

PROJECT TITLE: Garage Management System

College Name: PPG College of Arts and Science

College Code: bruap

TEAM ID: NM2025TMID25936

TEAM MEMBERS: 4

Team Leader Name: Avantika Nakshatra S

Email: 2328f0005.cas@ppg.edu.in

Team Member1: Barath P

Email: 2328f0006.cas@ppg.edu.in

Team Member2: Boomika G

Email: 2328f0007.cas@ppg.edu.in

Team Member3: Chandran P

Email: 2328f0008.cas@ppg.edu.in

INTRODUCTION

Project Overview

A Garage Management System is a software solution designed to streamline and automate the operations of an automobile garage or service center. It helps manage vehicle servicing, customer records, spare parts inventory, billing, and staff scheduling efficiently. The goal is to improve productivity, reduce manual paperwork, and provide better service to customers.

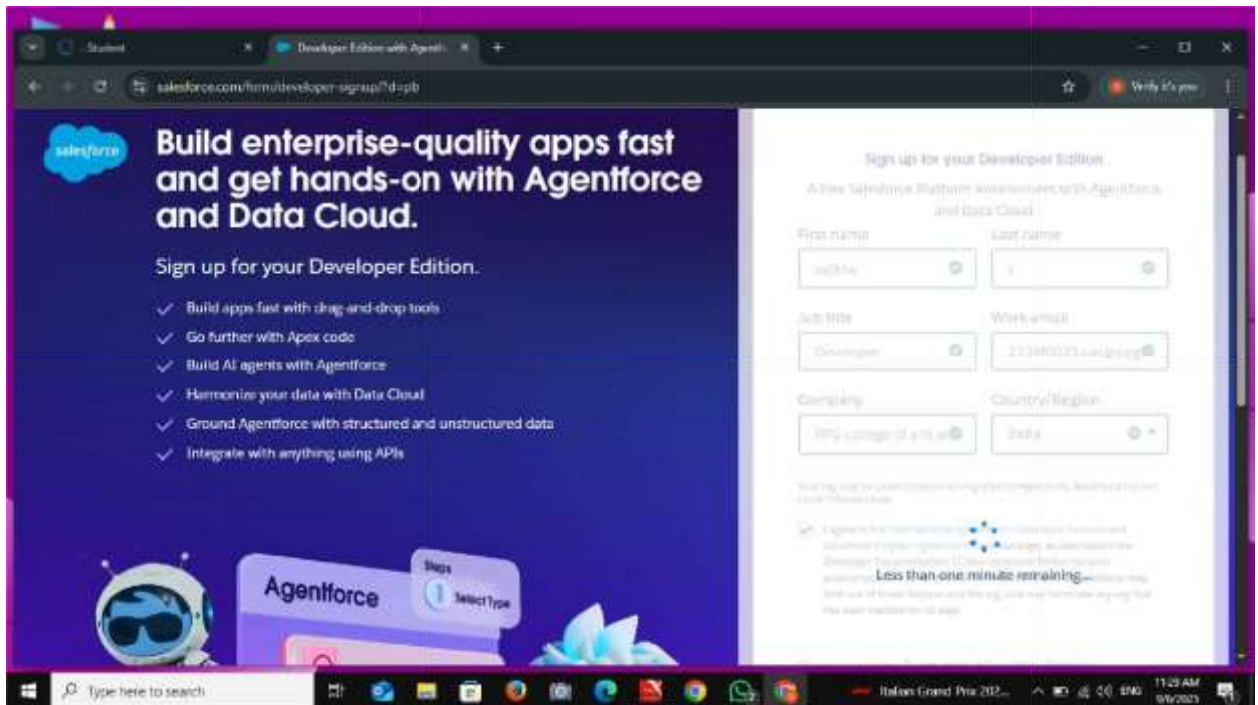
PURPOSE

The purpose of a Garage Management System is to automate and simplify the operations of an automobile garage or service center. It helps in maintaining customer and vehicle records, tracking service and repair activities, managing spare parts inventory, and generating accurate billing. The system ensures efficiency, reduces manual errors, and enhances customer satisfaction by providing timely and transparent services.

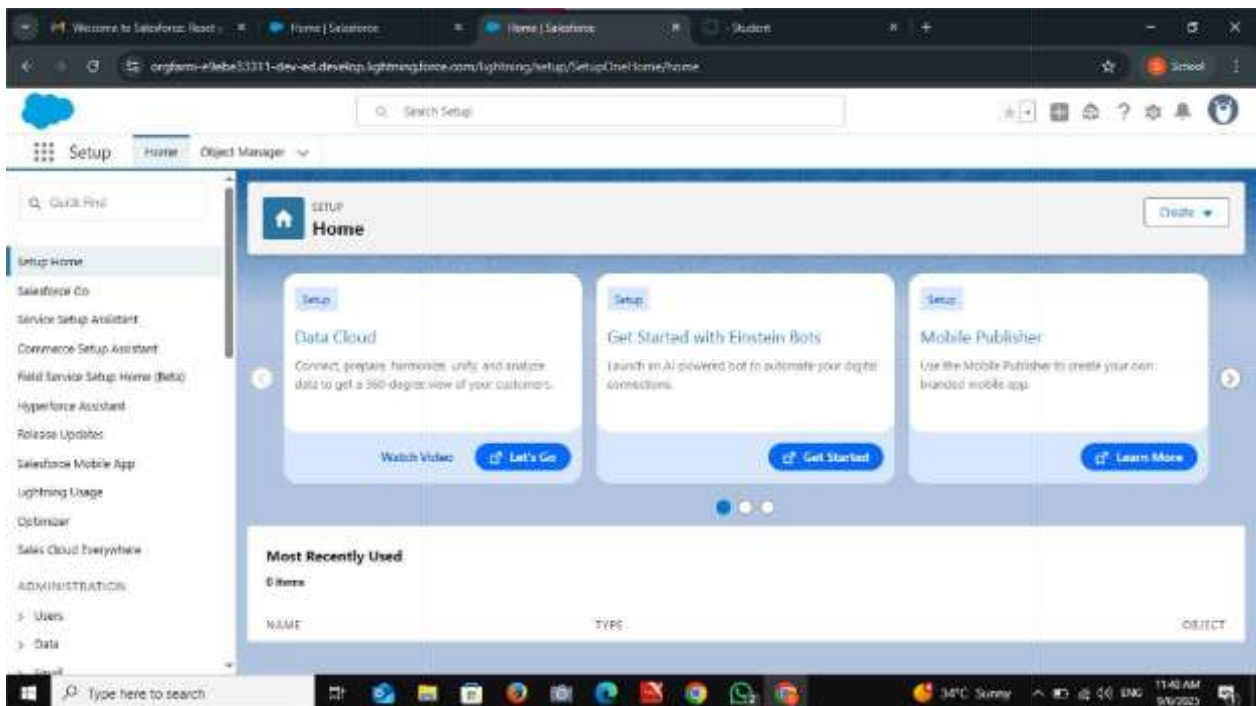
DEVELOPMENT PHASE

- Creating A Developer Account:

By using this URL- <https://www.salesforce.com/form/developer-signup/?d=pb>

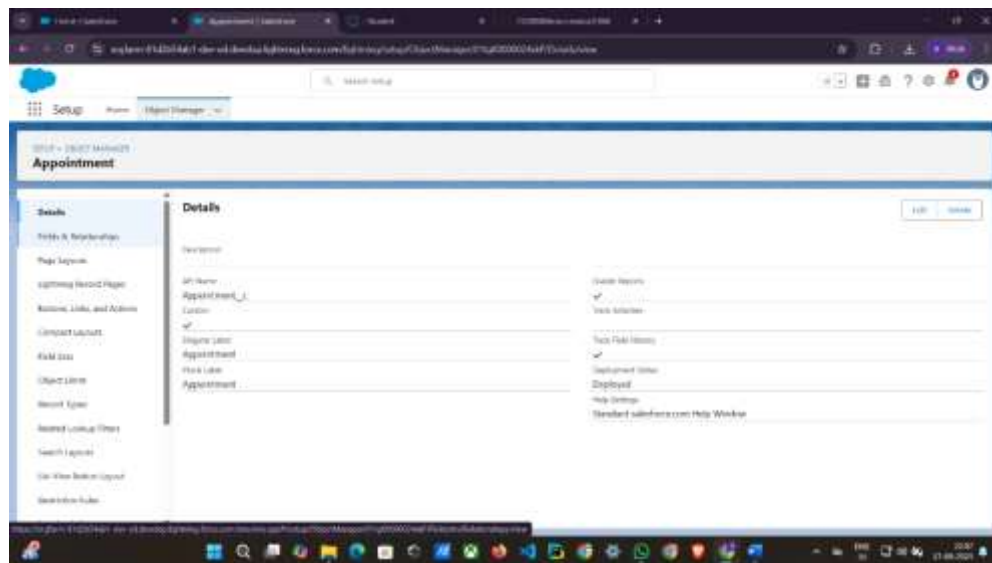
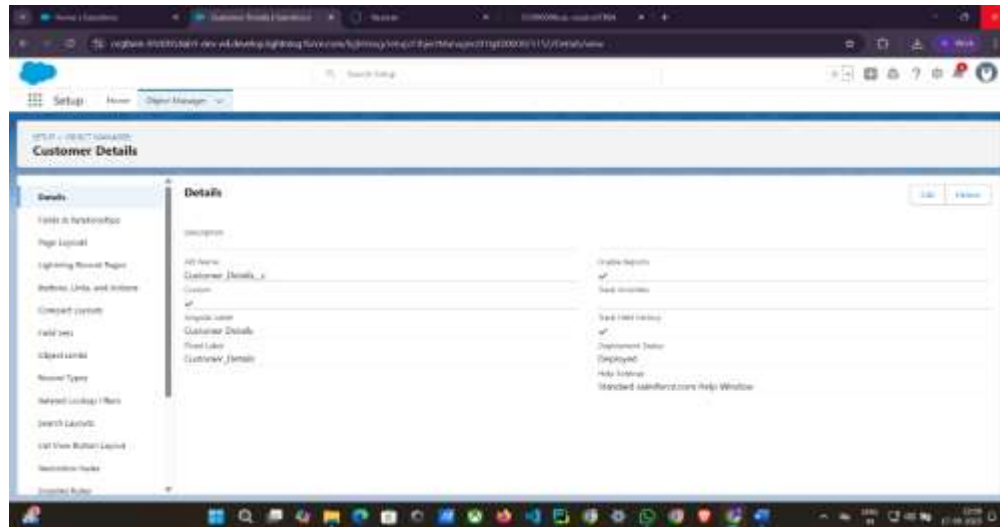


