

Title:- Garage Management System

Using Salesforce...

Project Overview :

The Garage Management System (GMS) is a software solution designed to streamline and automate the daily operations of an automobile garage or service center. Traditionally, garages rely on manual processes such as paper records, handwritten job cards, and verbal communication, which often lead to inefficiency, delays, and errors. This system addresses these challenges by providing a centralized digital platform that manages customer details, vehicle information, service records, spare parts inventory, billing, and reporting.

The system allows garage owners and staff to track service jobs from booking to completion, manage mechanics' assignments, and ensure efficient use of resources. Customers benefit from timely updates, accurate billing, and improved service quality, while garage owners gain insights into business performance through detailed reports and analytics.

By automating critical processes such as job scheduling, invoicing, and stock management, the GMS reduces paperwork, saves time, and minimizes human error. Ultimately, the project aims to increase operational efficiency, enhance customer satisfaction, and boost overall profitability of the garage.

Objectives :

1. Customer & Vehicle Management

- Maintain detailed records of customers and their vehicles (e.g., make, model, registration, service history).
- Enable quick retrieval of customer details for future service or repairs.

2. Service & Repair Management

- Schedule, track, and manage vehicle services and repair jobs.
- Assign jobs to mechanics/technicians and monitor progress.
- Generate job cards and service histories automatically.

3. Inventory & Spare Parts Management

- Track stock levels of spare parts and consumables.
- Automate purchase orders for low-stock items.
- Reduce delays caused by unavailable parts.

4. Billing & Invoicing

- Generate accurate service estimates, invoices, and receipts.
- Support multiple payment modes (cash, card, online).
- Apply discounts, taxes, and warranties systematically.

5. Reporting & Analytics

- Provide insights on garage performance (e.g., daily revenue, service frequency, part usage).
- Help owners make data-driven decisions for efficiency and profitability.

6. Workforce Management

- Assign tasks to mechanics and track labor hours.
- Monitor productivity and performance.

7. Customer Communication & Retention

- Send reminders for upcoming services or pending bills.
- Provide updates via SMS/email on job progress.
- Improve customer loyalty through timely follow-ups.

8. Automation & Efficiency

- Reduce paperwork and manual errors.
- Save time by automating repetitive administrative tasks.
- Ensure smooth workflow from job booking to completion.

Student Outcomes :

By completing the Garage Management System project, students will be able to:

1. Technical Skills

- Apply database management concepts to design and implement a relational database for customers, vehicles, services, and inventory.
- Develop a functional software application using appropriate programming languages, frameworks, and tools.
- Integrate modules such as job scheduling, billing, inventory tracking, and reporting into a unified system.

2. Problem-Solving Abilities

- Analyze real-world challenges faced by automobile garages and propose digital solutions.
- Translate business requirements into technical specifications.
- Optimize workflows to reduce redundancy and improve efficiency.

3. Project Management & Teamwork

- Collaborate in a team to divide tasks, coordinate development, and ensure timely completion.
- Apply project management practices like planning, documentation, and version control.

4. Analytical & Critical Thinking

- Evaluate system performance through testing and debugging.
- Assess the effectiveness of the system in improving garage operations.
- Draw insights from generated reports to support decision-making.

5. Professional & Communication Skills

- Prepare project documentation (objectives, overview, design, implementation, testing, outcomes).
- Present the project effectively to both technical and non-technical audiences.
- Demonstrate awareness of customer-centric software development.

System Requirements :

Hardware Requirements:

- * Computer with min/sum 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.

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:-

Garage Management System

The Garage Management System is a valuable tool for automotive repair facilities, helping them deliver top-notch service, increase operational efficiency, and build lasting customer relationships. With its user-friendly interface and powerful features, GMS empowers garages to thrive in a competitive market while ensuring a seamless and satisfying experience for both customers and staff.

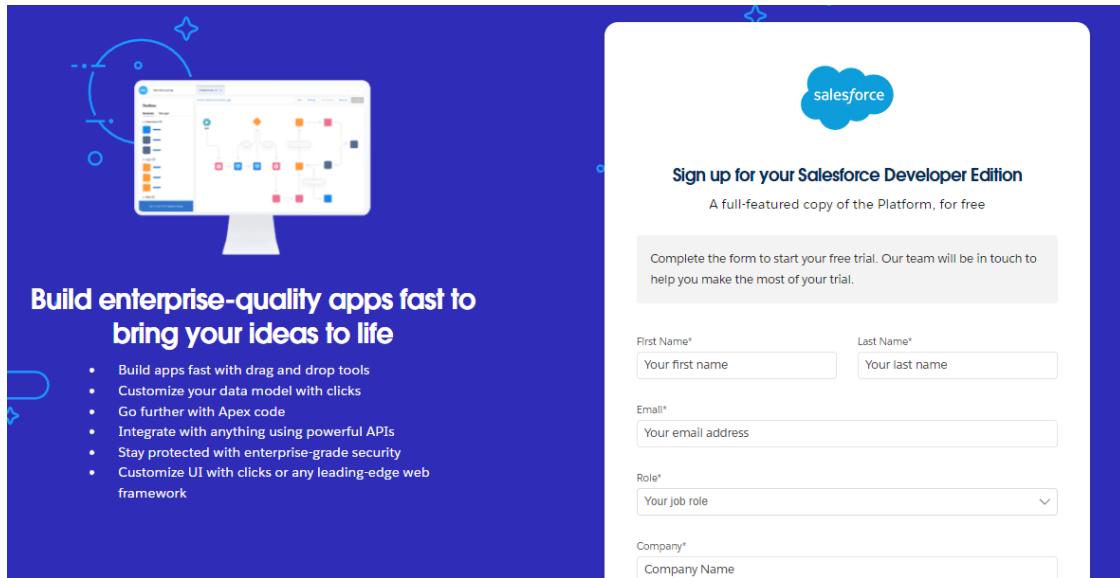
Phase 2: Salesforce Development – Backend & Configurations:-

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 :



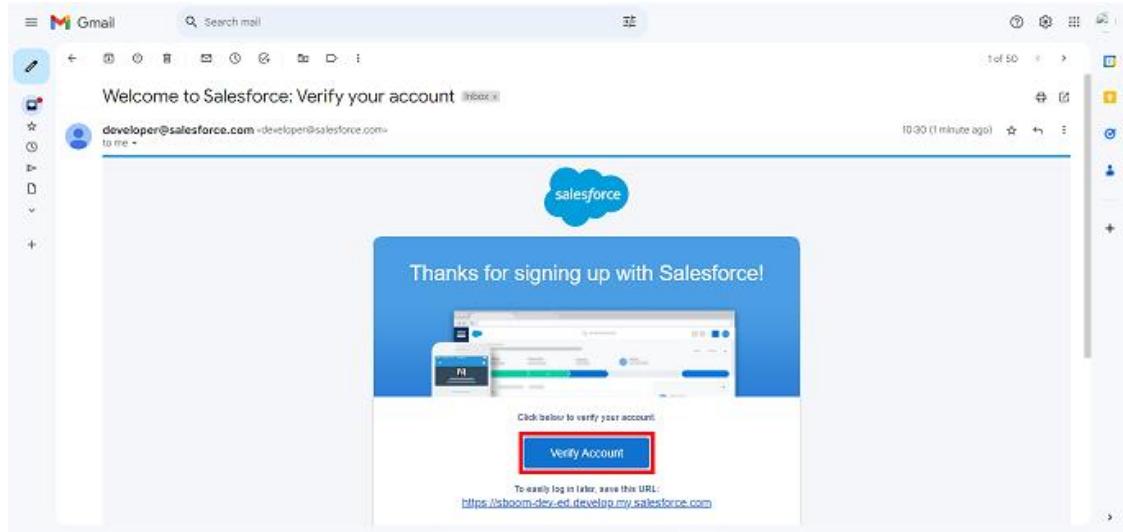
1. First name & Last name
2. Email
3. Role : Developer
4. Company : College Name
5. County : India
6. Postal Code : pin code
7. Username : should be a combination of your name and company

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.



2. Click on Verify Account
3. Give a password and answer a security question and click on change password.

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

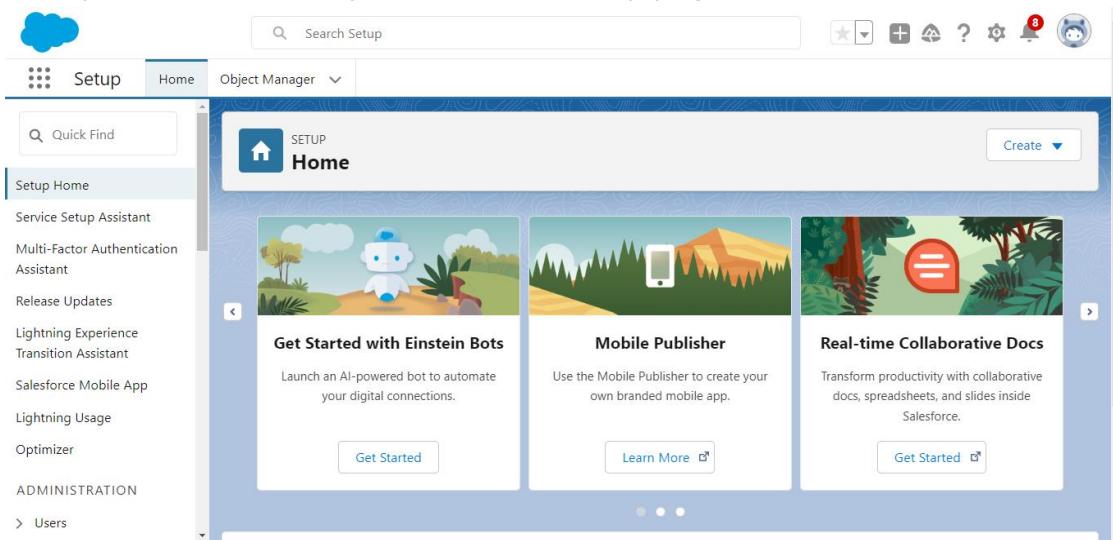
Security Question
▼ In what city were you born?

* Answer
asdfghjkl

Change Password

A screenshot of a "Change Your Password" form. It includes fields for "New Password" and "Confirm New Password", both of which are highlighted with a large red box. Below these is a "Security Question" field and an "Answer" field, also highlighted with a red box. At the bottom is a prominent blue "Change Password" button, which is also highlighted with a red box.

4. Then you will redirect to your salesforce setup page.



Milestone 2: Object

What Is an Object?

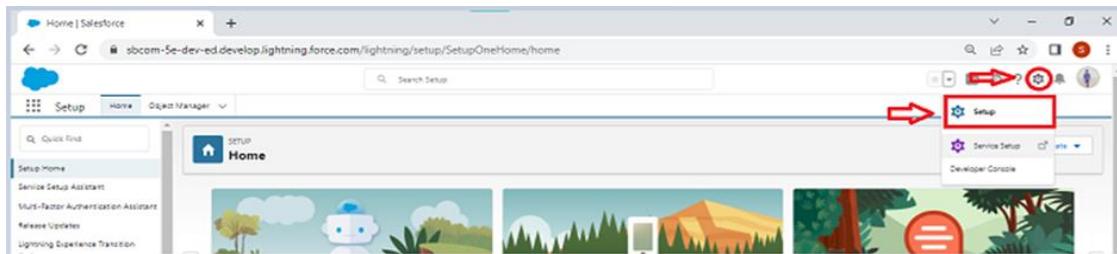
Salesforce objects are database tables that permit you to store data that is specific to an organization. What are the types of Salesforce objects

Salesforce objects are of two types:

- Standard Objects:** Standard objects are the kind of objects that are provided by salesforce.com such as users, contracts, reports, dashboards, etc.
- 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:

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



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

A detailed screenshot of the 'New Custom Object' configuration page. Several fields are highlighted with red arrows:

- The 'Label' field (containing 'Account') and the 'Plural Label' field (containing 'Accounts') are both circled with red boxes.
- The 'Object Name' field (containing 'Account') is also circled with a red box.
- In the 'Optional Features' section, the 'Allow Reports' checkbox is checked and circled with a red box.
- In the 'Search Status' section, the 'Allow Search' checkbox is checked and circled with a red box.
- At the bottom right, the 'Save' and 'Save & New' buttons are circled with red boxes, with a red arrow pointing upwards towards them.

