# Garage Management System

By

Kukkala Mahender [mahenderkukkala336@gmail.com](mailto:mahenderkukkala336@gmail.com)

Username: [kukkalamahendergamma@mru.in](mailto:kukkalamahendergamma@mru.in)

# Project Abstract

The Salesforce-based Garage Management System (GMS) is a sophisticated solution tailored for automotive repair facilities to elevate service quality, streamline operations, and cultivate enduring customer relationships. Leveraging Salesforce's powerful CRM capabilities, GMS oﬀers an intuitive interface and comprehensive features, allowing garages to thrive in a competitive marketplace. The system automates critical processes such as appointment scheduling, inventory management, billing, and customer communications. By integrating these functionalities within Salesforce, GMS ensures a seamless and satisfying experience for both customers and staﬀ. This advanced solution empowers automotive repair businesses to deliver top-notch service, optimize workﬂows, and maintain a competitive edge.

# INDEX PAGE

|  |  |
| --- | --- |
| Topics | Page no |
| Task 1 | 5 |
| Task 2 | 6 - 8 |
| Task 3 | 9 - 12 |
| Task 4 | 13 |
| Task 5 | 14 - 19 |
| Task 6 | 19 - 21 |
| Task 7 | 21 - 22 |
| Task 8 | 22 - 23 |
| Task 9 | 24 |
| Task 10 | 26 - 27 |
| Task 11 | 27 |

|  |  |
| --- | --- |
| Task 12 | 28 |
| Task 13 | 29 - 30 |
| Task 14 | 31 - 32 |
| Task 15 | 33 - 35 |
| Task 16 | 35 - 37 |

**INTRODUCTION**

Running an automotive repair shop involves juggling many tasks, from scheduling appointments to managing inventory and billing customers. The Salesforce-based Garage Management System (GMS) is designed to make these tasks easier and more eﬃcient.

This system uses Salesforce's powerful tools to help garages deliver better service and keep operations running smoothly. With GMS, repair shops can automate important tasks like booking appointments, tracking inventory, sending bills, and communicating with customers.

The user-friendly design of GMS ensures that both staﬀ and customers have a pleasant experience. By using this system, garages can stay ahead of the competition, provide exent service, and manage their business more eﬀectively. GMS not only simpliﬁes daily operations but also provides valuable insights to help make better business decisions.

# TASK 1:

* 1. **Creating The Developer Account:**

1. To create a developer account in Salesforce, follow these steps:
2. Go to Salesforce Developer Signup.

Fill in the signup form with the following details:

First name & Last name Email

Role: Developer Company: College Name Country: India Postal Code: Pin code

Username: A combination of your name and company (format: [username@organization.com,](mailto:username@organization.com) not an actual email id)

Click Sign me up.

# Activating the Account:

* + - Check the inbox of the email you used for signup.
    - Click on the veriﬁcation link to activate your account (the email may take 5-10 minutes to arrive).
    - Click on **Verify Account**.
    - Set a password and answer a security question.
    - Click on **Change Password**.
    - You will be redirected to your Salesforce setup page.

# TASK 2:

**Creating the Objects:**This report outlines the steps to create various custom objects in Salesforce, speciﬁcally for a Garage Management System. The objects include Customer Details, Appointments, Service Records, and Billing Details and Feedback. These objects will help streamline operations and improve data management within the system.

# Creating Custom Objects

Customer Details Object

Navigation Path: From the setup page, click on Object Manager >> Create >> Custom Object. Details: Label Name: Customer Details

Plural Label Name: Customer Details Record Name: Customer Name

Data Type: Text

Options: Allow reports, Track Field History, Allow search Save: Click Save to create the object.

# Appointment Object

Navigation Path: From the setup page, click on Object Manager >> Create >> Custom Object. Details:

Label Name: Appointment

Plural Label Name: Appointments Record Name: Appointment Name Data Type: Auto Number

Display Format: app-{000} Starting Number: 1

Options: Allow reports, Track Field History, Allow search Save: Click Save to create the object.

# Service Records Object

Navigation Path: From the setup page, click on Object Manager >> Create >> Custom Object. Details:

Label Name: Service Records

Plural Label Name: Service Records Record Name: Service Records Name Data Type: Auto Number Display Format: ser-

{000} Starting

Number: 1

Options: Allow reports, Track Field History, Allow search Save: Click Save to create the object.

# Billing Details and Feedback Object

Navigation Path: From the setup page, click on Object Manager >> Create >> Custom Object. Details:

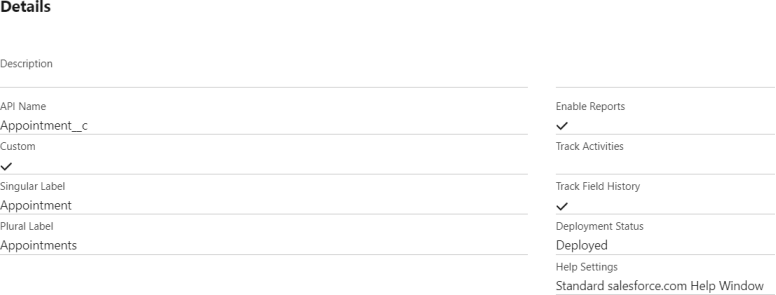
Label Name: Billing Details and Feedback

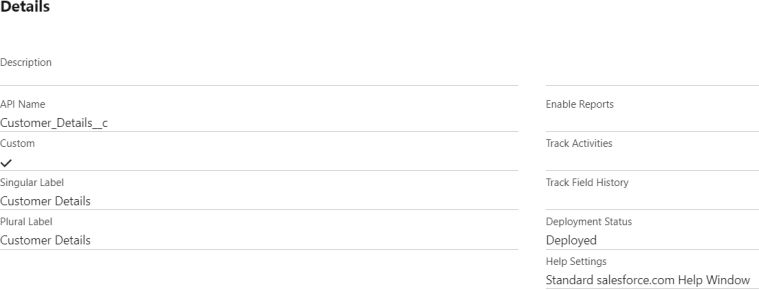
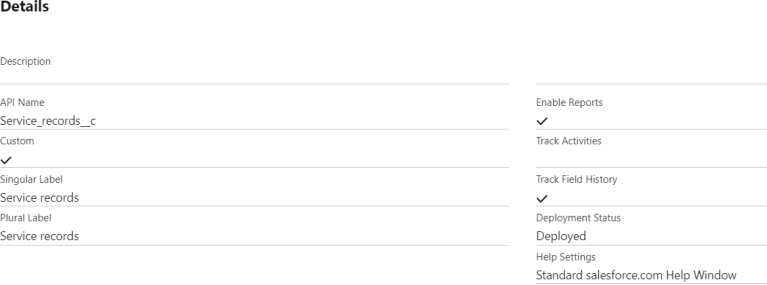
Plural Label Name: Billing Details and Feedback Record Name: Billing Details and Feedback Name Data Type: Auto Number Display Format: bill-

{000} Starting

Number: 1

Options: Allow reports, Track Field History, Allow search Save: Click Save to create the object.





# Task 3:

**Creating Tabs**

1. Creating a Custom Tab for Customer Details Navigation Path:

Go to the setup page.

Type "Tabs" in the Quick Find bar. Click on "Tabs".

Click on "New" under the Custom Object Tabs . Details:

Select Object: Customer Details

Select Tab Style: Choose a style that represents Customer Details. Click Next.

Add to Proﬁles Page: Keep the default settings. Click Next.