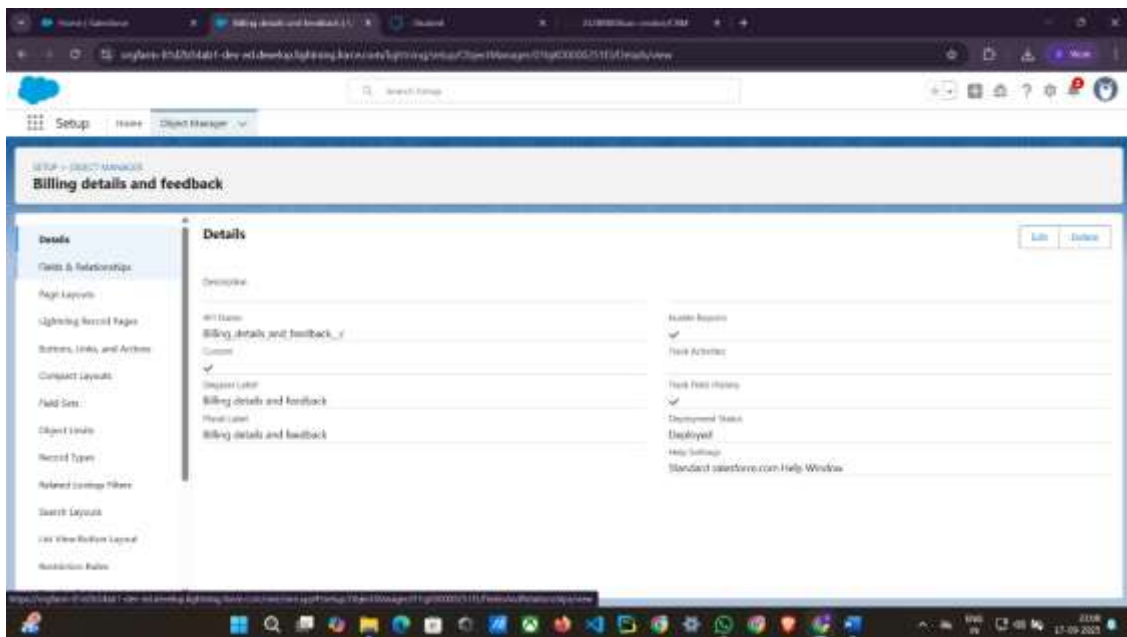
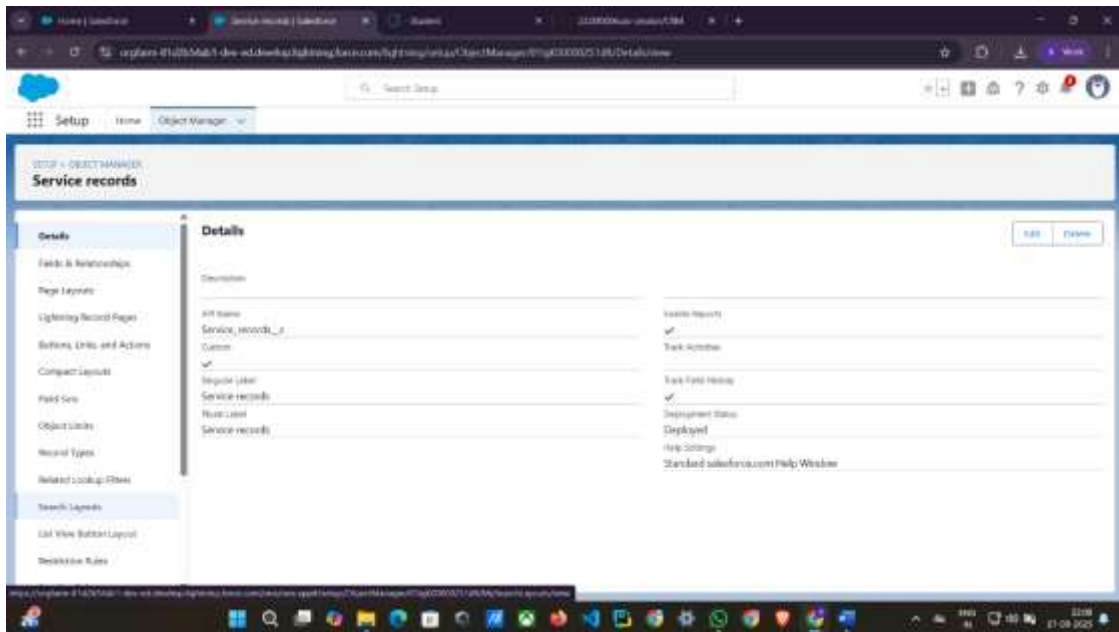
Account Activation:



Object

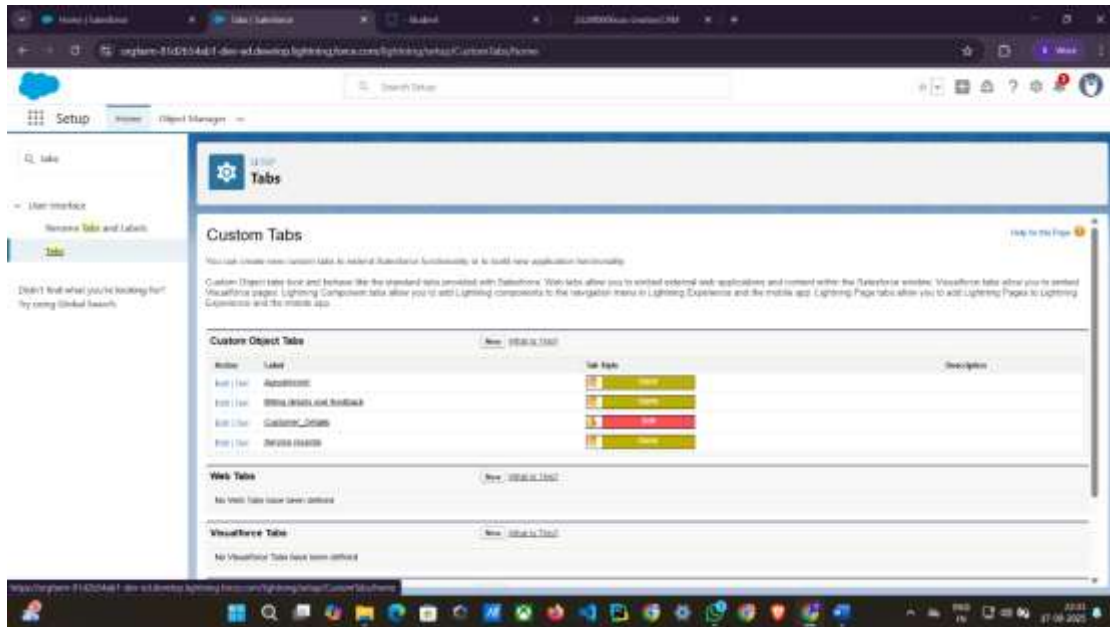
- Created objects: Customer Details, Appointment, Service records, Billing details and feedback