6. 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
2. Click on Allow reports and Track Field History,
3. 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
2. Click on Allow reports and Track Field History,
3. 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
2. Click on Allow reports and Track Field History,
3. Allow search >> Save.

Milestone 3: Tabs

What is Tab : A tab is like a user interface that is used to build records for objects and to view the records in the objects.

Types of Tabs:

1. Custom Tabs

Custom object tabs are the user interface for custom applications that you build in salesforce.com. They look and behave like standard salesforce.com tabs such as accounts, contacts, and opportunities.

2. Web Tabs

Web Tabs are custom tabs that display web content or applications embedded in the salesforce.com window. Web tabs make it easier for your users to quickly access content and applications they frequently use without leaving the salesforce.com application.

3. Visualforce Tabs

Visualforce Tabs are custom tabs that display a Visualforce page. Visualforce tabs look and behave like standard salesforce.com tabs such as accounts, contacts, and opportunities.

4. Lightning Component Tabs

Lightning Component tabs allow you to add Lightning components to the navigation menu in Lightning Experience and the mobile app.

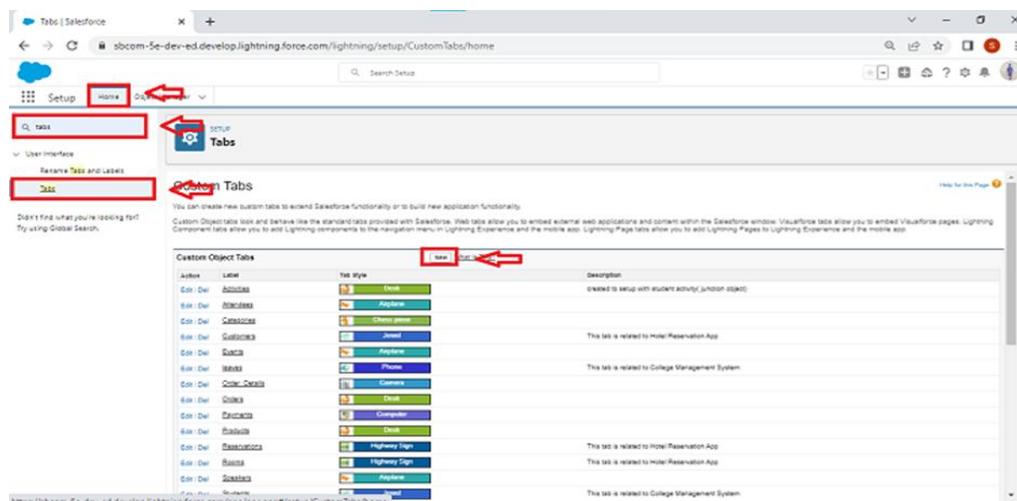
5. Lightning Page Tabs

Lightning Page Tabs let you add Lightning Pages to the mobile app navigation menu. Lightning Page tabs don't work like other custom tabs. Once created, they don't show up on the All Tabs page when you click the Plus icon that appears to the right of your current tabs. Lightning Page tabs also don't show up in the Available Tabs list when you customise the tabs for your apps.

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

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		Apple		Balls
	Bank[1]		Bell		Big top		Boat[1]
	Books		Bottle		Box		Bridge
	Building		Building Block		Caduceus		Camera
	Can		Car		Castle		CD/DVD
	Cell phone		Chalkboard		Chess piece		Chip
	Circle		Compass		Computer		Credit card
	CRT TV		Cup		Desk[1]		Diamond
	Dice		Factory		Fan		Flag
	Form		Gears		Globe		Guitar
	Hammer		Hands		Handsaw		Headset
	Heart[1]		Helicopter		Hexagon		Highway Sign
	Hot Air Balloon		Insect		IP Phone		Jewel
	Keys		Laptop		Leaf		Lightning

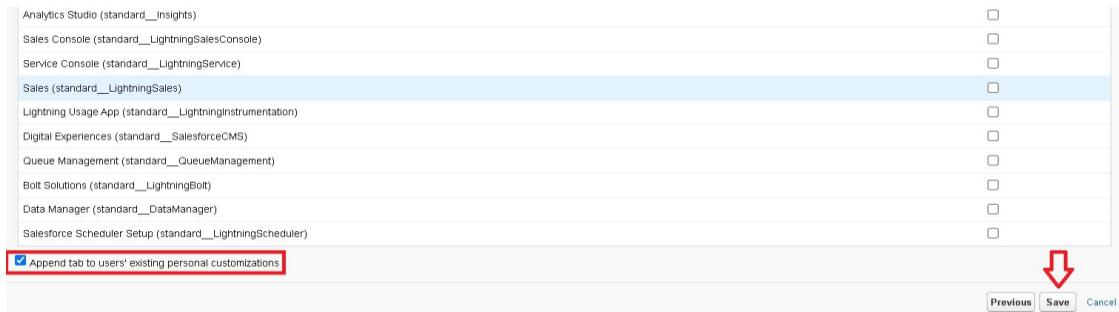
[Save](#) [Cancel](#)

Step 3. Add to Custom Apps

Step 3 of 3

Choose the custom apps for which the new custom tab will be available. You may also examine or alter the visibility of tabs from the detail and edit pages of each Custom App.

Custom App	<input checked="" type="checkbox"/> Include Tab
Platform (standard__Platform)	<input type="checkbox"/>
Sales (standard__Sales)	<input type="checkbox"/>
Service (standard__Service)	<input type="checkbox"/>
Marketing (standard__Marketing)	<input type="checkbox"/>
Sample Console (standard__ServiceConsole)	<input type="checkbox"/>
High Volume Customer Portal User	<input type="checkbox"/>
Authenticated Website User	<input type="checkbox"/>
App Launcher (standard__AppLauncher)	<input type="checkbox"/>



Activity 2: Creating Remaining Tabs

1. Now create the Tabs for the remaining Objects, they are “Appointments, Service records,Billing details and feedback”.
2. Follow the same steps as mentioned in Activity -1 .

Milestone 4: The Lightning App

An app is a collection of items that work together to serve a particular function. In Lightning Experience, Lightning apps give your users access to sets of objects, tabs, and other items all in one convenient bundle in the navigation bar.

Lightning apps let you brand your apps with a custom colour and logo. You can even include a utility bar and Lightning page tabs in your Lightning app. Members of your org can work more efficiently by easily switching between apps.

Activity 1: 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.

Lightning Experience App Manager

New Lightning App

App Name	Developer Name	Description	Last Modified	App Type	VL
All Tabs	AllTabSet		04/12/2022, 10:13 am	Classic	✓
Analytics Studio	Insights	Build CRM Analytics dashboards and apps	04/12/2022, 10:13 am	Classic	✓
App Launcher	AppLauncher	App Launcher tabs	04/12/2022, 10:13 am	Classic	✓
Bolt Solutions	LightningBolt	Discover and manage business solutions designed for your industry.	04/12/2022, 10:18 am	Lightning	✓
Chatter Desktop	Chatter/Desktop	Chatter Desktop is an Adobe AIR-based desktop application that lets Chatter users stay connected...	29/12/2022, 4:04 pm	Connected (Managed)	✓
Chatter Mobile for BlackBerry	ChatterForBlackBerry	The Salesforce.com Chatter Mobile app lets you access Chatter data on the go. Use it to view fe...	29/12/2022, 4:05 pm	Connected (Managed)	✓
College Management System	Hasseem	Demo app	08/12/2022, 4:18 pm	Lightning	✓
Community	Community	Salesforce CRM Communities	04/12/2022, 10:13 am	Classic	✓
Content	Content	Salesforce CRM Content	04/12/2022, 10:13 am	Classic	✓
Data Manager	DataManager	Use Data Manager to view limits, monitor usage, and manage recipes.	04/12/2022, 10:13 am	Lightning	✓

- Fill the app name in app details as Garage Management Application >> Next >> (App option page) keep it as default >> Next >> (Utility Items) keep it as default >> Next.

New Lightning App

App Details & Branding

App Name: Name your app... *

Developer Name: Enter a developer name...

Description: Enter a description...

Image: Upload

Primary Color Hex Value: #0070D2

Org Theme Options: Use the app's image and color instead of the org's custom theme

App Launcher Preview

Next

- To Add Navigation Items:

New Lightning App

Navigation Items

Choose the items to include in the app, and arrange the order in which they appear. Users can personalize the navigation to add or move items, but users can't remove or rename the items that you add. Some navigation items are available only for phone or only for desktop. These items are dropped from the navigation bar when the app is viewed in a format that the item doesn't support.

Available Items

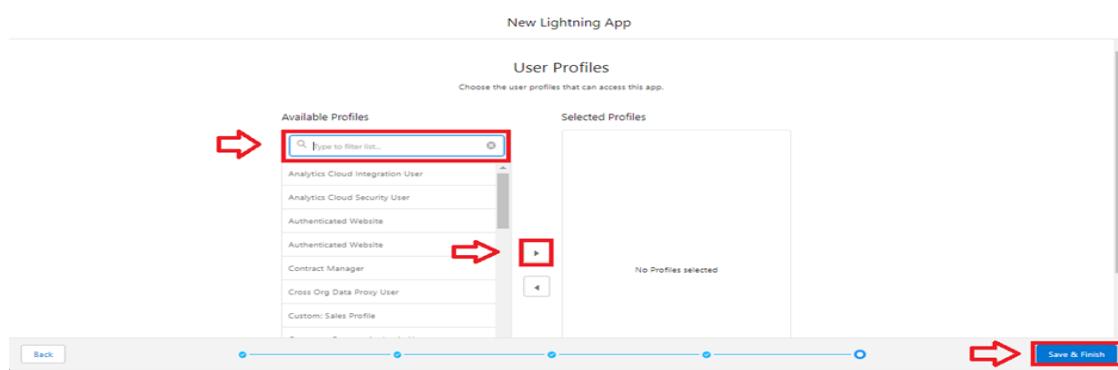
Selected Items

Back

Next

4. Select the items (Customer Details, Appointments, Service records, Billing details and feedback, Reports and Dashboards) from the search bar and move it using the arrow button >> Next.

5. To Add User Profiles:



Search profiles (System administrator) in the search bar >> click on the arrow button >> save & finish.

Milestone 5: Fields

When we talk about Salesforce, Fields represent the data stored in the columns of a relational database. It can also hold any valuable information that you require for a specific object. Hence, the overall searching, deletion, and editing of the records become simpler and quicker.

Types of Fields

1. Standard Fields
2. Custom Fields

Standard Fields:

As the name suggests, the Standard Fields are the predefined fields in Salesforce that perform a standard task. The main point is that you can't simply delete a Standard Field until it is a non-required standard field. Otherwise, users have the option to delete them at any point from the application freely. Moreover, we have some fields that you will find common in every Salesforce application. They are,

- Created By
- Owner
- Last Modified
- Field Made During object Creation

Custom Fields:

On the other side of the coin, Custom Fields are highly flexible, and users can change them according to requirements. Moreover, each organiser or company can use them if necessary. It means you need not always include them in the records, unlike Standard fields. Hence, the final decision depends on the user, and he can add/remove Custom Fields of any given form.

