GARAGE MANAGEMENT SYSTEM

By

Mayuri Verma

vermakrishna7525@gmail.com

Abstract

The **Garage Management System (GMS)** is designed to streamline operations within auto repair shops, helping businesses manage customer data, appointments, services, billing, and feedback efficiently. By leveraging Salesforce's powerful platform, GMS ensures a seamless workflow and enhances customer satisfaction by providing easy access to essential information, automating processes, and improving communication. The system includes custom objects like Customer Details, Appointments, Service Records, and Billing Feedback, along with automated flows, validation rules, and reports. The integration of Salesforce's robust features ensures real-time updates, accurate billing, and automated alerts, making GMS a valuable tool for any garage looking to optimize its services and customer interactions. This project highlights the importance of digital transformation in garage management by improving efficiency and customer experience.

INDEX

INTRODUCTION

The Garage Management System (GMS) is a helpful tool for auto repair shops. It makes it easier for them to provide great service, work more efficiently, and keep customers happy. With an easy-to-use interface and strong features, GMS helps garages succeed in a competitive market and ensures a smooth experience for both customers and employees. This project is built using Salesforce.

Task 1: Object Creation

So for this particular project we creates Four Objects named

1. Customer DetailsObject
2. Appointment Object
3. Service records Object
4. Billing details and feedback Object

Steps for creating the Customer Details object:-

To create an object:

1. From the setup page >> Click on Object Manager >> Click on Create >> Click on Custom Object.
    1. Enter the label name >> Customer Details
    2. Plural label name >> Customer Details
    3. Enter Record Name Label and Format
        Record Name >> Customer Name
        Data Type >> Text
2. Click on Allow reports and Track Field History,
3. Allow search >> Save.

Steps for Creating the Appointment Object:-

1. From the setup page >> Click on Object Manager >> Click on Create >> Click on Custom Object.
    1. Enter the label name >> Appointment
    2. Plural label name >> Appointments
    3. Enter Record Name Label and Format
        Record Name >> Appointment Name
        Data Type >> Auto Number
        Display Format >> app-{000}
        Starting number >> 1
2. Click on Allow reports and Track Field History,
3. Allow search >> Save.

Steps for creating Service Record Object:-

1. From the setup page >> Click on Object Manager >> Click on Create >> Click on Custom Object.
    1. Enter the label name >> Service records
    2. Plural label name >> Service records
    3. Enter Record Name Label and Format
        Record Name >>Service records Name

Data Type >> Auto Number
Display Format >> ser-{000}
Starting number >> 1

2. Click on Allow reports and Track Field History,
3. Allow search >> Save.

Steps for Creating Billing details and feedback Object

1. From the setup page >> Click on Object Manager >> Click on Create >> Click on Custom Object.
   1. Enter the label name >> Billing details and feedback
   2. Plural label name >> Billing details and feedback
   3. Enter Record Name Label and Format
      Record Name >> Billing details and feedback Name
      Data Type >> Auto Number
      Display Format >> bill-{000}
      Starting number >> 1
2. Click on Allow reports and Track Field History,
3. Allow search >> Save.

Task 2: Tabs

In this project we created 4 tabs for each objects.

Steps for creating Tab:(Customer Details)

1. Go to setup page >> type Tabs in Quick Find bar >> click on tabs >> New (under custom object tab)
2. Select Object (Customer Details) >> Select the tab style >> Next (Add to profiles page) keep it as default >> Next (Add to Custom App) uncheck the include tab.
3. Make sure that the Append tab to users' existing personal customizations is checked.
4. Click save.

Similar for other objects.

Task 3: Lightning App

For this project we created a Lightning App page named- "Garage Management Application".

Steps for Creating Lightning App: -

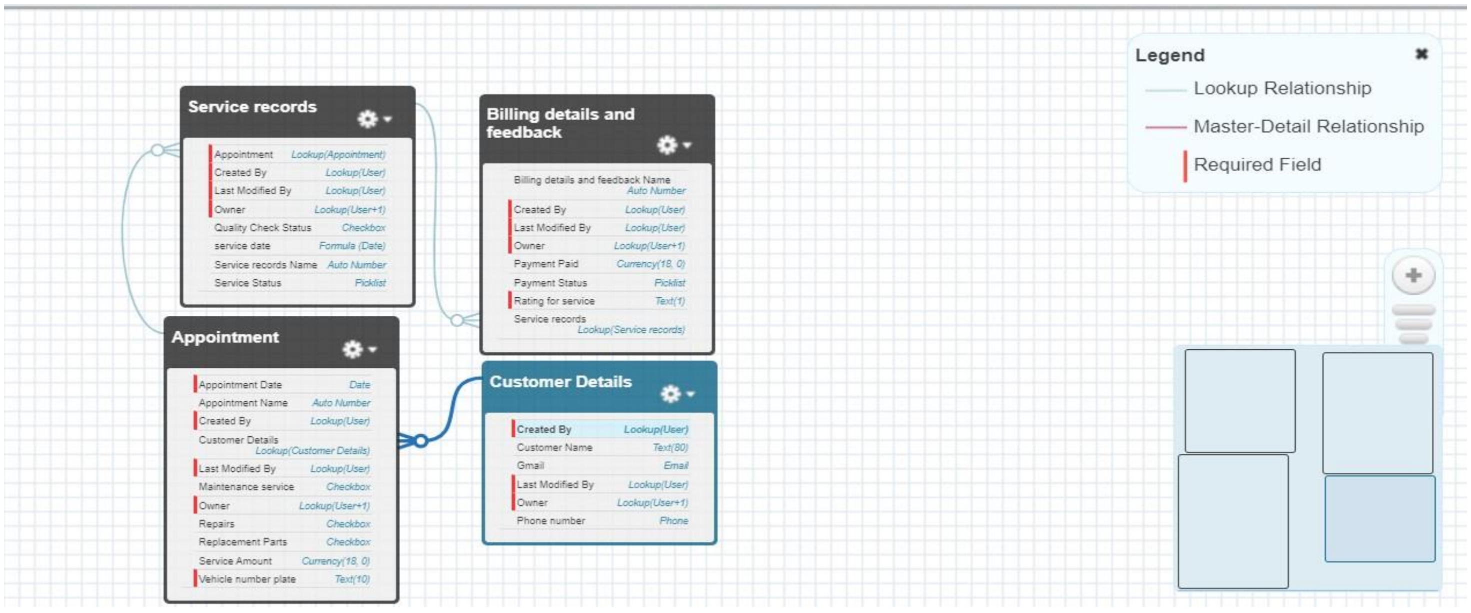
1. Go to setup page >> search “app manager” in quick find >> select “app manager” >> click on New lightning App.
2. Fill the app name in app details as Garage Management Application >> Next >> (App option page) keep it as default >> Next >> (Utility Items) keep it as default >> Next.
3. To Add Navigation Items:
4. Select the items (Customer Details, Appointments, Service records, Billing details and feedback, Reports and Dashboards) from the search bar and move it using the arrow button >> Next.
5. To Add User Profiles: Search profiles (System administrator) in the search bar >> click on the arrow button >> save & finish.

Task 4: Fields

Steps for creating fields: -

1. Go to setup >> click on Object Manager >> type object name(Customer Details) in search bar >> click on the object.
2. Now click on “Fields & Relationships” >> New
3. Select Data Type as a “Phone”
4. Click on next.
5. Fill the Above as following: • Field Label: Phone number • Field Name: gets auto generated • Click on Next >> Next >> Save and new.

For this project many types of field here is the image of all fields.



Service records

Appointment	Lookup(Appointment)
Created By	Lookup(User)
Last Modified By	Lookup(User)
Owner	Lookup(User+1)
Quality Check Status	Checkbox
service date	Formula (Date)
Service records Name	Auto Number
Service Status	Picklist

Billing details and feedback

Billing details and feedback Name	Auto Number
Created By	Lookup(User)
Last Modified By	Lookup(User)
Owner	Lookup(User+1)
Payment Paid	Currency(18, 0)
Payment Status	Picklist
Rating for service	Text(1)
Service records	Lookup(Service records)

Appointment

Appointment Date	Date
Appointment Name	Auto Number
Created By	Lookup(User)
Customer Details	Lookup(Customer Details)
Last Modified By	Lookup(User)
Maintenance service	Checkbox
Owner	Lookup(User+1)
Repairs	Checkbox
Replacement Parts	Checkbox
Service Amount	Currency(18, 0)
Vehicle number plate	Text(10)

Customer Details

Created By	Lookup(User)
Customer Name	Text(80)
Gmail	Email
Last Modified By	Lookup(User)
Owner	Lookup(User+1)
Phone number	Phone

Task 5: Validation Rule

In this project we created 3 Validation Rule.

Steps for creating Validation rule: -

1. Go to the setup page >> click on object manager >> From drop down click edit for Appointment object.
2. Click on the validation rule >> click New.
3. Enter the Rule name as “Vehicle”.
4. Insert the Error Condition Formula as: - NOT(REGEX(Vehicle_number_plate__c,"[A-Z]{2} [0-9]{2}[A-Z]{2} [0-9]{4}"))
5. Enter the Error Message as “Please enter vaild number”, select the Error location as Field and select the field as “Vehicle number plate”, and click Save.

