

Garage Management System

COLLEGE NAME: Sri Krishna Adithya College of Arts and Science

COLLEGE CODE: Bruag

TEAM ID: NM2025TMID22663

TEAM MEMBERS: 4

Team Leader Name: SHAHITH NAFAL D

Email: 23bsds153shahithnafald@skacas.ac.in

Team Member 1: SANJAY S

Email: 23bsds152sanjays@skacas.ac.in

Team Member 2: ADITHYA KRISHNA M

Email: 23bsds102adithyakrishnam@skacas.ac.in

Team Member 3: NITHISHVAR M

Email: 23bsds141nithishvarm@skacas.ac.in

Title:- Garage Management System

Project Overview:

The **Garage Management System** is a Salesforce-based solution designed to streamline the management of automotive service centers. It connects **customers, service advisors, mechanics, inventory managers, and billing staff** to ensure smooth operations. Using **custom objects, workflows, automation, and reports**, the system digitizes the entire process of **vehicle service booking, job tracking, spare parts management, and invoicing**.

The solution improves efficiency, enhances customer satisfaction, and ensures transparency in service history, cost estimation, and resource utilization.

Objectives:

1. Streamline Service Operations

- Manage vehicle service bookings, job assignments, and progress tracking digitally.

2. Enhance Customer Experience

- Provide real-time updates on service status, cost estimates, and delivery timelines.

3. Optimize Resource & Inventory Management

- Track spare parts availability, consumption, and procurement to reduce delays.

4. Improve Staff Coordination

- Automate task allocation, notifications, and approvals for mechanics and service advisors.

5. Ensure Transparency & Accountability

- Maintain detailed records of service history, invoices, and payments for future reference.

6. Enable Data-Driven Decisions

- Generate reports and dashboards to monitor revenue, service trends, and customer satisfaction.

Student Outcomes:

- **Hands-on Experience with Garage Operations Automation:** Students gain practical skills in configuring Salesforce objects, automating service workflows, and managing vehicle service records.
- **Understanding of Project Lifecycle in Automotive CRM:** Students learn the complete process from requirement gathering to deployment, enhancing their ability to deliver real-world Salesforce projects.
- **Enhanced Analytical and Problem-Solving Skills:** Students develop the ability to address operational challenges such as job tracking, inventory shortages, and billing errors.
- **Improved Collaboration Skills:** Students practice teamwork by coordinating tasks across modules like service scheduling, inventory management, and reporting.
- **Industry-Relevant Exposure:** Students explore practical use cases of Salesforce CRM in the automotive service industry, preparing them for careers in CRM and service management.

System Requirements:

Hardware Requirements:

- Computer with minimum 4 GB RAM, Dual-core processor
- Stable internet connection

Software Requirements:

- Salesforce Developer Edition Org
- Modern Web Browser (e.g., Google Chrome, Firefox)

Project Duration: 31 Hours

Phases Overview:

Phase No.	Phase Name	Description	Page Numbers
1	Requirement Analysis & Planning	Gathering requirements from donors, volunteers, and receivers; defining scope and goals; planning data model and workflows.	4
2	Salesforce Development – Backend & Configurations	Creating custom objects, fields, relationships; setting up Flows and Apex Triggers for automation.	4 - 11
3	UI/UX Development & Customization	Building Lightning App, customizing layouts, adding fields, implementing Flows, and developing UI logic.	11 - 28
4	Data Migration, Testing & Security	Creating Users, Profiles, Public Groups, Sharing Rules; configuring Report Types, Reports, Dashboards; testing functionalities and ensuring data security.	28 - 37
5	Deployment, Documentation & Maintenance	Designing and finalizing Home Page, deploying solution to live environment, preparing documentation, conclusion, and ongoing system maintenance.	37 - 40

Phase 1: Requirement Analysis & Planning:-

To make garage management system:

Utilizing Salesforce, our project streamlines vehicle service management, spare parts tracking, and customer coordination, ensuring efficiency and transparency.

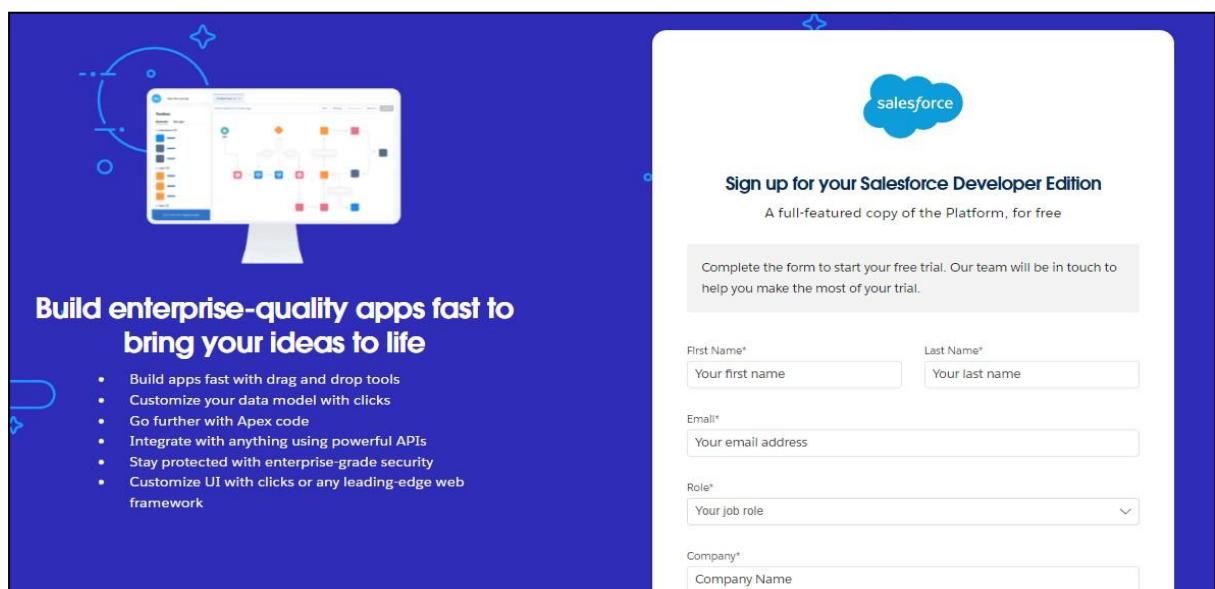
Phase 2: Salesforce Development – Backend & Configurations:-

Milestone 1: Salesforce developer account creation Activity

1: Creating Developer Account

Creating a developer org in salesforce.

1. Go to <https://developer.salesforce.com/signup>



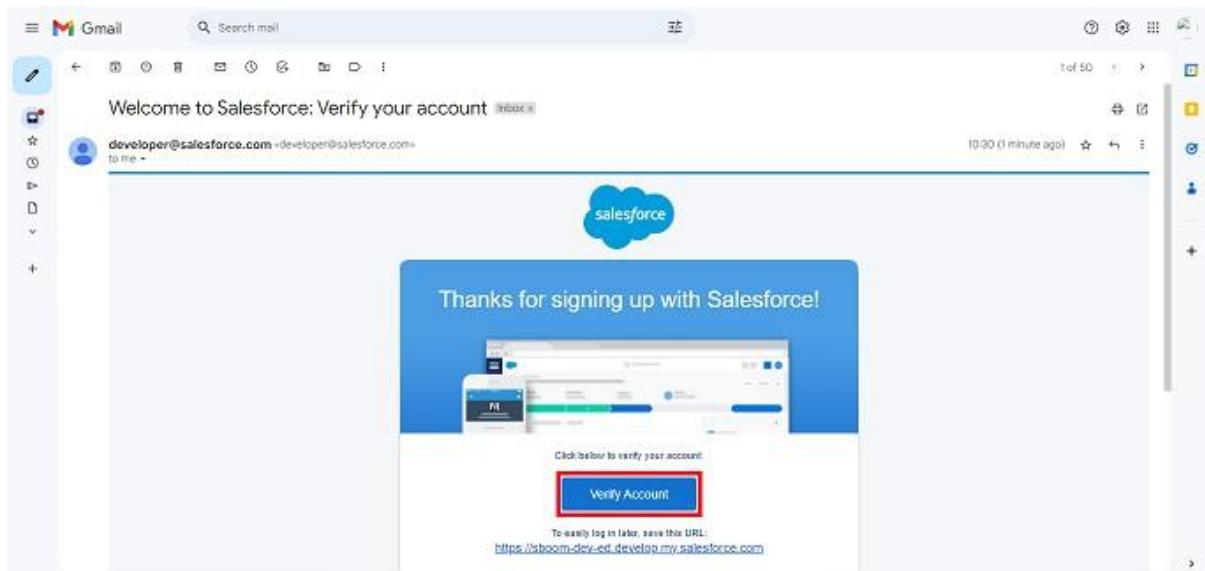
2. On the sign up form, enter the following details :

1. First name & Last name
2. Email
3. Role : Developer
4. Company : College or Company Name

5. County: India
 - G. Postal Code: pin code
 7. Username: should be a combination of your name and company This need not be an actual email id, you can give anything in the format: username@organization.com
- Click on sign me up after filling these.

Activity 2: Account Activation

1. Go to the inbox of the email that you used while signing up. Click on the verify account to activate your account. The email may take 5-10mins.



Change Your Password

Enter a new password for lead@sb.com.
Make sure to include at least:

- 8 characters
- 1 letter
- 1 number

* New Password
..... Good

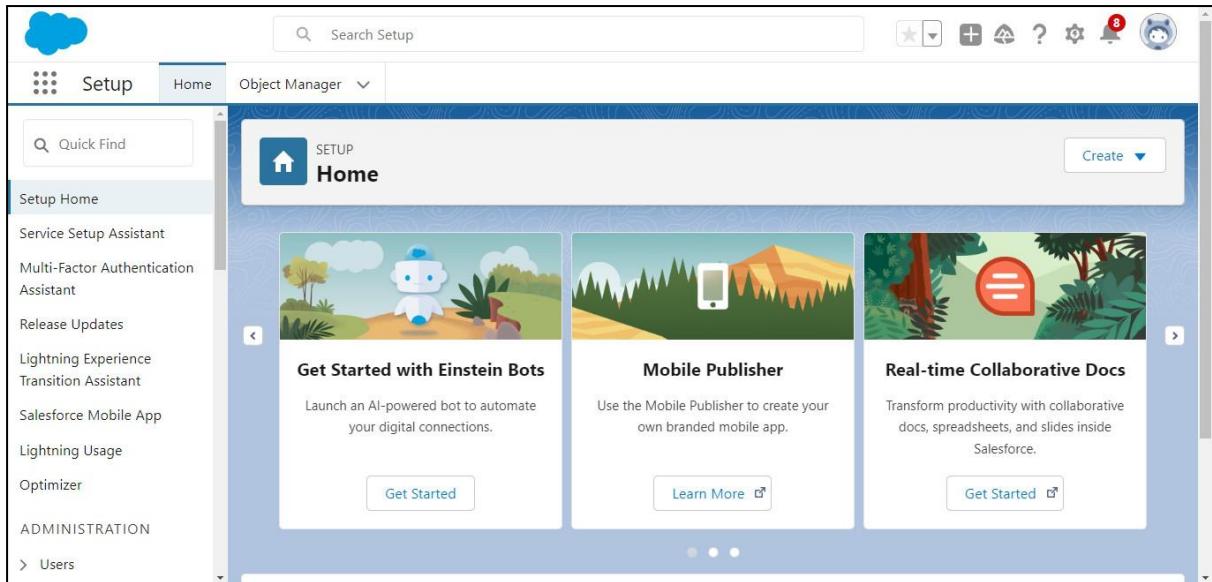
* Confirm New Password
..... Match

Security Question
In what city were you born?

* Answer
asdfghjkl

Change Password

1. Click on Verify Account
2. Give a password and answer a security question and click on change password.
3. Then you will redirect to your salesforce setup page.

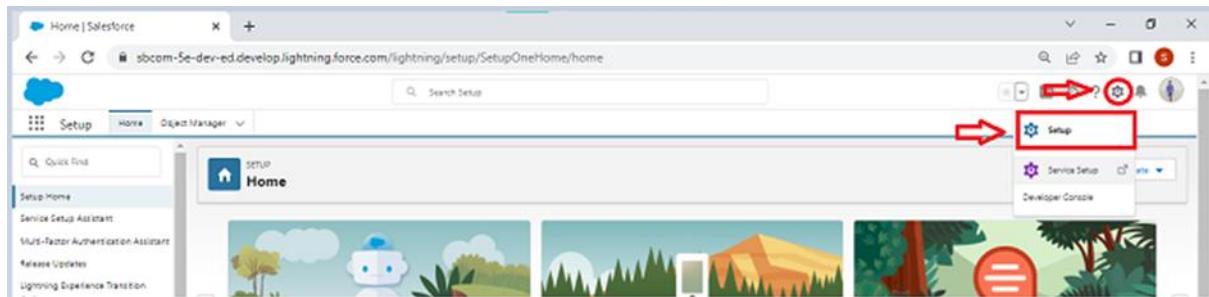


OBJECT:

1. **Standard Objects:** Standard objects are the kind of objects that are provided by salesforce.com such as users, contracts, reports, dashboards, etc.
2. **Custom Objects:** Custom objects are those objects that are created by users. They supply information that is unique and essential to their organization. They are the heart of any application and provide a structure for sharing data.