Activity 1: 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.

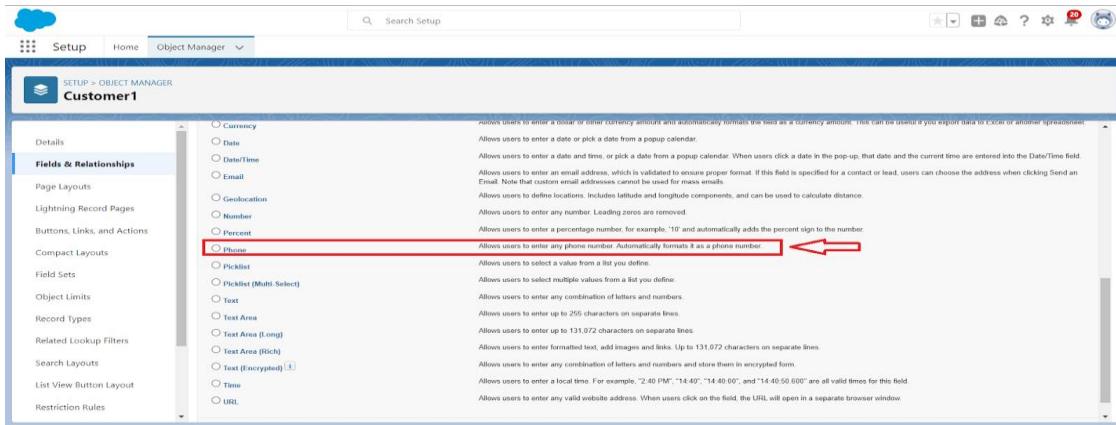
The screenshot shows the Salesforce Object Manager interface. At the top, there's a search bar with 'cus' and buttons for 'Schema Builder' and 'Create'. Below the header, a table lists objects: 'Customer' (Standard Object) and 'Customer Details' (Custom Object). The 'Customer Details' row is highlighted with a red border. The table columns are labeled: LABEL, API NAME, TYPE, DESCRIPTION, LAST MODIFIED, and DEPLOYED.

2. Now click on “Fields & Relationships” >> New

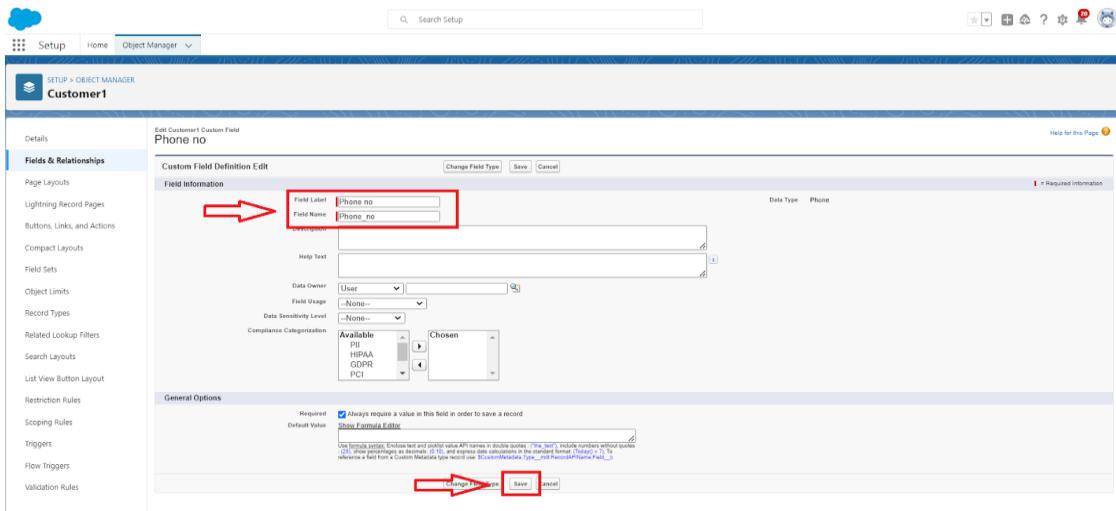
The screenshot shows the 'Fields & Relationships' page for the 'Customer1' object. At the top, there's a breadcrumb trail: SETUP > OBJECT MANAGER > Customer1. A red arrow points to the 'Customer1' label. On the left, a sidebar lists various setup categories like Page Layouts, Lightning Record Pages, etc. A red box highlights the 'Fields & Relationships' link in the sidebar, and a red arrow points to it. The main area shows a table titled 'Fields & Relationships' with 8 items. A red box highlights the 'New' button at the top right of the table, and a red arrow points to it. The table columns are: FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED.

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
current Status	current_Status__c	Picklist		
Customer Name	Name	Text(80)		
Email id	Email_id__c	Email (Unique)		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		
Permanent Address	Permanent_Address__c	Text Area(255)		
Phone no	Phone_no__c	Phone		

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 Fields

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.

The screenshot shows the Salesforce Object Manager interface. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. A search bar contains the text 'app'. Below the search bar, there are buttons for 'Schema Builder' and 'Create'. The main area displays a table with columns: LABEL, API NAME, TYPE, DESCRIPTION, LAST MODIFIED, and DEPLOYED. The 'Appointment' object is highlighted with a red box. Other objects listed are 'Appointment Category', 'Appointment Invitation', and 'Appointment Invittee'.

2. Now click on “Fields & Relationships” >> New

The screenshot shows the 'Fields & Relationships' section of the Appointment object's details. The left sidebar has tabs for 'Details', 'Fields & Relationships' (which is selected and highlighted with a red box), 'Page Layouts', and 'Lightning Record Pages'. The main area shows a table with columns: FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED. Two fields are listed: 'Appointment Date' and 'Appointment Name'. A 'New' button is highlighted with a red box at the top right of the table.

3. Select “Look-up relationship” as data type and click Next.

The screenshot shows the 'Data Type' selection screen. At the top, it says 'Specify the type of information that the custom field will contain.' Below is a table with rows for different data types:

DATA TYPE	DESCRIPTION
<input type="radio"/> None Selected	Select one of the data types below
<input type="radio"/> Auto Number	A system-generated sequence number that uses a display format you define. The number is automatically incremented for each new record.
<input type="radio"/> Formula	A read-only field that derives its value from a formula expression you define. The formula field is updated when any of the source fields change.
<input type="radio"/> Roll Up Summary	A read-only field that displays the sum, minimum, or maximum value of a field in a related list or the record count of all records listed in a related list.
<input checked="" type="radio"/> Lookups Relationship	Creates a relationship that links this object to another object. The relationship field allows users to click on a lookup icon to select a value from a popup list. The other object is the source of the values in the list.
<input type="radio"/> Master-Detail Relationship	Creates a special type of parent-child relationship between this object (the child, or "detail") and another object (the parent, or "master") where: <ul style="list-style-type: none"> • The relationship field is required on all detail records. • The ownership and sharing of a detail record are determined by the master record.

At the bottom right, there are 'Next' and 'Cancel' buttons, with a red arrow pointing to the 'Next' button.

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.

Field	Operator	Value / Field
Appointment: Appointment Date	less than	Appointment: Created Date

Filter Type

Required. The user-entered value must match filter criteria.
If it doesn't, display this error message on save:
Value does not exist or does not match filter criteria.

Optional. The user can remove the filter or enter values that don't match criteria.

Lookup Window Text Add this informational message to the lookup window.

Active Enable this filter.

10. Error Message : Value does not match the criteria.
11. Enable the filter by click on Active.
12. Next >> Next >> Save.

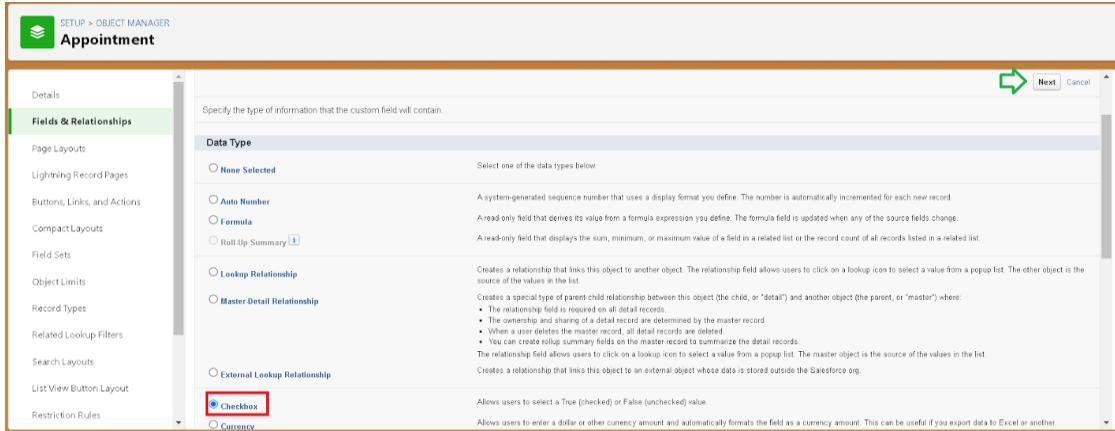
Creation of Lookup Field on Billing details and feedback Object :

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 Checkbox Fields

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

Appointment
New Custom Field

Help for this Page ?

Step 2. Enter the details Step 2 of 4

Previous Next Cancel

Field Label	Maintenance service
Default Value	<input checked="" type="radio"/> Checked <input type="radio"/> Unchecked
Field Name	Maintenance_service
Description	(empty)
Help Text	(empty)

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

Previous Next Cancel

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

Creation of Another Checkbox Field on Appointment Object :

1. Repeat the steps form 1 to 3.
2. Give the Field Label: Repairs
3. Field Nme : is auto populated
4. Default value : unchecked
5. Click on next >> next >> save.
6. Follow the same and create another checkbox with given names

7. Give the Field Label : Replacement Parts
8. Field Nme : is auto populated
9. Default value : unchecked
10. Click on next >> next >> save.

Creation of Checkbox 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 “Check box” as data type and click Next.
4. Give the Field Label : Quality Check Status
5. Field Name : is auto populated
6. Default value : unchecked
7. Click on next >> next >> save

Activity 4: 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.

Appointment
New Custom Field

Step 2. Enter the details Step 2 of 4

Help for this Page ?

Field Label:

Field Name:

Description:

Help Text:

Required: Always require a value in this field in order to save a record

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

Default Value: Show Formula Editor

Previous Next Cancel

Activity 5: Creation of Currency Fields

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

This screenshot shows the 'Step 2. Enter the details' screen for creating a new field. The field label is set to 'Service Amount'. The length is specified as 18 digits, and the decimal places are set to 0. The field name is 'Service_Amount'. There are input fields for 'Description' and 'Help Text'. Under 'Required', there is a checkbox for 'Always require a value in this field in order to save a record'. Below that, there is a checkbox for 'Auto add to custom report type' with an option to 'Add this field to existing custom report types that contain this entity'. Navigation buttons at the top right include 'Previous', 'Next', and 'Cancel'.

6. Click on next
7. Give read only for all the profiles in field level security for profile.

This screenshot shows the 'Step 3. Establish field-level security' screen. It displays the field details: Field Label 'Service Amounts', Data Type 'Currency', Field Name 'Service_Amounts', and Description. Below this, it asks to select profiles for field-level security. A table lists profiles: Analytics Cloud Integration User, Analytics Cloud Security User, Authenticated Website, Contract Manager, and Cross Org Data Proxy User. For each profile, there are checkboxes for 'Visible' and 'Read-Only'. A green arrow points to the 'Read-Only' column header. Navigation buttons at the top right include 'Previous', 'Next', and 'Cancel'.