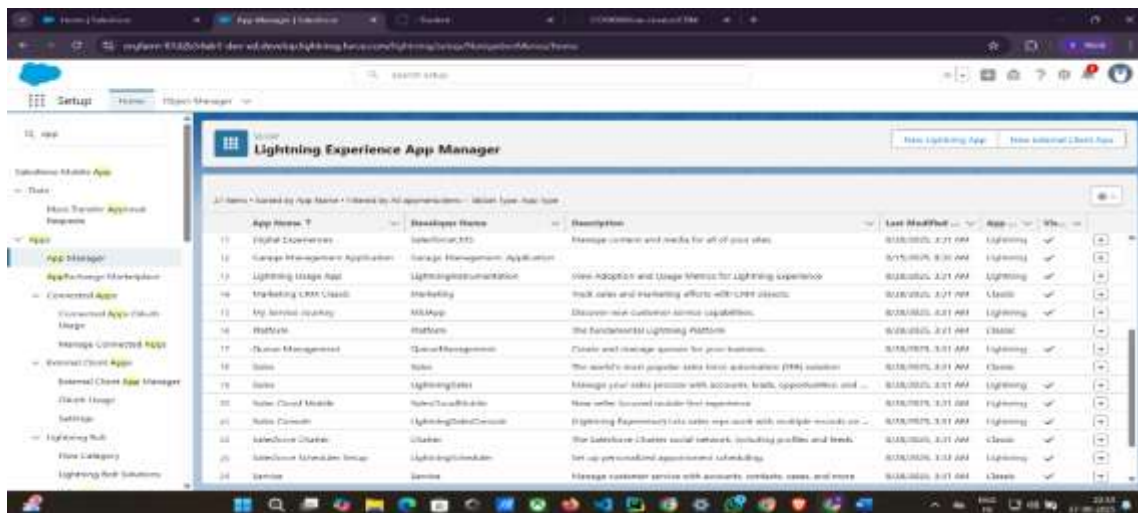


Tabs

- Created tabs: Custom Tab, Remaining Tabs

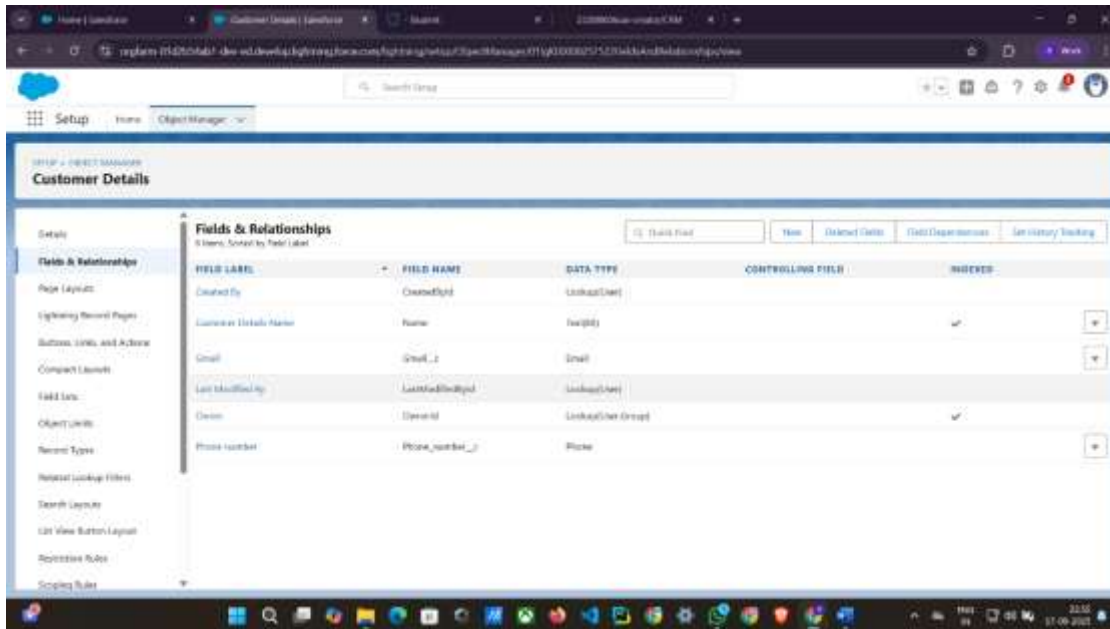


Developed Lightning App with relevant tabs

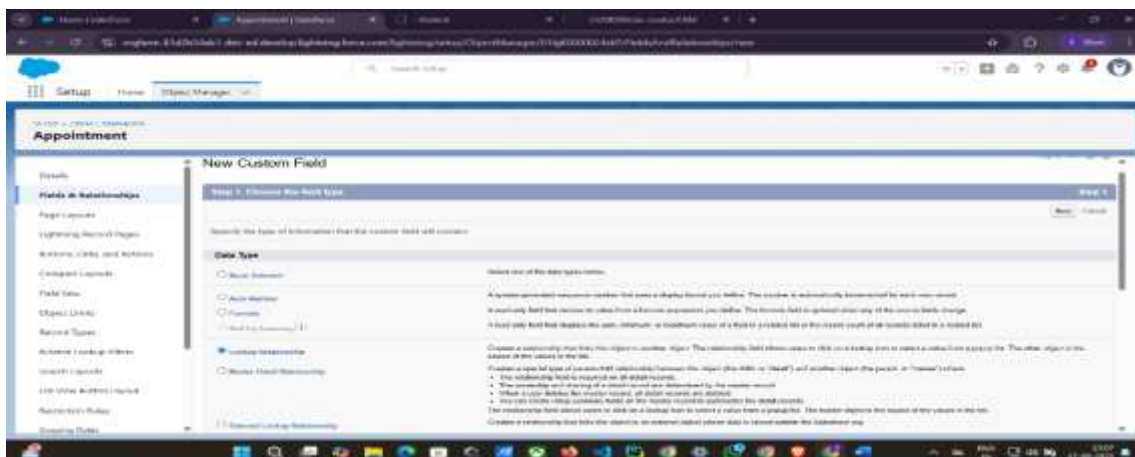


Fields

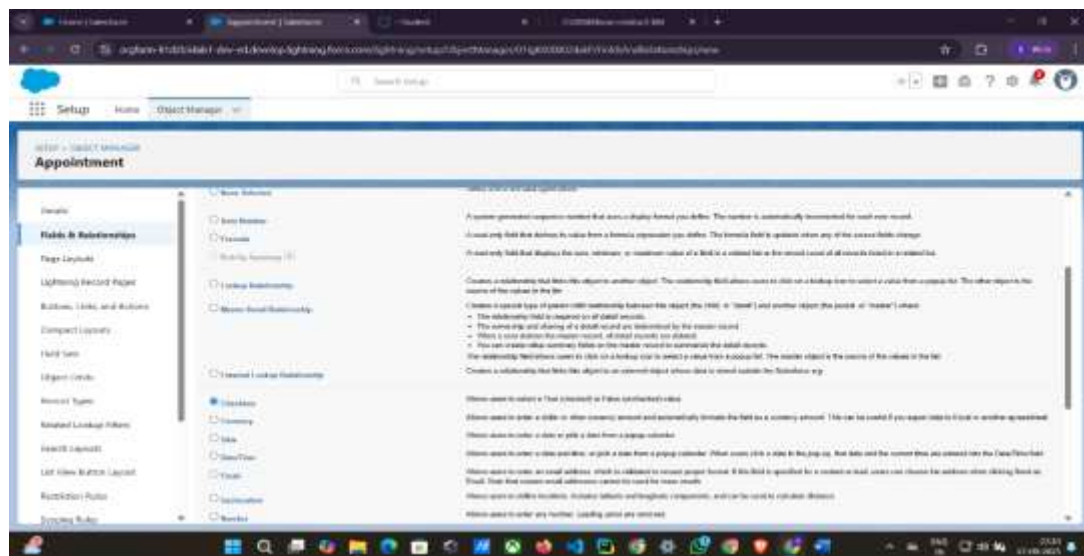
- Fields: Customer Details object



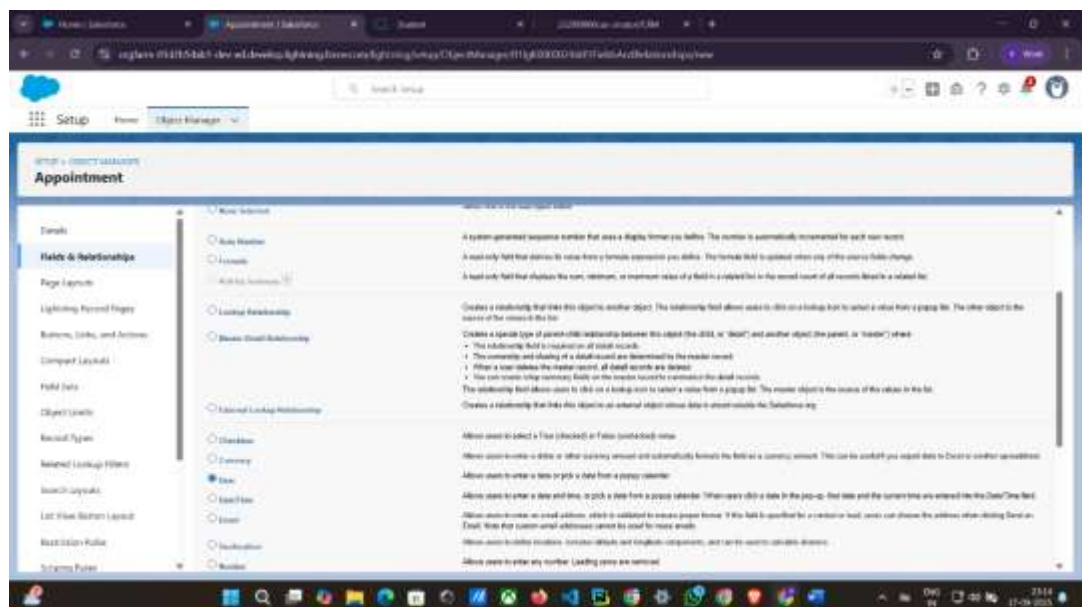
- Lookup Fields



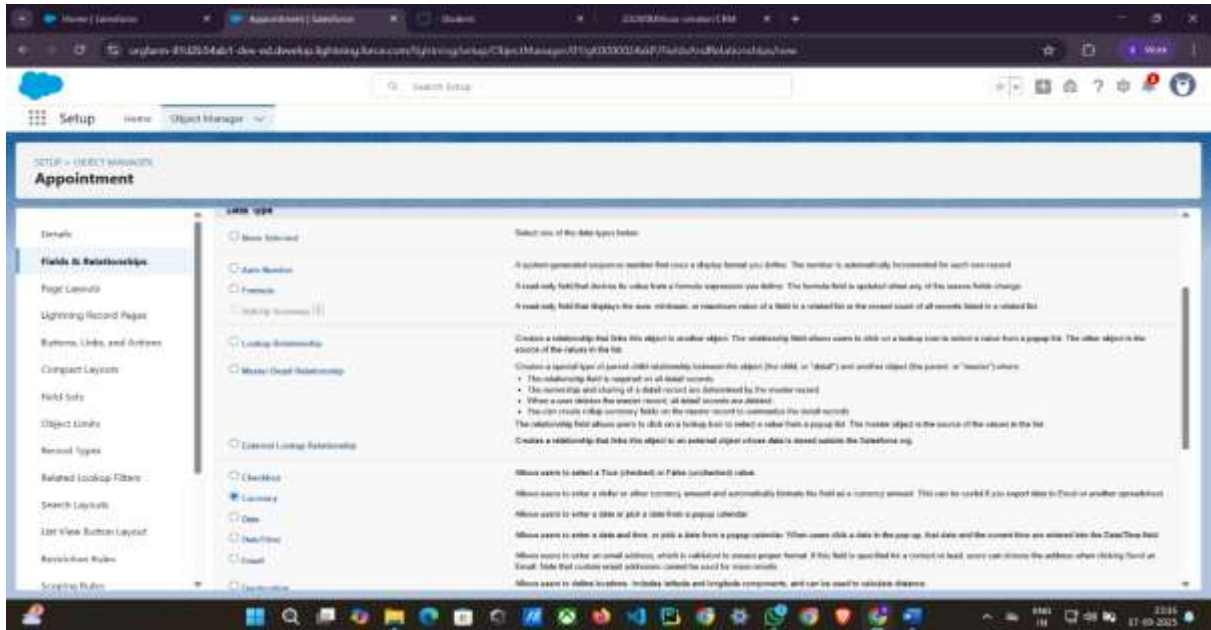
- Checkbox Fields



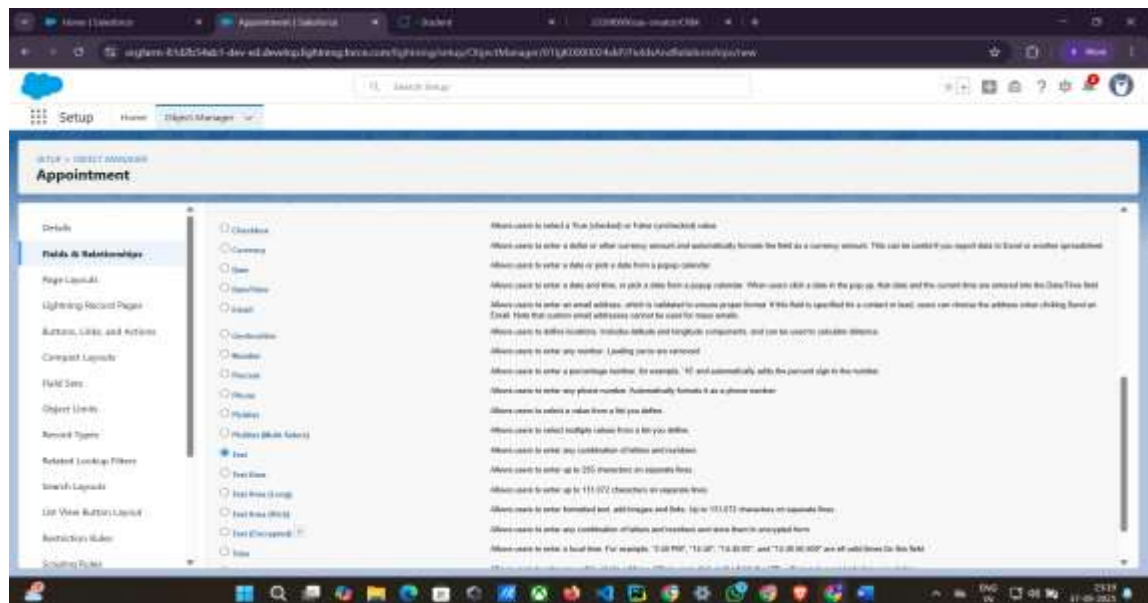
- Date Fields



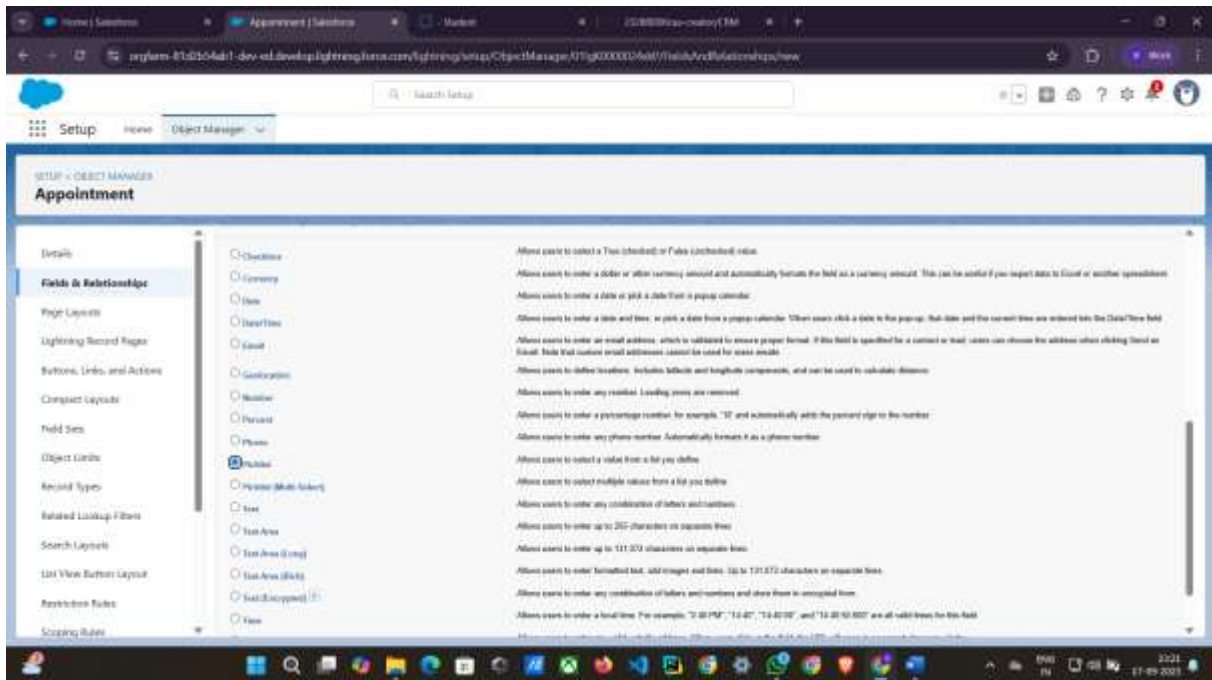
- Currency Field



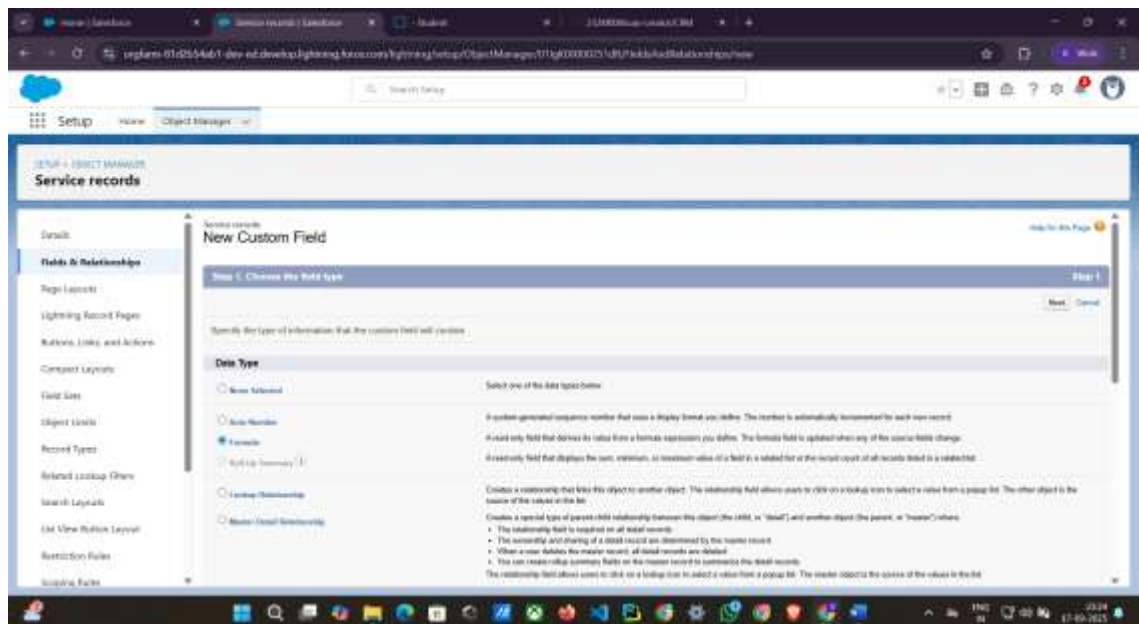
- Text Fields



- Picklist Fields

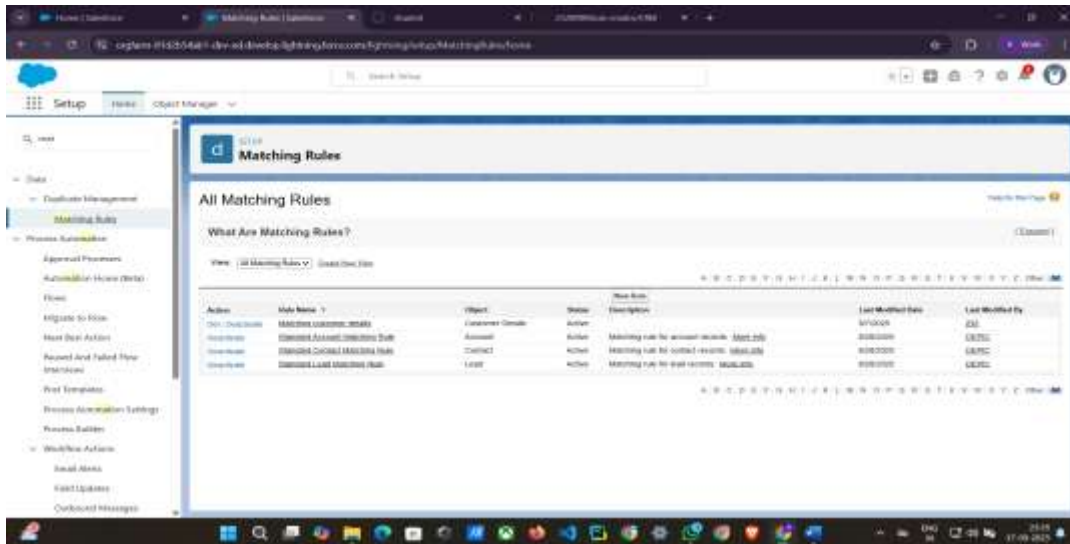