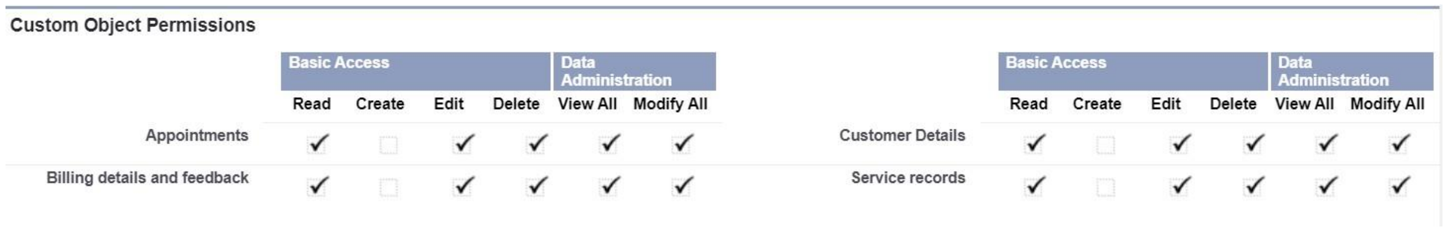
Similarly, we created 2 another validation rule for service record object and Billing details and feedback Object.

Task 6: Profile

For this project we mainly created a two profiles: - • Manager profile • Sales person profile

Steps to create a new profile:

1. Go to setup >> type profiles in quick find box >> click on profiles >> clone the desired profile (Standard User) >> enter profile name (Manager) >> Save.
2. While still on the profile page, then click Edit.
3. Select the Custom App settings as default for the Garage management.
4. Scroll down to Custom Object Permissions and Give access permissions for Appointments,Billing details and feedback , service records and customer details objects as mentioned in the below diagram.



Custom Object Permissions

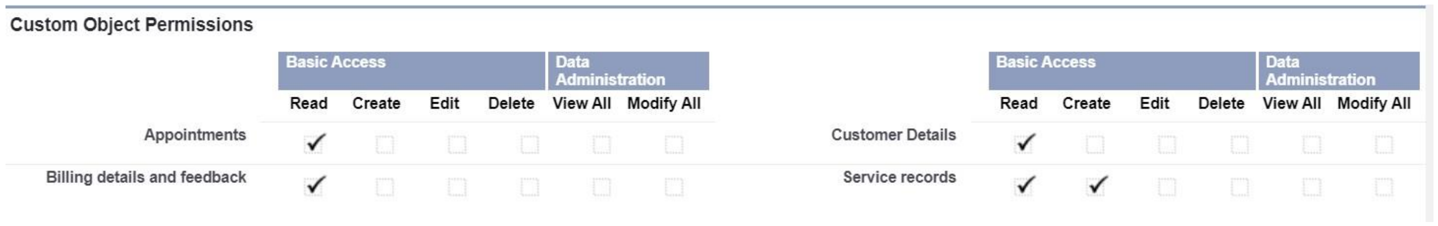
	Basic Access				Data Administration	
	Read	Create	Edit	Delete	View All	Modify All
Appointments	✓	☐	✓	✓	✓	✓
Billing details and feedback	✓	☐	✓	✓	✓	✓

	Basic Access				Data Administration	
	Read	Create	Edit	Delete	View All	Modify All
Customer Details	✓	☐	✓	✓	✓	✓
Service records	✓	☐	✓	✓	✓	✓

5. Changing the session times out after should be “ 8 hours of inactivity”.
6. Change the password policies as mentioned :
7. User passwords expire in should be “ never expires ”.
8. Minimum password length should be “ 8 ”, and click save.

Sales Person Profile: -

1. Go to setup >> type profiles in quick find box >> click on profiles >> clone the desired profile (Salesforce Platform User) >> enter profile name (sales person) >> Save.
2. While still on the profile page, then click Edit.
3. Select the Custom App settings as default for the Garage management.
4. Scroll down to Custom Object Permissions and Give access permissions for Appointments, Billing details and feedback, service records and customer details objects as mentioned in the below diagram.



Custom Object Permissions

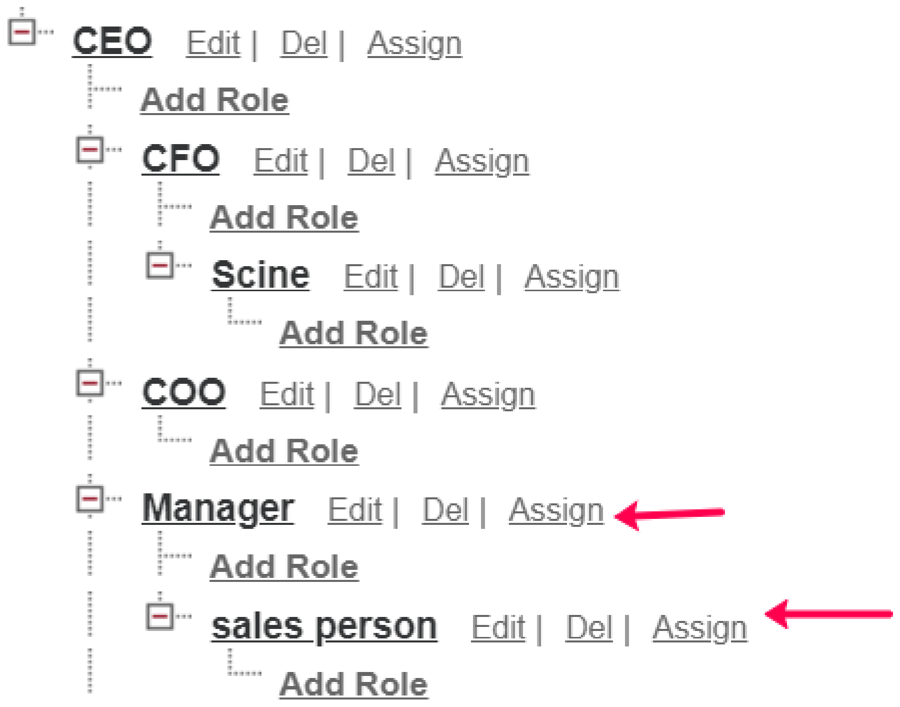
	Basic Access				Data Administration	
	Read	Create	Edit	Delete	View All	Modify All
Appointments	✓	☐	☐	☐	☐	☐
Billing details and feedback	✓	☐	☐	☐	☐	☐
Customer Details	✓	☐	☐	☐	☐	☐
Service records	✓	✓	☐	☐	☐	☐

5. And click save

Task 7: Role & Role Hierarchy

For this project we created two Roles: -

1. Manager Role Under CEO
2. Sales Person Role Under Manager

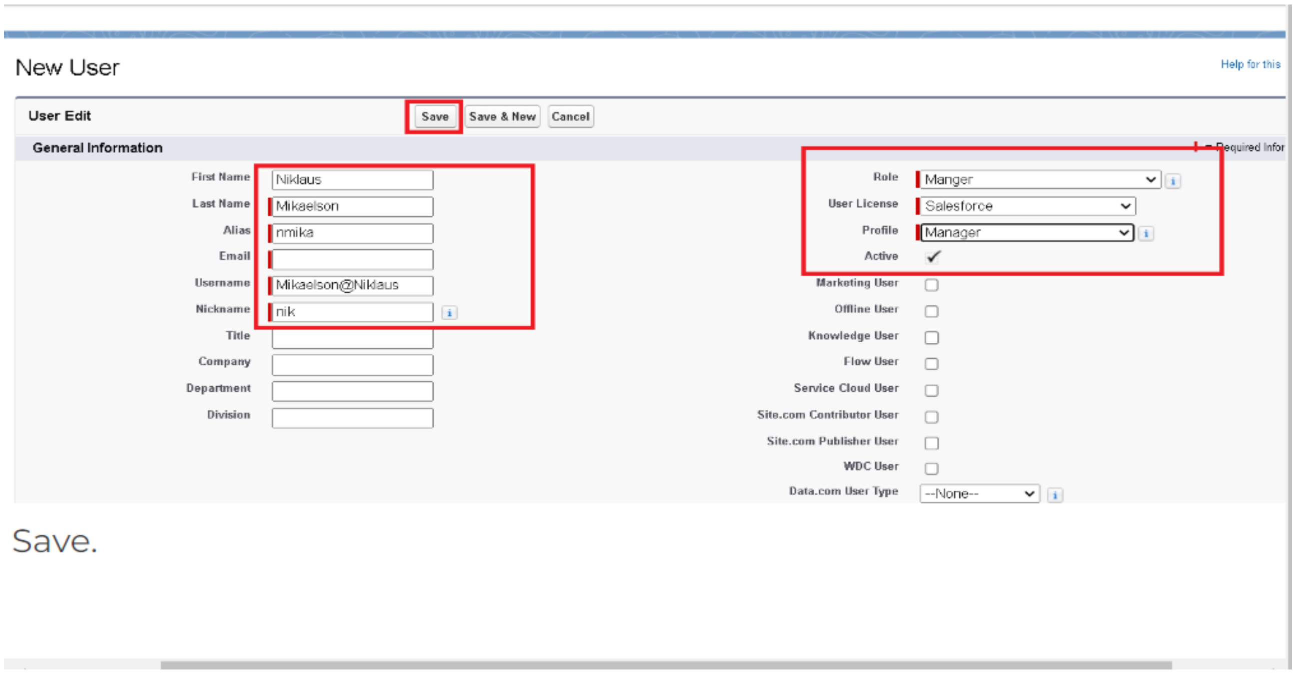


Task 8: User

Steps for creating User:

1. Go to setup >> type users in quick find box >> select users >> click New user.
2. Fill in the fields

   First Name : Niklaus
   Last Name : Mikaelson
   Alias : Give a Alias Name
   Email id : Give your Personal Email id
   Username : Username should be in this form: text@text.text
   Nick Name : Give a Nickname
   Role : Manager
   User licence : Salesforce
   Profiles : Manager



New User

Help for this

User Edit

General Information

= Required Infor

First Name	Niklaus	Role	Manger
Last Name	Mikaelson	User License	Salesforce
Alias	nmika	Profile	Manager
Email		Active	✓
Username	Mikaelson@Niklaus	Marketing User	☐
Nickname	nik	Offline User	☐
Title		Knowledge User	☐
Company		Flow User	☐
Department		Service Cloud User	☐
Division		Site.com Contributor User	☐
		Site.com Publisher User	☐
		WDC User	☐
		Data.com User Type	--None--

Save.