Add to Custom App: Uncheck "Include Tab".

Ensure "Append tab to users' existing personal customizations" is checked. Click Save.

Creating Remaining Tabs (Appointments, Service Records, Billing Details and Feedback)

Repeat the above steps for each remaining object: Appointments, Service Records, and Billing Details and Feedback.

Navigation Path:

Go to the setup page.

Type "Tabs" in the Quick Find bar. Click on "Tabs".

Click on "New" under the Custom Object Tabs . Details for Each Object:

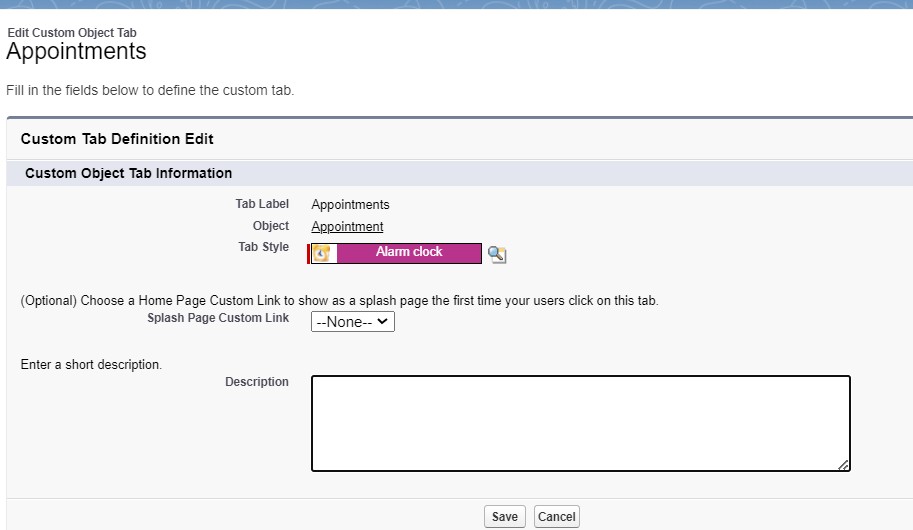
Select Object: Choose the respective object (Appointments, Service Records, Billing Details and Feedback).

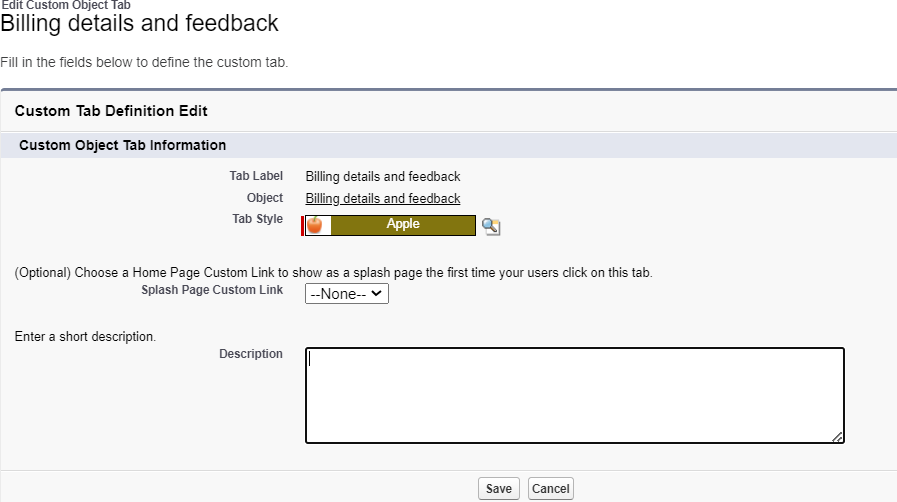
Select Tab Style: Choose a sui style for each object. Click Next.

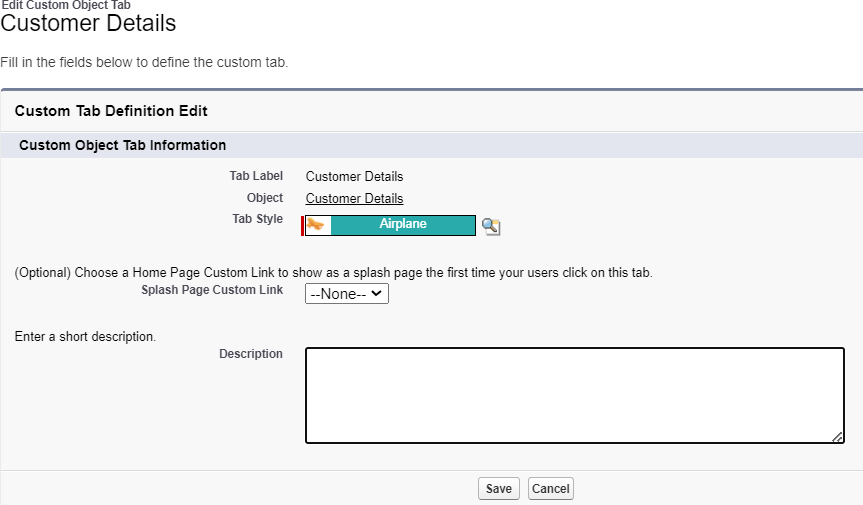
Add to Proﬁles Page: Keep the default settings. Click Next.

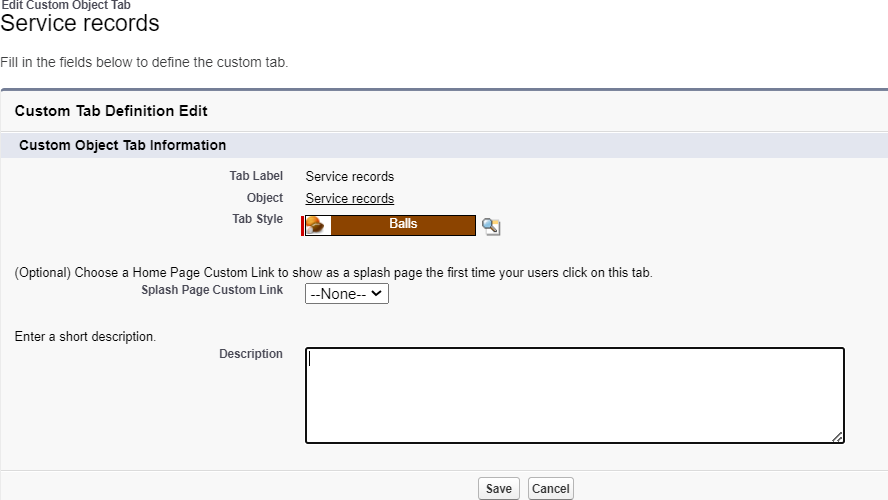
Add to Custom App: Uncheck "Include Tab".

Ensure "Append tab to users' existing personal customizations" is checked. Click Save.









# Task 4:

**Building Lignting App**

Creating a Lightning App in Salesforce for managing garage operations allows for streamlined processes and eﬃcient management of various business aspects. This guide outlines the steps to create the Garage Management Application, including adding essential navigation items and user proﬁles.

Steps to Create a Lightning App Accessing App Manager:

Begin by navigating to the setup page. In the Quick Find bar, search for "App Manager" and select it. Click on "New Lightning App" to start the creation process.

App Details:

Enter the name "Garage Management Application" in the app details . Proceed to the next steps, keeping all settings on the App Options and Utility Items pages as default.