To Navigate to Setup page:

Click on gear icon? click setup.



To create an object:

1. From the setup page?
2. Click on Object Manager?
3. Click on Create? Click on Custom Object.

The screenshot shows the Salesforce Setup interface. At the top, there's a navigation bar with 'Setup' and 'Object Manager'. A large black arrow points from the text 'Click on Create? Click on Custom Object.' to the 'Create' button, which is highlighted with a red box. To the right of the 'Create' button, another red box highlights the 'Custom Object' link in the top right corner of the main content area.

1. On Custom object defining page:
2. Enter the label name, plural label name, click on Allow reports, Allow search.

This screenshot shows the 'New Custom Object' setup page. Several fields are highlighted with red boxes and arrows:

- 'Label' field (singular label)
- 'Plural Label' field (plural label)
- 'Object Name' field (API name)
- 'Optional Features' section: 'Allow Reports' checkbox (circled with a red circle and arrow)

At the bottom, the 'Save' and 'Save & New' buttons are highlighted with red boxes and arrows.

This screenshot continues from the previous one, showing more of the 'New Custom Object' page:

- 'Optional Features' section: 'Allow Activities' checkbox (circled with a red circle and arrow)
- 'Object Classification' section: 'Allow Search' checkbox (circled with a red circle and arrow)
- 'Object Creation Options' section: 'Save' and 'Save & New' buttons at the bottom (highlighted with red boxes and arrows)

3. Click on Save.

Activity 1: Create 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
 - Click on Allow reports and Track Field History,
 - Allow search >> Save.

Activity 2: Create Appointment 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 >> 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.

Activity 3: Create Service Records 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 >> 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
- Click on Allow reports and Track Field History,
- Allow search >> Save.

Activity 4: Create Billing Details and Feedback Object:

To create an object:

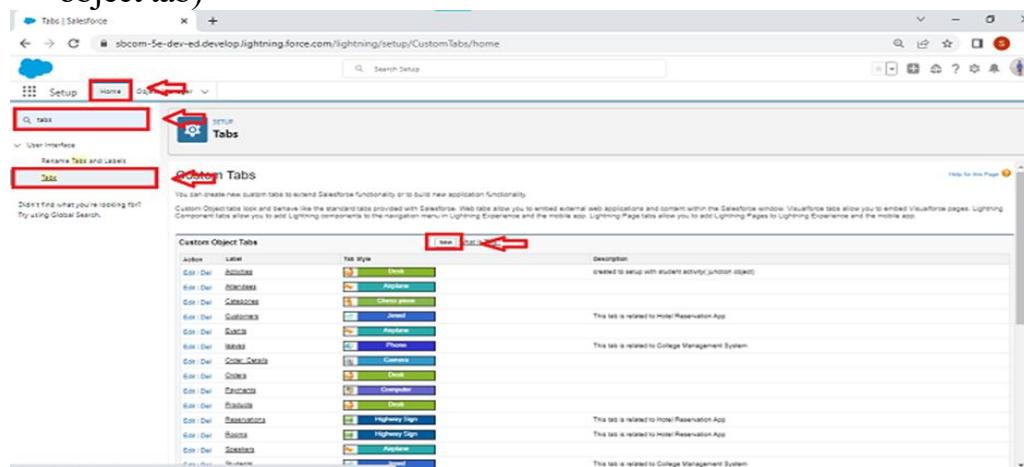
1. From the setup page >> Click on Object Manager >> Click on Create >> Click on Custom Object.
1. Enter the label name >> Billing details and feedback
2. Plural label name >> Billing details and feedback
3. Enter Record Name Label and Format
 - Record Name >> Billing details and feedback Name
 - Data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

TABS:

Activity 1: Creating a Custom Tab:

To create a 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.

New Custom Object Tab

Step 1. Enter the Details Step 1 of 3

Choose the custom object for this new custom tab. Fill in other details.

Select an existing custom object or [create a new custom object now](#).
Object: Customer Details
Tab Style:

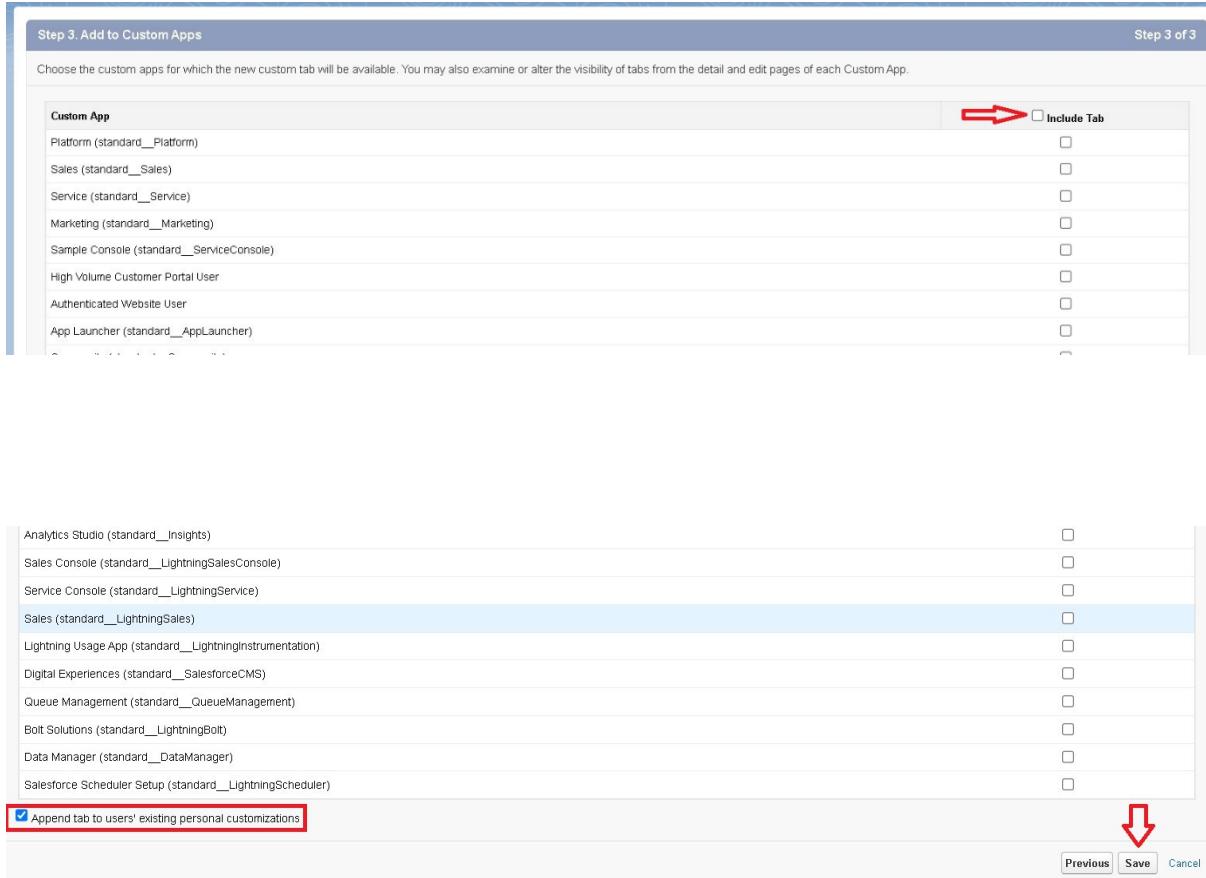
(Optional) Choose a Home Page Custom Link to show as a splash page the first time your users click on this tab.
Splash Page Custom Link: -None-

Enter a short description.
Description:

[Next](#) [Cancel](#)

Tab Style Selector		Create your own style	
Hide styles which are used on other tabs			
	Airplane		Alarm clock
	Bank[1]		Bell
	Books		Bottle
	Building		Building Block
	Can		Car
	Cell phone		Chalkboard
	Circle		Compass
	CRT TV		Cup
	Dice		Factory
	Form		Gears
	Hammer		Hands
	Heart[1]		Helicopter
	Hot Air Balloon		Insect
	Keys		Laptop
	Apple		Big top
	Box		Caduceus
	Castle		Chess piece
	Chip		Computer
	CD/DVD		Desk[1]
	Diamond		Fan
	Flag		Globe
	Guitar		Handsaw
	Headset		Hexagon
	Highway Sign		IP Phone
	Jewel		Leaf
	Lightning		

[Save](#) [Cancel](#)



Activity 2: Creating Remaining Tabs:-

1. Now create the Tabs for the remaining Objects, they are “Drop-Off Point, Task, Volunteer, Execution Details”.
2. Follow the same steps as mentioned in Activity -1.

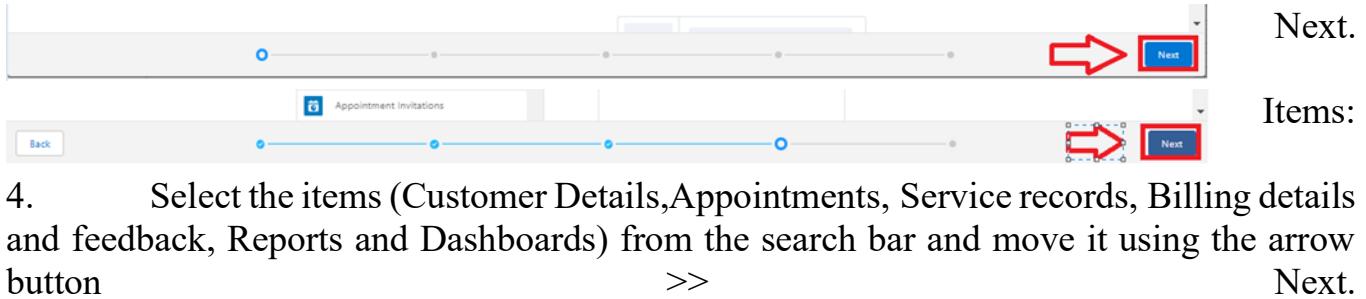
Phase 3: UI/UX Development & Customization:

THE LIGHTNING APP:

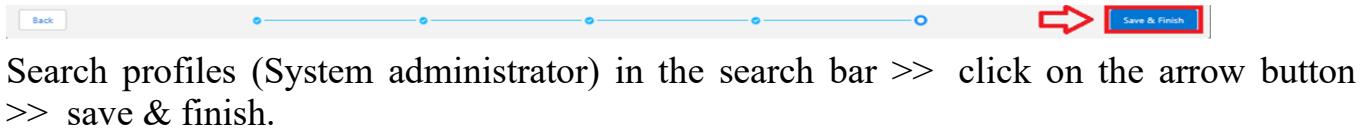
Create a Lightning App

To create a lightning app page:

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



5. To Add User Profiles:



FIELDS:

Activity 1: Creation of fields for the customer details object:

Creation of fields for the Customer Details object

1. To create fields in an object:

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.

Note: Follow the above steps for the remaining field for the same object.

2. To create another fields in an object:

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 “Email” and Click on Next
4. Fill the Above as following:
 - Field Label : Gmail
 - Field Name : gets auto generated
 - Click on Next >> Next >> Save and new.

Activity 2: Creation of Lookup Relationship Field on Volunteer Object:

Creation of Lookup Field on Appointment Object :

1. Go to setup >> click on Object Manager >> type object name(Appointment) in the search bar >> click on the object.



2. Now click on “Fields & Relationships” >> New
3. Select “Look-up relationship” as data type and click Next.

4. Select the related object “ Customer Details” and click next.
5. Next >> Next >> Save.

Note: Make sure you complete Activity 4 Before continuing.

Creation of Lookup Field on Service records Object :

1. Go to setup >> click on Object Manager >> type object name(Service records) in search bar >> click on the object.
2. Now click on “Fields & Relationships” >> New
3. Select “Look-up relationship” as data type and click Next.
4. Select the related object “ Appointment ” and click next.
5. Make it a required field so click on Required.
6. Scroll down for Lookup Filter and click on Show filter settings.
7. Now add the filter criteria.
8. Field : Appointment: Appointment Date >> Operator : less than >> select field >> Appointment: Created Date
9. Filter type should be Required.
10. Error Message : Value does not match the criteria.
11. Enable the filter by click on Active.
12. Next >> Next >> Save.