3. Save.

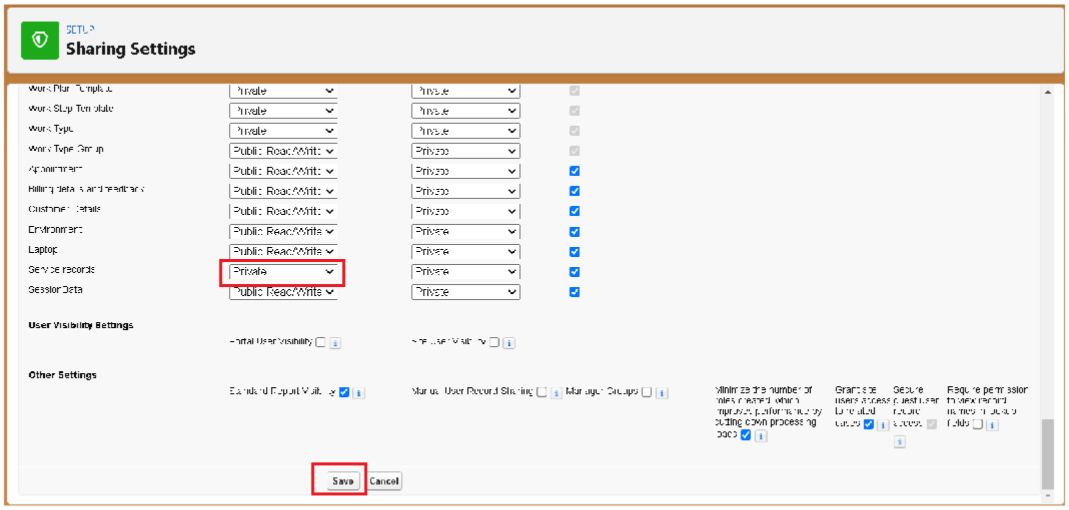
Repeat the steps and create another user using

- Role: sales person
- User licence: Salesforce Platform
- Profile: sales person

Task 9: Sharing Settings

Steps for creating Sharing setting:

1. Go to setup >> type users in quick find box >> select Sharing Settings >> click Edit.
2. Change the OWD setting of the Service records Object to private as shown in fig.



SETUP

Sharing Settings

Work Plan Template	Private	Private	✓
Work Step Template	Private	Private	✓
Work Type	Private	Private	✓
Work Type Group	Public Read/Write	Private	✓
Appointment	Public Read/Write	Private	✓
Billing details and feedbacks	Public Read/Write	Private	✓
Customer Details	Public Read/Write	Private	✓
Environment	Public Read/Write	Private	✓
Laptop	Public Read/Write	Private	✓
Service records	Private	Private	✓
Session Data	Public Read/Write	Private	✓

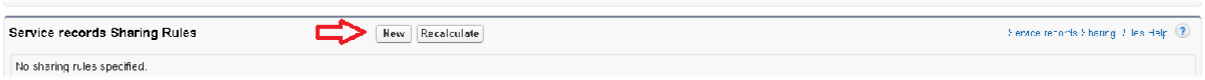
User Visibility Settings

Portal User Visibility ☐ · Site User Visibility ☐

Other Settings

Standard Report Visibility ✓ · Manual User Record Sharing ☐ · Manager Groups ☐ · Minimize the number of roles created which improves performance by cutting down processing loads ✓ · Grant site users access to related cases ✓ · Secure guest user record access ✓ · Require permission to view record names in lookup fields ☐

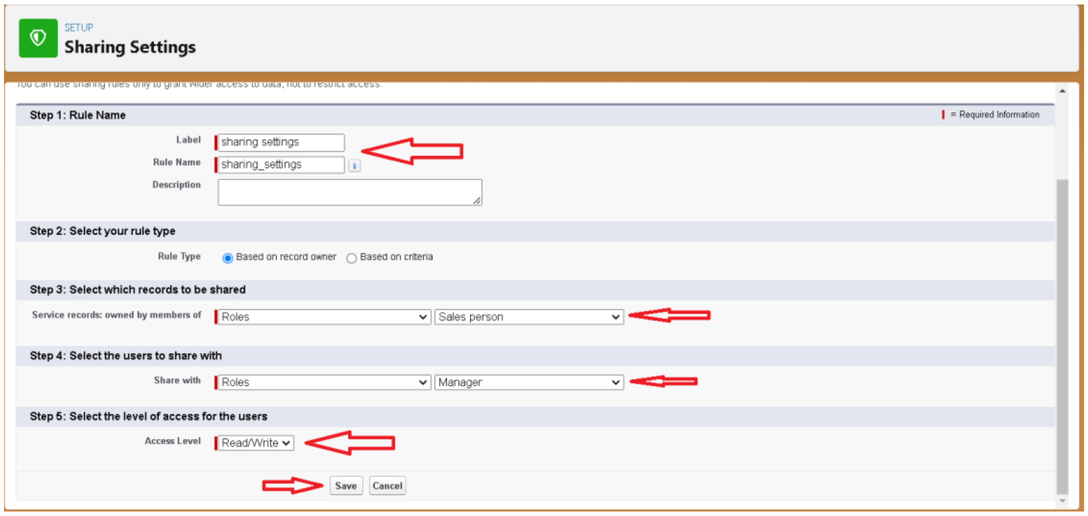
3. Click on save and refresh.
4. Scroll down a bit, Click new on Service records sharing Rules.



Service records Sharing Rules

No sharing rules specified.

6. Give the Label name as " Sharing setting"
7. Rule name is auto populated.
8. In step 3 : Select which records to be shared, members of " Roles " >> " Sales person"
9. In step 4: share with, select " Roles " >> " Manager "
10. In step 5 : Change the access level to " Read / write ".
11. Click on save.



SETUP

Sharing Settings

You can use sharing rules only to grant wider access to data, not to restrict access.

Step 1: Rule Name — = Required Information

Label: sharing settings

Rule Name: sharing_settings

Description:

Step 2: Select your rule type

Rule Type: [● Based on record owner] ○ Based on criteria

Step 3: Select which records to be shared

Service records: owned by members of: Roles — Sales person

Step 4: Select the users to share with

Share with: Roles — Manager

Step 5: Select the level of access for the users

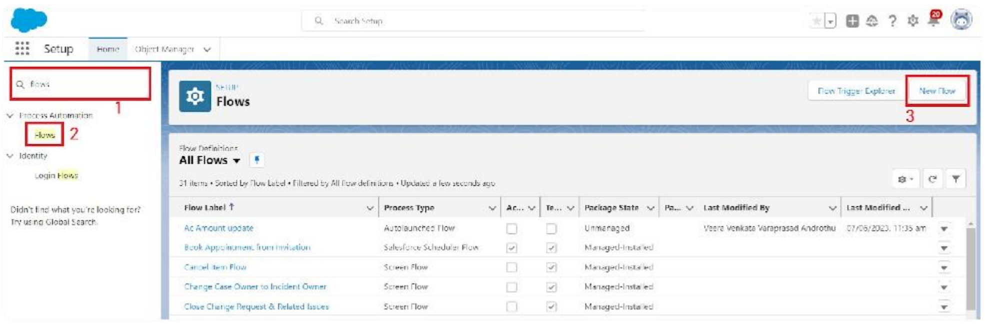
Access Level: Read/Write

Task 10: Flows

In this Project We use the Record Triggered Flow.

