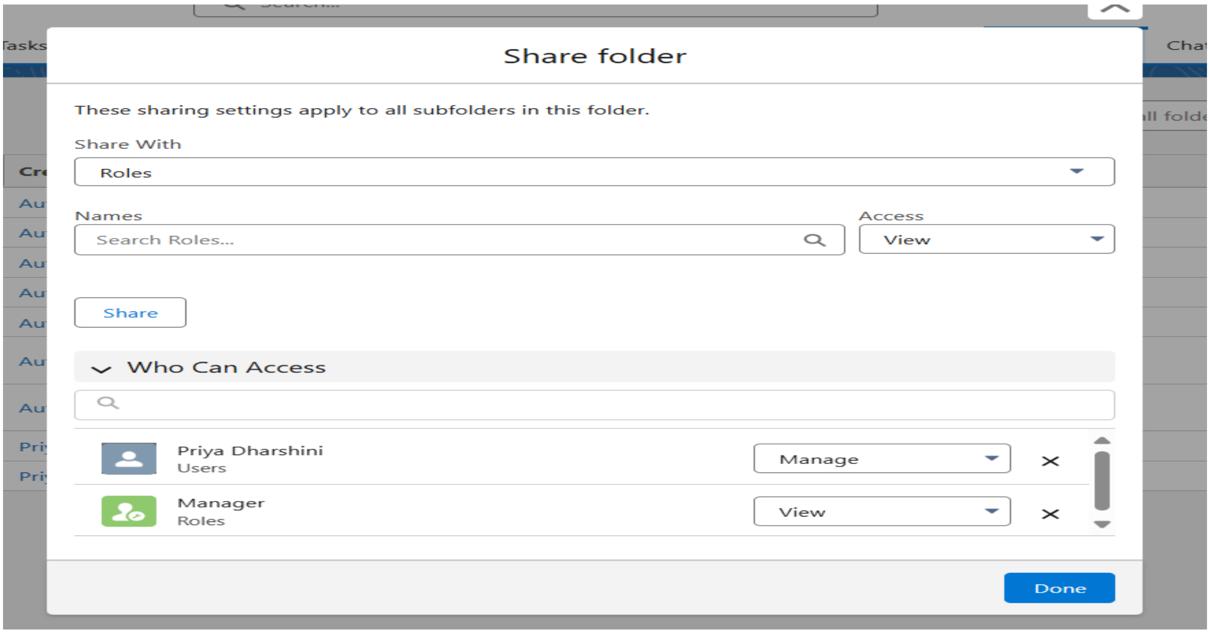
```
            trigger AmountDistribution on Appointments__c (before insert, before update) {
    if (Trigger.isBefore && (Trigger.isInsert || Trigger.isUpdate)) {
        AmountDistributionHandler.amountDist(Trigger.new);
    }
}
```