- Formula Field in Service records Object

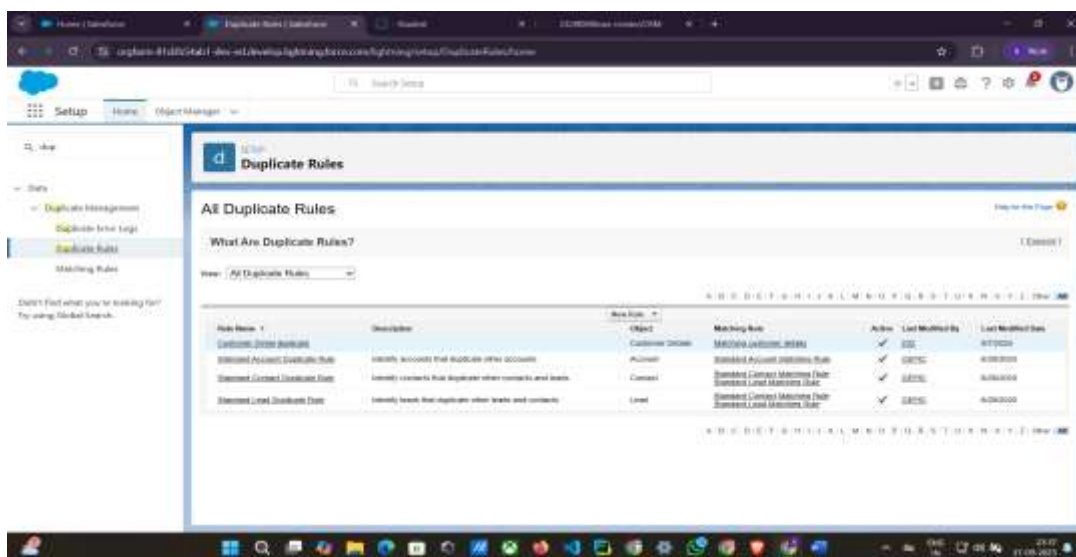


Validation rule

- Matching rule to an Customer details Object

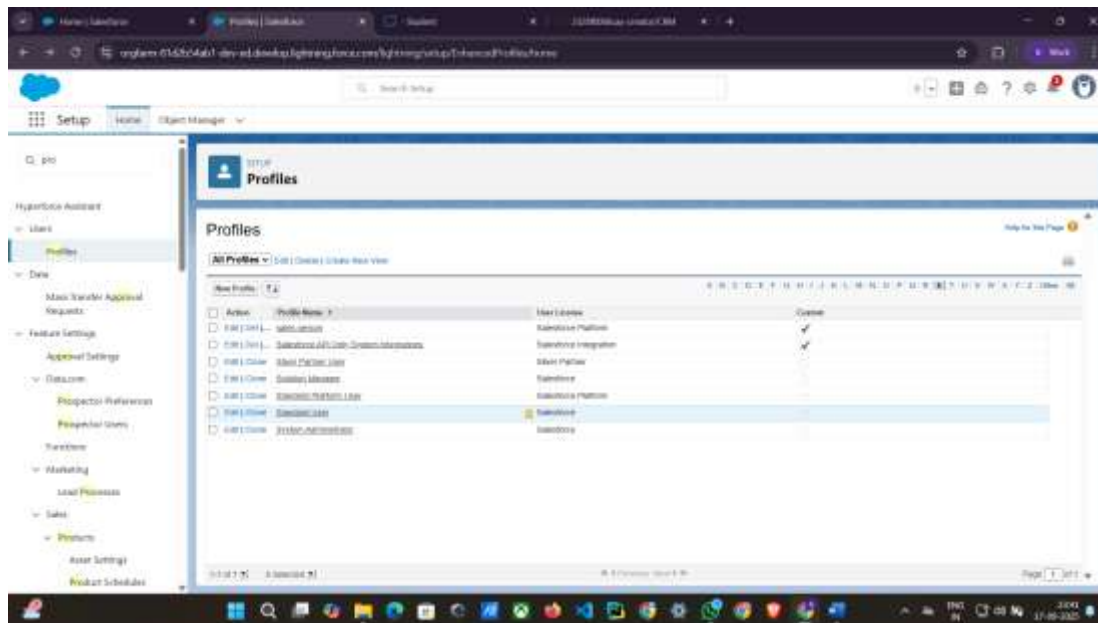


- Duplicate rule to an Customer details Object



Profiles

- Manager Profile



- sales person Profile

[illegible]

Sales person Profile

- Manager Role and another roles

Setup Roles

Understanding Roles

Set up your Role Hierarchy to control how your organization reports on and accesses data.

Sample Role Hierarchy

View other sample Role Hierarchies: Territory-based Sample

Executive Staff
CEO, President, CFO, VP Sales

Western Sales Director
Sales Rep

Eastern Sales Director
Sales Rep

International Sales Director
Sales Rep

Set up Roles

Don't show this page again

Your Organization's Role Hierarchy

Collapse All Expand All

Nick Enterprises

Add Role

CFO Edit Del Assign

Add Role

HR Edit Del Assign

Add Role

Manager Edit Del Assign

Add Role

On Site Emp Edit Del Assign

Add Role

Remote Emp Edit Del Assign

Add Role

SVP Customer Service & Support Edit Del Assign

[Collapse All](#) [Expand All](#)

 Thesmartbridge

Add Role

CEO Edit | Del | Assign

Add Role

 **CFO** Edit | Del | Assign

Add Role

 COO Edit | Del | Assign

Add Role

 **Manger** Edit | Del | Assign

[Add Role](#)