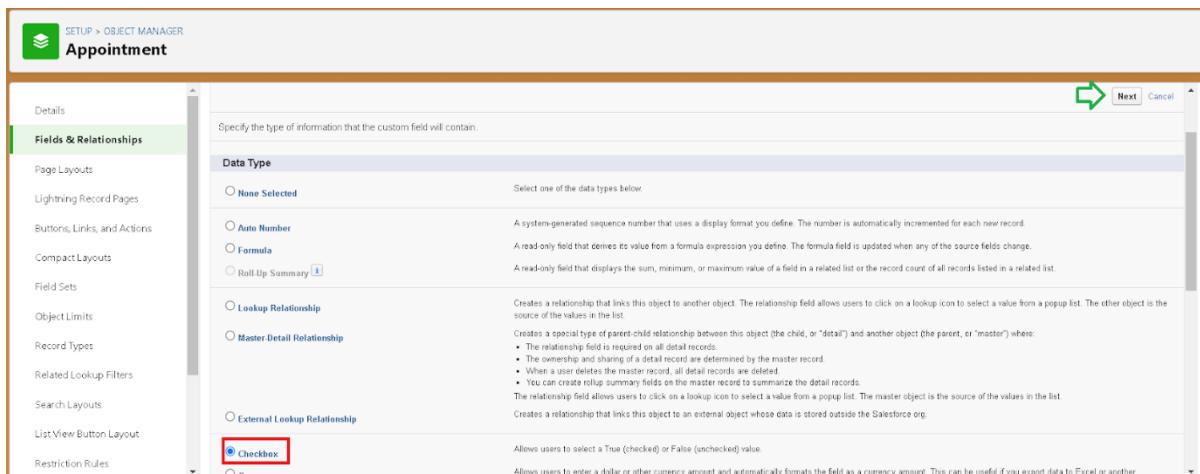
Creation of Lookup Field on Billing details and feedback Object :

1. Go to setup >> click on Object Manager >> type object name(Billing details and feedback) in search bar >> click on the object.
2. Now click on “Fields & Relationships” >> New.
3. Select “Look-up relationship” as data type and click Next.
4. Select the related object “ Service records” and click next.
5. Next >> Next >> Save & new.

Activity 3: Creation of Master-Detail Relationship Field on Execution Detail Object:

Creation of Checkbox Field on Appointment Object :

1. Go to setup >> click on Object Manager >> type object name(Appointment) in search bar >> click on the object.
2. Now click on “Fields & Relationships” >> New.
3. Select “Check box” as data type and click Next.



4. Give the Field Label : Maintenance service
5. Field Name : is auto populated
6. Default value : unchecked

Step 2 Enter the details

Field Label: Maintenance service

Default Value: Checked Unchecked

Field Name: Maintenance_service

Description:

Help Text:

Auto add to custom report type Add this field to existing custom report types that contain this entity

7. Click on next >> next >> save.

Activity 4: Creation of Date Field on Appointment Object:

Creation of date Fields

Creation of Date Field on Appointment Object :

1. Go to setup >> click on Object Manager >> type object name(Appointment) in the search bar >> click on the object.
2. Now click on “Fields & Relationships” >> New.
3. Select “Date” as data type and click Next.
4. Give the Field Label : Appointment Date
5. Field Nme : is auto populated
6. Make it as a Required field by click on the Required option.
7. Click on next >> next >> save.

Activity 5: Creation of Look-Up Relationship Field on Drop-off Point Object:

Creation of Currency Field on Appointment Object :

1. Go to setup >> click on Object Manager >> type object name(Appointment) in the search bar >> click on the object.
2. Now click on “Fields & Relationships” >> New.
3. Select “Currency” as data type and click Next.
4. Give the Field Label : Service Amount
5. Field Nme : is auto populated
6. Click on next
7. Give read only for all the profiles in field level security for profile.
8. Click on next >> save.

Activity 6: Creation of Lookup Relationship Field on Task Object:

Creation of Text Fields

1. Go to setup >> click on Object Manager >> type object name(Appointment) in the search bar >> click on the object.
2. Now click on “Fields & Relationships” >> New.
3. Select “Text” as data type and click Next.
4. Give the Field Label : Vehicle number plate
5. Field Name : is auto populated
6. Length : 10
7. Make field as Required and Unique.
8. Click on next >> next >> save.

Activity 7: Creation of Lookup Relationship Field on Task Object:

Creation of Picklist Fields in Service records object :

1. Go to setup >> click on Object Manager >> type object name(Service records) in search bar >> click on the object.
2. Click on fields & relationship >> click on New.
3. Select Data type as “Picklist” and click Next.
4. Enter Field Label as “Service Status”, under values select “Enter values, with each value separated by a new line” and enter values as shown below.
5. The values are: Started, Completed.
6. Click Next.
7. Next >> Next >> Save.

Activity 8: Creation of Fields for Venue Object:

Creating Formula Field in Service records Object

1. Go to setup >> click on Object Manager >> type object name(Service records) in search bar >> click on the object.
2. Click on fields & relationship >> click on New.
3. Select Data type as “Formula” and click Next.
4. Give Field Label and Field Name as “service date” and select formula return type as “Date” and click next.
5. Insert field formula should be : CreatedDate

6. click “Check Syntax” .
7. Click next >> next >> Save.

VALIDATION RULE:

Activity 2: Create a validation rule to an appointment object:

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.

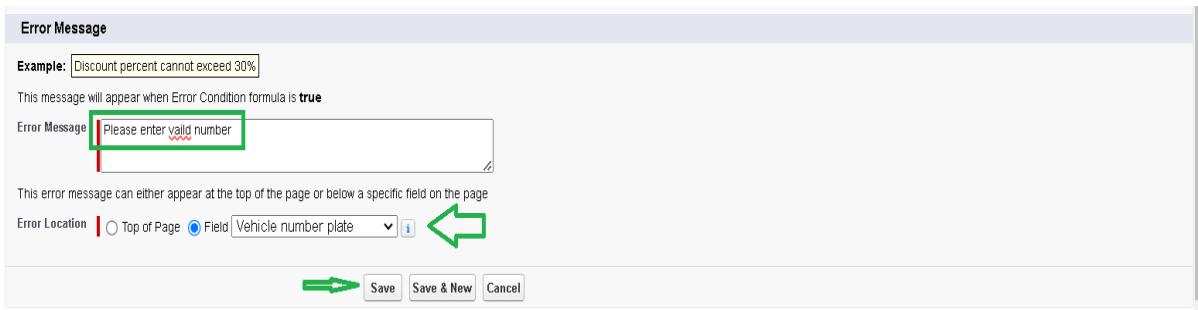
The screenshot shows the Salesforce Object Manager interface for the 'Appointment' object. On the left, there's a sidebar with various setup options like Buttons, Links, and Actions, Compact Layouts, Field Sets, etc. The 'Validation Rules' tab is selected and highlighted with a green box. In the main area, a table titled 'Validation Rules' lists one item: 'Vehicle'. The table has columns for Rule Name, Error Location, Error Message, Active status, and Modified By. A red box highlights the 'New' button in the top right corner of the table header.

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}"))`

The screenshot shows the 'Validation Rule Edit' screen for the 'Vehicle' rule. At the top, there are buttons for Save, Save & New, and Cancel. The 'Rule Name' field contains 'Vehicle'. The 'Active' checkbox is checked. The 'Description' field contains 'vehicle'. Below these, the 'Error Condition Formula' section shows the formula `NOT(REGEX(Vehicle_number_plate_c , "[A-Z]{2}[0-9]{2}[A-Z]{2}[0-9]{4}"))`. A green box highlights this formula. A red box highlights the 'Check Syntax' button at the bottom left. To the right, a dropdown menu for functions is open, showing options like ABS, ACOS, ADDMONTHS, AND, ASCII, ASIN, etc. A red box highlights the 'Insert Selected Function' button in the dropdown. A tooltip for the ABS function is visible, stating 'Returns the absolute value of a number, a number without its sign'.

5. Enter the Error Message as “Please enter valid number ”, select the Error location as Field and select the field as “Vehicle number plate”, and click Save.

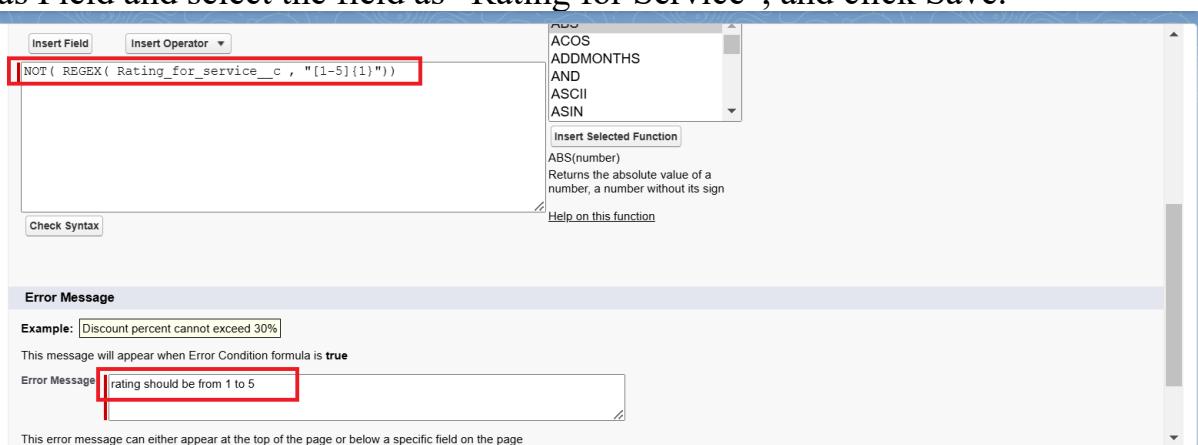


Activity 2: Create a validation rule for Billing details and feedback object:

1. Go to the setup page >> click on object manager >> From drop down click edit for Billing details and feedback object.
2. Click on the validation rule >> click New.
3. Enter the Rule name as “ rating_should_be_less_than_5”.
4. Insert the Error Condition Formula as :-
NOT(REGEX(Rating_for_service_c , "[1-5]{1}"))



5. Enter the Error Message as “rating should be from 1 to 5”, select the Error location as Field and select the field as “Rating for Service”, and click Save.



DUPLICATE RULE:

Activity 1: Create a matching rule to an customer details object:

To create a matching rule to an Customer details Object

1. Go to quick find box in setup and search for matching Rule.
2. Click on matching rule >> click on New Rule.

The screenshot shows the 'Matching Rules' page in the Salesforce Setup. The sidebar on the left has a 'Matching Rules' link with a green arrow pointing to it. The main area is titled 'All Matching Rules' and contains a table with columns: Action, Rule Name, Object, Status, Description, Last Modified Date, and Last Modified By. At the top of the table, there is a 'New Rule' button with a green arrow pointing to it. The URL in the browser bar is /setup/objects/matchingrules.

3. Select the object as Customer details and click Next.

The screenshot shows the 'Step 1: Select object' page. It has a header 'Step 1 of 2' and buttons 'Next' and 'Cancel'. Below the header, it says 'Select the object to which this matching rule applies.' and shows a dropdown menu labeled 'Object' with 'Customer Details' selected. There is also a green arrow pointing to the 'Next' button.

4. Give the Rule name : Matching customer details
5. Unique name : is auto populated
6. Define the matching criteria as
7. Field Matching Method
 1. Gmail
 2. Phone Number
8. Click save.
9. After Saving Click on Activate.

The screenshot shows the 'Rule Details' and 'Matching Criteria' pages. The 'Rule Details' section has fields for Object (Customer Details), Rule Name ('matching Customer detail' with a green arrow), Unique Name ('matching_Customer_det' with a green arrow), and Description. The 'Matching Criteria' section has a table with columns: Field, Matching Method, Match Blank Fields, and AND. It lists fields: Gmail (Exact), Phone Number (Exact), and three rows of '--None--' (Exact). There is a green arrow pointing to the 'Field' column in this section. At the bottom, there is a 'Save' button with a green arrow pointing to it.

Activity 2: Create a duplicate rule to an customer details object:

To create a Duplicate rule to an Customer details Object

1. Go to quick find box in setup and search for Duplicate rules.
2. Click on Duplicate rule >> click on New Rule >> select customer details object.

The screenshot shows the 'Duplicate Rules' page in Salesforce. The left sidebar has a 'Data' section with 'Duplicate Management', 'Duplicate Error Logs', and 'Duplicate Rules' (which is selected and highlighted with a green box). The main area shows a table of 'All Duplicate Rules'. One rule, 'Customer Detail duplicate', is highlighted with a green box. The table columns include 'Rule Name', 'Description', 'Matching Rule', 'Active', 'Last Modified By', and 'Last Modified Date'. The 'Customer Detail duplicate' rule is described as 'Identify accounts that duplicate other accounts' and uses the 'Matching Customer details' matching rule, last modified by user 'r2' on 10/10/2023.