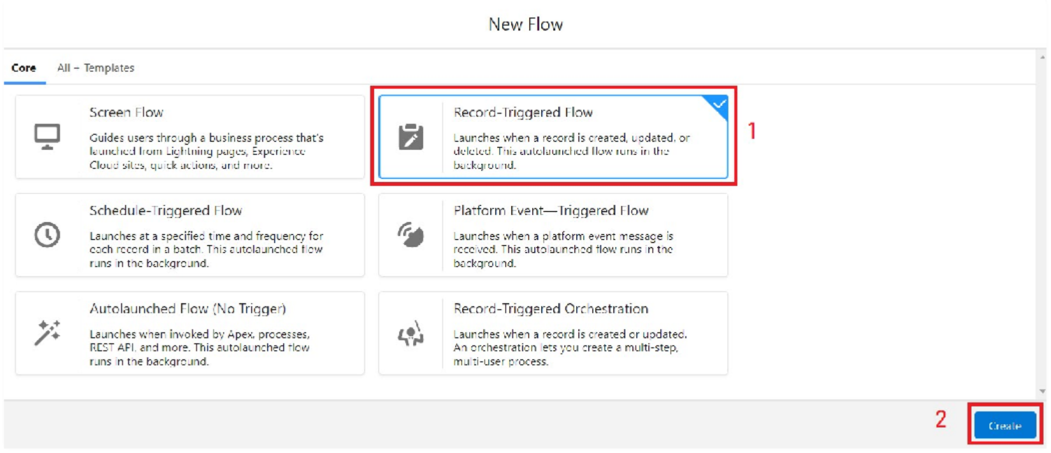
Steps for creating the Flow:-

1. Go to setup >> type Flow in quick find box >> Click on the Flow and Select the New Flow.

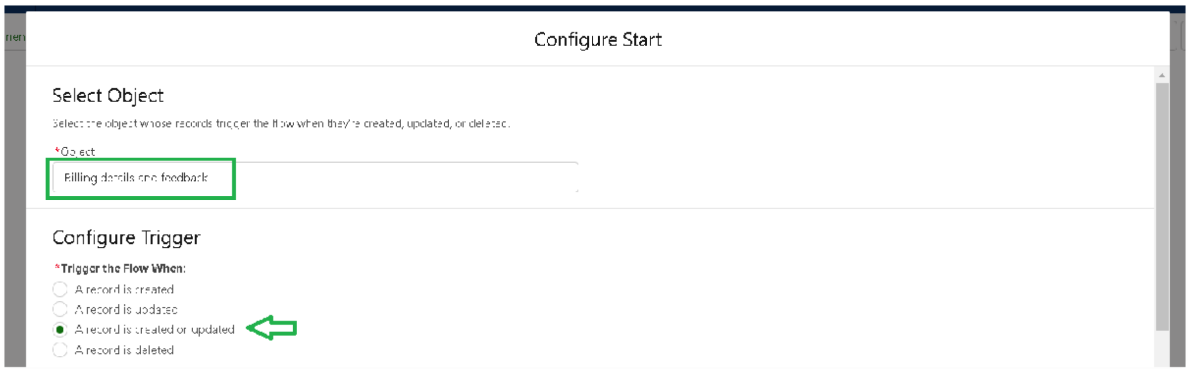

Setup
Home
Object Manager
Search Setup
flows
1
Process Automation
Flows
2
Identity
Login Flows
Didn't find what you're looking for? Try using Global Search.
SETUP
Flows
Flow Trigger Explorer
New Flow
3
Flow Definitions
All Flows
31 items • Sorted by Flow Label • Filtered by All flow definitions • Updated a few seconds ago

Flow Label	Process Type	Ac...	Te...	Package State	Pa...	Last Modified By	Last Modified ...
Ac Amount update	Autolaunched Flow	☐	☐	Unmanaged		Veera Venkata Varaprasad Androthu	07/06/2023, 11:35 am
Book Appointment from Invitation	Salesforce Scheduler Flow	✓	✓	Managed-Installed			
Cancel Item Flow	Screen Flow	☐	✓	Managed-Installed			
Change Case Owner to Incident Owner	Screen Flow	☐	✓	Managed-Installed			
Close Change Request & Related Issues	Screen Flow	☐	✓	Managed-Installed			

2. Select the Record-triggered flow and Click on Create.

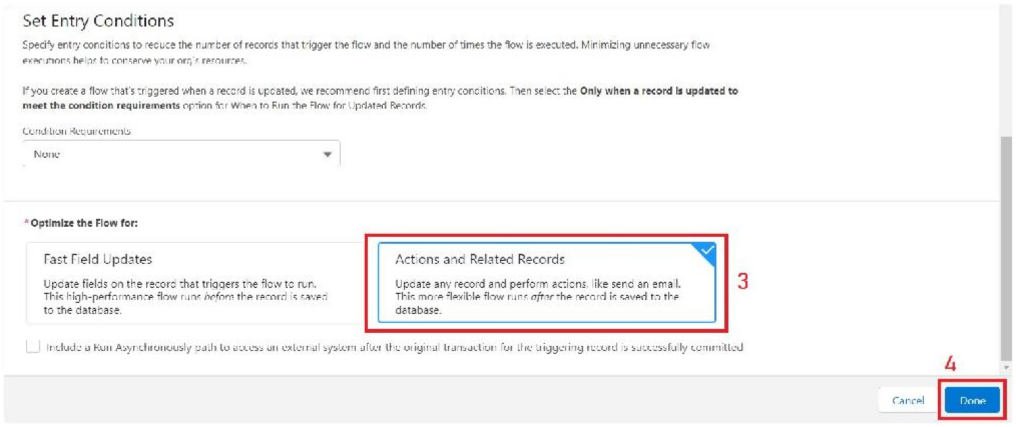

New Flow
Core
All + Templates
Screen Flow
Guides users through a business process that's launched from Lightning pages, Experience Cloud sites, quick actions, and more.
Record-Triggered Flow
Launches when a record is created, updated, or deleted. This autolaunched flow runs in the background.
1
Schedule-Triggered Flow
Launches at a specified time and frequency for each record in a batch. This autolaunched flow runs in the background.
Platform Event—Triggered Flow
Launches when a platform event message is received. This autolaunched flow runs in the background.
Autolaunched Flow (No Trigger)
Launches when invoked by Apex, processes, REST API, and more. This autolaunched flow runs in the background.
Record-Triggered Orchestration
Launches when a record is created or updated. An orchestration lets you create a multi-step, multi-user process.
2
Create

3. Select the Object as "Billing details and feedback"in the Drop down list.
4. Select the Trigger Flow when: "A record is Created or Updated".
5. Select the Optimize the flow for: "Actions and Related Records" and Click on Done.


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

*Object: Billing details and feedback

*Trigger the Flow When:
- ○ A record is created
- ○ A record is updated
- ● A record is created or updated
- ○ A record is deleted



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

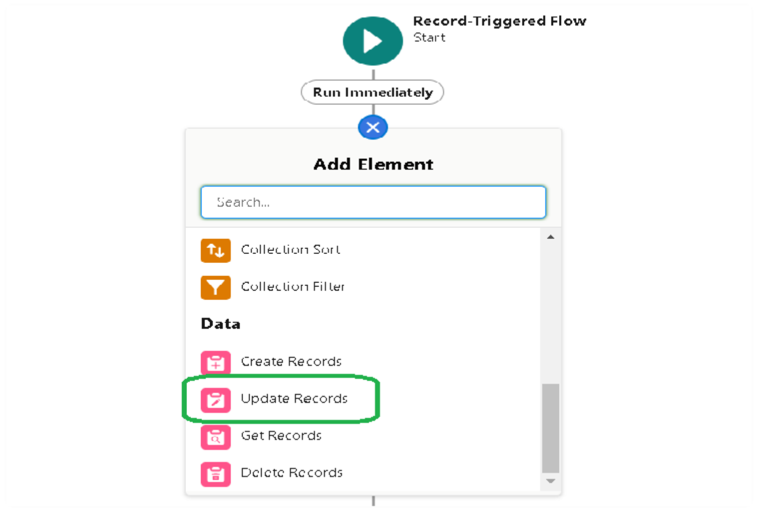
* **Optimize the Flow for:**

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.

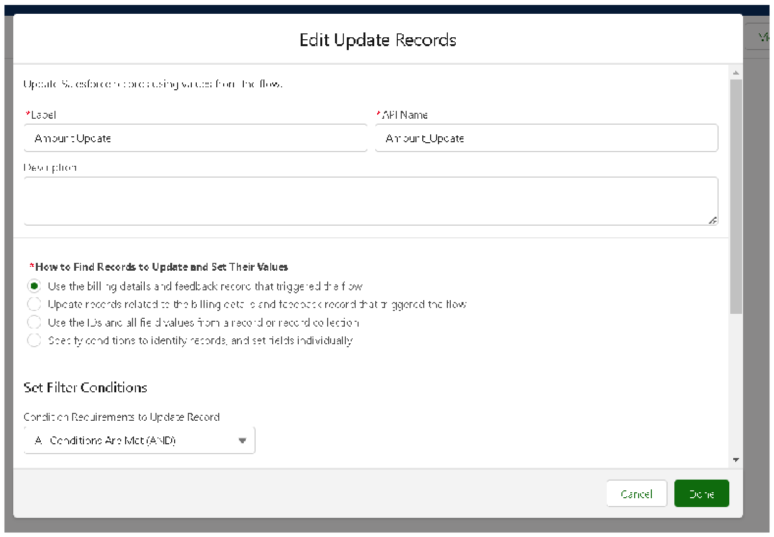
[✓ Actions and Related Records — Update any record and perform actions, like send an email. This more flexible flow runs *after* the record is saved to the database.]

☐ Include a Run Asynchronously path to access an external system after the original transaction for the triggering record is successfully committed

6. Under the Record-triggered Flow Click on "+" Symbol and In the Drop down List select the "Update records Element".



7. Give the Label Name : Amount Update
8. Api name : is auto populated



Edit Update Records

Update Salesforce records using values from the flow.