Milestone 15: Reports

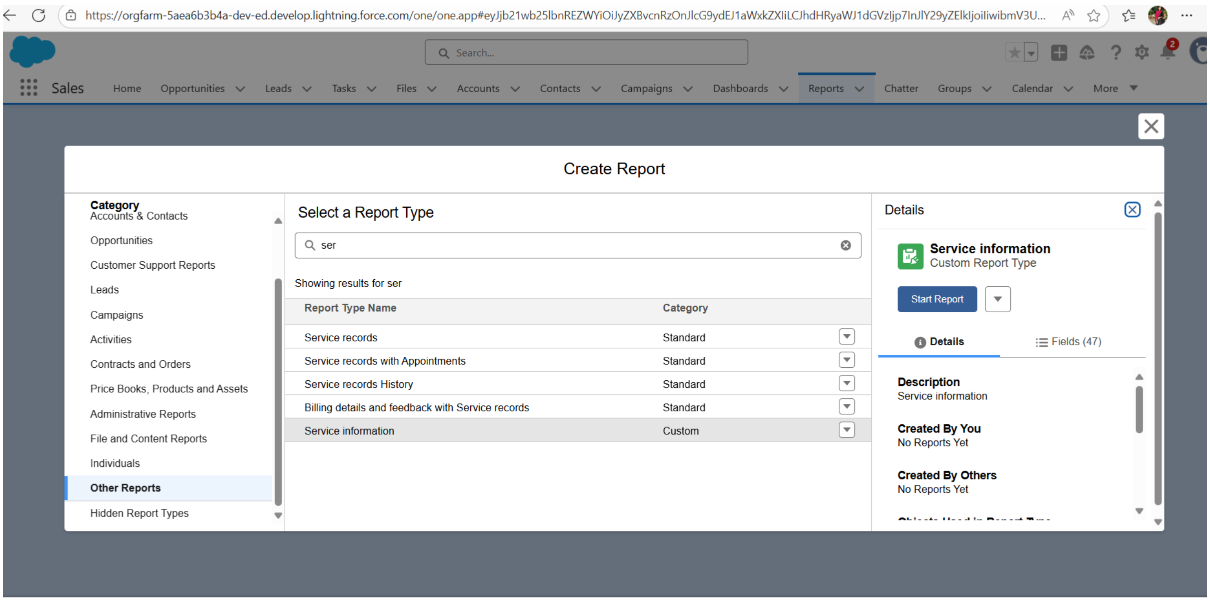
i)create a report folder



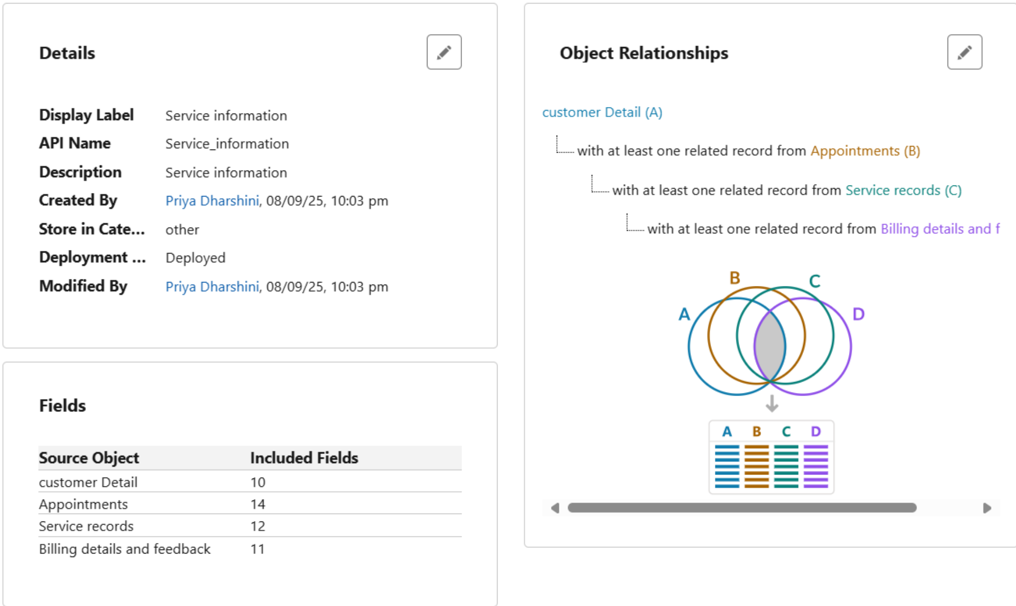
ii) Sharing a report folder