3. Give the Rule name as : Customer Detail duplicate
4. Scroll a little in Matching rule section
5. Select the matching rule : Matching customer details
6. And Click on save.
7. After saving the Duplicate Rule, Click on Activate.

The screenshot shows the 'Edit Duplicate Rule' page for 'Customer Detail duplicate'. The 'Rule Details' section includes fields for 'Rule Name' (Customer Detail duplicate), 'Description' (Customer Details), 'Object' (Customer Details), and 'Record-Level Security' (Enforce sharing rules selected). The 'Actions' section defines actions on create and edit. The 'Matching Rules' section shows 'Compare Customer Details With' set to 'Customer Details' and 'Matching Rule' set to 'matching Customer details'. The 'Conditions' section displays a table of filter logic with four rows, each with 'Field' (all 'None'), 'Operator' (all '--None--'), and 'Value' (empty) columns, followed by an 'AND' operator. At the bottom are 'Save' and 'Save & New' buttons.

PROFILES:

Activity 1: Create a manager profile:

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.

The screenshot shows the 'Clone Profile' screen in the Salesforce Setup. The left sidebar has a 'Profiles' link highlighted with a red arrow. The main form has a dropdown 'Existing Profile' set to 'Standard User', a 'User License' field showing 'Salesforce', and a 'Profile Name' field containing 'Manager'. A red box surrounds these three fields. A red arrow points to the 'Save' button at the bottom right of the form.

2. While still on the profile page, then click Edit.

The screenshot shows the 'Profile Detail' page for the 'Manager' profile. The 'Edit' button is highlighted with a red box. Other visible fields include 'Name' (Manager), 'User License' (Salesforce), 'Description', 'Created By' (sunny_1, 13/06/2023, 2:40 pm), and 'Modified By' (sunny_1, 13/06/2023, 2:40 pm). A red box also highlights the 'Custom Profile' checkbox which is checked.

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.

The screenshot shows the 'Custom Object Permissions' page. It displays permission matrices for various objects: Appointments, Billing details and feedback, Customer Details, Environments, Laptops, Service records, and SessionData. The columns represent basic access (Read, Create, Edit, Delete) and data administration (View All, Modify All). Checkmarks indicate granted permissions, with some highlighted with red boxes.

	Basic Access				Data Administration	
	Read	Create	Edit	Delete	View All	Modify All
Appointments	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Billing details and feedback	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Customer Details	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Environments	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Laptops	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Service records	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
SessionData	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

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.

Activity 2: Create a salesperson 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	<input checked="" type="checkbox"/>	<input type="checkbox"/>				
Billing details and feedback	<input checked="" type="checkbox"/>	<input type="checkbox"/>				
Customer Details	<input checked="" type="checkbox"/>	<input type="checkbox"/>				
Environments	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Custom Object Permissions						
	Basic Access				Data Administration	
	Read	Create	Edit	Delete	View All	Modify All
Laptops	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Service records	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
SessionData	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

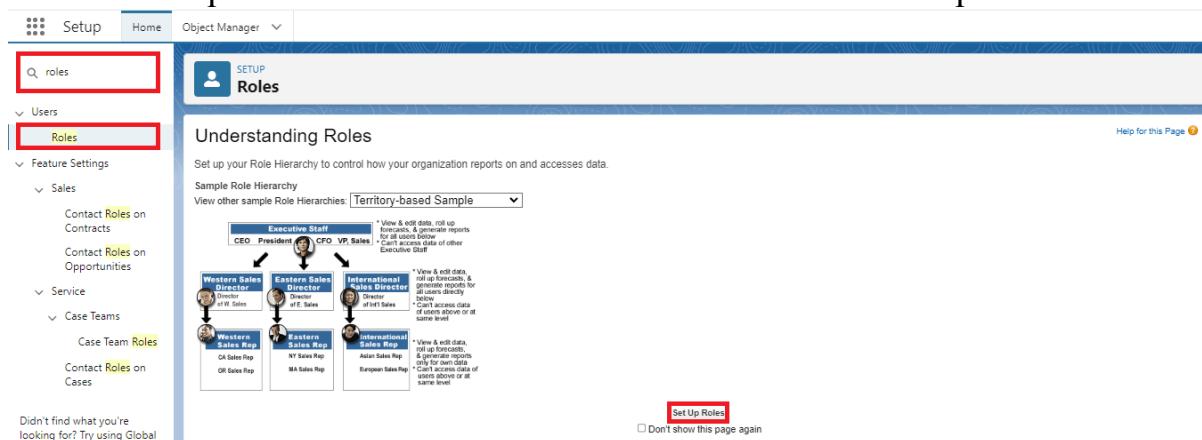
5. And click save.

ROLE & Role HIERARCHY:

Activity 1: Creating Manager Role:

Creating Manager Role:

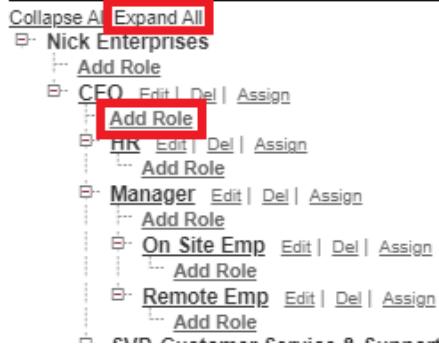
1. Go to quick find >> Search for Roles >> click on set up roles.



The screenshot shows the Salesforce Setup interface with the 'Roles' page selected. The left sidebar has 'roles' highlighted under 'Users'. The main content area displays a 'Territory-based Sample' role hierarchy diagram. At the top level are 'Executive Staff' (CEO, President, CPO, VP_Sales). Below them are 'Western Sales Director', 'Eastern Sales Director', and 'International Sales Director'. Under each director are specific sales representatives like 'Western Sales Rep' and 'Eastern Sales Rep'. A legend explains the icons: a blue square for 'View & edit data, not see forecasts & generate reports for all users below', a green square for 'View & edit data, & see forecasts for all users below', and a yellow square for 'View & edit data, & all up forecasts & generate reports for own data only'. A 'Set Up Roles' button is located at the bottom right of the page.

2. Click on Expand All and click on add role under whom this role works.

Your Organization's Role Hierarchy



3. Give Label as “Manager” and Role name gets auto populated. Then click on Save.

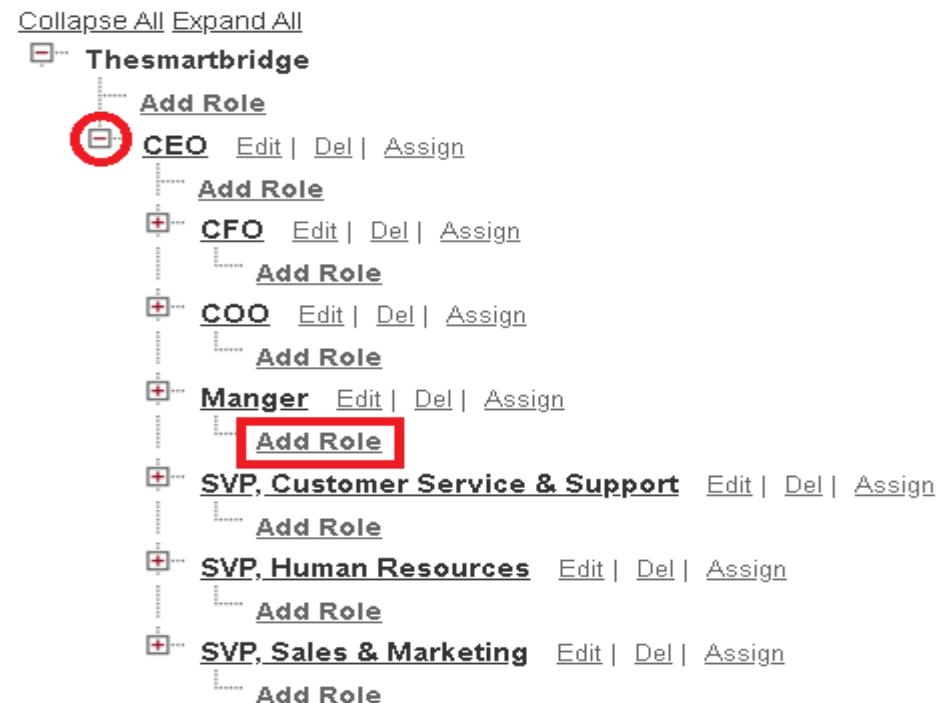
The screenshot shows the 'Role Edit' dialog box. It contains fields for 'Label' (set to 'Manger'), 'Role Name' (auto-filled with 'Manger'), and 'This role reports to' (set to 'CEO'). Below these fields is a note 'Role Name as displayed on reports'. At the bottom are 'Save', 'Save & New', and 'Cancel' buttons. Red arrows point to the 'Label' field and the 'Save' button.

Label	Manger
Role Name	Manger
This role reports to	CEO
Role Name as displayed on reports	
<input type="button" value="Save"/> <input type="button" value="Save & New"/> <input type="button" value="Cancel"/>	

Activity 2: Creating another roles:

Creating another two roles under manager

1. Go to quick find >> Search for Roles >> click on set up roles.
2. Click plus on CEO role, and click add role under manager.



3. Give Label as “sales person” and Role name gets auto populated. Then click on Save.

USERS:

Activity 1: Create Users:

1. Go to setup >> type users in quick find box >> select users >> click New user.
2. Fill in the fields
1. First Name : Niklaus
2. Last Name : Mikaelson
3. Alias : Give a Alias Name
4. Email id : Give your Personal Email id
5. Username : Username should be in this form: text@text.text
6. Nick Name : Give a Nickname
7. Role : Manager
8. User licence : Salesforce
9. Profiles : Manager

New User

User Edit Save Save & New Cancel

General Information

First Name	Niklaus
Last Name	Mikaelson
Alias	nmika
Email	
Username	Mikaelson@Niklaus
Nickname	nik
Title	
Company	
Department	
Division	

Required Information

Role	Manager
User License	Salesforce
Profile	Manager
Active	<input checked="" type="checkbox"/>
Marketing User	<input type="checkbox"/>
Offline User	<input type="checkbox"/>
Knowledge User	<input type="checkbox"/>
Flow User	<input type="checkbox"/>
Service Cloud User	<input type="checkbox"/>
Site.com Contributor User	<input type="checkbox"/>
Site.com Publisher User	<input type="checkbox"/>
WDC User	<input type="checkbox"/>
Data.com User Type	-None-

3. Save.

Activity 2: Creating Another User:

Repeat the steps and create another user using

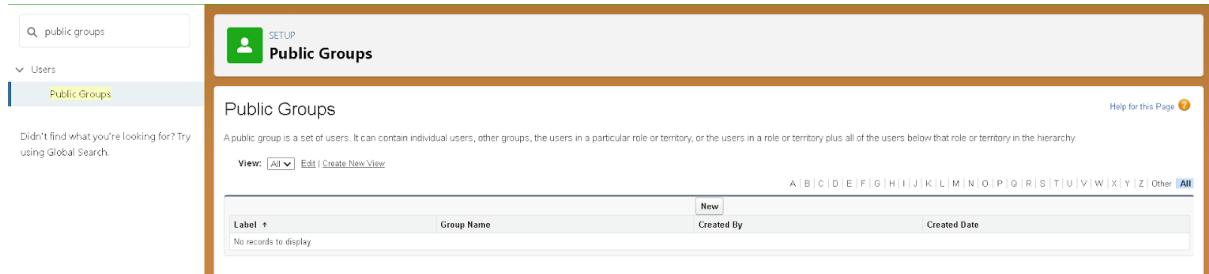
- a. Role : sales person
- b. User licence : Salesforce Platform
- c. Profile : sales person

Note : create atleast 3 users with these permissions.