*Label: Amount Update

*API Name: Amount_Update

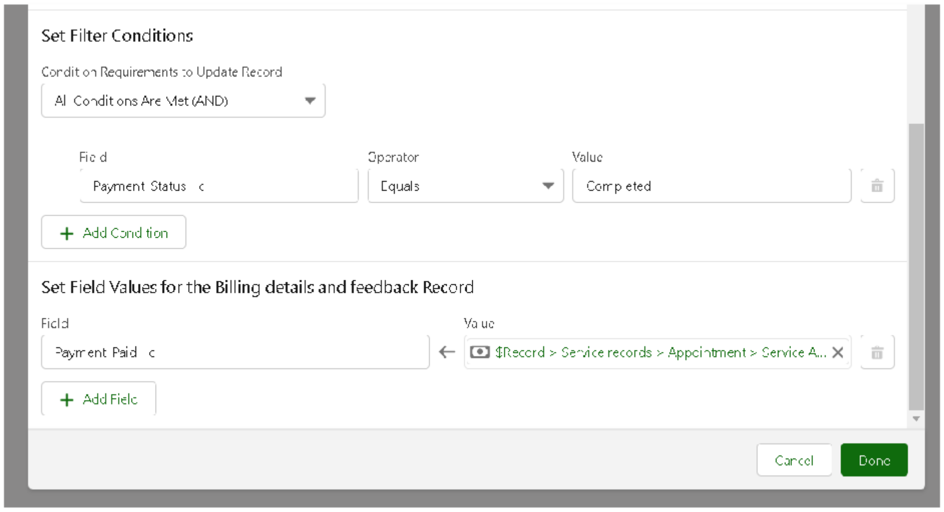
Description:

***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)


Set Filter Conditions

Condition Requirements to Update Record

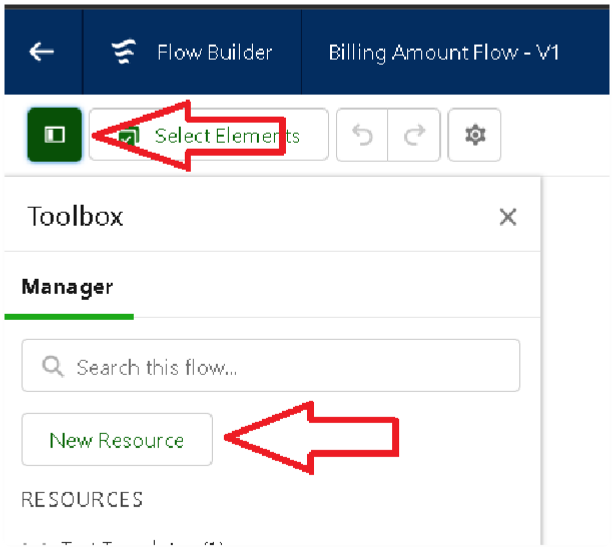
All Conditions Are Met (AND)

Field	Operator	Value
Payment_Status__c	Equals	Completed

Set Field Values for the Billing details and feedback Record

Field	Value
Payment_Paid__c	$Record > Service records > Appointment > Service A...

9. Set a filter condition : All Conditions are met(AND)
10. Field : Payment_Status__c
11. Operator : Equals
12. Value : Completed
13. And Set Field Values for the Billing details and feedback Record
14. Field : Payment_Paid__c
15. Value : {!$Record.Service_records__r.Appointment__r.Service_Amount__c}
16. Click On Done.
17. Before creating another Element. Create a New Resource form Toolbox form top left.



18. Click on the New Resource, And select Variable.
19. Select the resource type as text template.
20. Enter the API name as " alert".
21. Change the view as Rich Text ? View to Plain Text.

22. In body field paste the syntax that given below.

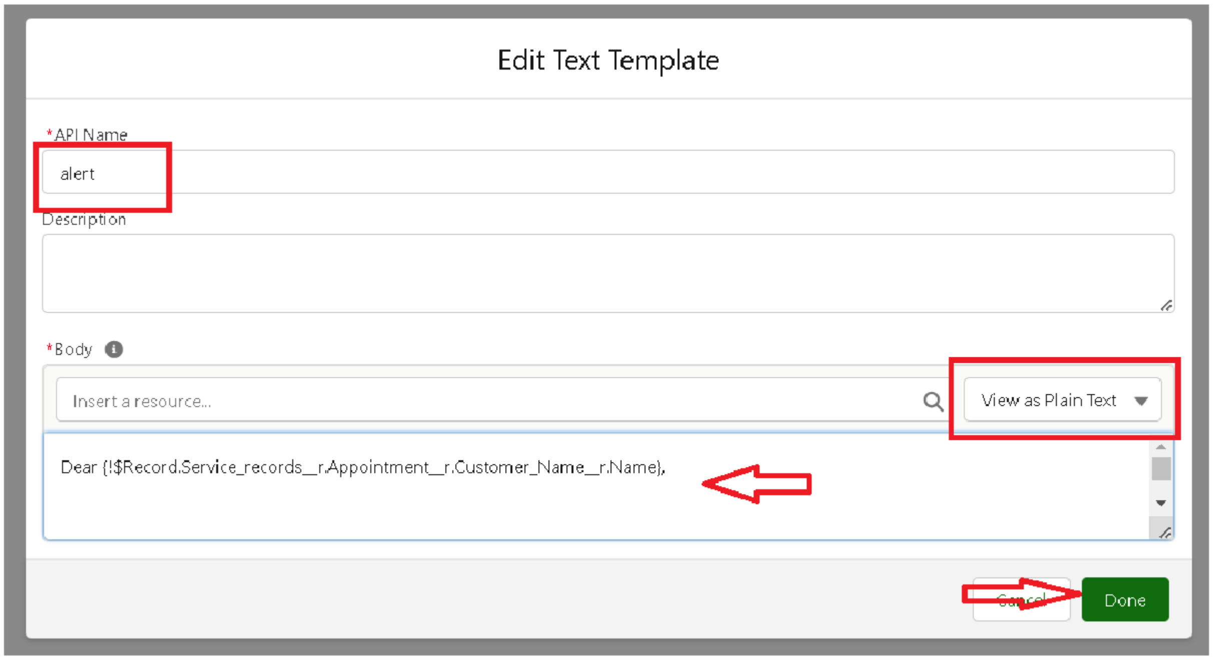
Dear {!$Record.Service_records__r.Appointment__r.Customer_Name__r.Name},

I hope this message finds you well. I wanted to take a moment to express my sincere gratitude for your recent payment for the services provided by our garage management team. Your prompt payment is greatly appreciated, and it helps us continue to provide top-notch services to you and all our valued customers.

Amount paid : {!$Record.Payment_Paid__c}

Thank you for Coming .

23. Click done.


Edit Text Template

*API Name

alert

Description

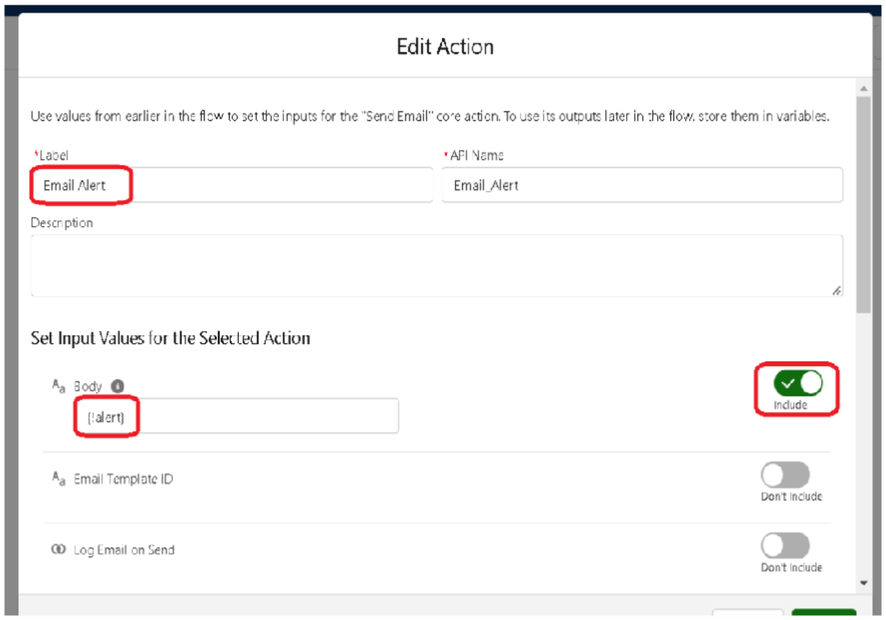
*Body

Insert a resource...

View as Plain Text

Dear {!$Record.Service_records__r.Appointment__r.Customer_Name__r.Name},

24. Now Click on Add Element , select Action.
25. Their action bar will be opened in that search for “ send email ” and click on it.
26. Give the label name as “ Email Alert”
27. API name will be auto populated.
28. Enable the body in set input values for the selected action.
29. Select the text template that created , Body : {!alert}
30. Include recipient address list select the email form the record.
31. RecipientAddressList: {!$Record.Service_records__r.Appointment__r.Customer_Name__r.Gmail__c}
32. Include subject as “ Thank You for Your Payment - Garage Management”.
33. Click done.



Edit Action

Use values from earlier in the flow to set the inputs for the "Send Email" core action. To use its outputs later in the flow, store them in variables.