Adding Navigation Items:

To enhance navigation within the app, select essential items such as Customer Details, Appointments,

Service Records, Billing Details and Feedback, Reports, and Dashboards from the search bar. Move these items using the ar button to include them in the app’s navigation menu.

Assigning User Profiles:

To ensure appropriate access and functionality, search for the "System Administrator" proﬁle in the search bar. Add this proﬁle by clicking the ar button. Complete the setup by saving and ﬁnishing the configuration.

# Task 5:

**Creating Fields**

* 1. Customer Details Object Fields

\*Phone Field:

Created a phone ﬁeld labeled "Phone Number" to store customer contact numbers.

\*Email Field:

Added an email ﬁeld labeled "Gmail" to capture customer email addresses. 2)Lookup Fields

\*Appointment Object:

Established a lookup relationship ﬁeld to link appointments to customer details, ensuring appointments are associated with the correct customer.

\*Service Records Object:

Created a lookup relationship ﬁeld to link service records to appointments. Added a ﬁlter to ensure the appointment date is less than the created date, making it a required ﬁeld with error validation.

\*Billing Details and Feedback Object:

Added a lookup relationship ﬁeld to connect billing details and feedback with service records. 3)Checkbox Fields

\*Appointment Object:

Added multiple checkbox ﬁelds to indicate diﬀerent services, including "Maintenance Service," "Repairs," and "Replacement Parts," all with default values set to unchecked.

\*Service Records Object:

Created a checkbox ﬁeld labeled "Quality Check Status" to track the quality check status of the service records.

1. Date Fields

\*Appointment Object:

Added a date ﬁeld labeled "Appointment Date" and marked it as required to ensure each appointment has a speciﬁed date.

1. Currency Fields

\*Appointment Object:

Created a currency ﬁeld labeled "Service Amount" to record the cost of services provided, with read- only access for all proﬁles.

\*Billing Details and Feedback Object:

Added a currency ﬁeld labeled "Payment Paid" to track payments made by customers.

1. Text Fields

\*Appointment Object:

Created a text ﬁeld labeled "Vehicle Number Plate" with a length of 10 characters, marked as required and unique for vehicle identiﬁcation.

\*Billing Details and Feedback Object:

Added a text ﬁeld labeled "Rating for Service" with a length of 1 character, required and unique to capture customer ratings.

1. Picklist Fields

Service Records Object:

Created a picklist ﬁeld labeled "Service Status" with values "Started" and "Completed" to track the progress of services.

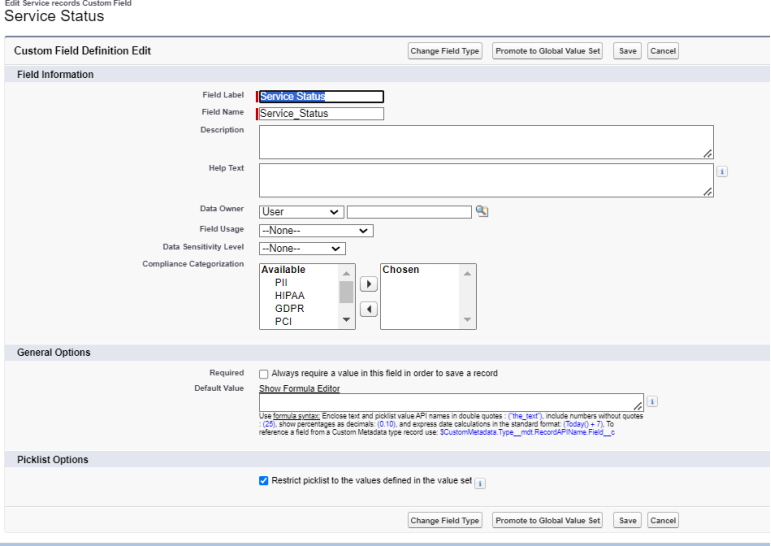
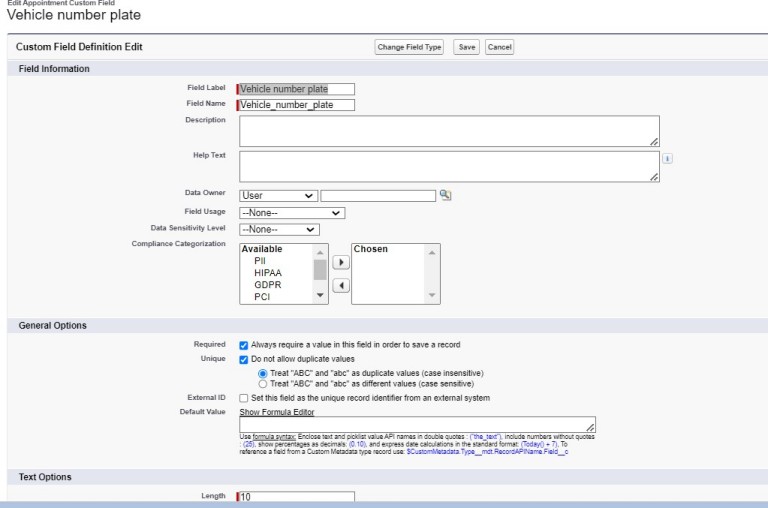
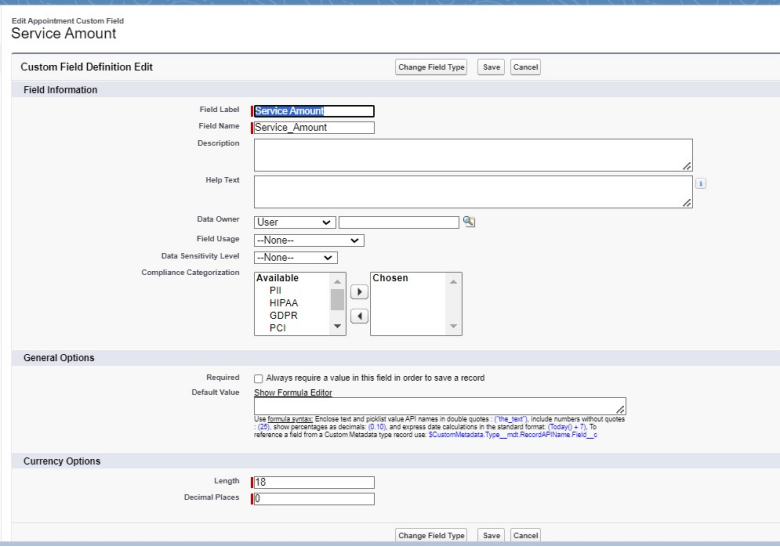
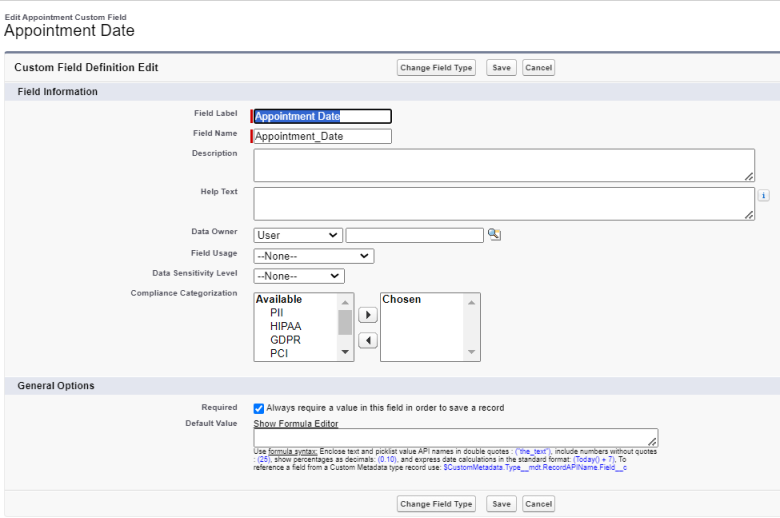
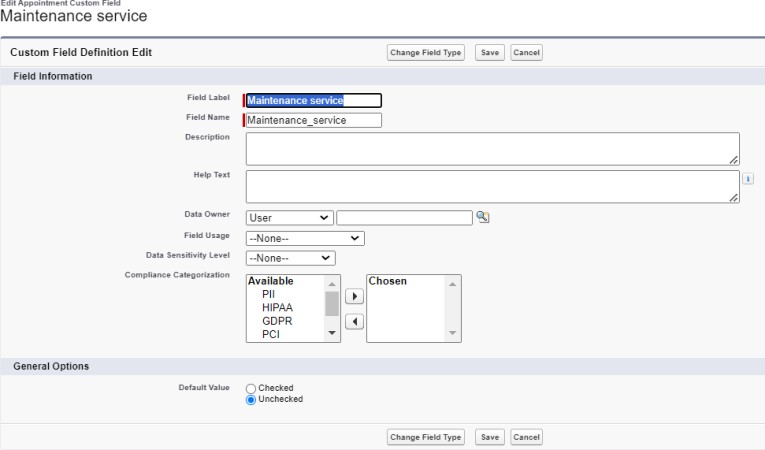
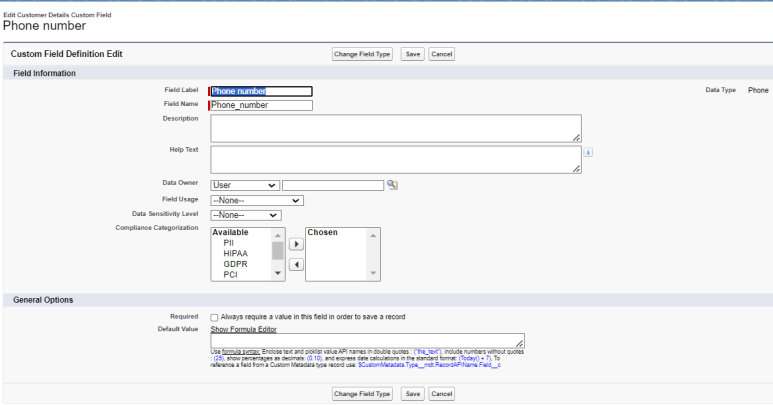
\*Billing Details and Feedback Object:

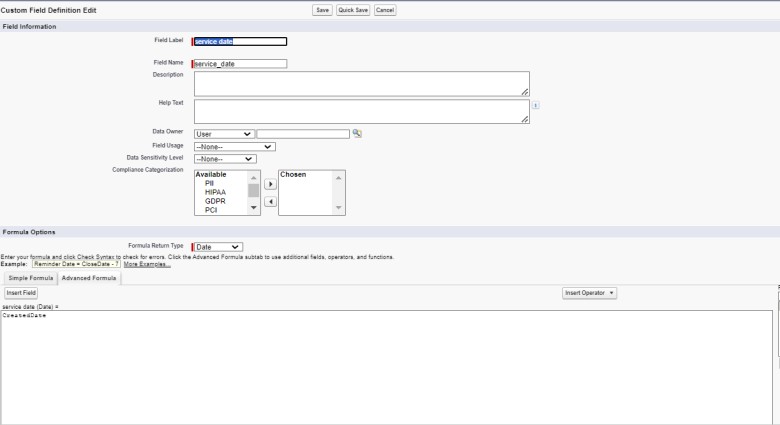
Added a picklist ﬁeld labeled "Payment Status" with values "Pending" and "Completed" to monitor payment statuses.

1. Formula Fields

\*Service Records Object:

Created a formula ﬁeld labeled "Service Date" to automatically populate with the created date of the record.





# Task 6:

**Validation Rules:**

# 1)Validation Rule for Appointment Object

Rule Name: Vehicle

Purpose: Ensure that the vehicle number plate follows a speciﬁc format. Error Condition Formula:

# NOT(REGEX(Vehicle\_number\_plate c , "[A-Z]{2}[0-9]{2}[A-Z]{2}[0-9]{4}"))

Error Message: "Please enter a valid number" Error Location: Field - Vehicle number plate **2)Validation Rule for Service Records Object** Rule Name: service\_status\_note

Purpose: Ensure that the service status is marked as "Completed" before the record can be saved. Error Condition Formula:

# NOT(ISPICKVAL(Service\_Status c, "Completed"))

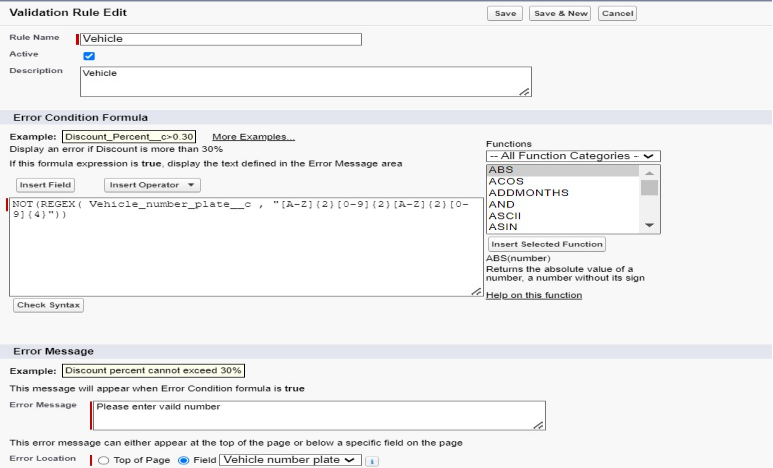
Error Message: "Still it is pending"

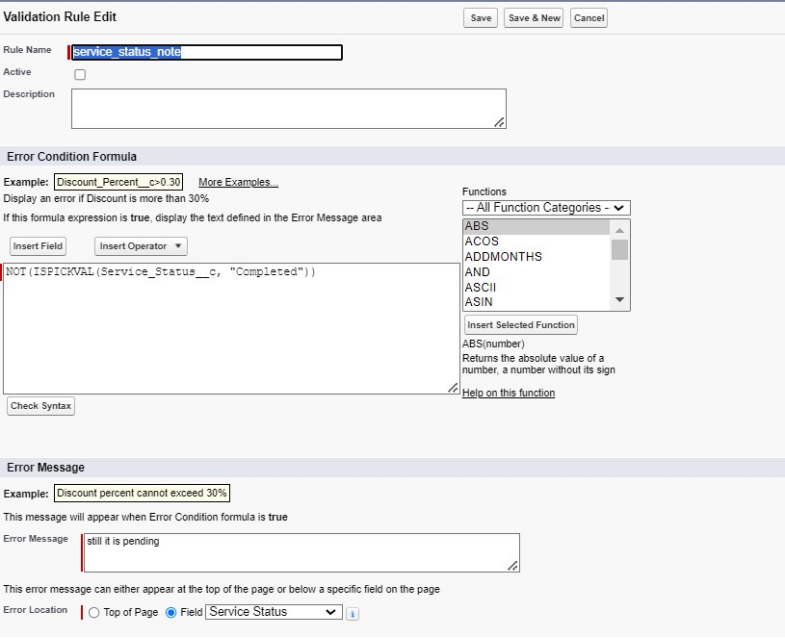
Error Location: Field - Service status

1. Validation Rule for Billing Details and Feedback Object Rule Name: rating\_should\_be\_less\_than\_5

Purpose: Ensure that the rating for service is between 1 and 5. Error Condition Formula:

**NOT(REGEX(Rating\_for\_service c, "[1-5]{1}"))** Error Message: "Rating should be from 1 to 5" Error Location: Field - Rating for Service





# Task 7:

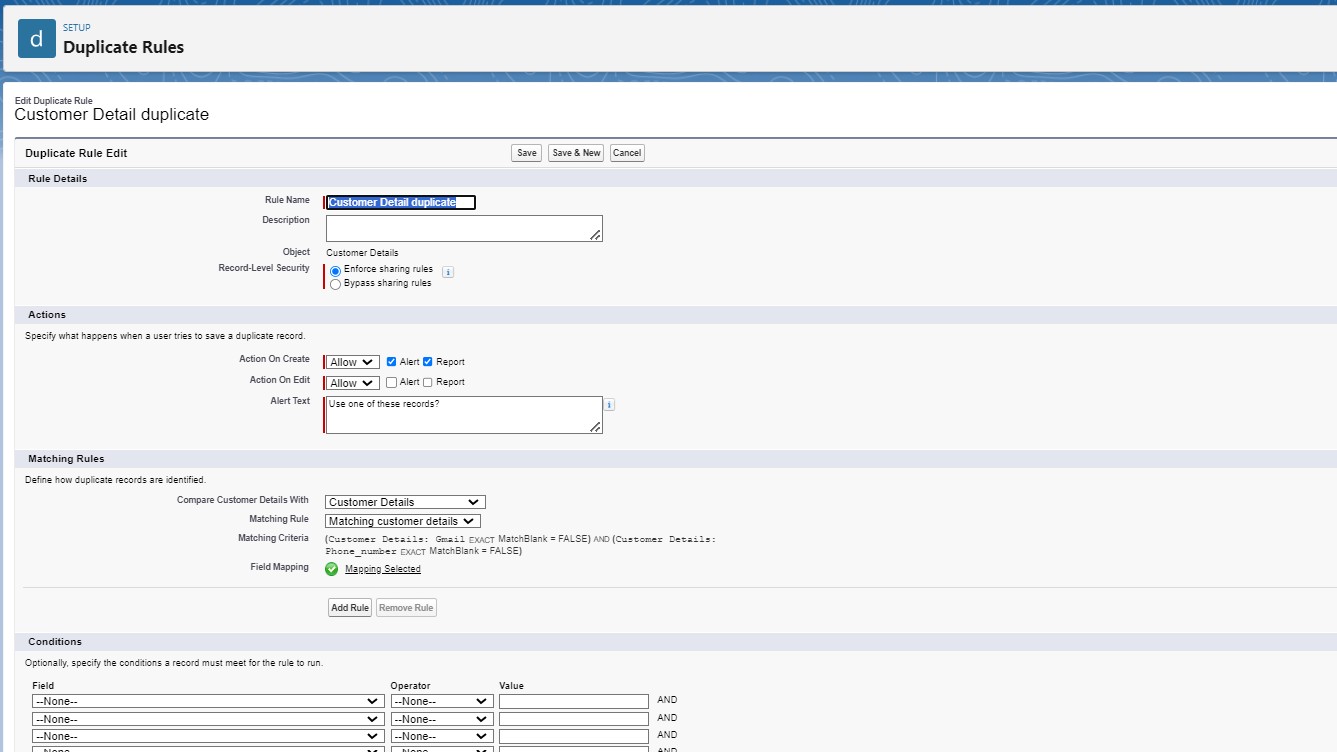
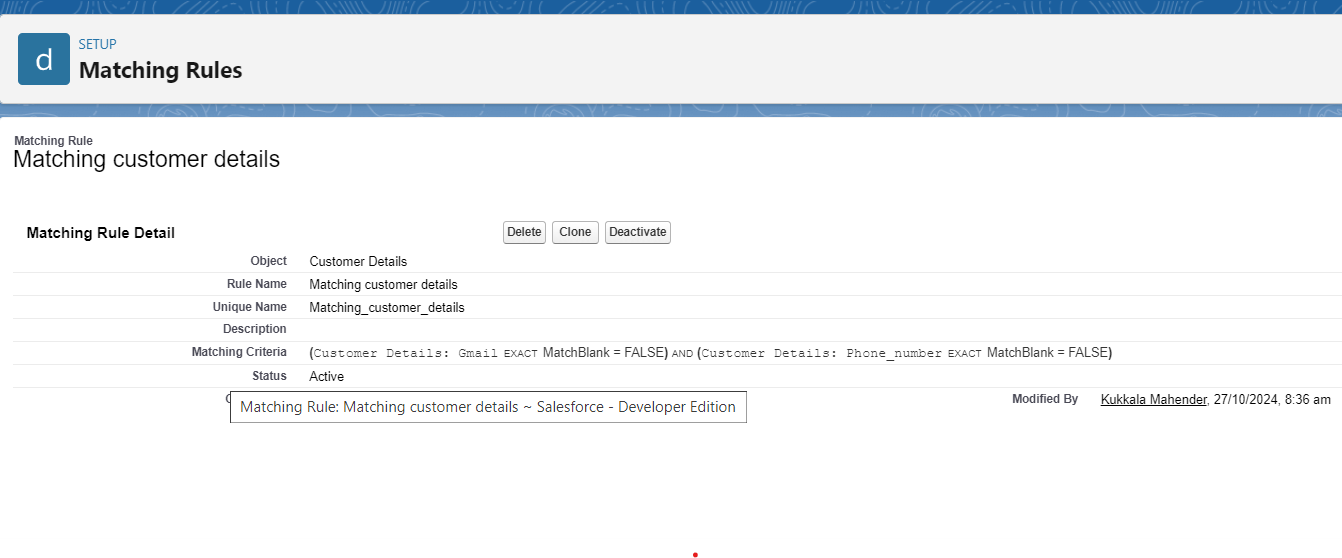
**Duplicate Rules:**

# Creating a Matching Rule for Customer Details Object

* 1. **Rule Name**: Matching Customer Details **Object**: Customer Details **Matching Criteria:**
     + **Gmail**: Exact match
     + **Phone Number**: Exact match
* **Activation**: Once the rule is saved, it is activated to start matching customer details based on the speciﬁed criteria.

# Creating a Duplicate Rule for Customer Details Object

1. **Rule Name**: Customer Detail Duplicate
   1. **Object**: Customer Details
   2. **Matching Rule**: Utilizes the previously created "Matching Customer Details" rule.
   3. **Activation**: After saving the duplicate rule, it is activated to begin identifying duplicate customer details based on the matching criteria.



# Task 8:

**Profiles:**

# Creating the Manager Profile

1. **Proﬁle Cloning**:
   * Clone the Standard User proﬁle to create a new proﬁle named "Manager."

# Custom App Settings:

* + Set the Garage Management Application as the default custom app for the Manager profile.

# Custom Object Permissions:

* + Grant access permissions for the following objects:
    - Appointments
    - Billing Details and Feedback
    - Service Records
    - Customer Details

# Session Timeout:

* + Conﬁgure the session timeout to occur after 8 hours of inactivity.