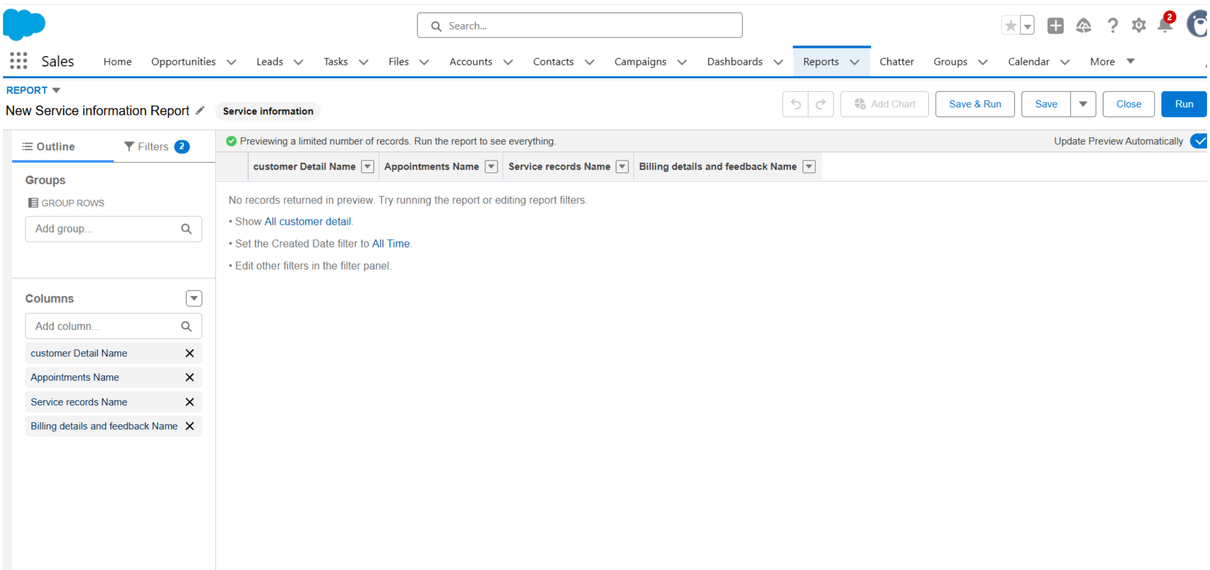


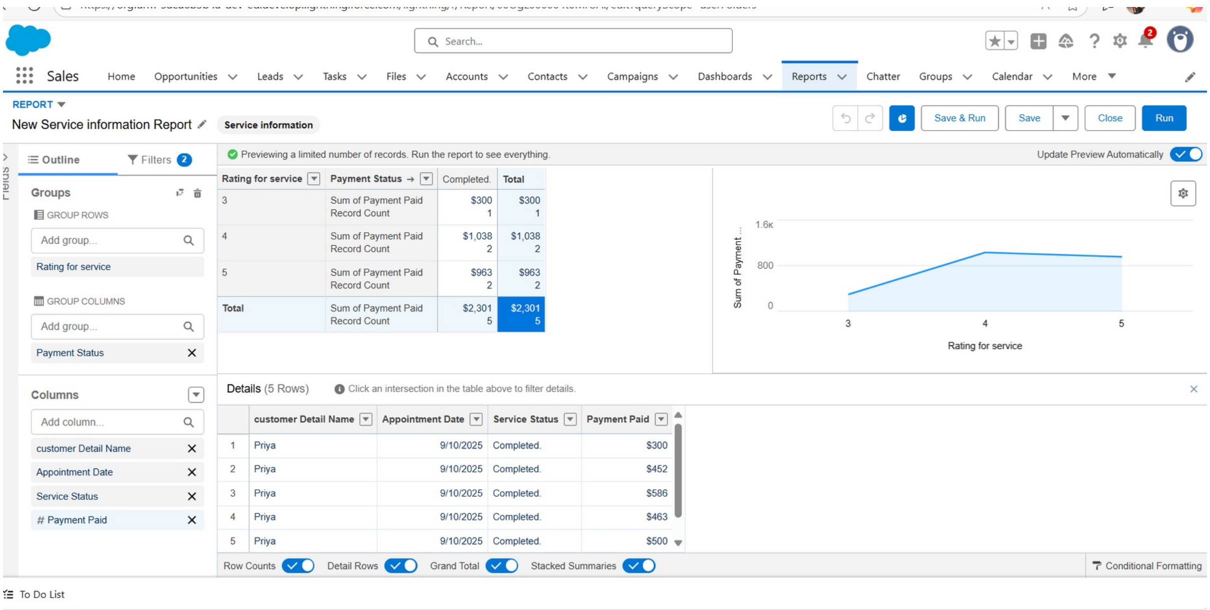
iii) Create report type

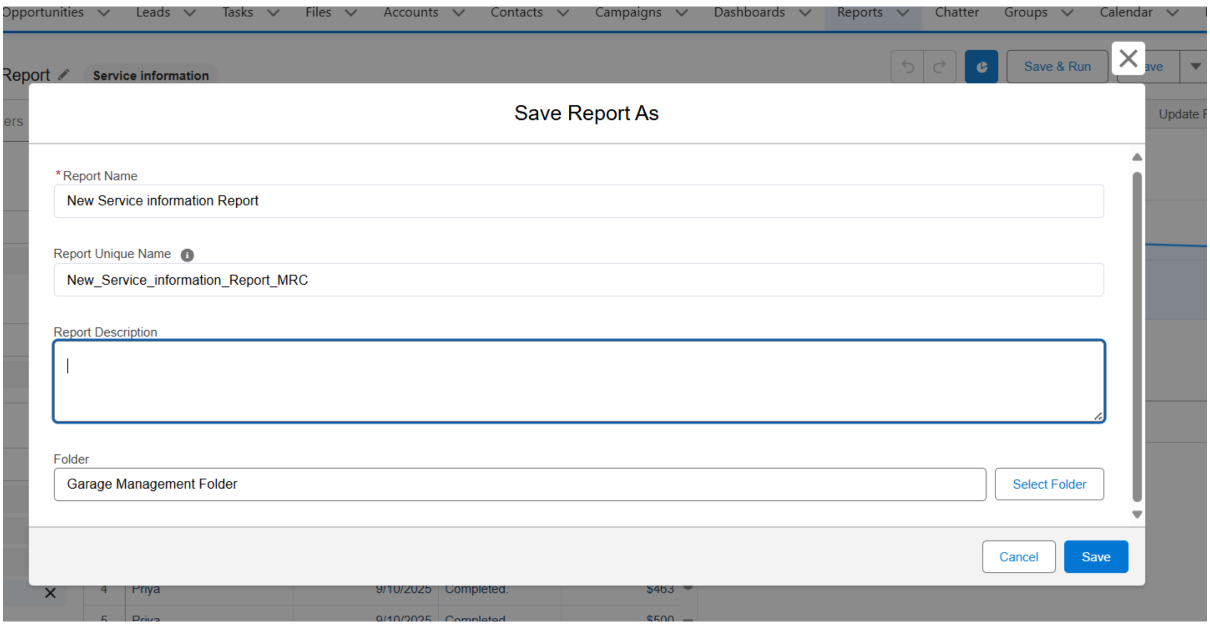