8. Click on next >> save.

Creation of Currency Field on Billing details and feedback Object :

1. Follow the same steps as mentioned above in Billing details and feedback Object.
2. Change the label name as mentioned.
3. Give the Field Label : Payment Paid
4. Field Name : is auto populated

Activity 6: 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.

Step 2. Enter the details Step 2 of 4

Field Label: Vehicle number plate

Length: 10

Field Name: Vehicle_number_plate

Description:

Help Text:

Required: Always require a value in this field in order to save a record

Unique: Do not allow duplicate values

External ID: Set this field as the unique record identifier from an external system

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

8. Click on next >> next >> save.

Creation of Text Fields in 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 “text” as data type and click Next.
4. Give the Field Label : Rating for service
5. Field Name : is auto populated
6. Length :1
7. Make field as Required.
8. Click on next >> next >> save

Activity 7: Creation of Picklist Fields

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.

New Custom Field

Step 2. Enter the details Step 2 of 4

Field Label [i](#)

Values Use global picklist value set
 Enter values, with each value separated by a new line

Display values alphabetically, not in the order entered
 Use first value as default value [i](#)
 Restrict picklist to the values defined in the value set [i](#)

Field Name [i](#)

Description

Previous Next Cancel

6. Click Next.
7. Next >> Next >> Save.

Creation of Picklist Fields in 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. Click on fields & relationship >> click on New.
3. Select Data type as "Picklist" and click Next.
4. Enter Field Label as "Payment Status", under values select "Enter values, with each value separated by a new line" and enter values as shown below.
5. The values are: Pending, Completed.
6. Click Next.
7. Next >> Next >> Save.

Activity 8: 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.

Step 2. Choose output type

Field Label Field Name Up

Add this field to existing custom report types that contain this entity.

Formula Return Type

None Selected

Checkbox

Currency

Date

Date/Time

Select one of the data types below.

Calculate a boolean value
Example: `|TODAY() > CloseDate|`

Calculate a dollar or other currency amount and automatically format the field as a currency amount.
Example: `Gross Margin = Amount - Cost__c`

Calculate a date, for example, by adding or subtracting days to other dates.
Example: `Reminder Date = CloseDate - 7`

Calculate a datetime, for example, by adding a number of hours or days to another datetime.
Example: `Next = NOW() + 1`

5. Insert field formula should be : Created Date

Insert Field

Select a field, then click Insert. Labels followed by a ">" indicate that there are more fields available.

Service records >

- \$Api >
- \$Label >
- \$Organization >
- \$Profile >
- \$System >
- \$User >
- \$UserRole >

Appointment >

- Created By >
- Created By ID

Created Date

Last Activity Date

Last Modified By >

Last Modified By ID

Last Modified Date

You have selected:
CreatedDate
Type: Date/Time
API Name: CreatedDate

Insert

Close

Step 3. Enter formula

Enter your formula and click Check Syntax to check for errors. Click the Advanced Formula subtab to use additional fields, operators, and functions.

Example: `(Reminder Date = CloseDate - 7)` [More Examples...](#)

Simple Formula Advanced Formula

Insert Field Insert Operator

service_date (Date) = **CreatedDate** Left Arrow

Functions

ABS
ACOS
ADDMONTHS
AND
ASCII

Up Quick Tips

- * Getting Started
- * Operators & Functions

6. click "Check Syntax".
7. Click next >> next >> Save.

Milestone 6: Validation rule

Validation rules are applied when a user tries to save a record and are used to check if the data meets specified criteria. If the criteria are not met, the validation rule triggers an error message and prevents the user from saving the record until the issues are resolved.

Activity 1: To 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 'Object Manager' page for the 'Appointment' object. In the sidebar, under 'Buttons, Links, and Actions', the 'Validation Rules' link is highlighted with a green box. At the top right of the main content area, there is a 'New' button highlighted with a red box.

RULE NAME	ERROR LOCATION	ERROR MESSAGE	ACTIVE	MODIFIED BY
Vehicle	Vehicle number plate	Please enter valid number	✓	project 2, 25/09/2023, 11:56 am

3. Enter the Rule name as " Vehicle ".
4. Insert the Error Condition Formula as :-

The screenshot shows the 'Validation Rule Edit' screen for a rule named 'Vehicle'. The 'Rule Name' field contains 'Vehicle'. The 'Active' checkbox is checked. The 'Description' field contains 'vehicle'. The 'Error Condition Formula' field contains 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. A yellow box highlights the 'Insert Field' and 'Insert Operator' buttons. A blue box highlights the 'Functions' dropdown menu on the right, which lists various functions like ABS, ACOS, ADDMONTHS, AND, ASCII, ASIN, etc. A yellow box highlights the 'Insert Selected Function' button.

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.

Error Message

Example: Discount percent cannot exceed 30%

This message will appear when Error Condition formula is true

Error Message: Please enter valid number

This error message can either appear at the top of the page or below a specific field on the page

Error Location: Top of Page Field Vehicle number plate

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

Activity 2: To create a validation rule to an 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}"))

Validation Rule Edit

Rule Name: rating_should_be_less_than_5

Active:

Description:

Error Condition Formula: NOT(REGEX(Rating_for_service__c , "[1-5]{1}"))

Quick Tips

- Operators & Functions

R = Required Information

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.

Insert Field Insert Operator ▾

NOT(REGEX(Rating_for_service__c , "[1-5]{1}"))

Check Syntax

Error Message

Example: Discount percent cannot exceed 30%

This message will appear when Error Condition formula is true

Error Message: rating should be from 1 to 5

This error message can either appear at the top of the page or below a specific field on the page

Milestone 7: Duplicate rule

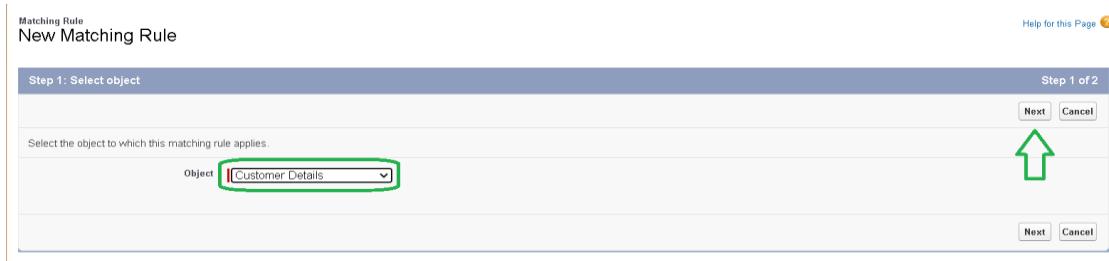
Activity 1: To create a matching rule to an Customer details Object

1. Go to quick find box in setup and search for matching Rule.

2. Click on matching rule >> click on New Rule.

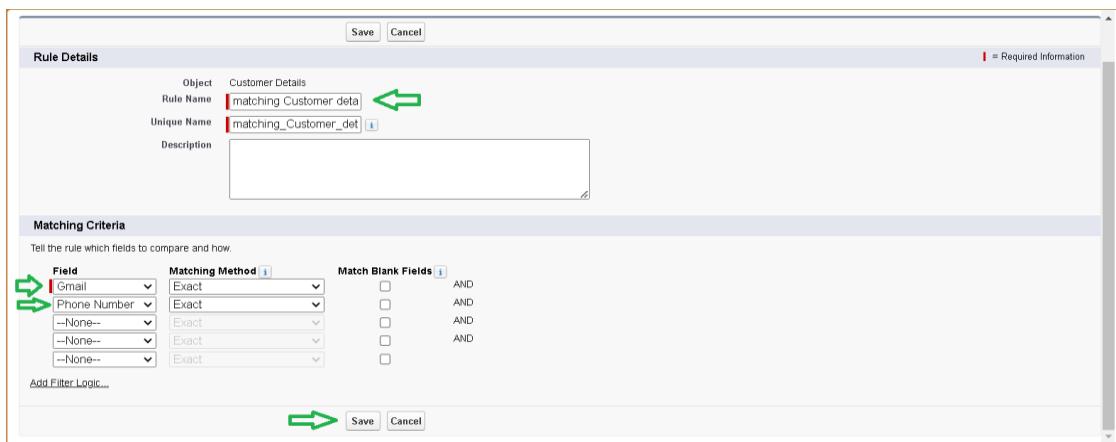


3. Select the object as Customer details and click Next.



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	Exact
2. Phone Number	Exact
8. Click save.
9. After Saving Click on Activate.



Matching Rule
matching Customer details

Help for this Page ?

Matching Rule Detail

Object	Customer Details
Rule Name	matching Customer details
Unique Name	matching_Customer_details
Description	
Matching Criteria	(Customer_Details: Gmail EXACT MatchBlank = FALSE) AND (Customer_Details: Phone_Number EXACT MatchBlank = FALSE)
Status	Inactive
Created By	project2, 25/09/2023, 10:15 am
Modified By	project2, 10/10/2023, 3:32 pm

Activity 2: 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.

Q: Duplicat

SETUP

Duplicate Management

Duplicate Error Logs

Duplicate Rules

All Duplicate Rules

What Are Duplicate Rules?

View: All Duplicate Rules

Rule Name	Description	New Rule	Matching Rule	Active	Last Modified By	Last Modified Date
Customer_Detail_duplicate	Identify accounts that duplicate other accounts	Customer Details	Matching Customer details	<input type="checkbox"/>	q2	10/10/2023
Standard Account Duplicate Rule	Identify accounts that duplicate other accounts	Account	Standard Account Matching Rule	<input checked="" type="checkbox"/>	q2	24/09/2023
Standard Contact Duplicate Rule	Identify contacts that duplicate other contacts and leads	Contact	Standard Lead Matching Rule	<input checked="" type="checkbox"/>	q2	24/09/2023
Standard Lead Duplicate Rule	Identify leads that duplicate other leads and contacts	Environment	Standard Contact Matching Rule	<input checked="" type="checkbox"/>	q2	24/09/2023
		Individual	Standard Lead Matching Rule	<input checked="" type="checkbox"/>	q2	24/09/2023
		Laptop	Standard Contact Matching Rule	<input checked="" type="checkbox"/>	q2	24/09/2023
		Lead				

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.

Edit Duplicate Rule
Customer Detail duplicate

Help for this Page ?

Duplicate Rule Edit

Save | Save & New | Cancel

Rule Details

Rule Name: Customer Detail duplicate

Description:

Object: Customer Details

Record Level Security: Enforce sharing rules (radio button selected)

Actions

Specify what happens when a user tries to save a duplicate record.

Action On Create	Allow <input checked="" type="checkbox"/> Alert <input type="checkbox"/> Report <input type="checkbox"/>
Action On Edit	Allow <input checked="" type="checkbox"/> Alert <input type="checkbox"/> Report <input type="checkbox"/>
Alert Text	Use one of these records?

Milestone 8: Profiles

A profile is a group/collection of settings and permissions that define what a user can do in salesforce. Profile controls “Object permissions, Field permissions, User permissions, Tab settings, App settings, Apex class access, Visualforce page access, Page layouts, Record Types, Login hours & Login IP ranges. You can define profiles by the user's job function. For example System Administrator, Developer, Sales Representative.

Types of profiles in salesforce

1. Standard profiles:

By default salesforce provides below standard profiles.

- Contract Manager
- Read Only
- Marketing User
- Solutions Manager
- Standard User
- System Administrator.

We cannot deleted standard ones

Each of these standard ones includes a default set of permissions for all of the standard objects available on the platform.