# Password Policies:

* + Set user passwords to never expire.
  + Establish a minimum password length of 8 characters.

# Creating the Sales Person Profile

1. **Proﬁle Cloning:**
   * Clone the Salesforce Platform User proﬁle to create a new proﬁle named "Sales Person."

# Custom App Settings:

* + Set the Garage Management Application as the default custom app for the Sales Person profile.

# Custom Object Permissions:

* + Grant access permissions for the following objects:
    - Appointments
    - Billing Details and Feedback
    - Service Records

# Task 9:

* Customer Details

# Role And Role Hierarch Creating the Manager Role

1. **Access Roles Setup**:
   * Navigate to the roles setup section in Salesforce.

# Expand Role Hierarchy:

* + Expand all roles and add a new role under the appropriate superior role.

# Deﬁne Role Details:

* + Label the new role as "Manager" and save the configuration.

# Creating Sales Person Role

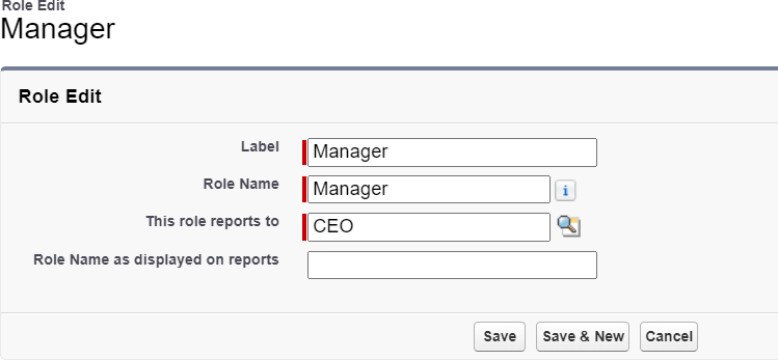
1. **Access Roles Setup**:
   * Navigate to the roles setup section in Salesforce.

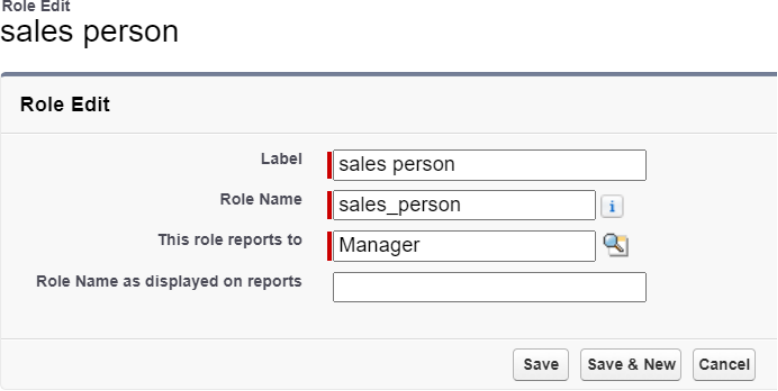
# Add Role Under Manager:

* + Expand the CEO role and add a new role under the Manager role.

# Deﬁne Role Details:

* + Label the new role as "Sales Person" and save the configuration.





# Task 10:

**Users:**

# Creating a Manager User

1. **Access User Setup:**
   * Navigate to the users section in Salesforce setup.

# New User Details:

* + Fill in the required ﬁelds with the user's details, such as ﬁrst name, last name, alias, email, username, and nickname.
  + Assign the role of "Manager."
  + Set the user license to "Salesforce."
  + Assign the proﬁle of "Manager."

# Save the Configuration:

* + Save the user details to create the new Manager user.

# Creating Sales Person Users

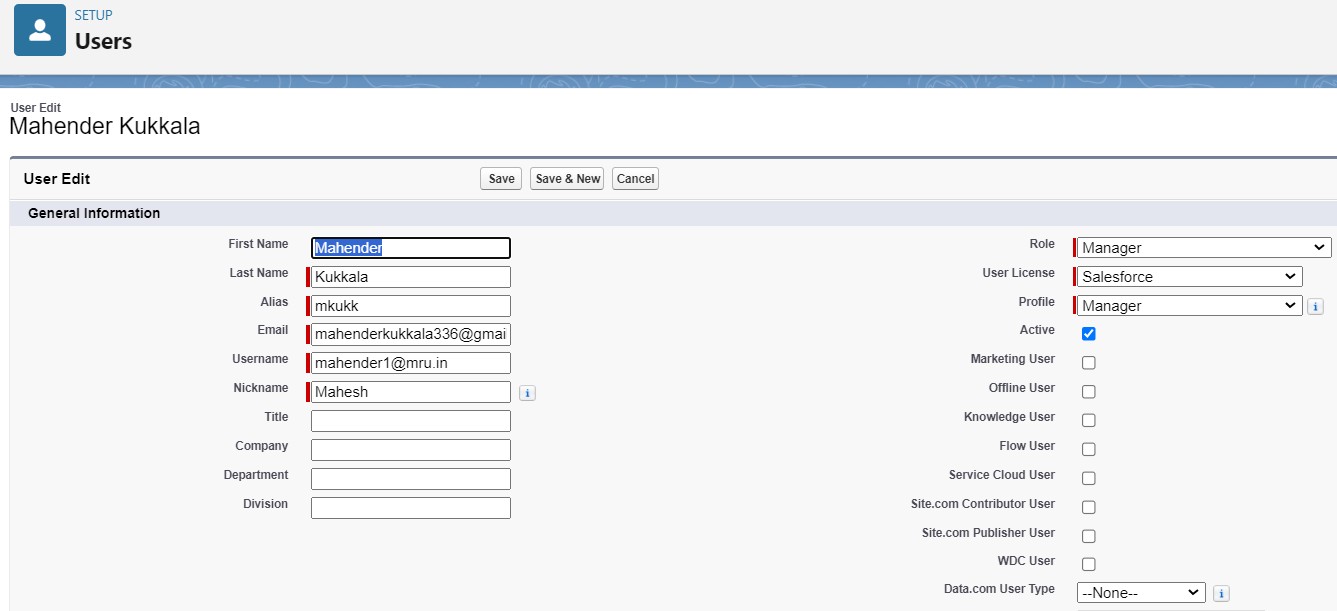
1. **Access User Setup**:
   * Navigate to the users section in Salesforce setup.

# New User Details:

* + Fill in the required ﬁelds with the user's details.
  + Assign the role of "Sales Person."
  + Set the user license to "Salesforce Platform."
  + Assign the proﬁle of "Sales Person."

# Save and Repeat:

* + Save the user details and repeat the process to create at least three users with the Sales Person role and proﬁle.



# Task 11:

**Public Groups:**

# Creating the Sales Team Public Group

1. **Access Public Groups Setup**:
   * Navigate to the public groups section in Salesforce setup.