PUBLIC GROUPS:

Activity 1: Creating new public groups:

1. Go to setup >> type users in quick find box >> select public groups >> click New.

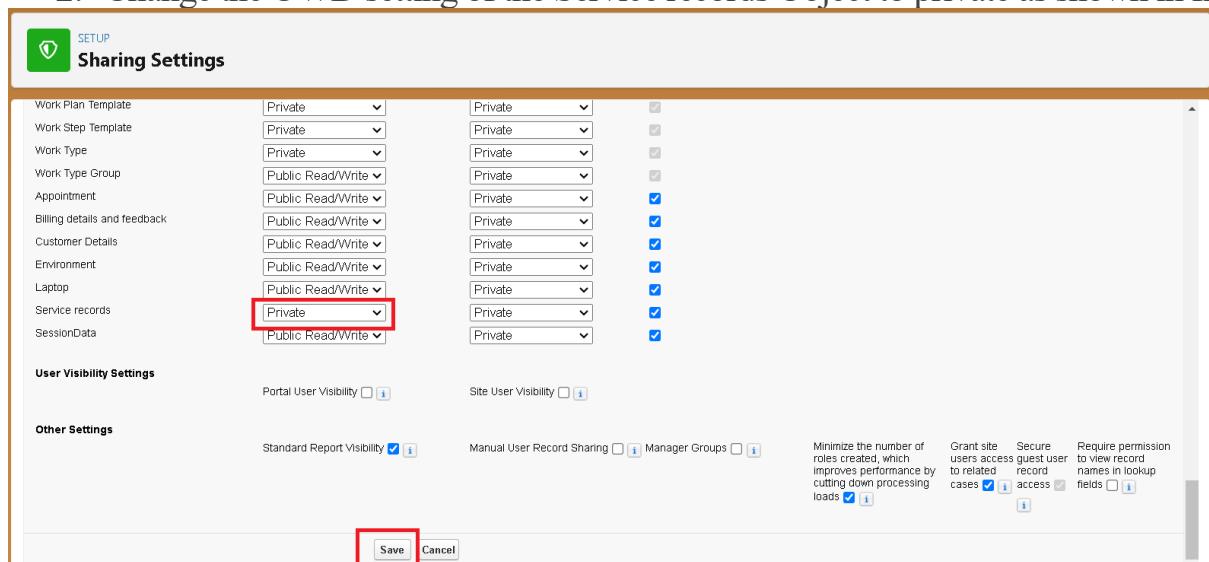


2. Give the Label as “sales team”.
3. Group name is autopopulated.
4. Search for Roles.
5. In Available Members select Sales person and click on add it will be moved to selected member.
6. Click on save.

SHARING SETTINGS:

Activity 1: Create the sharing settings:

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.



3. Click on save and refresh.

4. Scroll down a bit, Click new on Service records sharing Rules.

5.

Service records Sharing Rules

New Recalculate

No sharing rules specified.

Service records Sharing Rules Help ?

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

Label: sharing settings

Rule Name: sharing_settings

Description:

Step 2: Select your rule type

Rule Type: Based on record owner

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

Save Cancel

FLOWs:

Activity 1: Create a flow:

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

1

2

3

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

3.

1

2

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.
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.

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.

6.

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

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.

3

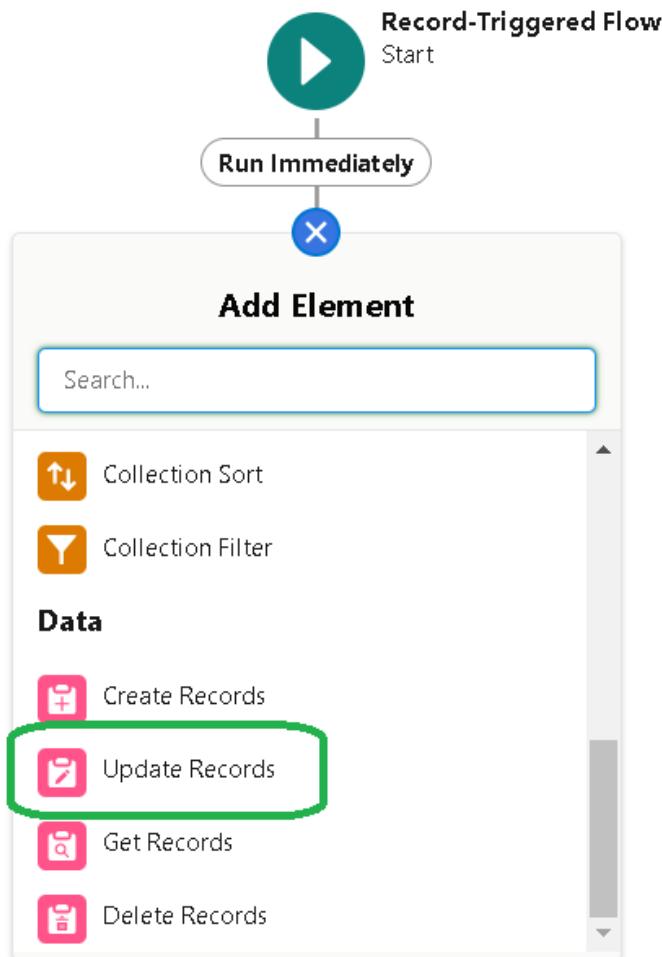
Include a Run Asynchronously path to access an external system after the original transaction for the triggering record is successfully committed

4

Cancel

Done

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	*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

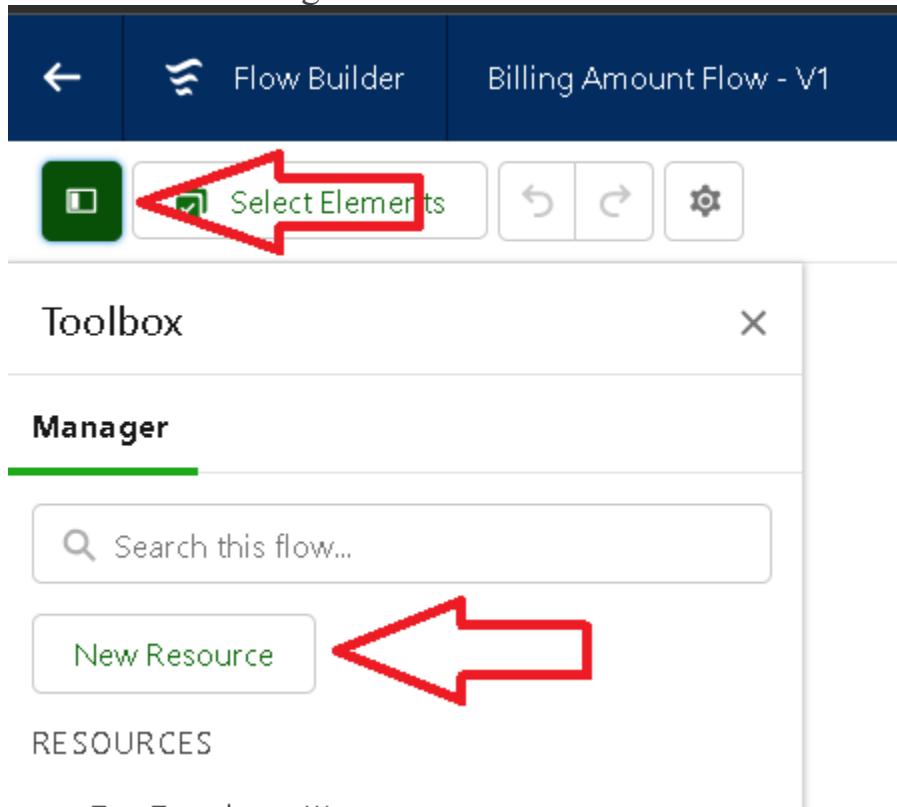
Field	Value
Payment_Paid__c	\$Record > Service records > Appointment > Service A... X

+ Add Field

Cancel Done

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.

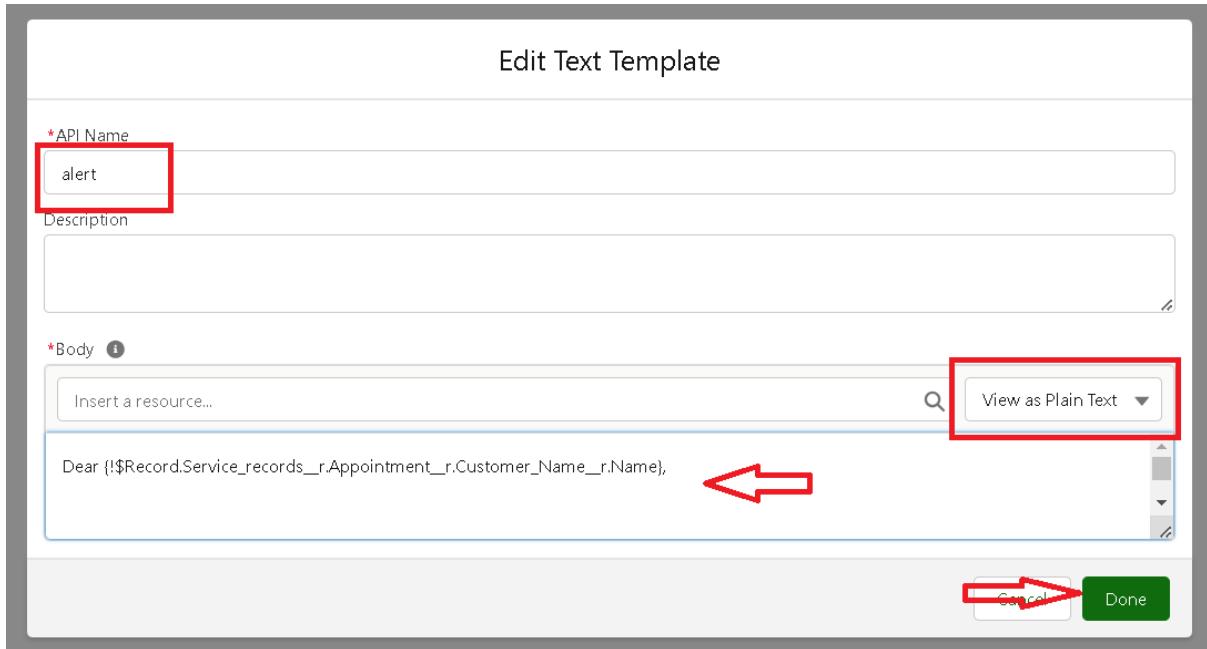
Dear {!\$Record.Service_records__r.Appointment__r.Customer_Name__r.Name},

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.



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 <div style="border: 1px solid #ccc; height: 50px; width: 100%;"></div>	

Set Input Values for the Selected Action

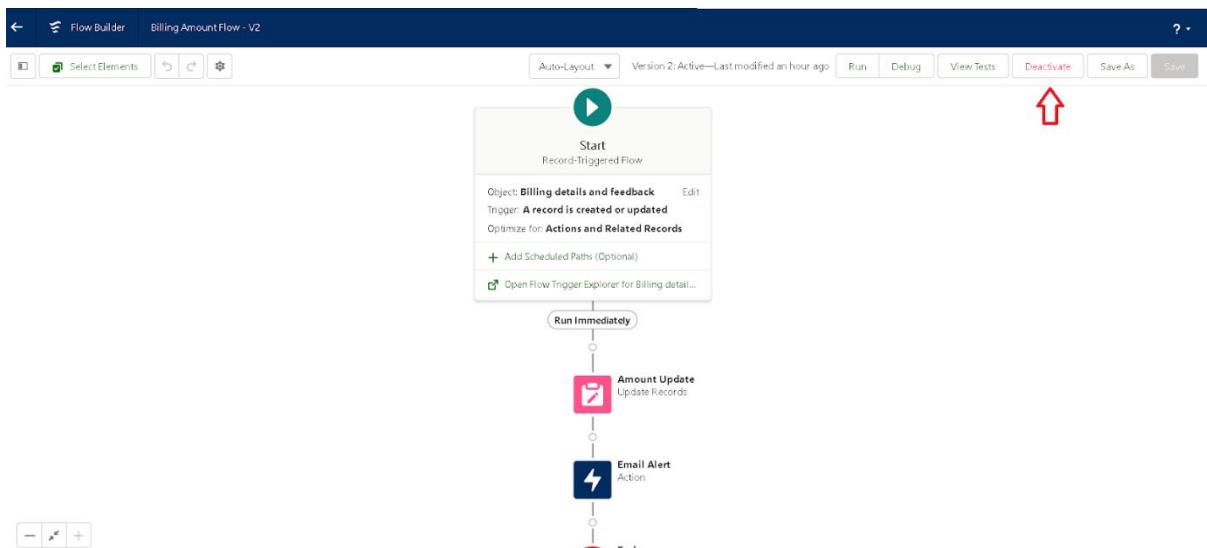
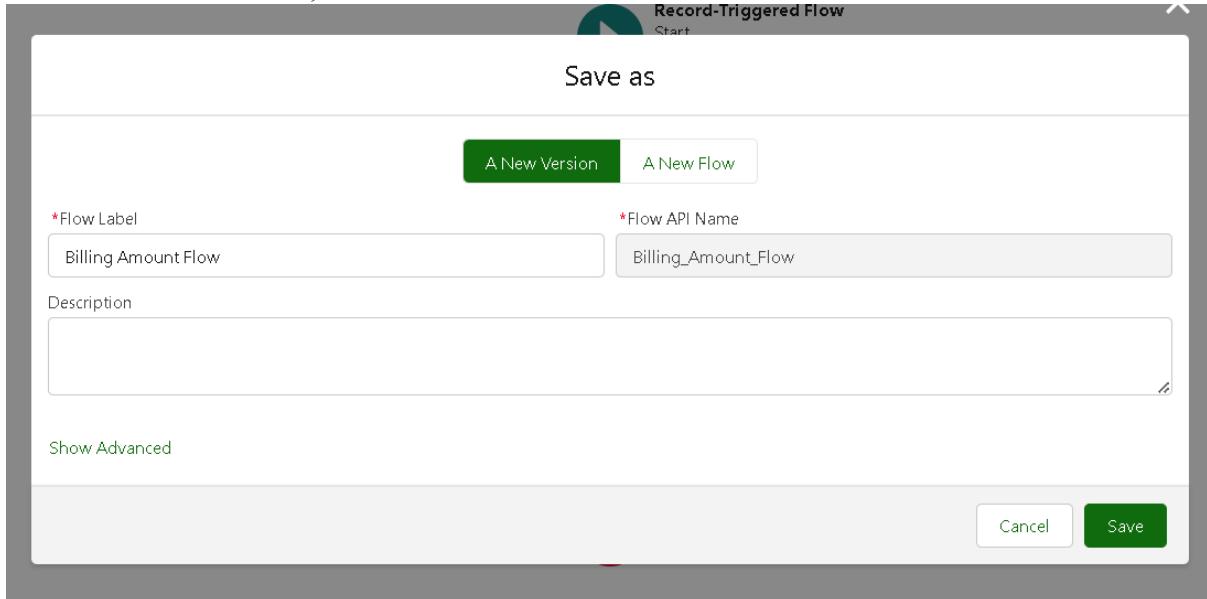
A_a Body {!alert}	<input checked="" type="checkbox"/> Include <input type="checkbox"/> Don't Include
A_a Email Template ID	<input type="checkbox"/> Don't Include
A_a Log Email on Send	<input type="checkbox"/> Don't Include