iv) Create report



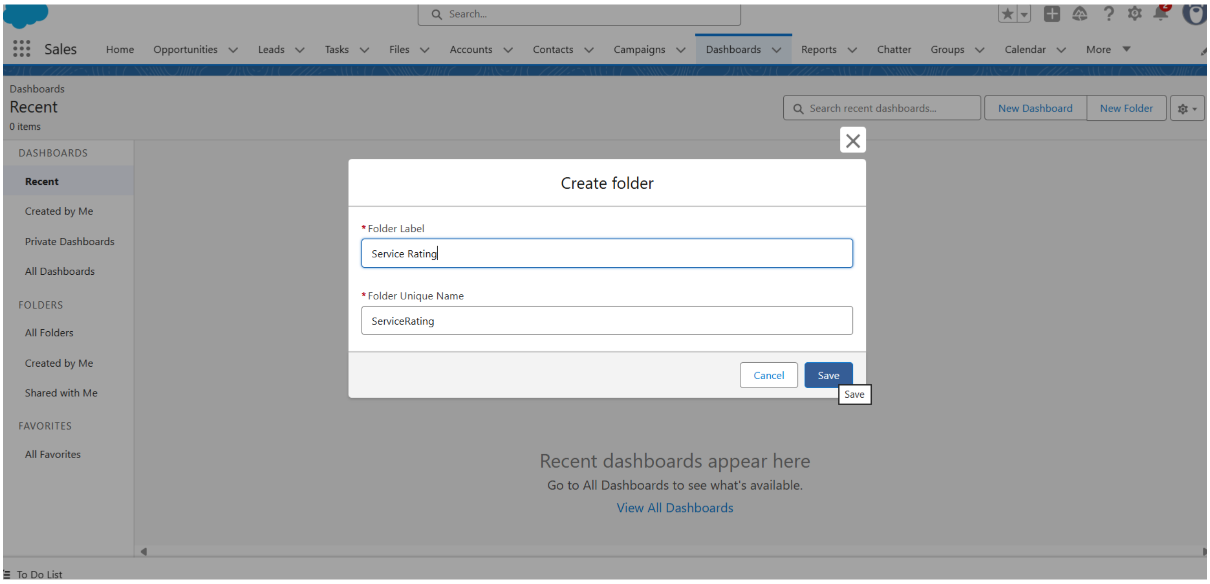




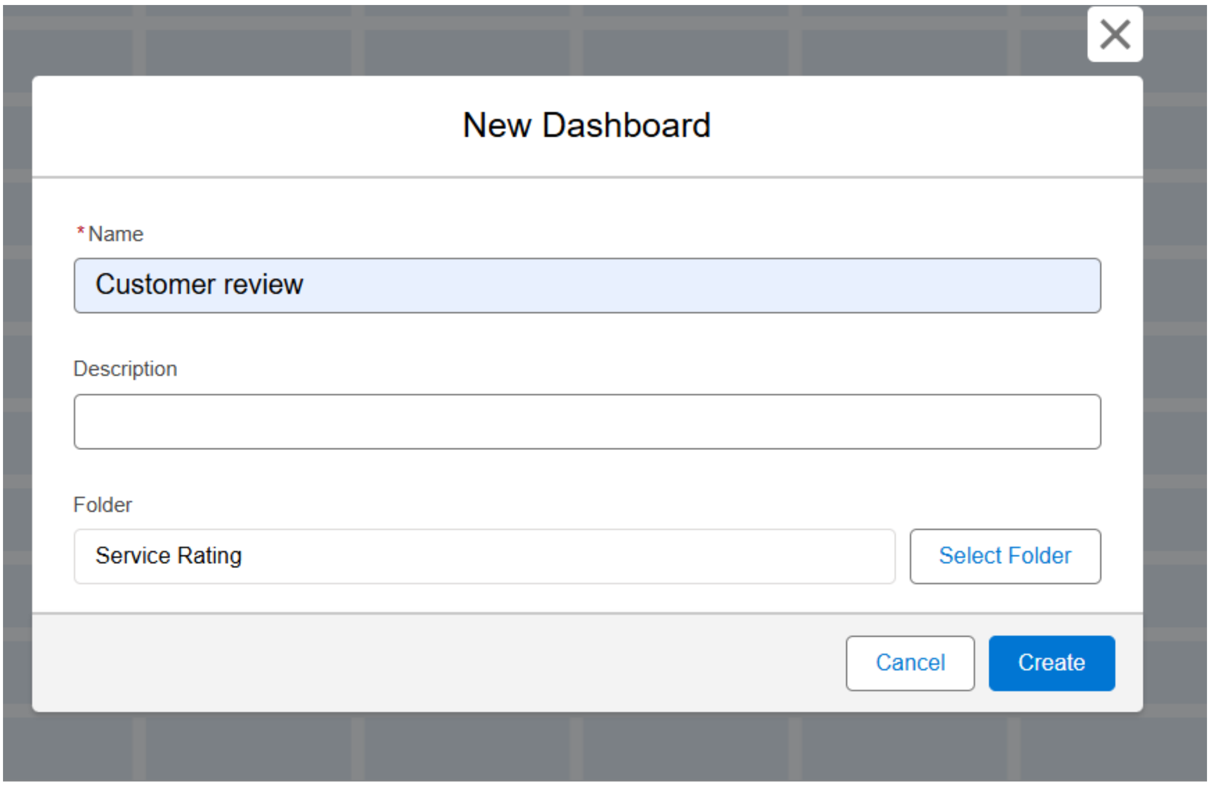

Save Report As
*Report Name
New Service information Report
Report Unique Name
New_Service_information_Report_MRC
Report Description
Folder
Garage Management Folder
Select Folder
Cancel
Save