# Deﬁne Group Details:

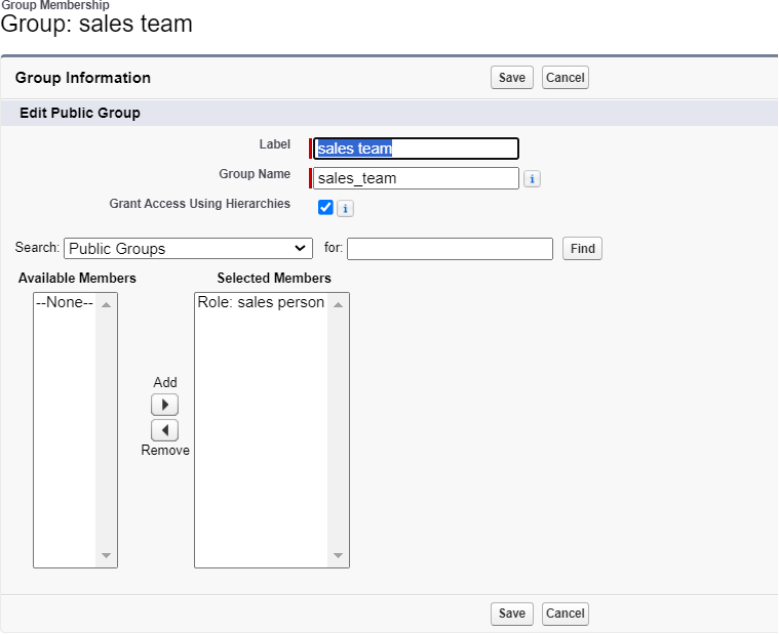
* + Provide a label for the new group, such as "Sales Team." The group name is automatically generated based on the label.

# Add Members:

* + Search for roles in the available members section.
  + Select the "Sales Person" role and add it to the selected members list.

# Save the Group:

* + Save the new public group configuration.



# Task 12:

**Sharing Settings**

# Conﬁguring Sharing Settings for Service Records

1. **Access Sharing Settings**:
   * Navigate to the Sharing Settings section in Salesforce setup.

# Set Object-Wide Default (OWD):

* + Change the OWD setting for the Service Records object to "Private" to restrict access to records by default.

# Create Sharing Rules:

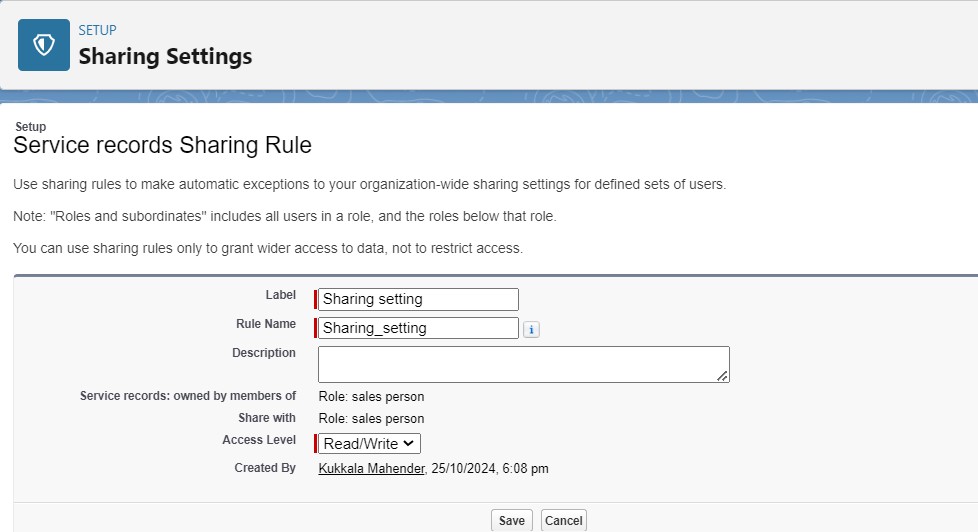
* + Deﬁne a new sharing rule to grant access to speciﬁc roles.

# Deﬁne Sharing Rule Details:

* + Label the sharing rule appropriately, such as "Sharing Setting."
  + Specify the members of the role to be shared, selecting "Sales Person."
  + Determine who will receive the access, selecting "Manager."
  + Set the access level to "Read/Write."

# Save and Refresh:

* + Save the sharing rule and refresh the settings to apply changes.



# Task 13:

**Flows:**

# Creating the Flow

1. **Initiate Flow Creation**:
   * Access the Flow setup page and start a new Record-Triggered Flow.

# Conﬁgure Flow Trigger:

* + Select "Billing Details and Feedback" as the object.
  + Set the ﬂow to trigger when a record is created or updated.
  + Choose "Actions and Related Records" to optimize the flow.

# Add Update Records Element:

* + Label the update element as “Amount Update.”
  + Conﬁgure it to update records where the ﬁeld Payment\_Status c equals "Completed."
  + Set the ﬁeld value for Payment\_Paid c to the service amount from the related Appointment record.

# Create and Conﬁgure Email Alert:

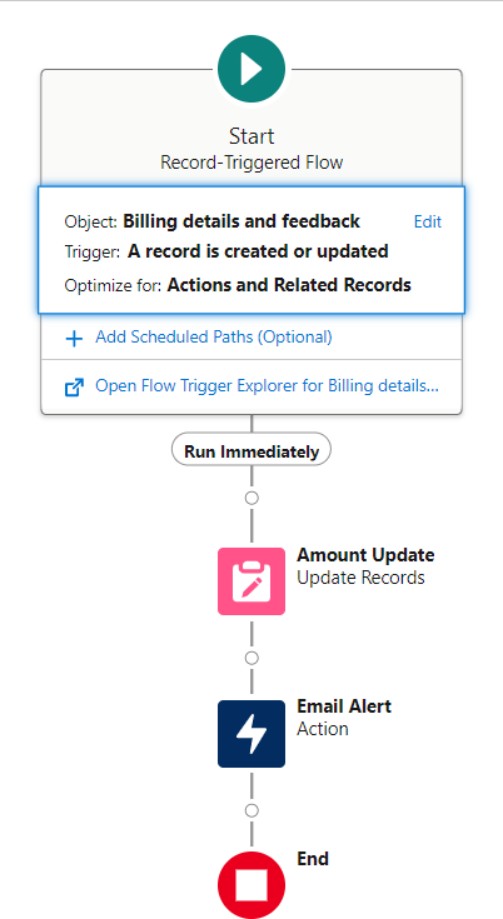
* + Deﬁne a new resource as a text template to construct the email body.
  + Use rich text format for clarity and include placeholders for dynamic content.
  + The email body should express gratitude for the payment and include the payment amount.

# Add Email Action:

* + Insert an "Action" element to send an email.
  + Utilize the previously created text template for the email body.
  + Set the recipient address dynamically from the related record.
  + Deﬁne the subject of the email as "Thank You for Your Payment - Garage Management."

# Save and Activate the Flow:

* + Save the ﬂow with an appropriate label and auto-populated API name.
  + Activate the ﬂow to make it operational.



# Task 14:

**Apex Triggers**

# Apex Handler: AmountDistributionHandler

**Purpose**: This Apex class calculates and updates the service amount based on selected services in the Appointment c object.

# Class Definition:

* + **Class Name**: AmountDistributionHandler
  + **Method**: amountDist
  + **Parameters**: List<Appointment c> – A list of appointment records to process.

# Functionality:

* + Iterates through the list of appointments.
  + Updates the Service\_Amount c ﬁeld based on the combination of services selected:
    - All three services: 10000
    - Maintenance and Repairs: 5000
    - Maintenance and Replacement Parts: 8000
    - Repairs and Replacement Parts: 7000
    - Maintenance only: 2000
    - Repairs only: 3000
    - Replacement Parts only: 5000

# Trigger: AmountDistribution

**Purpose**: This trigger invokes the AmountDistributionHandler to update the Service\_Amount c ﬁeld whenever an appointment record is inserted or updated.