Edit Action

A_a Recipient Address List ={!\$Record.Service_records__r.Appointment__r.Cus}	<input checked="" type="checkbox"/> Include <input type="checkbox"/> Don't Include
A_a Recipient ID	<input type="checkbox"/> Don't Include
A_a Related Record ID	<input type="checkbox"/> Don't Include
A_a Rich-Text-Formatted Body	<input type="checkbox"/> Don't Include
A_a Sender Email Address	<input type="checkbox"/> Don't Include
A_a Sender Type	<input type="checkbox"/> Don't Include
A_a Subject Thank You for Your Payment - Garage Manageme	<input checked="" type="checkbox"/> Include <input type="checkbox"/> Don't Include

34. Click on save. Give the Flow label , Flow Api name will be autopopulated.

35. And click save, and click on activate.



Activity 2: Create another flow:

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

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

3. Select the Object as “Service records” in the Drop down list.
4. Select the Trigger Flow when: “A record is Created or Updated”.
5. Select the Optimise the flow for: “Actions and Related Records” and Click on Done.
6. Under the Record-triggered Flow Click on “+” Symbol and In the Drop down List select the “Update records Element”.
7. Set a filter condition : All Conditions are met(AND)
8. Field : Quality_Check_Status__c
9. Operator : Equals
10. Value : True
11. And Set Field Values for the Billing details and feedback Record
12. Field : Service_Status__c
13. Value : Completed

Set Filter Conditions

Condition Requirements to Update Record

All Conditions Are Met (AND)

Field

Quality_Check_Status__c

Operator

Equals

Value

True

+ Add Condition

Set Field Values for the Service record Record

Field

Service_Status__c

Value

Completed

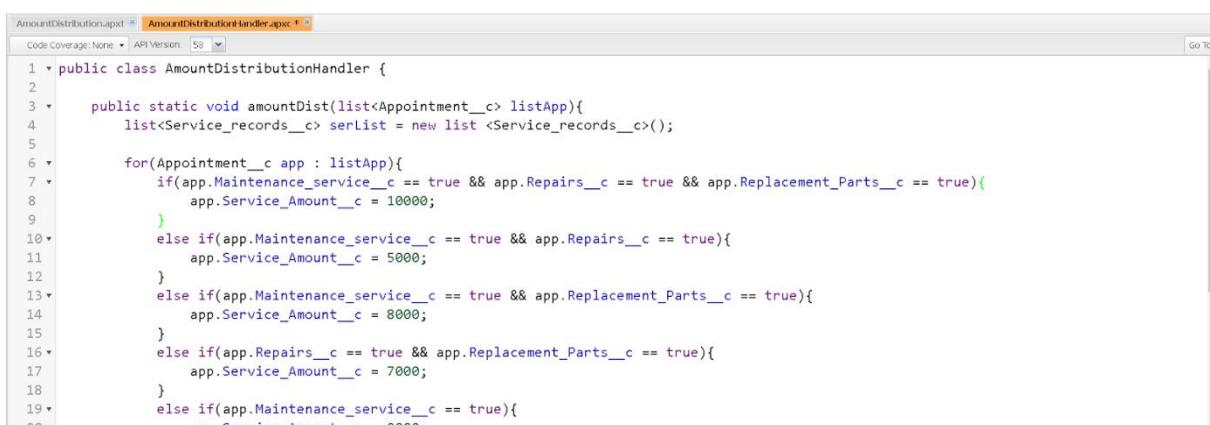
+ Add Field

14. Click On Done.
15. Click on save
16. Given the Flow label as Update Service Status , Flow Api name will be auto populated.
17. And click save, and click on activate

APEX TRIGGER:

Activity 1: Create a apex handler:

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”.



```
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}
```

```

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 }