*Label: Email Alert

*API Name: Email_Alert

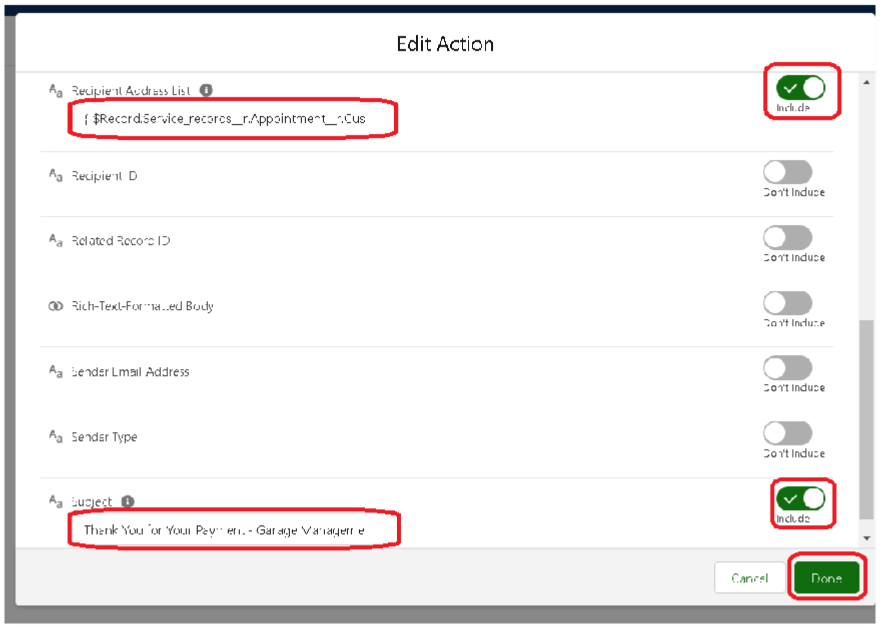
Description

Set Input Values for the Selected Action

Body: {!alert} — Include

Email Template ID — Don't Include

Log Email on Send — Don't Include



Edit Action

Recipient Address List: {!$Record.Service_records__r.Appointment__r.Cus — Include

Recipient ID — Don't Include

Related Record ID — Don't Include

Rich-Text-Formatted Body — Don't Include

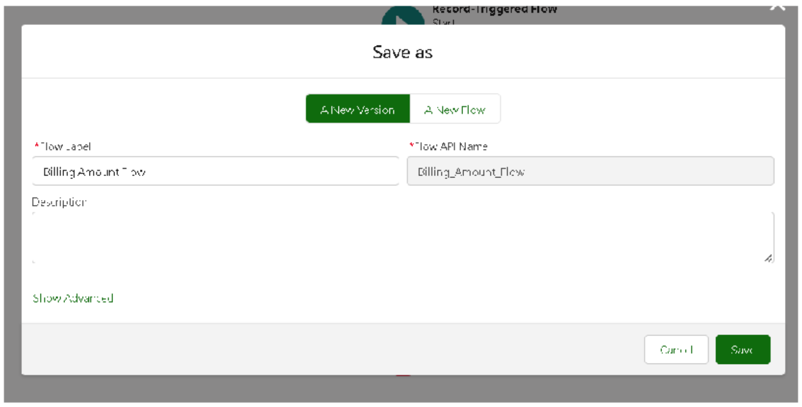
Sender Email Address — Don't Include

Sender Type — Don't Include

Subject: Thank You for Your Payment - Garage Manageme — Include

Cancel | Done

34. Click on save. Give the Flow label , Flow Api name will be autopopulated.
35. And click save, and click on activate.



Record-Triggered Flow
Start

Save as

A New Version | A New Flow

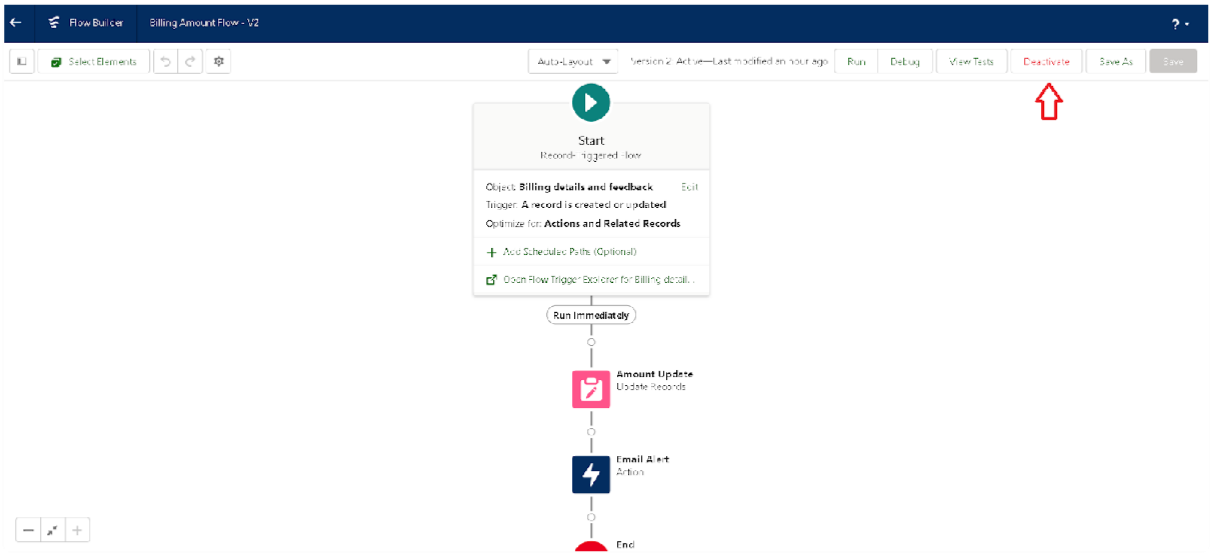
*Flow Label: Billing Amount Flow

*Flow API Name: Billing_Amount_Flow

Description

Show Advanced

Cancel | Save



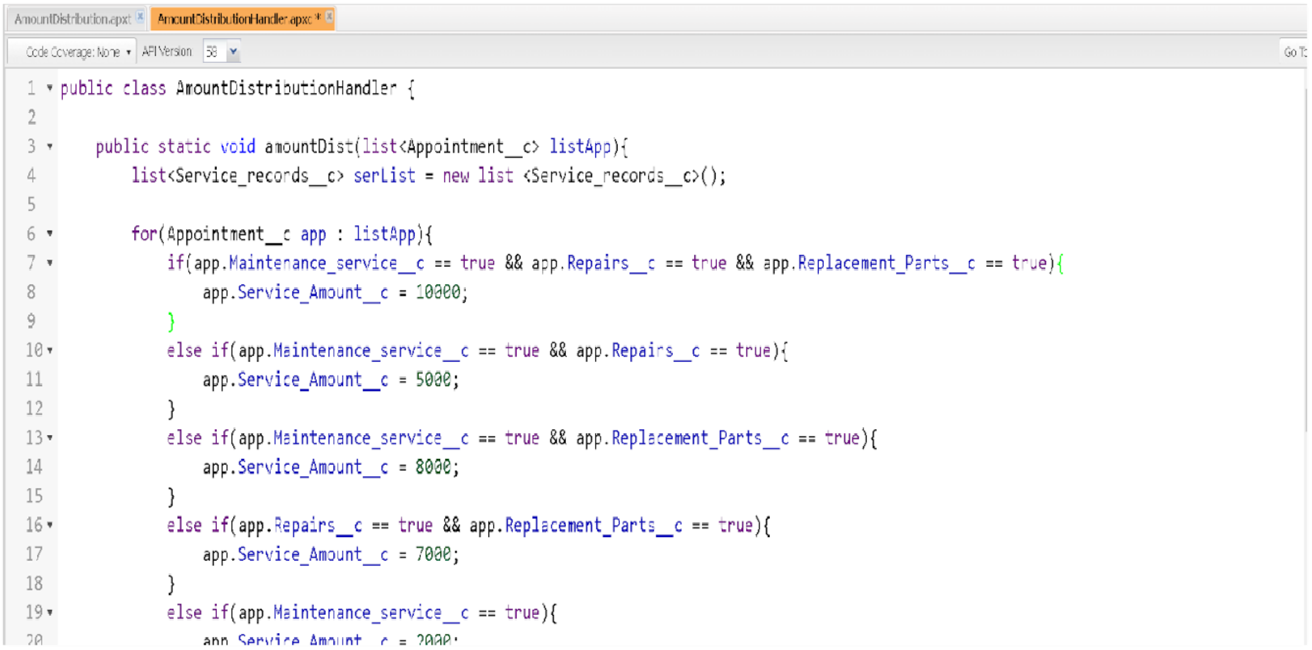
Object: **Billing details and feedback**
Trigger: **A record is created or updated**
Optimize for: **Actions and Related Records**

Task 11: Apex Trigger

This use case works for Amount Distribution for each Service the customer selected for there Vehicle.

Steps for creating Apex Trigger are:-

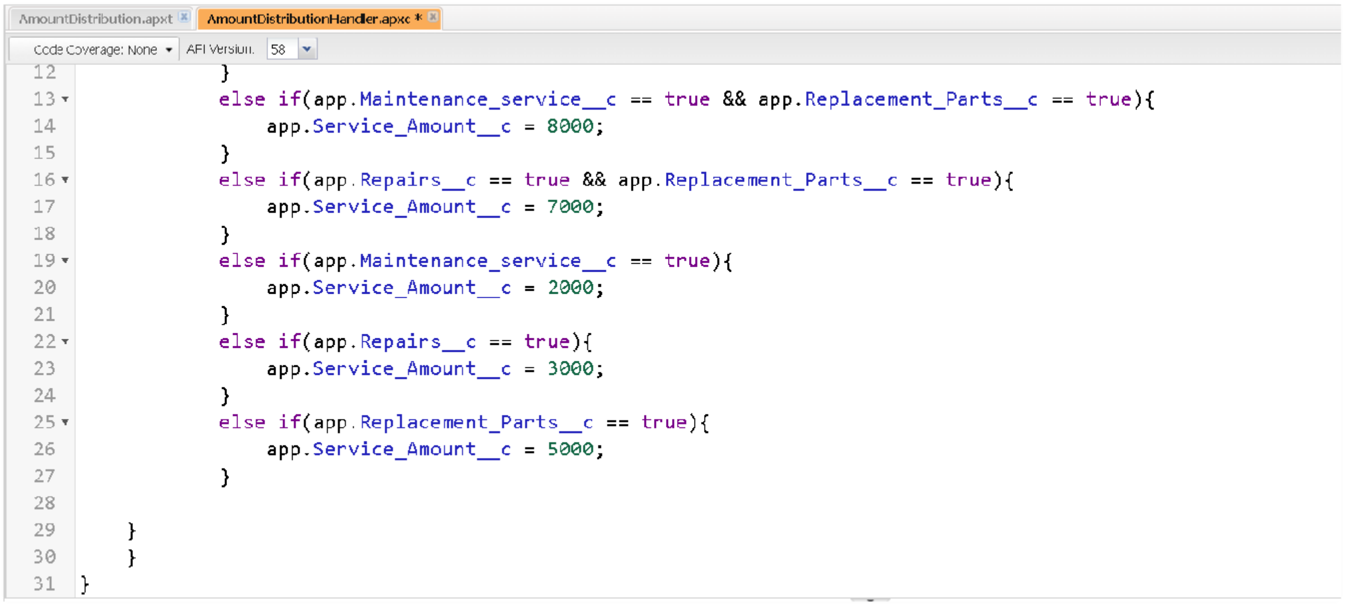
1. Login to the respective trailhead account and navigate to the gear icon in the top right corner.
2. Click on the Developer console. Now you will see a new console window.
3. In the toolbar, you can see FILE. Click on it and navigate to new and create New apex class.
4. Name the class as “AmountDistributionHandler ”.



```
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;
```