# Trigger Definition:

* + **Trigger Name**: AmountDistribution
  + **sObject**: Appointment c
  + **Events**: before insert, before update

# Trigger Logic:

* + Checks if the trigger event is before insert or before update.
  + Calls the amountDist method from AmountDistributionHandler to perform the required updates on the Appointment c records.

# Code:

trigger AmountDistribution on Appointment c (before insert, before update)

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

}

}

# Task 15:

**Reports:**

# Creating a Report Folder

1. **Objective**: Organize reports by creating a dedicated folder.

# Process:

* + Access the Reports tab through the app launcher.
  + Click on **New Folder** and name it **"Garage Management Folder"**. The folder's unique name is automatically generated.
  + Save the new folder to store and manage related reports.

# Sharing the Report Folder

1. **Objective**: Control access to the report folder.

# Process:

* + Navigate to the Reports tab and locate the **"Garage Management Folder"**.
  + Click on the dropdown menu next to the folder name and select **Share**.
  + Choose **Roles** as the sharing option, search for the **"Manager"** role, and assign **VieW**

access.

* + Conﬁrm the sharing settings and ﬁnalize by clicking

# Done. Creating a Custom Report Type

1. **Objective**: Deﬁne a report type to include multiple related objects.

# Process:

* + Access **Setup** and search for **Report Types**.
  + Click on **New Custom Report Type**.
  + Set the **Primary Object** to **Customer Details**.
  + Name the report type **"Service Information"** and conﬁrm the auto-populated name.
  + Categorize under **Other Reports** and set the deployment status to **Deployed**.
  + Click **Next** to deﬁne related objects:

# Appointment Object

* + - **Service Records**

# Billing Details and Feedback

* + Save the new report type.

# Creating a Report

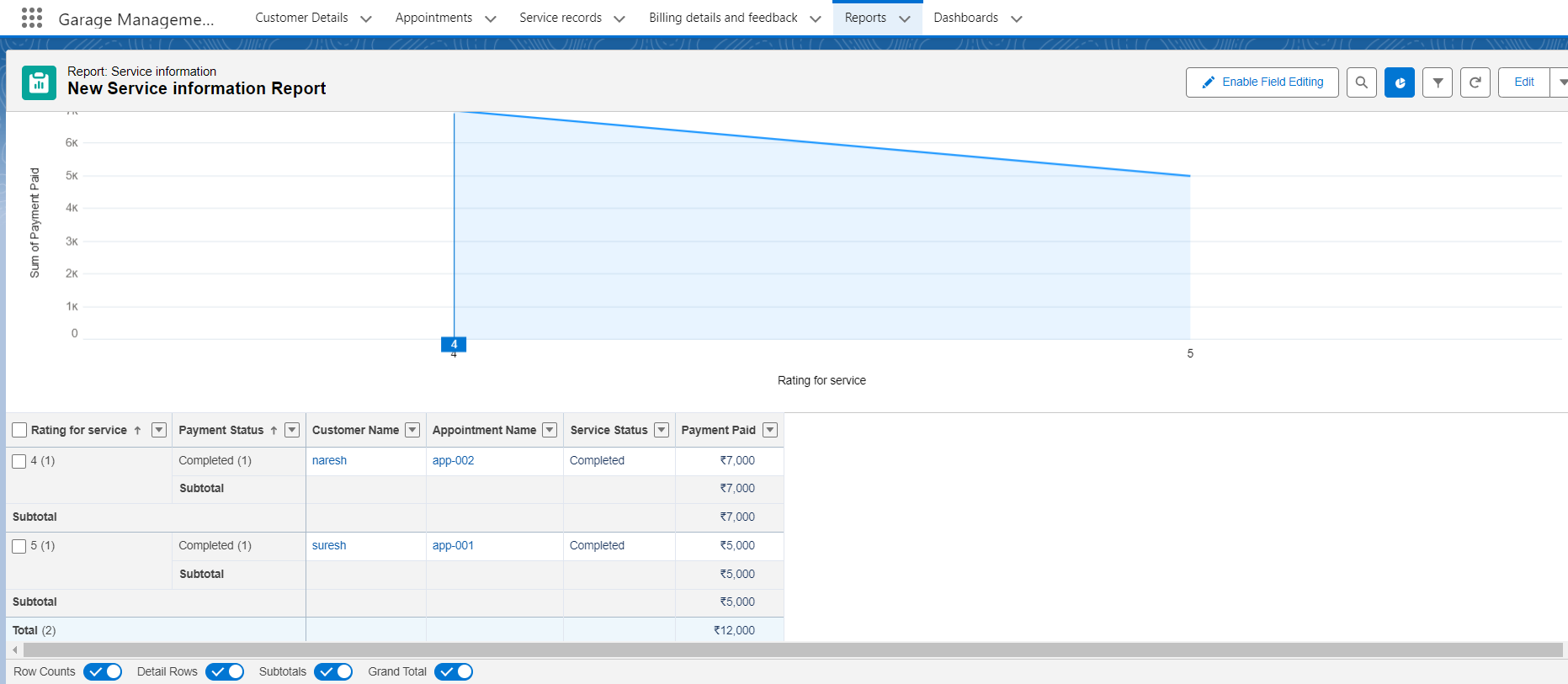
1. **Objective**: Generate a detailed report based on the created report type.

# Preparation:

* + Ensure to create at least 10 records for each object to ensure comprehensive data representation.

# Process:

* + Access the Reports tab and click on **New Report**.
  + Select the category **Other Reports** and choose **Service Information** as the report type.
  + Start the report builder and conﬁgure the fields:
    - **Columns**: Customer Name, Appointment Date, Service Status, Payment Paid.
    - **Group Rows**: Rating for Service, Payment Status.
  + Add a Line Chart to visualize the data.
  + Save the report with the name **"New Service Information Report"**, ensuring the unique name is auto-populated and the report is saved in the **"Garage Management Folder"**.



# Task 16:

**Dashboards:**

# Creating a Dashboard Folder

1. **Objective**: Organize dashboards by creating a speciﬁc folder.

# Process:

* + Navigate to the **Dashboards** tab via the app launcher.
  + Click **New Folder** and name it **"Service Rating Dashboard"**. The unique name is auto- generated.
  + Save the folder to group and manage related dashboards efficiently.

# Sharing the Dashboard Folder

1. **Objective**: Set access permissions for the dashboard folder.

# Process:

* + Locate the **"Service Rating Dashboard"** folder in the Dashboards tab.
  + Click on the dropdown menu for the folder and select **Share**.
  + Assign appropriate permissions based on user roles, ensuring that the relevant team members have access to view or edit the dashboards as needed.
  + Conﬁrm the sharing settings to finalize.

# Creating a Dashboard

1. **Objective**: Build a dashboard to visualize data and insights.

# Process:

* + Access the **Dashboards** tab and click **New Dashboard**.
  + Enter a name for the dashboard and select the **"Service Rating Dashboard"**

folder created earlier.

* + Click **Create** to start building the dashboard.
  + Add a new component by selecting **Add Component**.
  + Choose a report to base the dashboard on, and select a **Line Chart** for visualization.
  + Customize the chart theme as desired.
  + Click **Add** to include the component in the dashboard, then click **Save** and

# Done. Subscribing to the Dashboard

**1. Objective**: Automate the delivery of dashboard updates.

**2 Process:**

* After saving the dashboard, click **Subscribe** at the top right.
* Set the subscription frequency to **Weekly**.
* Choose **Monday** as the day for the report to be sent.
* Click **Save** to activate the subscription and ensure regular updates.