SVP, Customer Service & Support Edit | Del | Assign

Add Role

 SVP, Human Resources Edit | Del | Assign

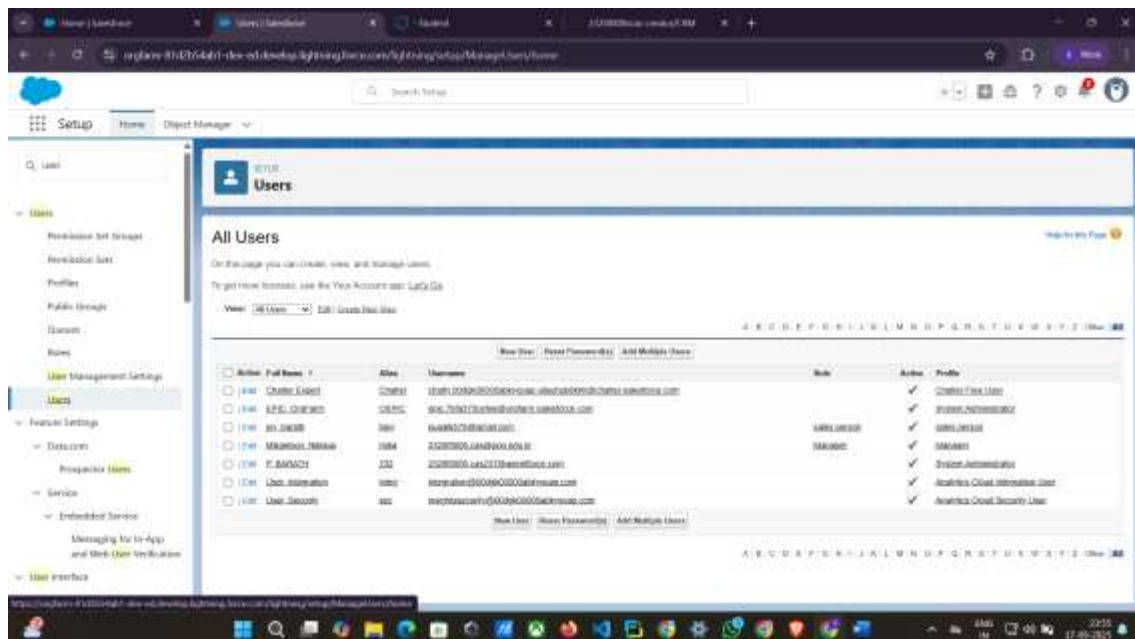
Add Role

 SVP, Sales & Marketing [Edit](#) | [Del](#) | [Assign](#)

Add Role

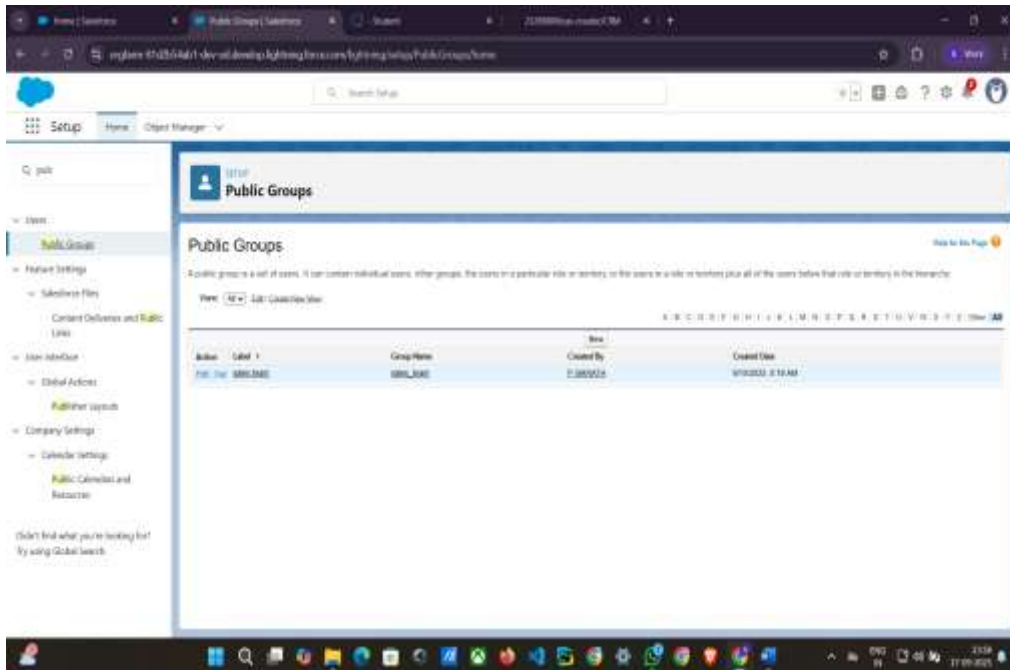
Users

- User and another user

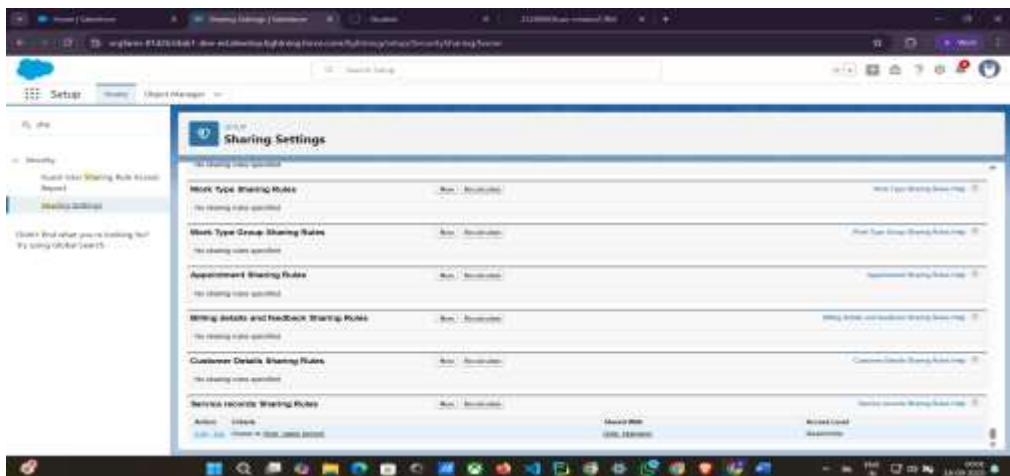


Public groups

- New Public Group

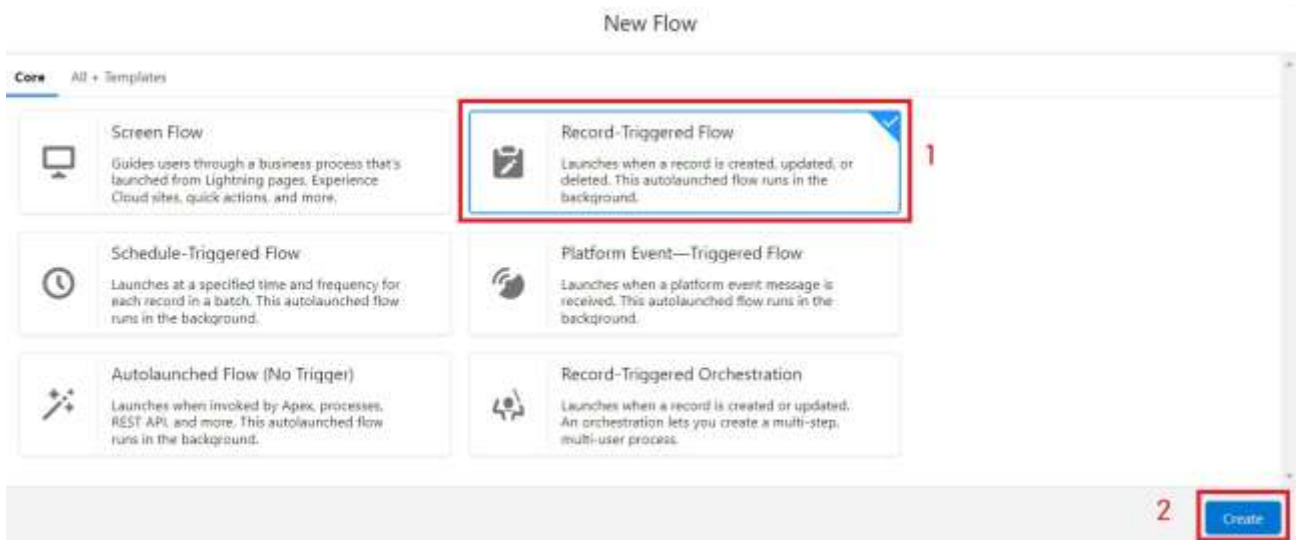
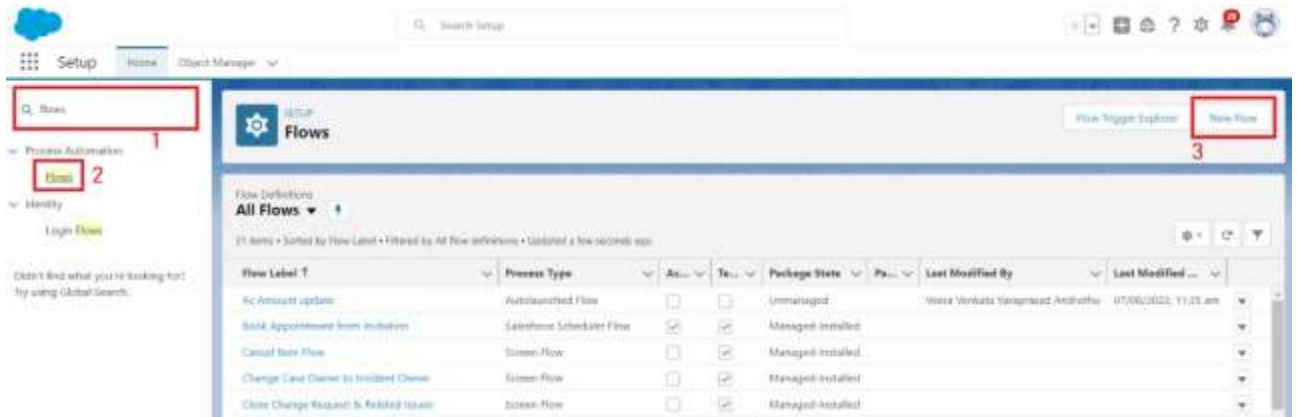


Sharing Setting



Flows

- flow and another flow



Configure Start

Select Object

Select the object whose records trigger the flow when they're created, updated, or deleted.