2. Custom Profiles:

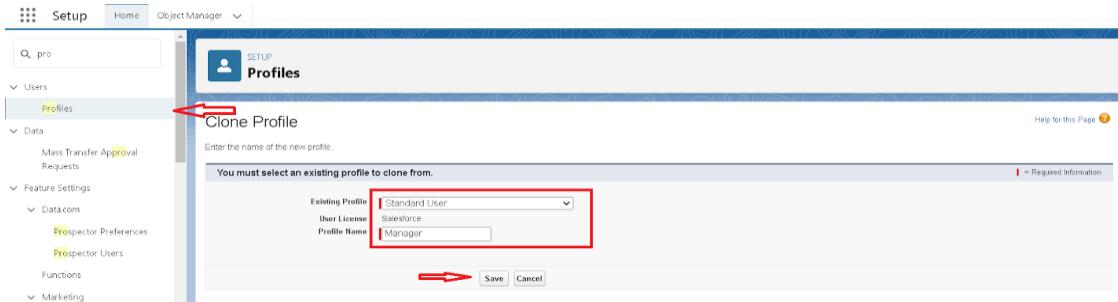
Custom ones defined by us.

They can be deleted if there are no users assigned with that particular one.

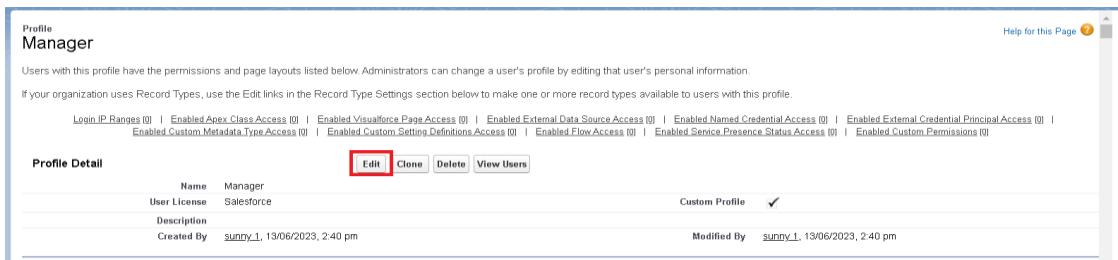
Activity 1: Manager Profile

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.



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	Read	Create	Edit	Delete	View All
Appointments		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>							
Billing details and feedback		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>							
Customer Details		<input checked="" type="checkbox"/>	<input 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"/>	<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"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Service records		<input checked="" type="checkbox"/>	<input 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"/>	<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: Sales person Profile

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

Custom Object Permissions						
	Basic Access		Data Administration			
	Read	Create	Edit	Delete	View All	Modify All
Appointments	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Billing details and feedback	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Customer Details	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input 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 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.

Milestone 9: Role & Role Hierarchy

A role in Salesforce defines a user's visibility access at the record level. Roles may be used to specify the types of access that people in your Salesforce organization can have to data. Simply put, it describes what a user could see within the Salesforce organization.

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. In the top left, there's a quick find bar with 'roles' typed in, and a sidebar with sections like 'Users' and 'Roles'. The 'Roles' section is expanded, showing a 'Territory-based Sample' role hierarchy. At the bottom right of the main content area, there's a red box around the 'Set Up Roles' button.

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



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

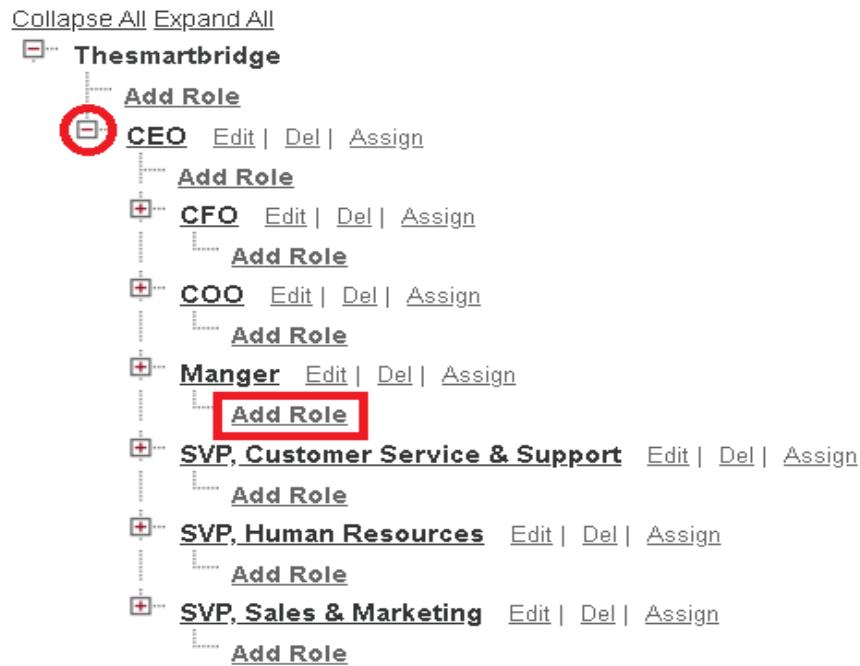
Role Edit	
Label	Manger
Role Name	Manger
This role reports to	CEO
Role Name as displayed on reports	
<input type="button" value="Save"/> <input type="button" value="Save & New"/> <input type="button" value="Cancel"/>	

Activity 2: Creating another roles

Creating another 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.



Milestone 10: Users

A user is anyone who logs in to Salesforce. Users are employees at your company, such as sales reps, managers, and IT specialists, who need access to the company's records. Every user in Salesforce has a user account. The user account identifies the user, and the user account settings determine what features and records the user can access.

Activity 1: Create User

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

3. Save.

Activity 2: Creating another users

1. 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.

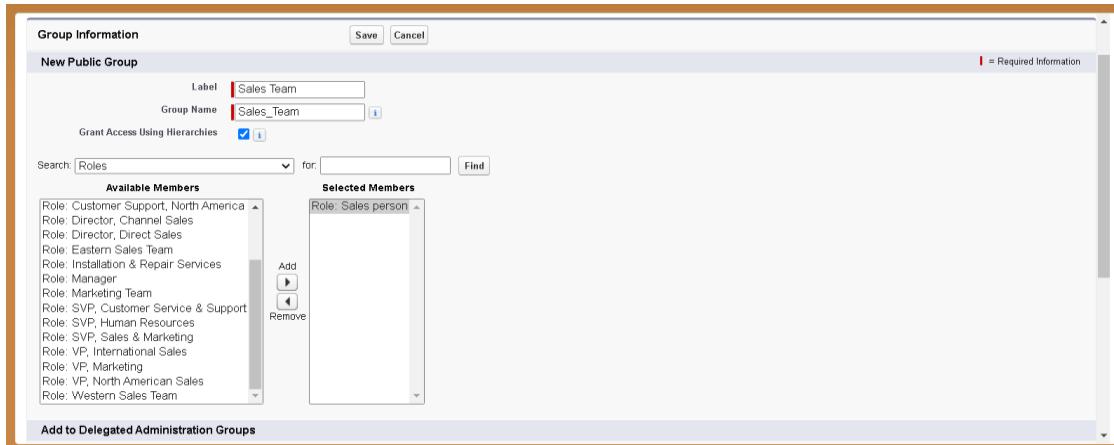
Milestone 11: Public groups

Public groups are a valuable tool for Salesforce administrators and developers to streamline user management, data access, and security settings. By creating and using public groups effectively, you can maintain a secure and organized Salesforce environment while ensuring that users have appropriate access to the resources they need.

Activity 1: Creating New Public Group

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.



Milestone 12: Sharing Setting

Salesforce allows you to configure sharing settings to control how records are accessed and shared within your organization. These settings are crucial for maintaining data security and privacy. Salesforce provides a variety of tools and mechanisms to define and enforce sharing rules, such as:

Organization-Wide Default (OWD) Settings:

These settings define the default level of access for all objects within your Salesforce org. OWD settings include Private, Public Read-Only, Public Read/Write, and Controlled by Parent.

OWD settings can be configured for each standard and custom object.

Role Hierarchy:

Salesforce uses a role hierarchy to determine record access.

Users at higher levels in the hierarchy have greater access to records owned by or shared with users lower in the hierarchy.

The role hierarchy is often used in combination with OWD settings to grant different levels of access.

Profiles and Permission Sets:

Profiles and permission sets allow administrators to specify object-level and field-level permissions for users.

Profiles are typically used to grant general object and field access, while permission sets can be used to extend those permissions to specific users.

Sharing Rules:

Sharing rules are used to extend access to records for users who meet specific criteria. They can be used to grant read-only or read-write access to records owned by other users.

Manual Sharing:

Administrators and record owners can manually share specific records with other users or groups.

Activity 1: Creating 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.

Object	Default sharing level	Sharing rule 1	Sharing rule 2	Sharing rule 3
Work Plan Template	Private	Private	✓	
Work Step Template	Private	Private	✓	
Work Type	Private	Private	✓	
Work Type Group	Public ReadWrite	Private	✓	
Appointment	Public ReadWrite	Private	✓	
Billing details and feedback	Public ReadWrite	Private	✓	
Customer Details	Public ReadWrite	Private	✓	
Environment	Public ReadWrite	Private	✓	
Laptop	Public ReadWrite	Private	✓	
Service records	Private	Private	✓	
SessionData	Public ReadWrite	Private	✓	

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

Sharing Rule	Action
No sharing rules specified.	

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

Step 1: Rule Name

Label: sharing settings
Rule Name: sharing_settings

Step 2: Select your rule type

Rule Type: Based on record owner Based on criteria

Step 3: Select which records to be shared

Service records: owned by members of Roles: Sales person

Step 4: Select the users to share with

Share with Roles: Manager

Step 5: Select the level of access for the users

Access Level: Read/Write

Buttons: Save

Milestone 13: Flows

In Salesforce, a flow is a powerful tool that allows you to automate business processes, collect and update data, and guide users through a series of screens or steps. Flows are built using a visual interface and can be created without any coding knowledge.

Activity 1: Create a Flow

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

Setup Home Object Manager

Q Flows 1

Process Automation 2

Flows 2

Identity

Login Flows

Didn't find what you're looking for? Try using Global Search.