```

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;
    }

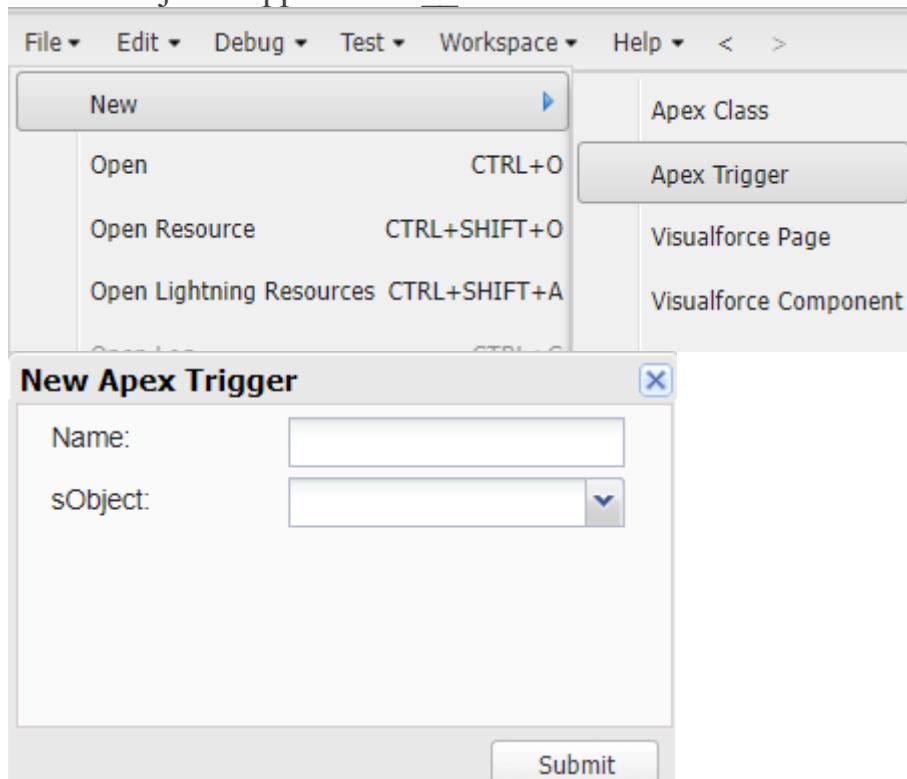
}
}
}

```

Trigger Handler :

How to create a new trigger :

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 :

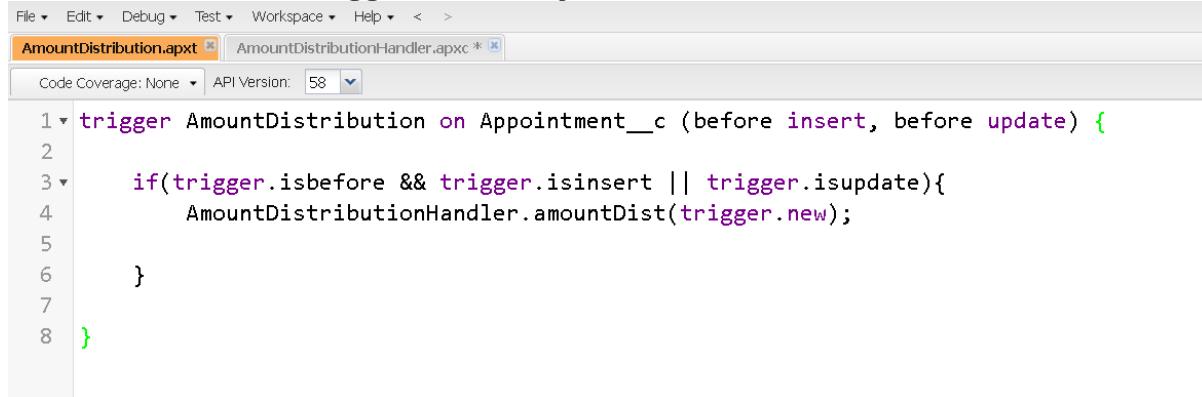
The syntax for creating trigger is :

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



The screenshot shows the Salesforce IDE interface with the following details:

- File menu: File, Edit, Debug, Test, Workspace, Help.
- Toolbar: Standard icons for file operations.
- Tab bar: AmountDistribution.apxt (highlighted in orange), AmountDistributionHandler.apxc *.
- Status bar: Code Coverage: None, API Version: 58.
- Code editor:

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

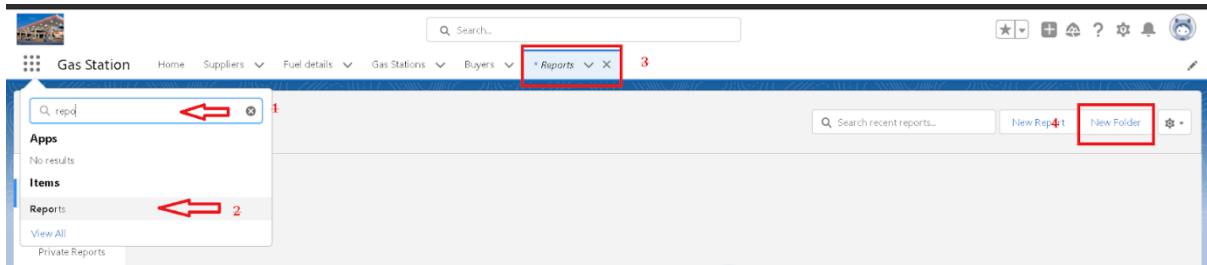
Code:

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

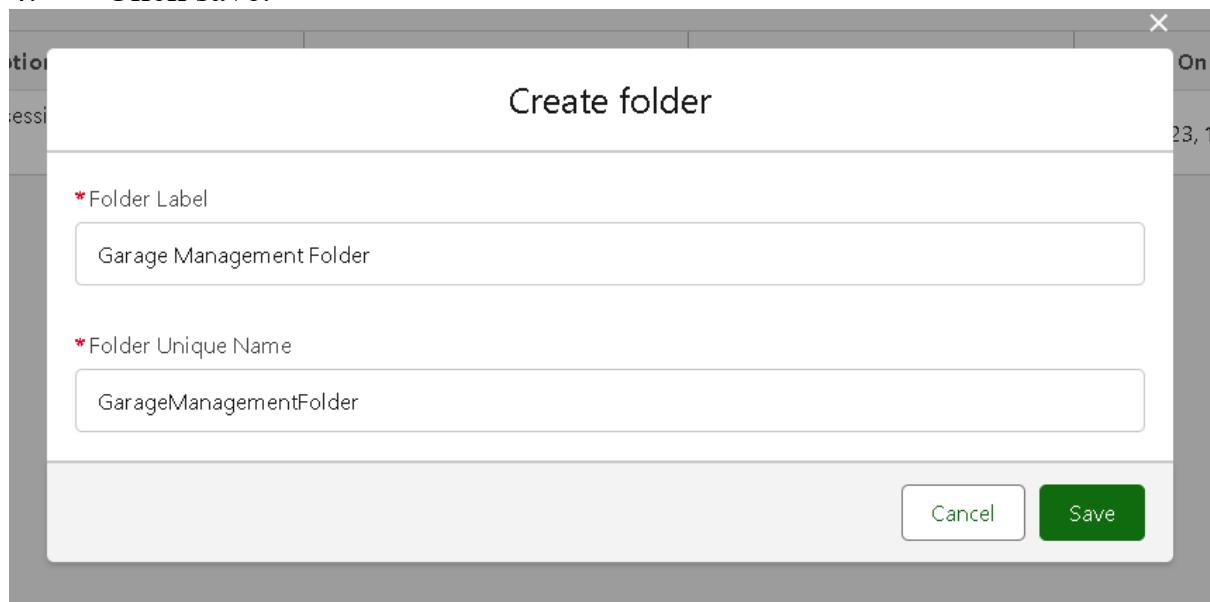
REPORTS:

Activity 1: Create a report folder:

1. Click on the app launcher and search for reports.
2. Click on the report tab, click on new folder.

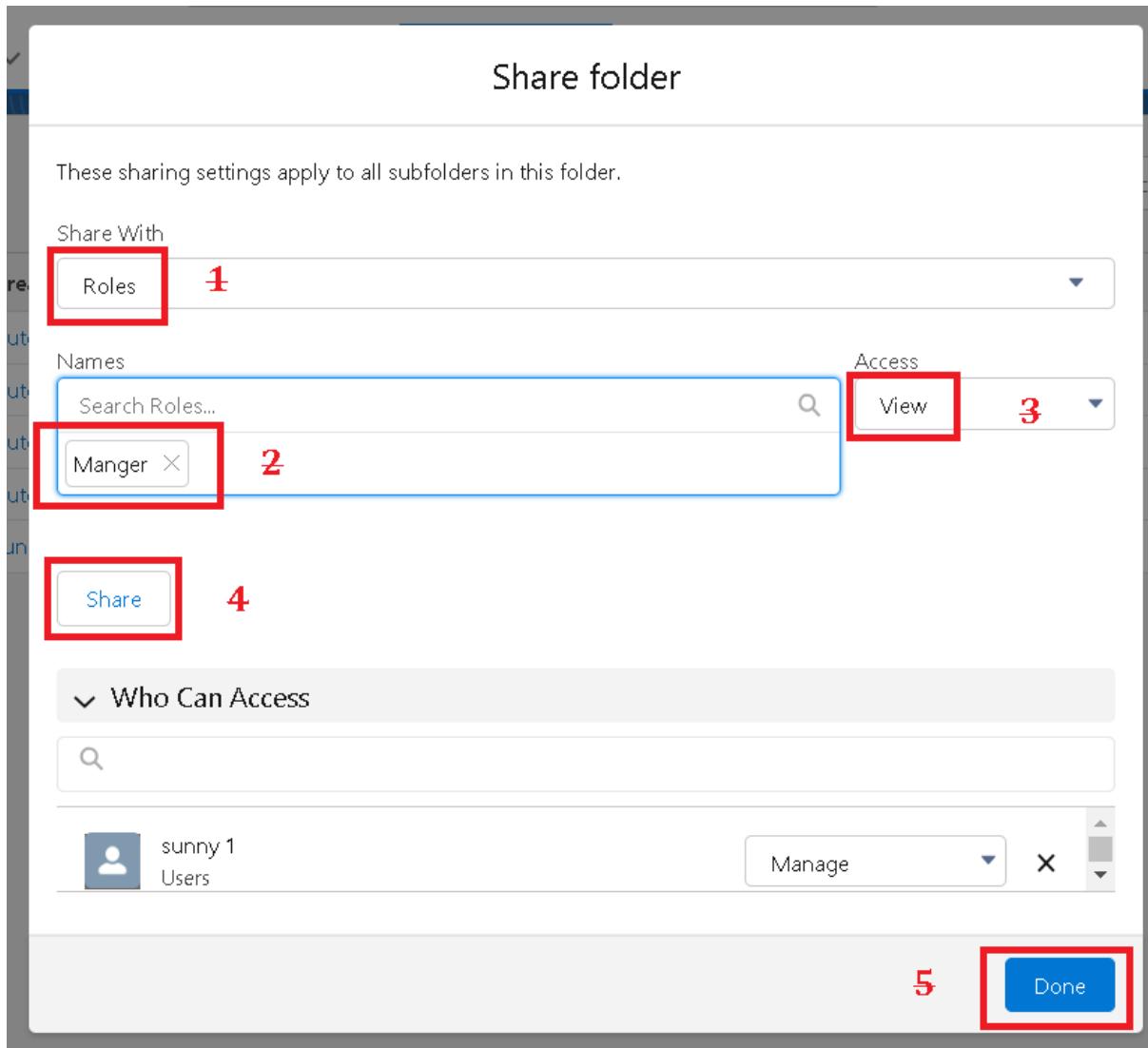


3. Give the Folder label as “Garage Management Folder”, Folder unique name will be auto populated.
4. Click save.



Activity 2: 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.



Activity 3: Create 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.

The screenshot shows the 'Report Types' page under the 'Analytics' section of the setup menu. It displays a list of existing report types and a 'New Custom Report Type' button. A green arrow points to the 'Report Types' link in the sidebar, and another green arrow points to the 'New Custom Report Type' button.

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 “ Deployed ”, click on Next.

Report Type Focus

Specify what type of records (rows) will be the focus of reports generated by this report type.

Example: If reporting on "Contacts with Opportunities with Partners," select "Contacts" as the primary object.

Identification

- Primary Object: Customer Details
- Report Type Label: Service information
- Report Type Name: Service_information
- Description: Service information
- Store in Category: Other Reports

Deployment

A report type with deployed status is available for use in the report wizard. While in development, report types are visible only to authorized administrators and their delegates.

- Deployment Status: Deployed

Next **Cancel**

9. now , Click on Related object box.
10. Click on Select Object, choose Appointment Object as shown in fig.

New Custom Report Type
Service information

Step 2. Define Report Records Set Step 2 of 2

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 Select Object

At one related "B" record, related "B" records.

Activities
Appointments

Diagram: A Venn diagram with two overlapping circles labeled A and B. Below it, a downward arrow points to a grid divided into two sections, A and B, each containing horizontal bars.

Buttons: Previous, Save, Cancel

Step 2. Define Report Records Set

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.

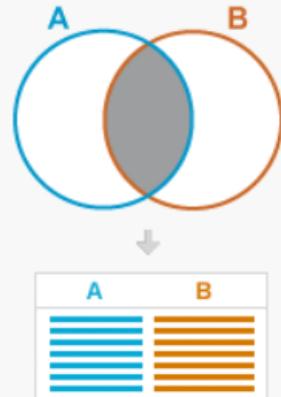
A Customer Details Primary Object

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.

(Click to relate another object)



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.

SETUP 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

Activity 4: Create a report:

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

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

4. Their outline pane is opened already, select the fields that mentioned below in column section.

- Customer name
- Appointment Date
- Service Status
- Payment paid

5. Remove the unnecessary fields.

6. Select the fields that mentioned below in GROUP ROWS section.

- Rating for Service

7. Select the fields that mentioned below in GROUP ROWS section.

- 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.

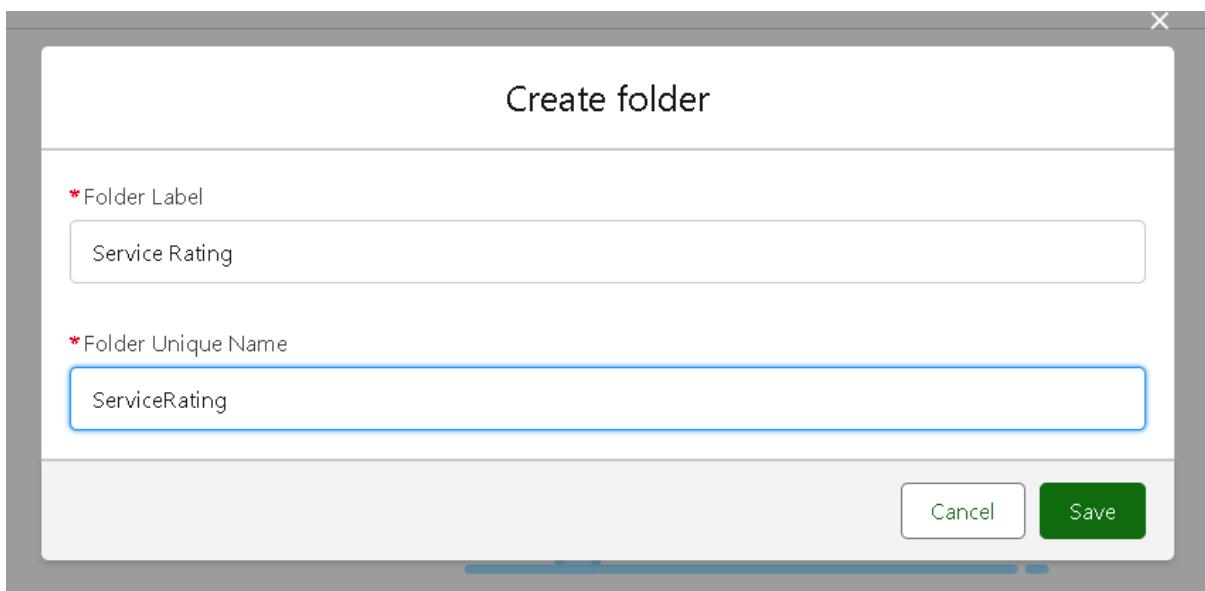
The screenshot shows a report preview interface with various sections:

- REPORT**: New Service information Report
- Fields** section: Groups (GROUP ROWS, Add group...), Rating for service, Payment Status.
- Columns** section: Customer Name, Appointment Date, Service Status, # Payment Paid.
- Table Preview**: A table showing data grouped by Rating for service (4, 5) with columns: Sum of Payment Paid, Record Count, Total.
- Chart**: A line chart titled "Sum of Payment Paid" vs "Rating for service".
- Save Report** dialog box:
 - *Report Name: New Service information Report (highlighted with a green arrow).
 - Report Unique Name: New_Service_information_Report_oVu
 - Report Description: (empty text area)
 - Folder: Garage Management Folder (highlighted with a green arrow).
 - Buttons: Cancel, Save.

DASHBOARDS:

Activity 1: Create dashboard folder:

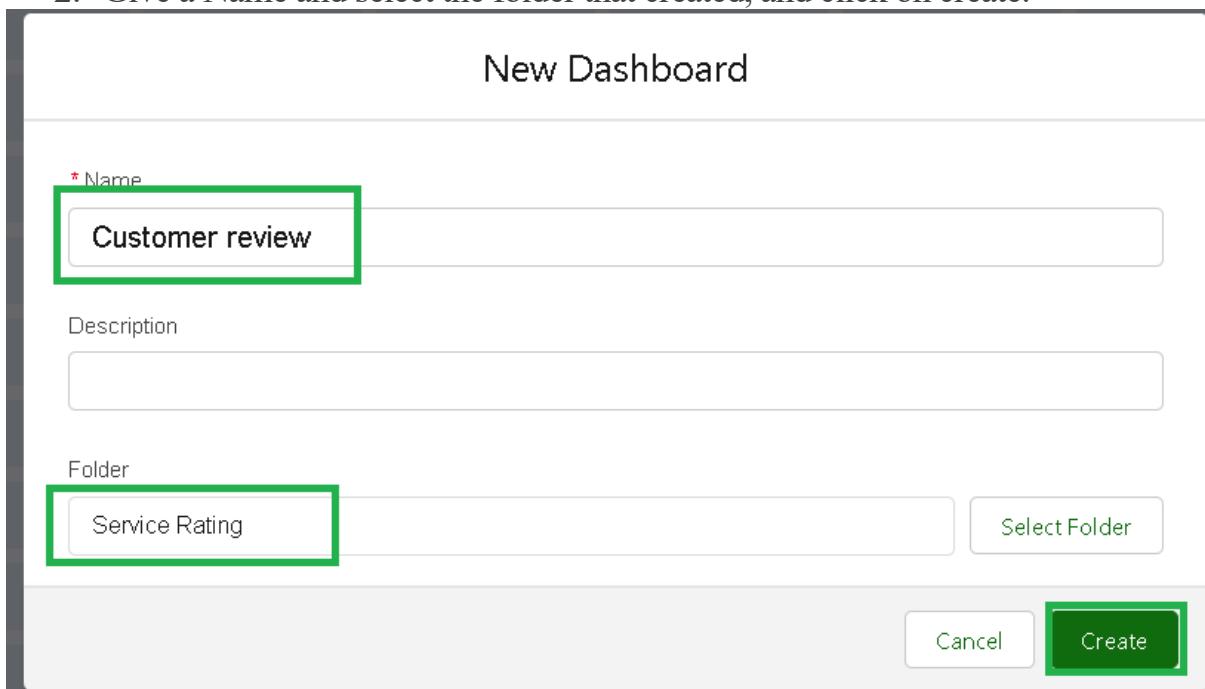
1. Click on the app launcher and search for dashboard.
2. Click on dashboard tab.