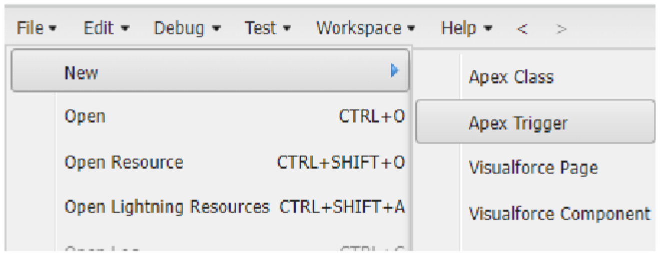


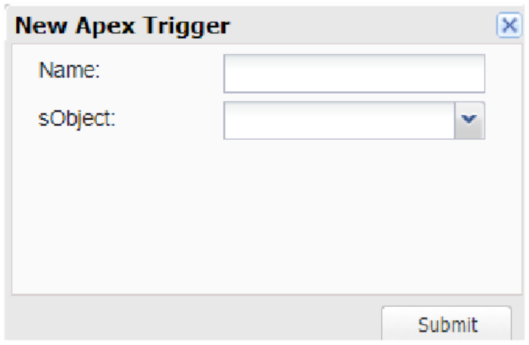
```
            }
            else if(app.Maintenance_service__c == true && app.Replacement_Parts__c == true){
                app.Service_Amount__c = 8000;
            }
            else if(app.Repairs__c == true && app.Replacement_Parts__c == true){
                app.Service_Amount__c = 7000;
            }
            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;
            }

        }
    }
}
```

Steps for creating Trigger Handler:-

1. While still in the trailhead account, navigate to the gear icon in the top right corner.
2. Click on developer console and you will be navigated to a new console window.
3. Click on File menu in the tool bar, and click on new? Trigger.
4. Enter the trigger name and the object to be triggered.
5. Name : AmountDistribution
6. sObject : Appointment__c





Syntax For creating trigger :

The syntax for creating trigger is :

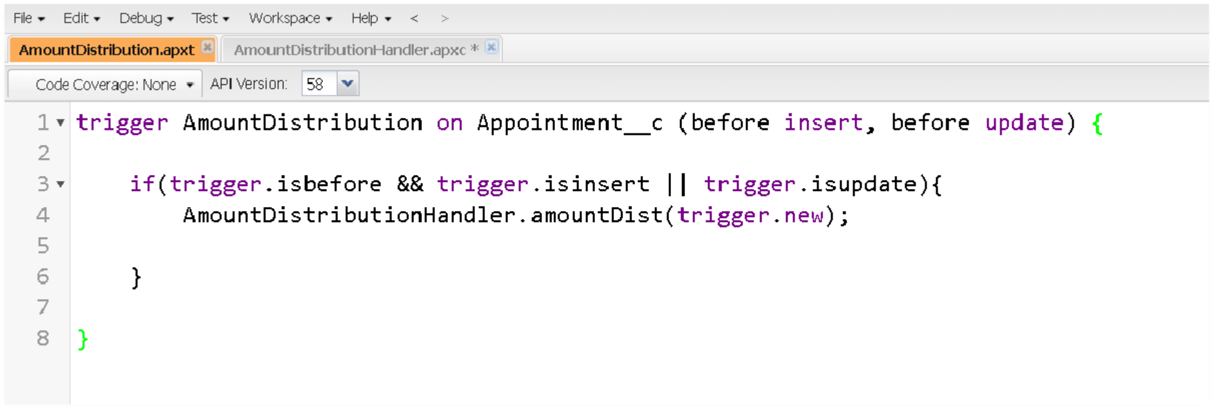
Trigger [trigger name] on [object name](Before/After event)

{

}

In this project , trigger is called whenever the particular records sum exceed the threshold i.e minimum business requirement value. Then the code in the trigger will get executed.

1. Handler for the Appointment Object



```
trigger AmountDistribution on Appointment__c (before insert, before update) {

    if(trigger.isbefore && trigger.isinsert || trigger.isupdate){
        AmountDistributionHandler.amountDist(trigger.new);

    }

}
```

Task 11: Report

Steps for creating the Report folder:-

1. Click on the app launcher and search for reports.
2. Click on the report tab, click on new folder.
3. Give the Folder label as “Garage Management Folder”, Folder unique name will be auto populated.
4. Click save.

Steps for Sharing a Report folder:-

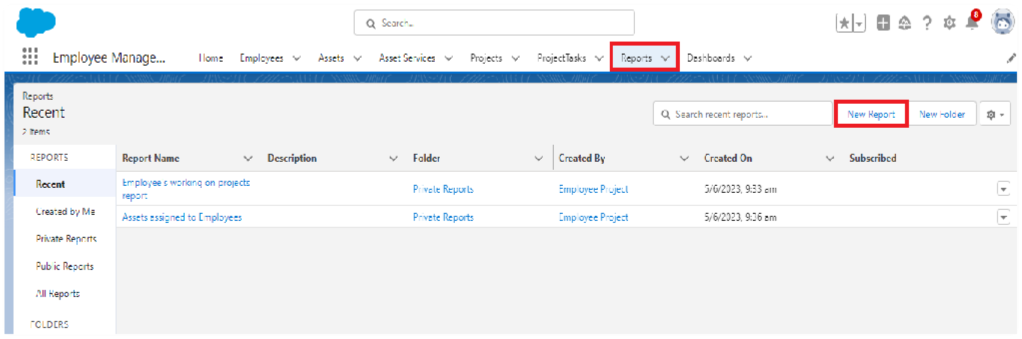
1. Go to the app >> click on the reports tab.
2. Click on the All folder , click on the Drop down arrow for Garage Management folder, and Click on share.
3. Select the share with as “roles”, in name field search for “manager”, give “view” as access for that role.
4. Then click share, and click on Done.

Steps for Creating a Report Type:-

1. Go to setup >> type users in quick find box >> select Report Type >> click on Continue.
2. Click on new custom report type.
3. Select the Primary object as " Customer details" .
4. Give the Report type Label as " Service information "
5. Report type Name is autopopulated.
6. Keep the Description as same.
7. Select Store in Category as " other Reports "
8. Select the deployment status as " Depolyed ", click on Next.
9. now , Click on Related object box.
10. Click on Select Object, choose Appointment Object as shown in fig.
11. Again Click to relate another object.
12. And select the related object as " service records".
13. Repeat the process and select the related object as " Billing details and feedback".
14. And click on save.

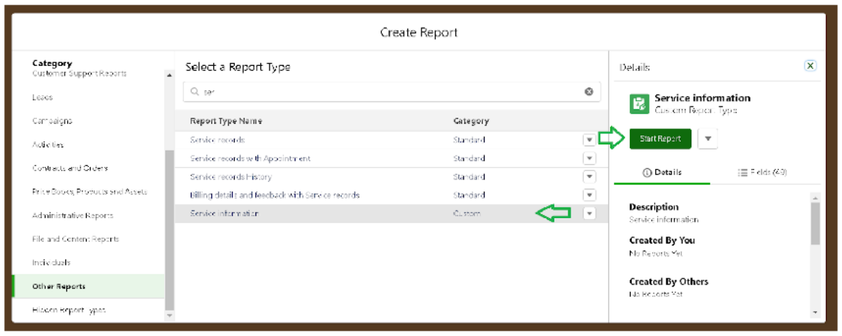
Steps for Creating A Report:-

1. Go to the app >> click on the reports tab
2. Click New Report.



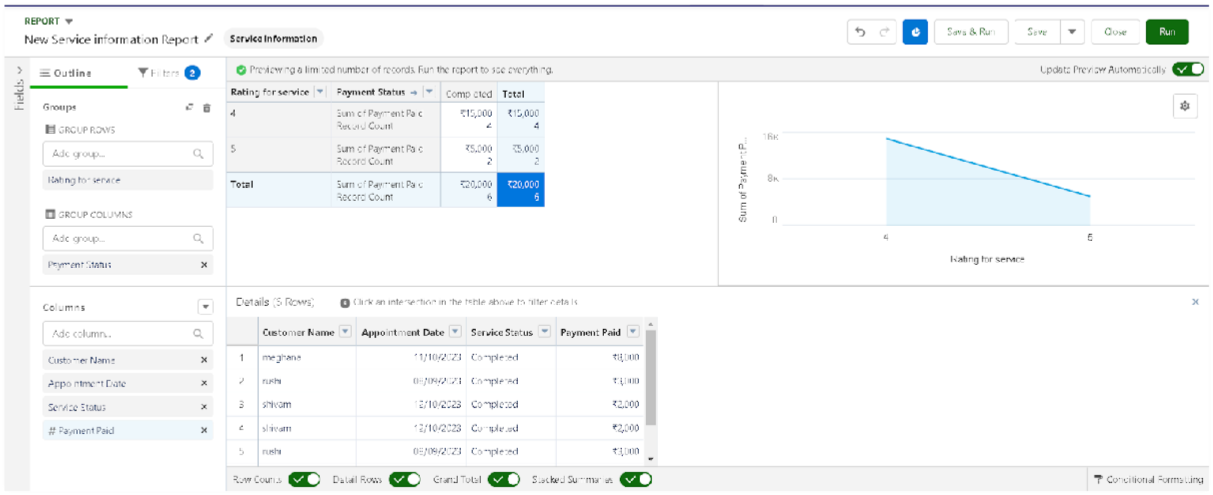
Report Name	Description	Folder	Created By	Created On	Subscribed
Employee s working on projects report		Private Reports	Employee Project	5/6/2023, 9:33 am	
Assets assigned to Employees		Private Reports	Employee Project	5/6/2023, 9:36 am	