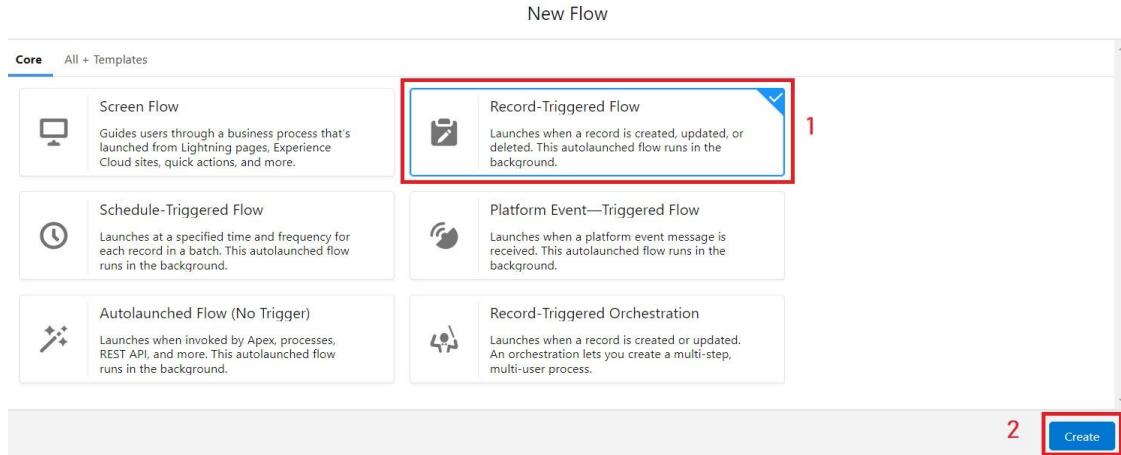
SETUP Flows

Flow Trigger Explorer New Flow 3

All Flows

Flow Label	Process Type	Ac...	Te...	Package State	Pa...	Last Modified By	Last Modified ...
Ac Amount update	Auto-launched Flow	<input type="checkbox"/>	<input type="checkbox"/>	Unmanaged		Veera Venkata Varaprasad Androthu	07/06/2023, 11:35 am
Book Appointment from Invitation	Salesforce Scheduler Flow	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Managed-installed			
Cancel Item Flow	Screen Flow	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Managed-installed			
Change Case Owner to Incident Owner	Screen Flow	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Managed-installed			
Close Change Request & Related Issues	Screen Flow	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Managed-installed			

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



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

Configure Start

Select Object

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

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:

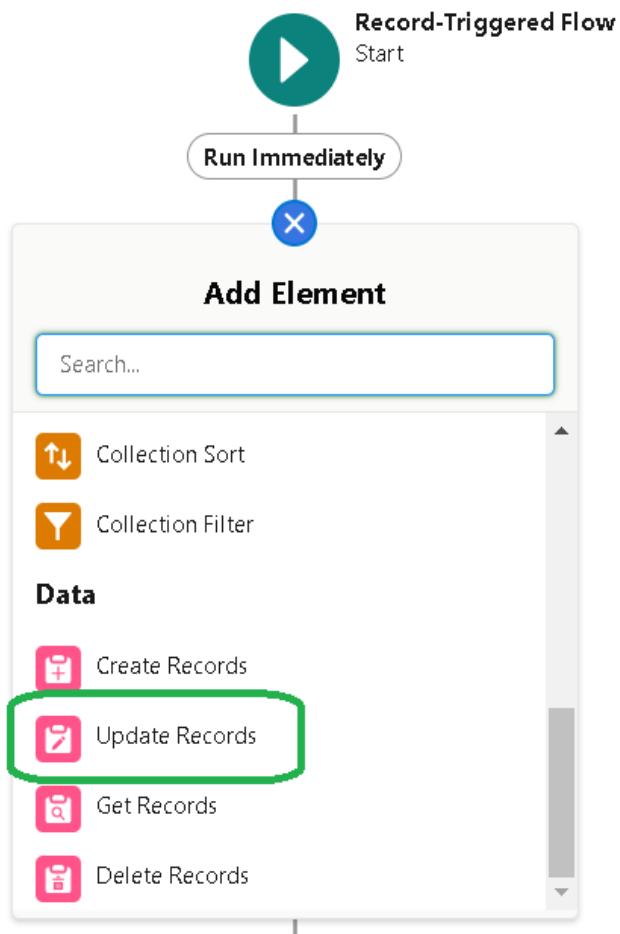
Fast Field Updates

Update fields on the record that triggers the flow to run. This high-performance flow runs *before* the record is saved to the database.

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

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

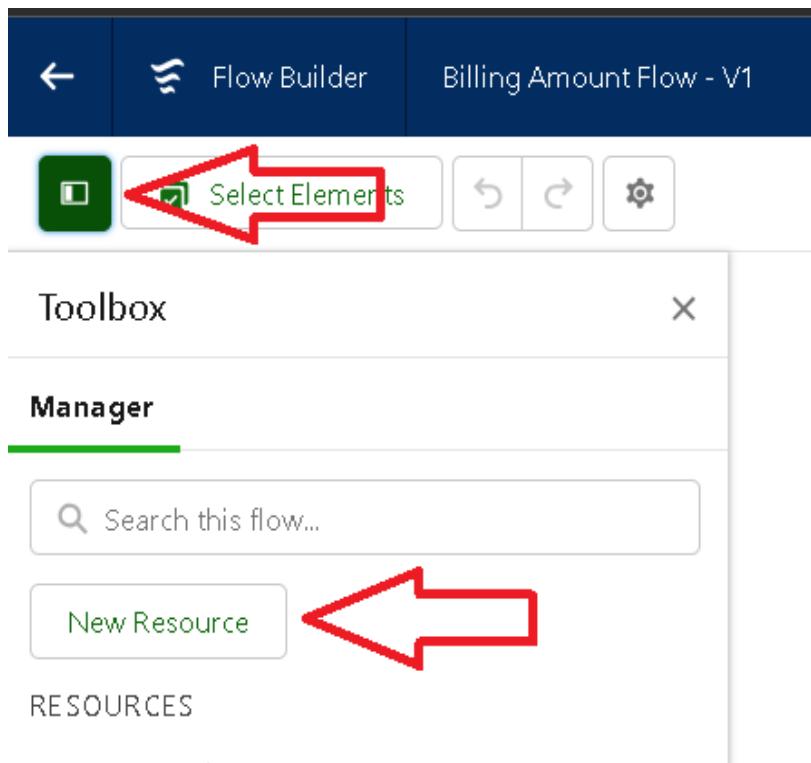
Field	Value
Payment_Paid__c	\$Record > Service records > Appointment > Service A... X

+ 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.

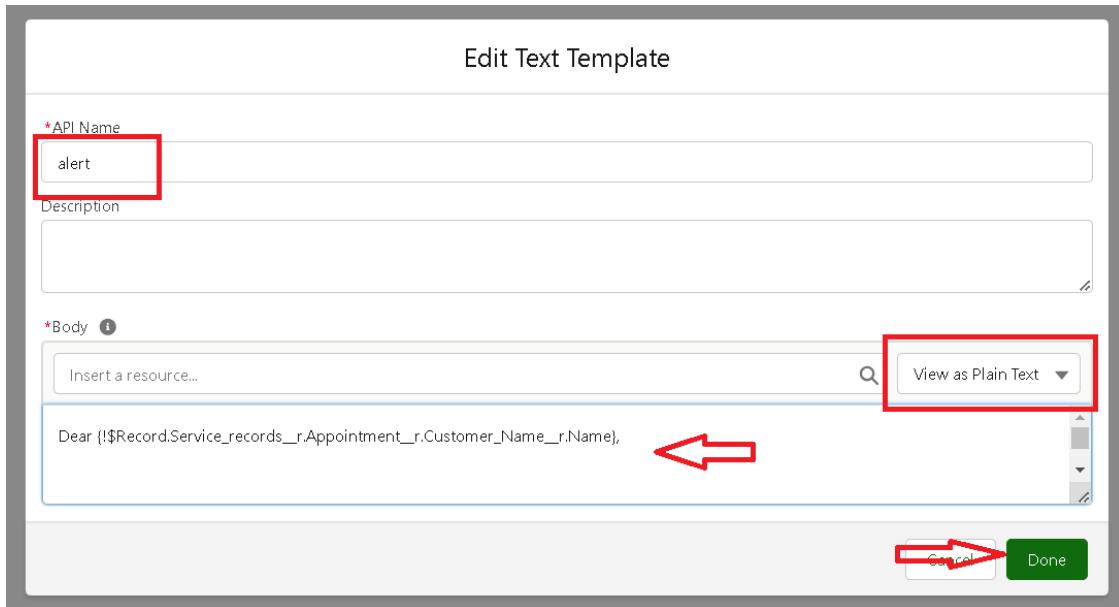
Dear {\$Record.Service_records__r.Appointment__r.Customer_Name__r.Name},

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	*API Name
Email Alert	Email_Alert
Description	
<div style="border: 1px solid #ccc; height: 40px; margin-bottom: 10px;"></div>	

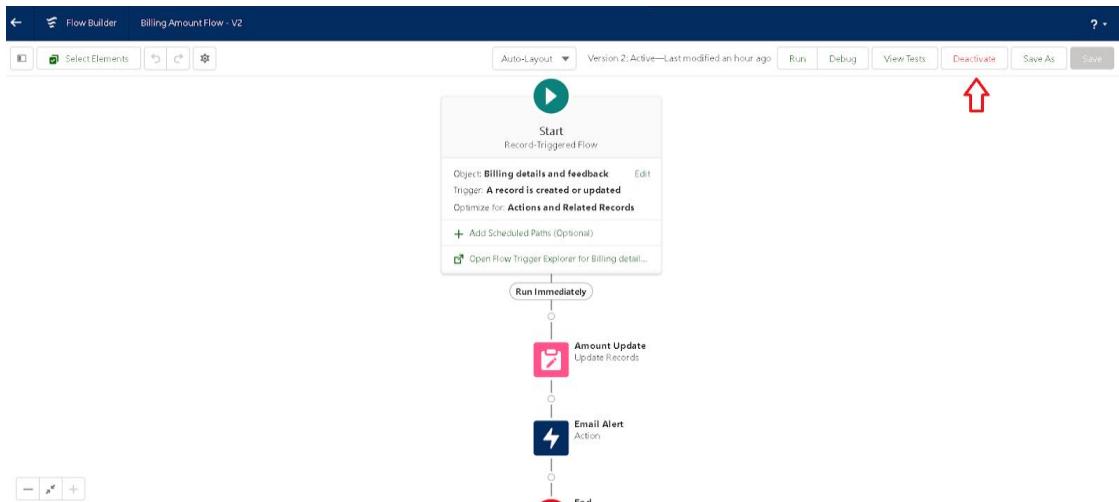
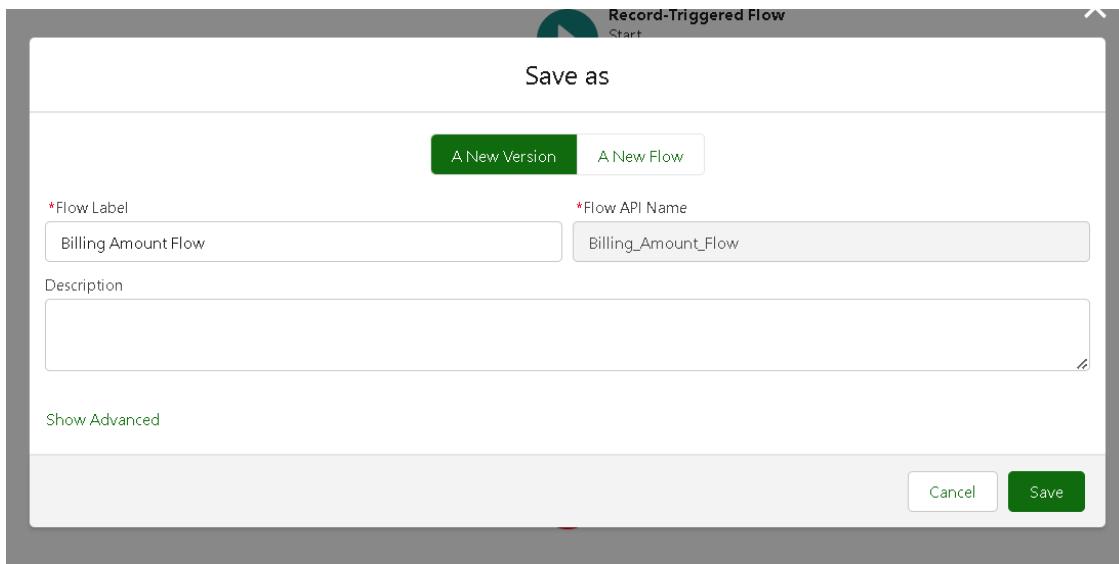
Set Input Values for the Selected Action

A_a Body	<input type="text" value="={!alert}"/> <div style="margin-left: 20px;"> <input checked="" type="checkbox"/> Include <input type="checkbox"/> Don't Include </div>
A_a Email Template ID	<input type="text"/> <div style="margin-left: 20px;"> <input type="checkbox"/> Don't Include </div>
A_a Log Email on Send	<input type="text"/> <div style="margin-left: 20px;"> <input type="checkbox"/> Don't Include </div>