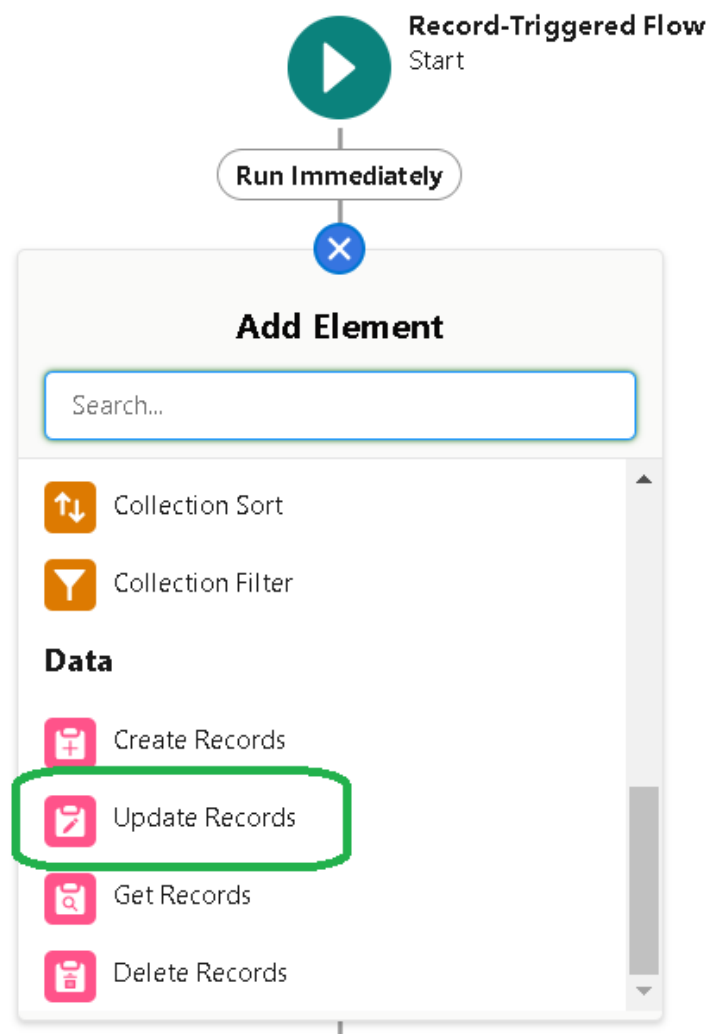
*Object

Billing details and feedback

Configure Trigger

*Trigger the Flow When:

- ☐ A record is created
- ☐ A record is updated
- ☒ A record is created or updated
- ☐ A record is deleted



Edit Update Records

Update Salesforce records using values from the flow.

*Label	*API Name
Amount Update	Amount_Update

Description

***How to Find Records to Update and Set Their Values**

- ☒ Use the billing details and feedback record that triggered the flow
- ☐ Update records related to the billing details and feedback record that triggered the flow
- ☐ Use the IDs and all field values from a record or record collection
- ☐ Specify conditions to identify records, and set fields individually

Set Filter Conditions

Condition Requirements to Update Record

All Conditions Are Met (AND) ▼

Cancel Done

Set Filter Conditions

Condition Requirements to Update Record

All Conditions Are Met (AND) ▼

Field	Operator	Value
Payment_Status__c	Equals ▼	Completed

+ Add Condition

Set Field Values for the Billing details and feedback Record

Field	Value
Payment_Paid__c	← \$Record > Service records > Appointment > Service A... X

+ Add Field

Cancel Done

Apex Trigger

- Apex handler

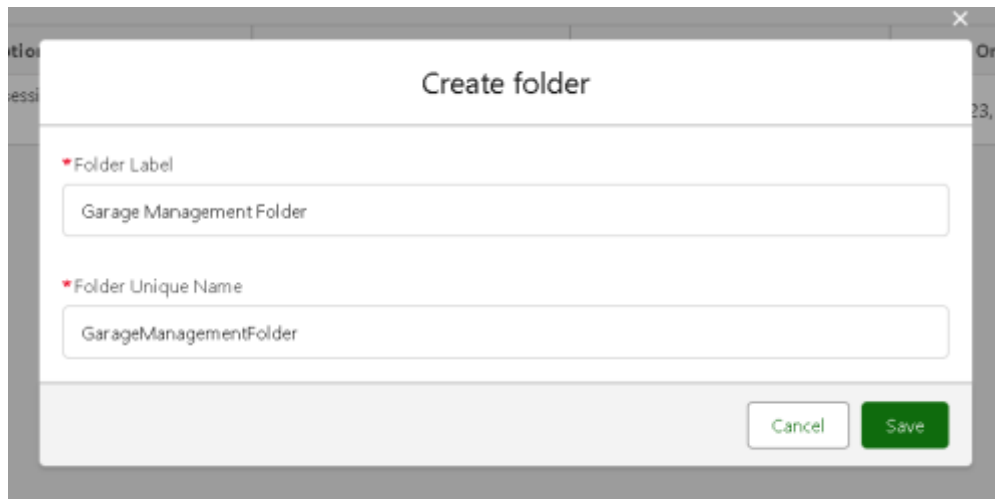
```
AmountDistribution.apex - AmountDistributionHandler.apex
Code Coverage: None | API Version: 58
12 }
13 * else if(app.Maintenance_service__c == true && app.Replacement_Parts__c == true){
14     app.Service_Amount__c = 8000;
15 }
16 * else if(app.Repairs__c == true && app.Replacement_Parts__c == true){
17     app.Service_Amount__c = 7000;
18 }
19 * else if(app.Maintenance_service__c == true){
20     app.Service_Amount__c = 2000;
21 }
22 * else if(app.Repairs__c == true){
23     app.Service_Amount__c = 3000;
24 }
25 * else if(app.Replacement_Parts__c == true){
26     app.Service_Amount__c = 5000;
27 }
28
29 }
30 }
31 }
```

```
AmountDistribution.apex - AmountDistributionHandler.apex
Code Coverage: None | API Version: 58
1 * public class AmountDistributionHandler {
2
3 *     public static void amountDist(list<Appointment__c> listApp){
4         list<Service_records__c> serList = new list<Service_records__c>();
5
6 *         for(Appointment__c app : listApp){
7 *             if(app.Maintenance_service__c == true && app.Repairs__c == true && app.Replacement_Parts__c == true){
8                 app.Service_Amount__c = 10000;
9
10 *             else if(app.Maintenance_service__c == true && app.Repairs__c == true){
11                 app.Service_Amount__c = 5000;
12             }
13 *             else if(app.Maintenance_service__c == true && app.Replacement_Parts__c == true){
14                 app.Service_Amount__c = 8000;
15             }
16 *             else if(app.Repairs__c == true && app.Replacement_Parts__c == true){
17                 app.Service_Amount__c = 7000;
18             }
19 *             else if(app.Maintenance_service__c == true){
20                 app.Service_Amount__c = 2000;
21             }
22         }
23     }
24 }
```

```
File - Edit - Debug - Test - Workspace - Help -
AmountDistribution.apex - AmountDistributionHandler.apex
Code Coverage: None | API Version: 58
1 * trigger AmountDistribution on Appointment__c (before insert, before update) {
2
3 *     if(trigger.isbefore && trigger.isinsert || trigger.isupdate){
4         AmountDistributionHandler.amountDist(trigger.new);
5
6     }
7
8 }
```