3. Select the Category as other reports, search for Service Information, select that report, click on it. And click on start report.



Report Type Name	Category
Service records	Standard
Service records with Appointment	Standard
Service records History	Standard
Billing details and feedback with Service records	Standard
Service information	Custom

4. Their outline pane is opened alredy, select the fields that mentioned below in column section.
    1. Customer name
    2. Appointment Date
    3. Service Status
    4. Payment paid
5. Remove the unnecessary fields.
6. Select the fields that mentioned below in GROUP ROWS section.
    1. Rating for Service
7. Select the fields that mentioned below in GROUP ROWS section.
    1. Payment Status
8. Click on Add Chart , Select the Line Chart.
9. Click on save, Give the report Name : New Service information Report
10. Report unique Name is auto populated.
11. Select the folder the created and Click on save.


REPORT

New Service information Report

Service Information

Previewing a limited number of records. Run the report to see everything.

Update Preview Automatically

Groups

GROUP ROWS: Rating for service

GROUP COLUMNS: Payment Status

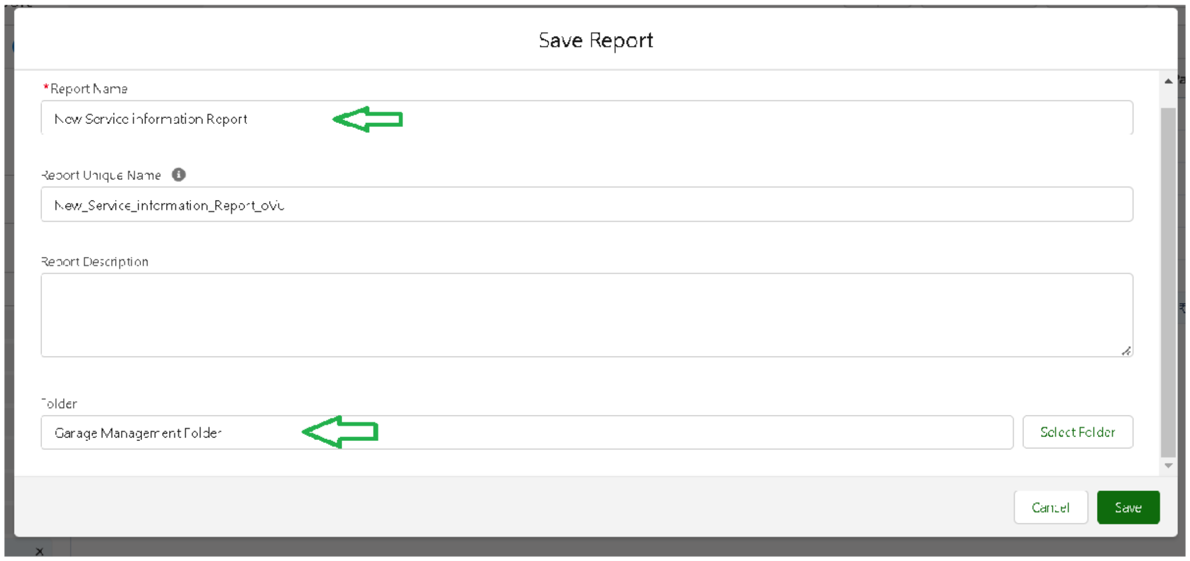
Columns: Customer Name, Appointment Date, Service Status, # Payment Paid

Rating for service	Payment Status →	Completed	Total
4	Sum of Payment Paid Record Count	₹15,000 4	₹15,000 4
5	Sum of Payment Paid Record Count	₹5,000 2	₹5,000 2
Total	Sum of Payment Paid Record Count	₹20,000 6	₹20,000 6

Details (5 Rows) Click an intersection in the table above to filter details

	Customer Name	Appointment Date	Service Status	Payment Paid
1	meghana	11/10/2023	Completed	₹8,000
2	rushi	08/09/2023	Completed	₹3,000
3	shivam	12/10/2023	Completed	₹2,000
4	shivam	12/10/2023	Completed	₹2,000
5	rushi	08/09/2023	Completed	₹3,000

Row Counts ✓ Detail Rows ✓ Grand Total ✓ Stacked Summaries ✓


Save Report

*Report Name: New Service information Report

Report Unique Name: New_Service_information_Report_oVL

Report Description:

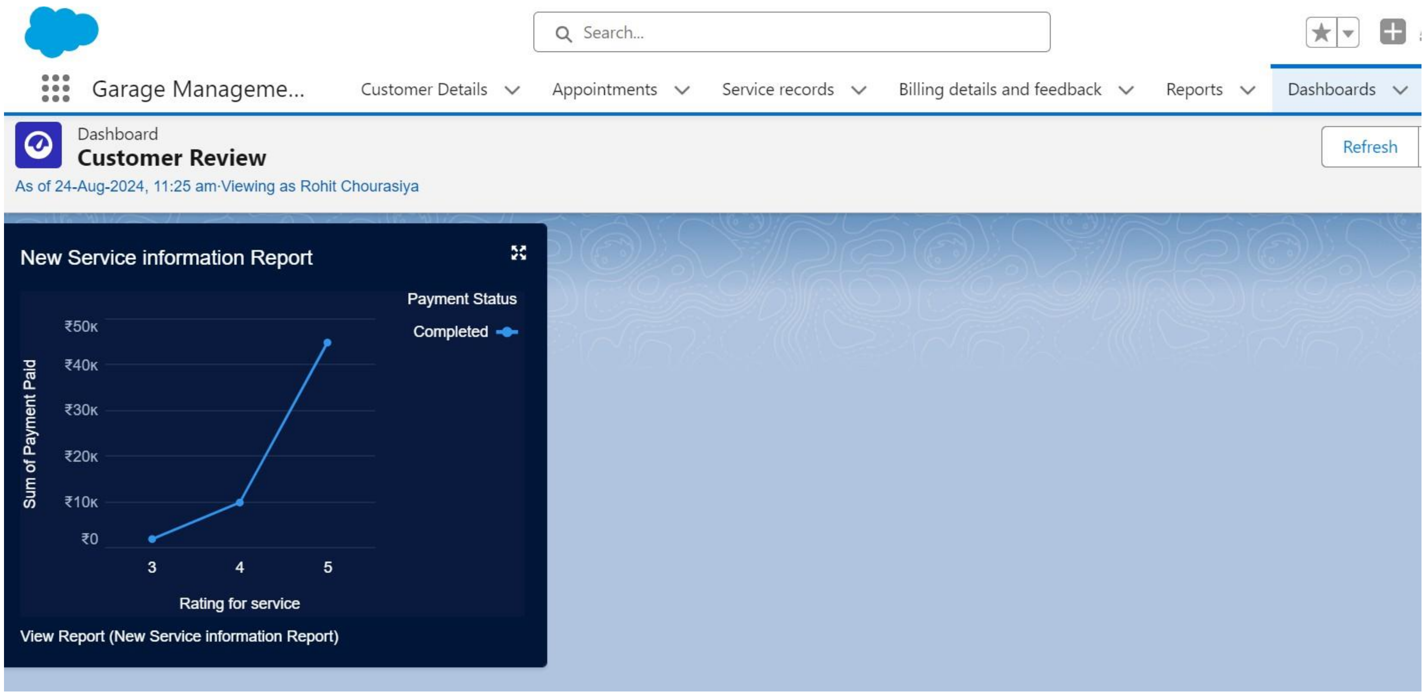
Folder: Garage Management Folder

Task 12: Dashboard

Dashboard is simply a visual representation of the Report.

Steps for creating the Dashboard are:-

1. Go to the app >> click on the Dashboards tabs.
2. Give a Name and select the folder that created, and click on create.
3. Select add component.
4. Select a Report and click on select.
5. Select the Line Chart. Change the theme.
6. Click Add then click on Save and then click on Done.
7. Preview is shown below
8. After that Click on Subcribe on top right.
9. Set the Frequency as " weekly ".
10. Set a day as monday.
11. And Click on save.



Dashboard

Customer Review

As of 24-Aug-2024, 11:25 am·Viewing as Rohit Chourasiya

View Report (New Service information Report)

Thank You