Edit Action

A_a Recipient Address List	<input type="text" value="={!\$Record.Service_records__r.Appointment__r.Cus}"/> <div style="margin-left: 20px;"> <input checked="" type="checkbox"/> Include <input type="checkbox"/> Don't Include </div>
A_a Recipient ID	<input type="text"/> <div style="margin-left: 20px;"> <input type="checkbox"/> Don't Include </div>
A_a Related Record ID	<input type="text"/> <div style="margin-left: 20px;"> <input type="checkbox"/> Don't Include </div>
A_a Rich-Text-Formatted Body	<input type="text"/> <div style="margin-left: 20px;"> <input type="checkbox"/> Don't Include </div>
A_a Sender Email Address	<input type="text"/> <div style="margin-left: 20px;"> <input type="checkbox"/> Don't Include </div>
A_a Sender Type	<input type="text"/> <div style="margin-left: 20px;"> <input type="checkbox"/> Don't Include </div>
A_a Subject	<input type="text" value="Thank You for Your Payment - Garage Manageme"/> <div style="margin-left: 20px;"> <input checked="" type="checkbox"/> Include <input type="checkbox"/> Don't Include </div>

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**

The screenshot shows the Salesforce Flow builder interface. At the top, it says "Set Filter Conditions". Below that, under "Condition Requirements to Update Record", there is a dropdown menu set to "All Conditions Are Met (AND)". Under "Set Filter Conditions", there is a row with "Field: Quality_Check_Status__c", "Operator: Equals", and "Value: True". A "Delete" button is also present. Below this, there is a blue "Add Condition" button. Under "Set Field Values for the Service record Record", there is a row with "Field: Service_Status__c" and "Value: Completed". A "Delete" button is also present. Below this, there is a blue "Add Field" button.

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**.

Milestone 14: Apex Trigger

Apex can be invoked by using triggers. Apex triggers enable you to perform customer actions before or after changes to Salesforce records, such as insertions, updates, or deletions.

A trigger is Apex code that executes before or after the following types of operations:

- insert
- update
- delete
- merge
- upsert
- undelete

For example, you can have a trigger run before an object's records are inserted into the database, after records have been deleted, or even after a record is restored from the Recycle Bin.

You can define triggers for top-level standard objects that support triggers, such as a Contact or an Account, some standard child objects, such as a CaseComment, and custom objects. To define a trigger, from the object management settings for the object whose triggers you want to access, go to Triggers.

There are primarily two types of Apex Triggers:

Before Trigger: This type of trigger in Salesforce is used either to update or validate the values of a record before they can be saved into the database. So, basically, the before trigger validates the record first and then saves it. Some criteria or code can be set to check data before it gets ready to be inserted into the database.

After Trigger: This type of trigger in Salesforce is used to access the field values set by the system and affect any change in the record. In other words, the after trigger makes changes to the value from the data inserted in some other record.

Activity 1: Apex handler

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

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”.



```
AmountDistributionApex * AmountDistributionHandler.apex * 
Code Coverage: None API Version: 50 Go To
1 * public class AmountDistributionHandler {
2
3 *     public static void amountDist(list<Appointment__c> listApp){
4 *         list<Service_records__c> serlist = new list <Service_records__c>();
5
6 *         for(Appointment__c app : listApp){
7 *             if(app.Maintenance_service__c == true && app.Repairs__c == true && app.Replacement_Parts__c == true){
8 *                 app.Service_Amount__c = 10000;
9 *             }
10 *             else if(app.Maintenance_service__c == true && app.Repairs__c == true){
11 *                 app.Service_Amount__c = 5000;
12 *             }
13 *             else if(app.Maintenance_service__c == true && app.Replacement_Parts__c == true){
14 *                 app.Service_Amount__c = 8000;
15 *             }
16 *             else if(app.Repairs__c == true && app.Replacement_Parts__c == true){
17 *                 app.Service_Amount__c = 7000;
18 *             }
19 *             else if(app.Maintenance_service__c == true){
20 *                 app.Service_Amount__c = 2000;
21 *             }
22 *         }
23 *     }
24 }
```

```
AmountDistribution.apxt AmountDistributionHandler.apxc * [1]
Code Coverage: None | API Version: 58
12     }
13     else if(app.Maintenance_service__c == true && app.Replacement_Parts__c == true){
14         app.Service_Amount__c = 8000;
15     }
16     else if(app.Repairs__c == true && app.Replacement_Parts__c == true){
17         app.Service_Amount__c = 7000;
18     }
19     else if(app.Maintenance_service__c == true){
20         app.Service_Amount__c = 2000;
21     }
22     else if(app.Repairs__c == true){
23         app.Service_Amount__c = 3000;
24     }
25     else if(app.Replacement_Parts__c == true){
26         app.Service_Amount__c = 5000;
27     }
28 }
29 }
30 }
31 }
```

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;
            }
        }
    }
}
```

```
}
```

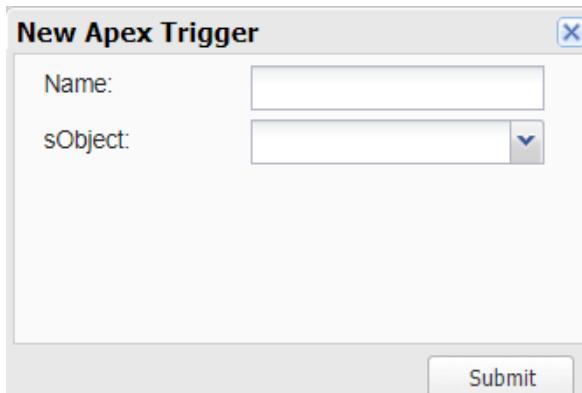
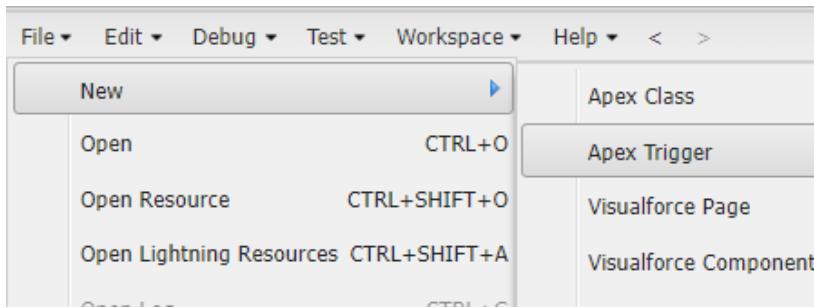
```
}
```

```
}
```

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 :

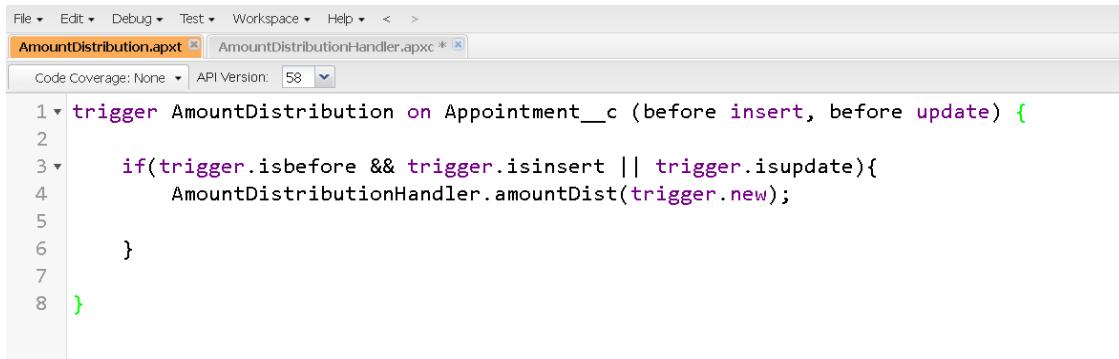
Trigger [trigger name] on [object name](Before/After event)

```
{
```

}

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 ▾ < >
- Tab bar: AmountDistribution.apxt (selected) and AmountDistributionHandler.apxc *
- Code Coverage: None
- API Version: 58
- Trigger code:

```
1 trigger AmountDistribution on Appointment__c (before insert, before update) {
2
3     if(trigger.isbefore && trigger.isinsert || trigger.isupdate){
4         AmountDistributionHandler.amountDist(trigger.new);
5
6     }
7
8 }
```

Code:

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

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

    }

}
```

Milestone 15: Reports

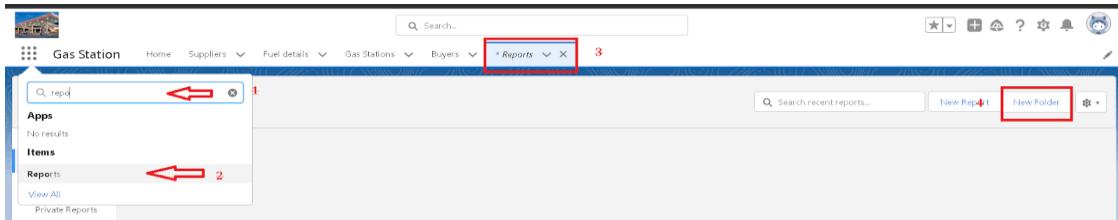
Reports give you access to your Salesforce data. You can examine your Salesforce data in almost infinite combinations, display it in easy-to-understand formats, and share the resulting insights with others. Before building, reading, and sharing reports, review these reporting basics.

Types of Reports in Salesforce

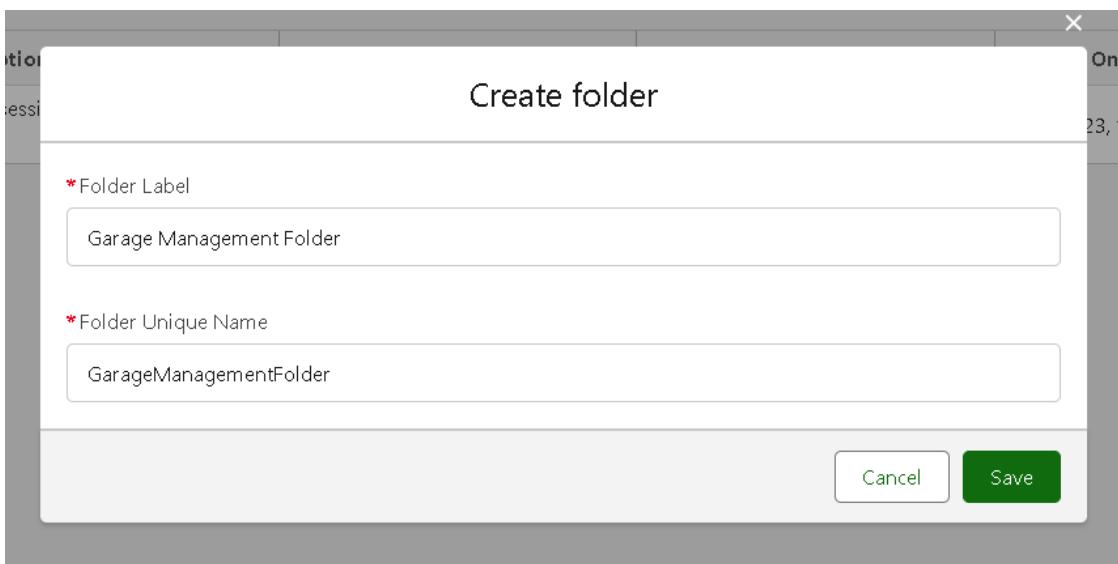
1. Tabular
2. Summary
3. Matrix
4. Joined Reports

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.