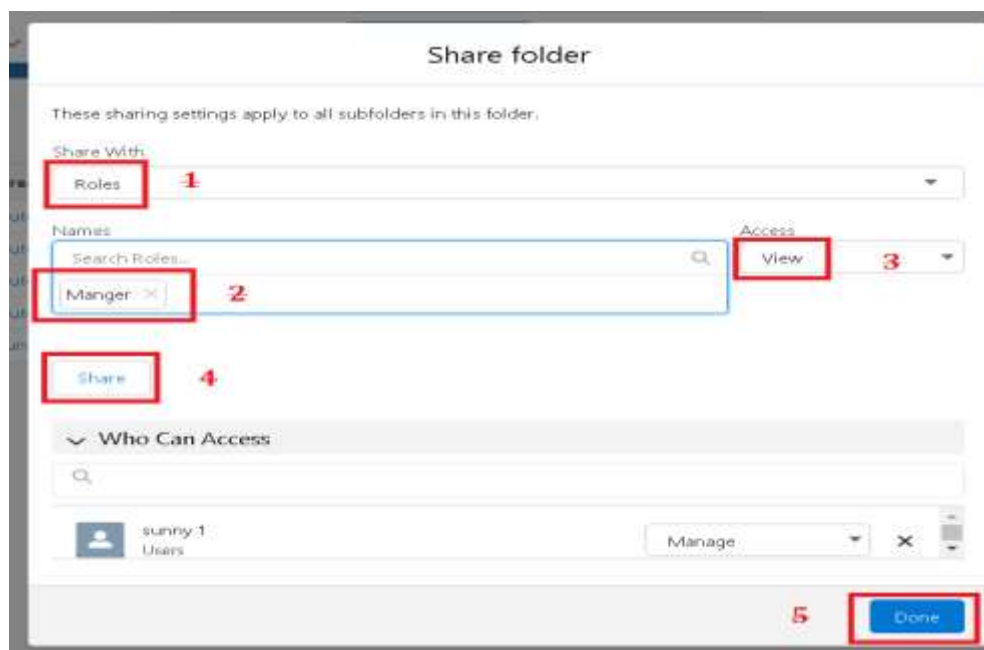
Reports

- report folder



A dialog box titled "Create folder" with a close button (X) in the top right corner. It contains two input fields: "Folder Label" with the text "Garage Management Folder" and "Folder Unique Name" with the text "GarageManagementFolder". At the bottom right are "Cancel" and "Save" buttons.

- Sharing folder



A dialog box titled "Share folder" with a close button (X) in the top right corner. It contains the following elements:

- A message: "These sharing settings apply to all subfolders in this folder."
- A "Share With" dropdown menu set to "Roles", marked with a red box and the number 1.
- A "Names" section with a search bar "Search Roles..." and a list containing "Manger" (with a red box and the number 2).
- An "Access" dropdown menu set to "View", marked with a red box and the number 3.
- A "Share" button, marked with a red box and the number 4.
- A section titled "Who Can Access" with a search bar and a list containing "sunny.1 Users".
- A "Manage" button next to the user list.
- A "Done" button at the bottom right, marked with a red box and the number 5.

- Report Type

Report Types

This report type will generate reports about Customer Details. You may define which related records from other objects are returned in report results by choosing a relationship to another object.

A Customer Details
Primary Object

B Appointments
A to B Relationship:
☒ Each "A" record must have at least one related "B" record.
☐ "A" records may or may not have related "B" records.

C Service records
B to C Relationship:
☒ Each "B" record must have at least one related "C" record.
☐ "B" records may or may not have related "C" records.

D Billing details and feedback
C to D Relationship:
☒ Each "C" record must have at least one related "D" record.
☐ "C" records may or may not have related "D" records.

Object Limit Reached
You can associate up to four objects to a custom report type.

Previous Save Cancel

- Report

Save Report

*Report Name
New Service information Report

Report Unique Name ⓘ
New_Service_information_Report_oVu

Report Description

Folder
Garage Management Folder

Select Folder

Cancel Save

Dashboards

- Dashboard Folder

New Dashboard

*

Name

Customer review

Description

Folder

Service Rating

Select Folder

Cancel

Create

Select Report

Reports

Recent

Create & by Me

Private Reports

Public Reports

All Reports

Folders

Create & by Me

Share with Me

Select Report

Search Reports and Folders

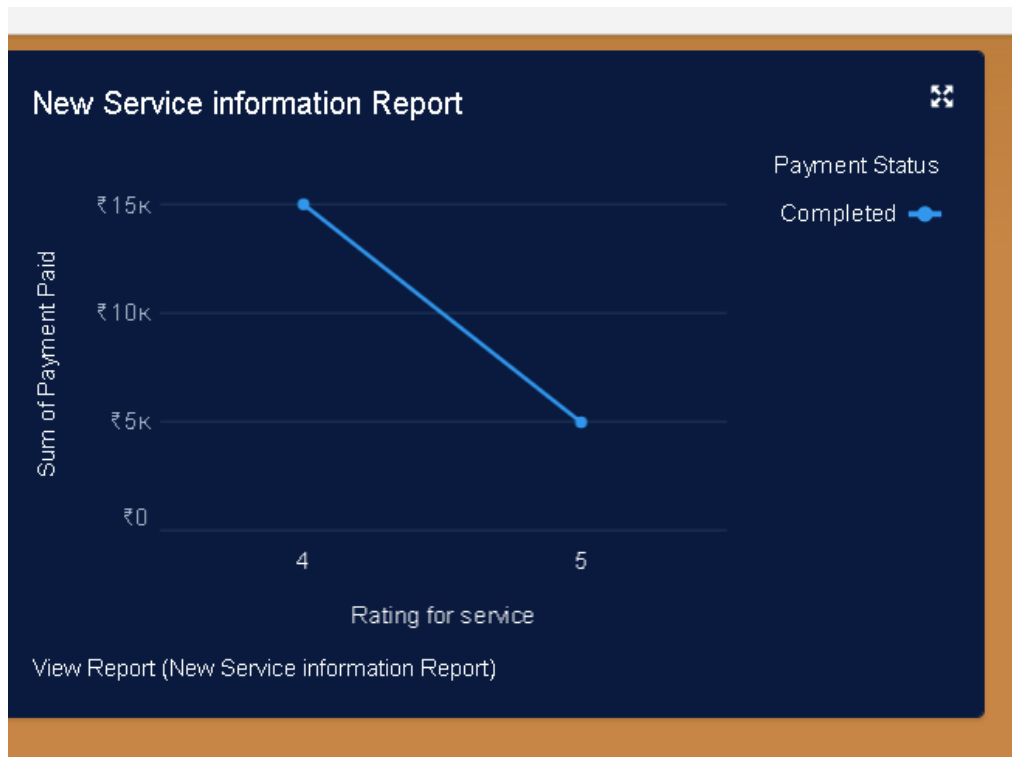
Reports and Folders

new Service Information Report
Impact 2 - 19-Oct-2023 5:25 pm - Garage Management Folder

AI Day Sessions Last 20 days
Automatic Process - 28-Aug-2023 13:08 pm - Enquire Bot Reports Summer'23

Cancel

Select



Edit Subscription

Schedule dashboard refresher and subscribe to receive results.

Settings

Frequency: ☐ Daily ☒ Weekly ☐ Monthly

Days: ☐ Sun ☒ Mon ☐ Tue ☐ Wed ☐ Thu ☐ Fri ☐ Sat

Time:

Recipients

☒ Receive new results by email when dashboard is refreshed.

Send email to: Me

User Adoption

- records

New Customer Detail

* = Required Information

Information

* Customer Name

Mac


Phone number

5678765567

Gmail

mac@gmail.com

Owner

 Annapurna SmartBridge

Cancel

Save & New

Save

Garage Manage...

Customer Details

Appointments

Service records

Billing details and feedback

Reports

Dashboards


Appointment

app-016


Appointment Name

app-016

Owner

 Annapurna SmartBridge

Customer Details

 Mac

* Appointment Date

13/11/2024

Maintenance service

☒

Repairs

☒

Replacement Parts

☐

Service Amount

* Vehicle number plate

TS30EU0443

Cancel

Save

New Service record

* = Required Information

Information

Service Record Name

Owner

* Appointment

Quality Check Status

Service Status

app-016

Started

Cancel

Save & New

Save

Service Record Name

ser-010

Owner

* Appointment

Quality Check Status

Service Status

app-016

Started

service date

18/11/2024

This field is calculated upon save

Created By

Cancel

Save

Created By

Related

Details

Service Record Name

ser-010

Owner

 [Annapurna SmartBridge](#)



Appointment

[app-016](#)



Quality Check Status



Service Status

Completed



service date

18/11/2024

Created By

 [Annapurna SmartBridge](#), 18/11/2024, 4:32 pm

Last Modified By

 [Annapurna SmartBridge](#), 18/11/2024, 4:34 pm

CONCLUSION:

The Garage Management System successfully streamlines the daily operations of automobile service centers by digitizing customer management, vehicle tracking, service scheduling, inventory control, and billing processes. It reduces manual effort, minimizes errors, and ensures efficient workflow for both staff and management. By providing accurate records, timely reminders, and quick service delivery, the system improves overall productivity and enhances customer satisfaction. This project demonstrates how automation in garages can lead to better resource utilization, cost savings, and long-term business growth.

APPENDIX:

SOURCE CODE:

```
public class AmountDistributionHandler {

    public static void amountDist(list<Appointment__c> listApp){
        list<Service_records__c> serList = new list <Service_records__c>();

        for(Appointment__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;
            }
        }
    }
}
```

Trigger Handler:

Code:

```
trigger AmountDistribution on Appointment__c (before insert, before update) {  
  
    if(trigger.isbefore && trigger.isinsert || trigger.isupdate){  
        AmountDistributionHandler.amountDist(trigger.new);  
    }  
  
}
```