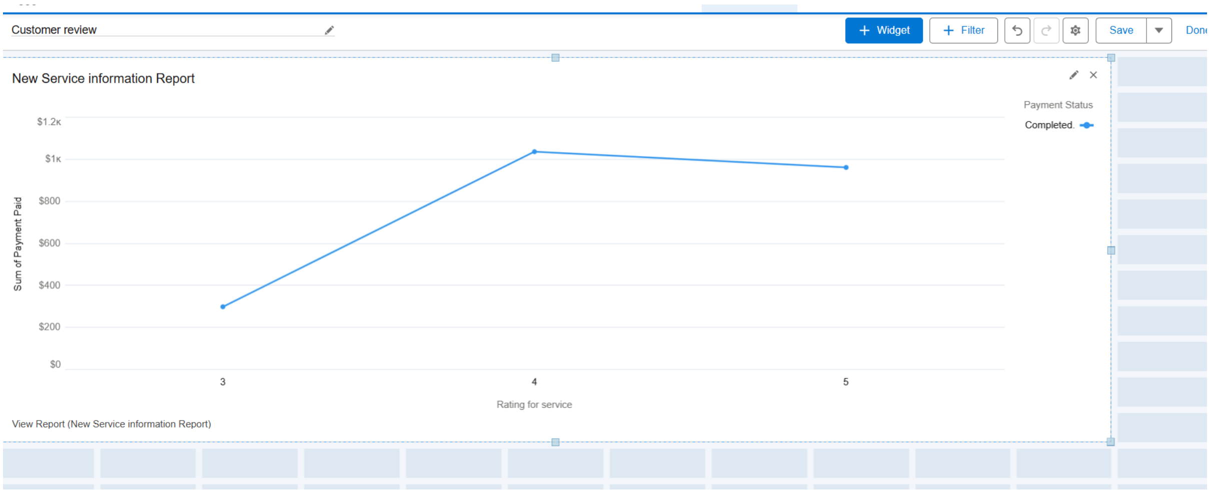
Milestone 16: Dashboards

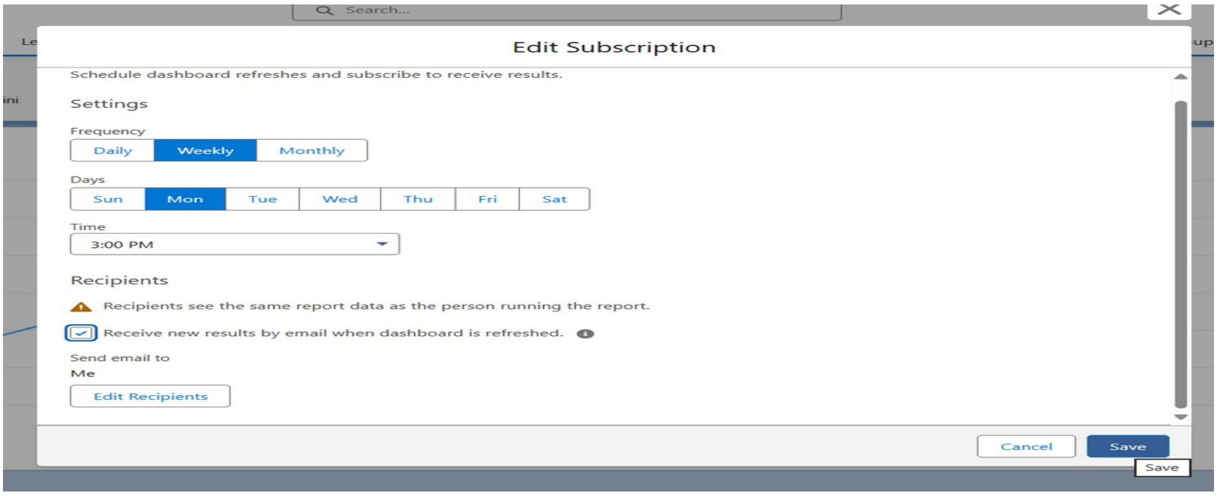
i) Create Dashboard Folder



ii) Create Dashboard

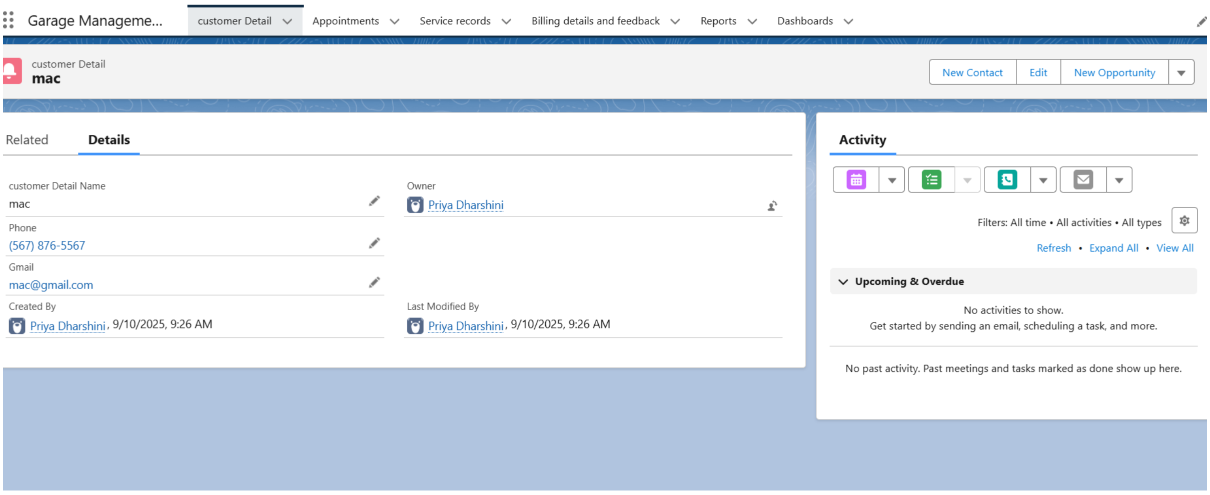


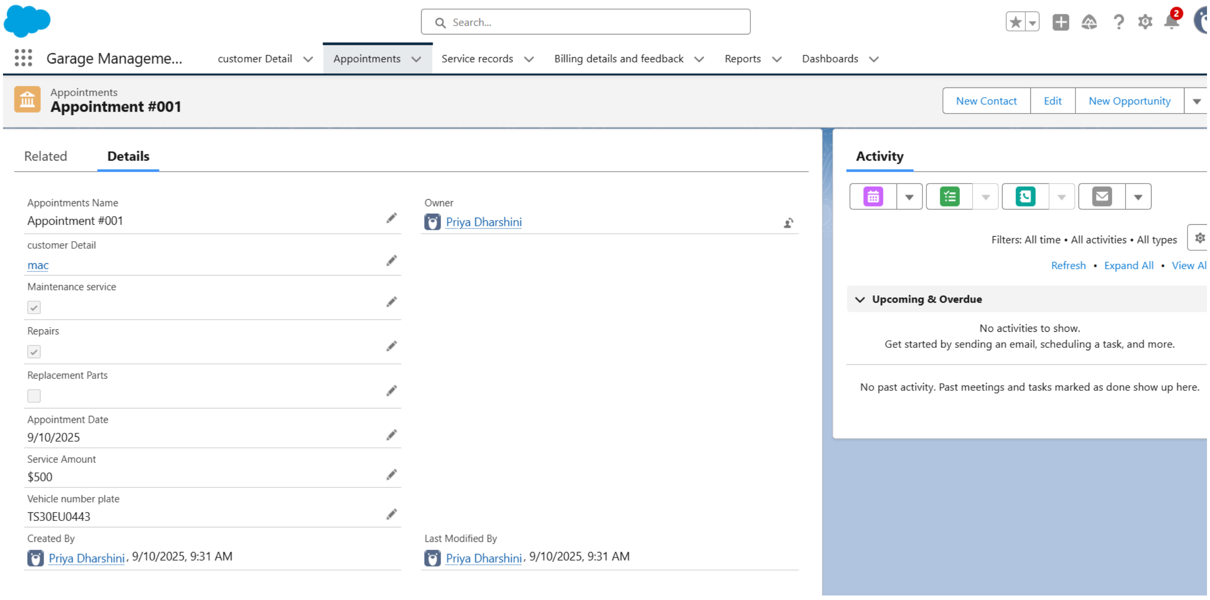


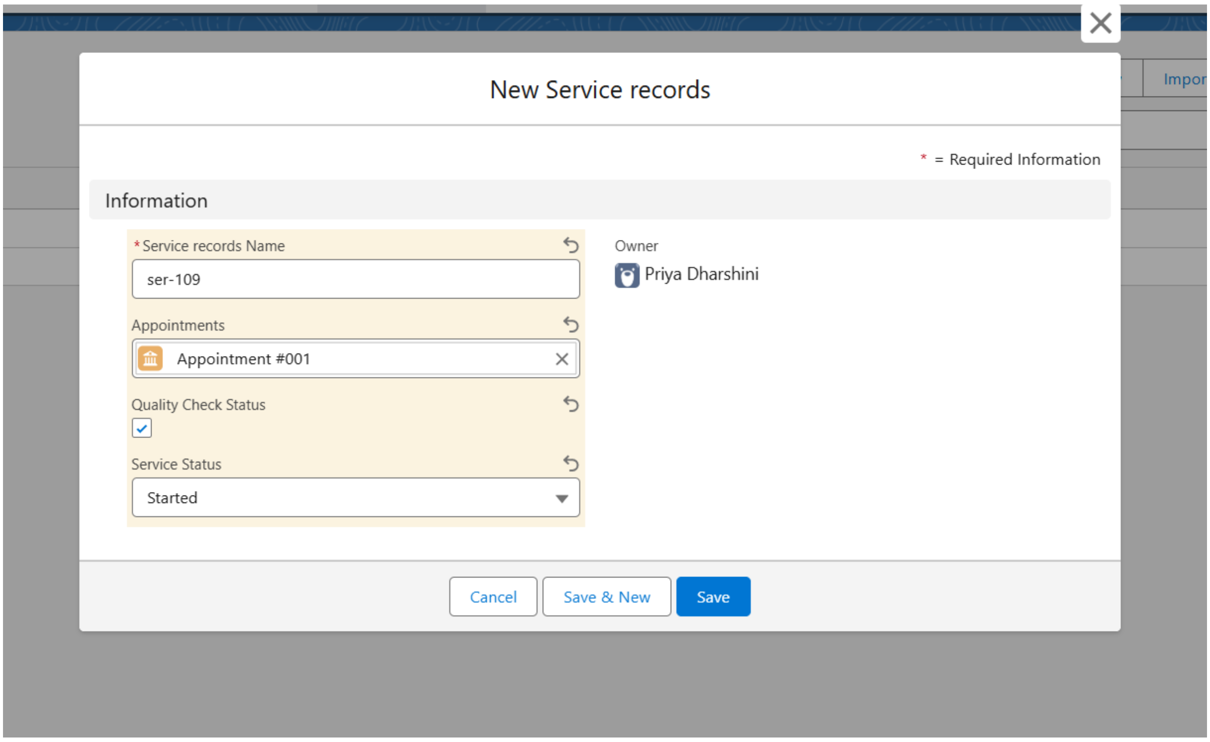


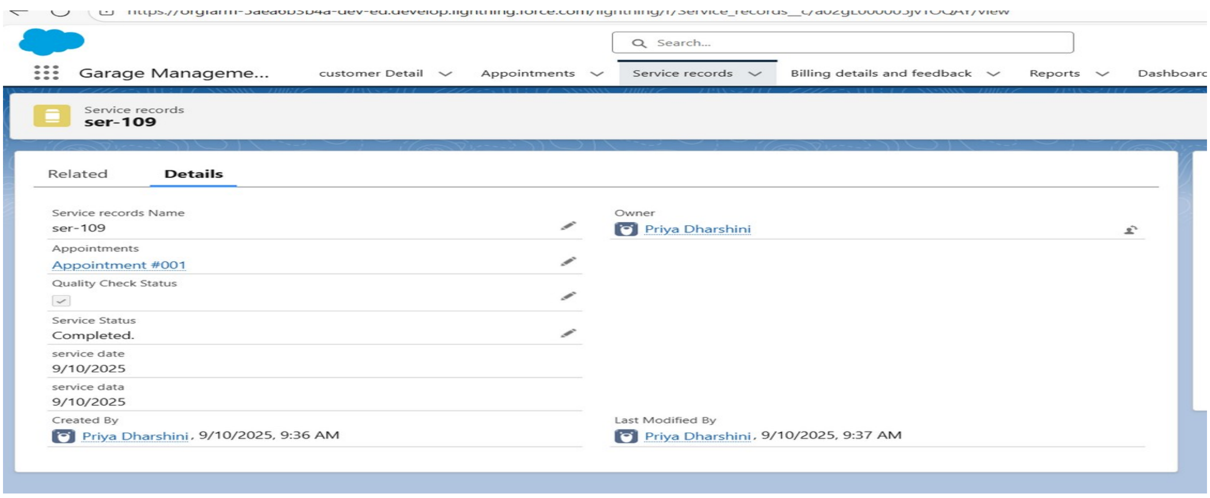
Milestone 17: User Adoption

i)creating records






New Service records
* = Required Information
Information
*Service records Name
ser-109
Appointments
Appointment #001
Quality Check Status
Service Status
Started
Owner
Priya Dharshini
Cancel
Save & New
Save


Garage Manageme...
customer Detail
Appointments
Service records
Billing details and feedback
Reports
Service records
ser-109
Related
Details
Service records Name
ser-109
Appointments
Appointment #001
Quality Check Status
Service Status
Completed.
service date
9/10/2025
service data
9/10/2025
Created By
Priya Dharshini, 9/10/2025, 9:36 AM
Owner
Priya Dharshini
Last Modified By
Priya Dharshini, 9/10/2025, 9:37 AM

Advantages:

- Centralized customer and service data.
- Automated reminders and workflows.
- Real-time reporting and analytics.
- Scalable for growing businesses.
- Mobile access and system integration.
- Better customer communication.

Disadvantages:

- High setup and licensing costs.
- Requires technical setup and training.
- May be too complex for small garages.
- Needs reliable internet connection.

Conclusion:

Salesforce-based Garage Management Systems offer powerful tools for automation and growth, ideal for medium to large garages. However, small garages may find it too costly or complex for their needs.