3. Click new folder, give the folder label as “ Service Rating dashboard”.
4. Folder unique name will be auto populated.
5. Click save.



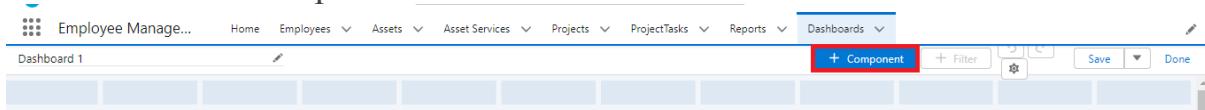
6. Follow the same steps, from Reports Milestone and Activity 2, and provide the sharing settings for the folder that was just created.

Activity 1: Create a dashboard:

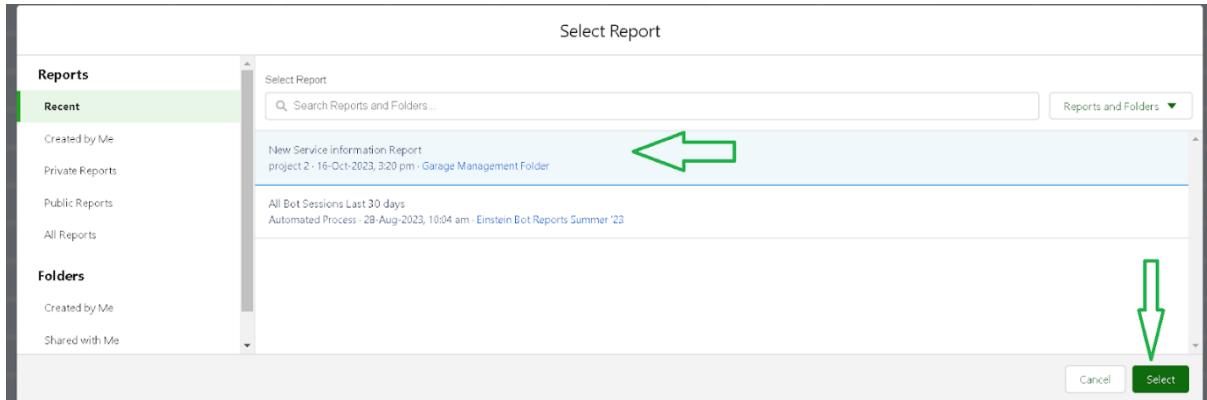
1. Go to the app >> click on the Dashboards tabs.
2. Give a Name and select the folder that created, and click on create.



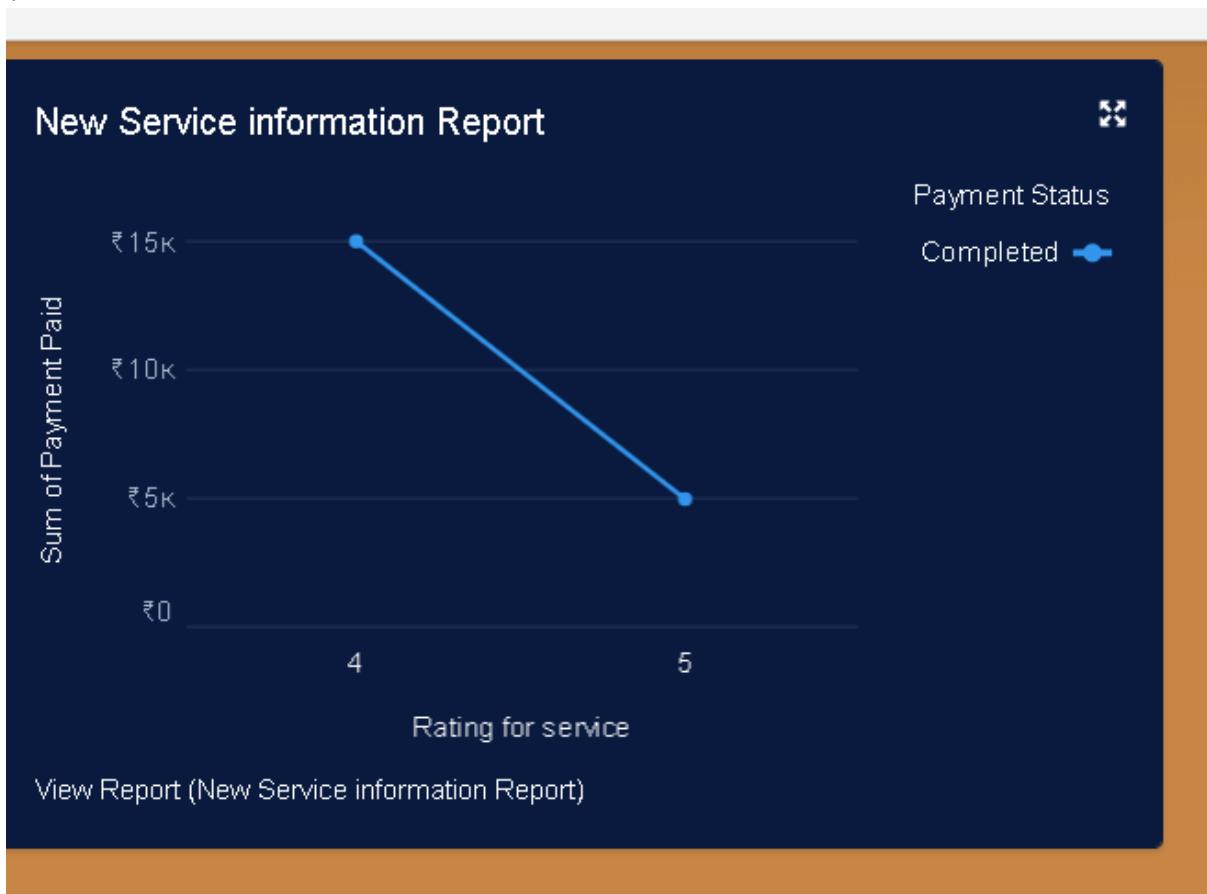
3. Select add component.



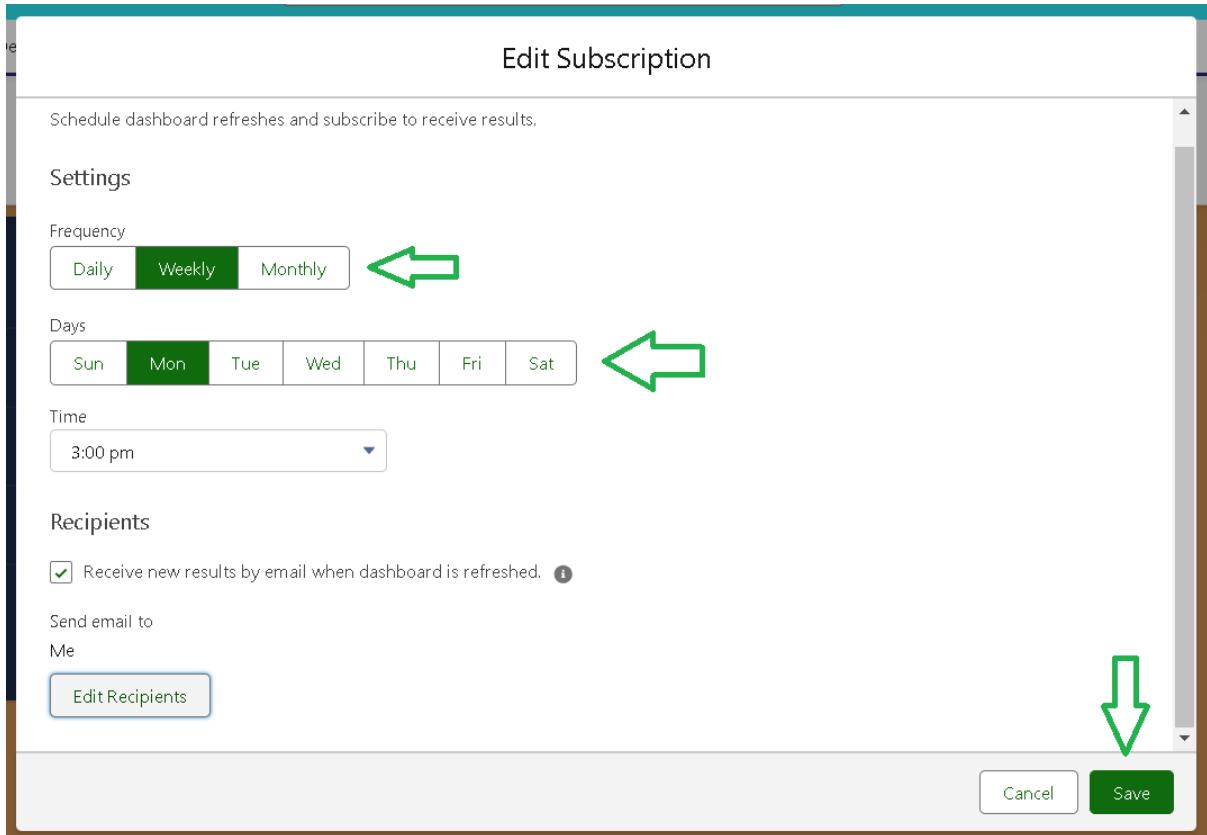
4. Select a Report and click on select.



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 Subscribe on top right.
9. Set the Frequency as “ weekly ”.
10. Set a day as monday.
11. And Click on save.



USER ADOPTION:

Activity 1: Creating Records:

To create a record in the follow objects follow these steps

1. Click on the app launcher located at the left side of the screen.
2. Search for “**Garage Management**” and click on it.
3. Click on the “**Consumer details tab**”.
4. Click on new and fill the details as shown below figs, and click save.

New Customer Detail

* = Required Information

Information

* Customer Name	Mac	Owner	Annapurna SmartBridge
Phone number	5678765567		
Gmail	mac@gmail.com		

Cancel **Save & New** **Save**

Now, Create the Appointment Record

1. Click on the “**Appointment** tab”.
2. Enter the customer details as created, while entering Appointment Date enter the date less than the created date.
3. Match the validation while entering the vehicle number plate.
4. Select the services you need.
5. Click on save to see the Service Amount.

Garage Management... 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	<input checked="" type="checkbox"/>		
Repairs	<input checked="" type="checkbox"/>		
Replacement Parts	<input type="checkbox"/>		
Service Amount			
* Vehicle number plate	TS30EU0443		
Created By	Annapurna SmartBridge, 18/11/2024, 3:28 pm	Edited By	Annapurna SmartBridge, 18/11/2024, 3:28 pm

Cancel **Save**

Now, Create a service Record

1. Click on the “**Service record** tab”.

2. Enter the Appointment, and started is selected as default.
3. Click on save.

New Service record

* = Required Information

Information

Service Record Name

Owner

* Appointment

app-016

Quality Check Status

Service Status

Started

Cancel Save & New Save

4. Open the record and click on Quality check status as true.
5. Click on save.

Service Record Name

Owner

ser-010

Annapurna SmartBridge

* Appointment

app-016

Quality Check Status

Service Status

Started

service date

18/11/2024

This field is calculated upon save

Created By

Updated By

Annapurna SmartBridge, 18/11/2024, 4:32 pm

Annapurna SmartBridge, 18/11/2024, 4:32 pm

Cancel Save

6. Now automatically Service status will be moved to completed.

Conclusion:

By leveraging the Salesforce platform, the project successfully established a streamlined and transparent system for managing garage operations. Through efficient coordination between customers, service advisors, mechanics, and inventory managers, the project effectively addressed common challenges such as job tracking, spare parts shortages, and billing delays.

The project “Garage Management System using Salesforce” has been successfully implemented and demonstrates the practical application of Salesforce CRM in the automotive service industry.

Project Achievements:

- Streamlined the process of vehicle service booking, job allocation, and tracking.
- Ensured real-time coordination among customers, mechanics, and service advisors.
- Automated workflows for service requests, approvals, and billing through custom objects, fields, Flows, and Apex triggers.
- Improved transparency with service history, invoices, and payment records using reports and dashboards.
- Enhanced usability with Lightning Apps, role-based security, and customized home pages for different stakeholders.

Student Learning Outcomes:

- Gained hands-on skills in Salesforce development, CRM customization, and automation.
- Developed problem-solving abilities by implementing real-time use cases in service management.
- Improved team collaboration in requirement gathering, development, and testing.
- Acquired industry-relevant exposure to CRM applications in the automotive sector, including project lifecycle management.

Future Scope:

- Mobile Integration: Enable customers to book services and track status via mobile apps.
- AI and Analytics: Use predictive analytics to forecast demand for spare parts and optimize resource allocation.
- Integration with IoT: Connect with vehicle diagnostic systems for automated service recommendations.
- Multi-Branch Expansion: Extend the system to manage operations across multiple garage locations.
- Customer Engagement: Introduce loyalty programs, feedback systems, and personalized offers to enhance customer satisfaction.