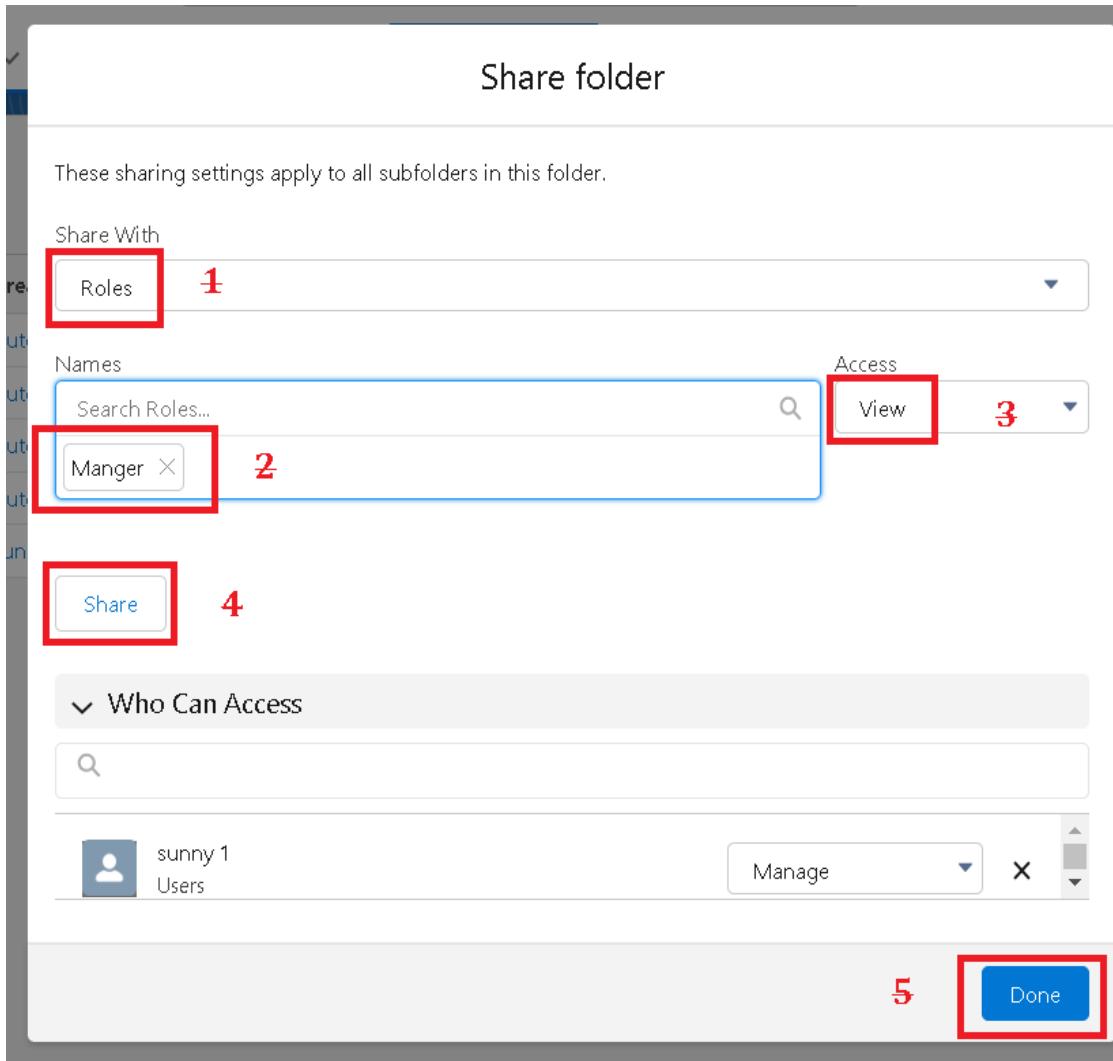


3. Give the Folder label as "Garage Management Folder", Folder unique name will be auto populated.
4. Click save.



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 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 in the Salesforce Setup. In the left sidebar, there's a 'Report Types' link with a green arrow pointing to it. At the top of the main content area, there's a 'New Custom Report Type' button with another green arrow pointing to it. The main table lists several report types, each with an 'Action' column containing 'Edit | Del' links and a 'Label' column showing names like 'Bot Metrics Daily Summer '23', 'Bot Metrics Hourly Summer '23', 'Screen Flows', and 'Session Metrics Summer '23'. The 'Description' column provides brief details for each. The 'Category' column shows 'Other Reports' for all entries. The 'Deployed' column has checked boxes. The 'Created By Alias' and 'Created Date' columns show 'autoproc' and the date '28/09/2023' respectively. A navigation bar at the bottom includes letters from A to Z and an 'All' option.

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.

Primary Object: Customer Details

Identification

Report Type Label: Service information

Report Type Name: Service_Information

Description: Service information

Note: Description will be visible to users who create reports.

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 Selected Object

Activities

Appointments

Duplicate Record Items

Help for this Page ?

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 Report

Note : Before creating report, create latest “10” records in every object.

Try to fill every field in each record for better experience.

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

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

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

4. Their outline pane is opened already, select the fields that mentioned below in column section.
 - a. Customer name
 - b. Appointment Date
 - c. Service Status
 - d. Payment paid
5. Remove the unnecessary fields.
6. Select the fields that mentioned below in GROUP ROWS section.
 - a. Rating for Service
7. Select the fields that mentioned below in GROUP ROWS section.
 - a. 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 the ServiceNow report builder interface. At the top, there's a toolbar with buttons for Save & Run, Save, Close, and Run. Below the toolbar is a preview area showing a chart titled "Rating for service" with a blue line graph. The chart has "Rating for service" on the x-axis (values 4 and 5) and "Sum of Payment Paid" on the y-axis (values 0, 8k, 16k). The chart shows two data points: (4, ~15k) and (5, ~5k). To the left of the chart is a table with columns: Rating for service, Payment Status, Completed, and Total. Rows show: Sum of Payment Paid (₹15,000), Record Count (4), Sum of Payment Paid (₹5,000), Record Count (2); and a summary row "Total" with Sum of Payment Paid (₹20,000) and Record Count (6). Below the chart is a table titled "Details (6 Rows)" with columns: Customer Name, Appointment Date, Service Status, and Payment Paid. Rows list appointments for meghana, rushi, shivam, and another rushi. At the bottom of the preview area are buttons for Row Counts, Detail Rows, Grand Total, and Stacked Summaries.

Save Report

*Report Name: New Service information Report

Report Unique Name: New_Service_information_Report_oVu

Report Description:

Folder: Garage Management Folder

Cancel Save

Milestone 16: Dashboards

Dashboards help you visually understand changing business conditions so you can make decisions based on the real-time data you've gathered with reports. Use dashboards to help users identify trends, sort out quantities, and measure the impact of their activities. Before building, reading, and sharing dashboards, review these dashboard basics.

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.

Create folder

* Folder Label
Service Rating

* Folder Unique Name
ServiceRating

Cancel **Save**

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

Activity 2: Create Dashboard

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

New Dashboard

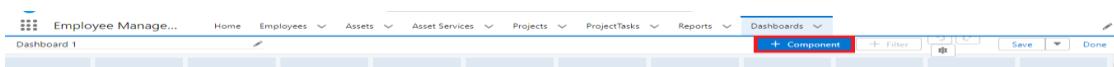
* Name
Customer review

Description

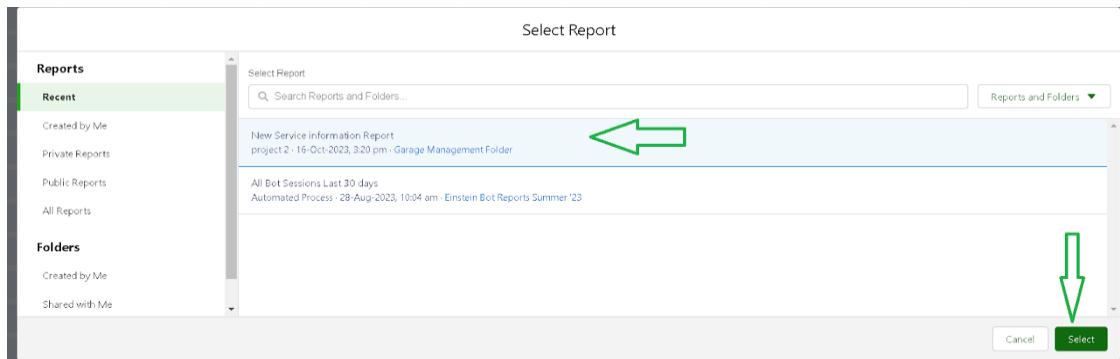
Folder
Service Rating **Select Folder**

Cancel **Create**

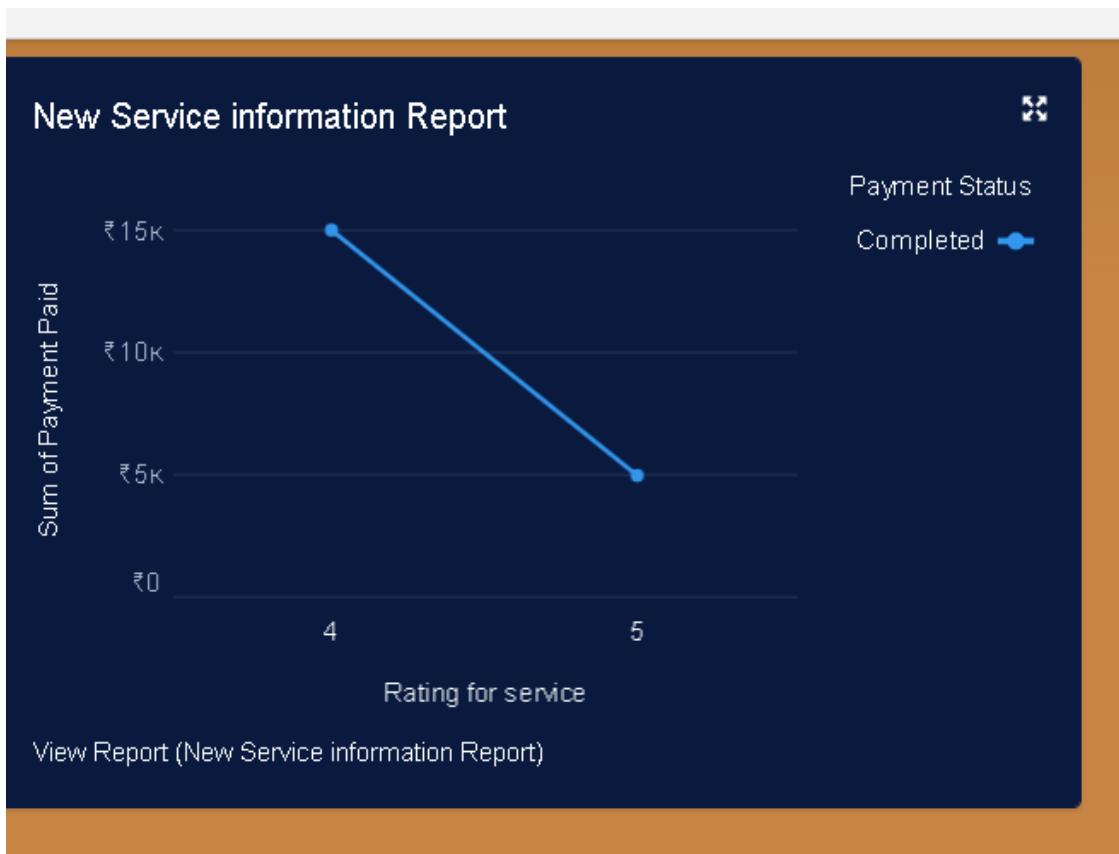
- Select add component.



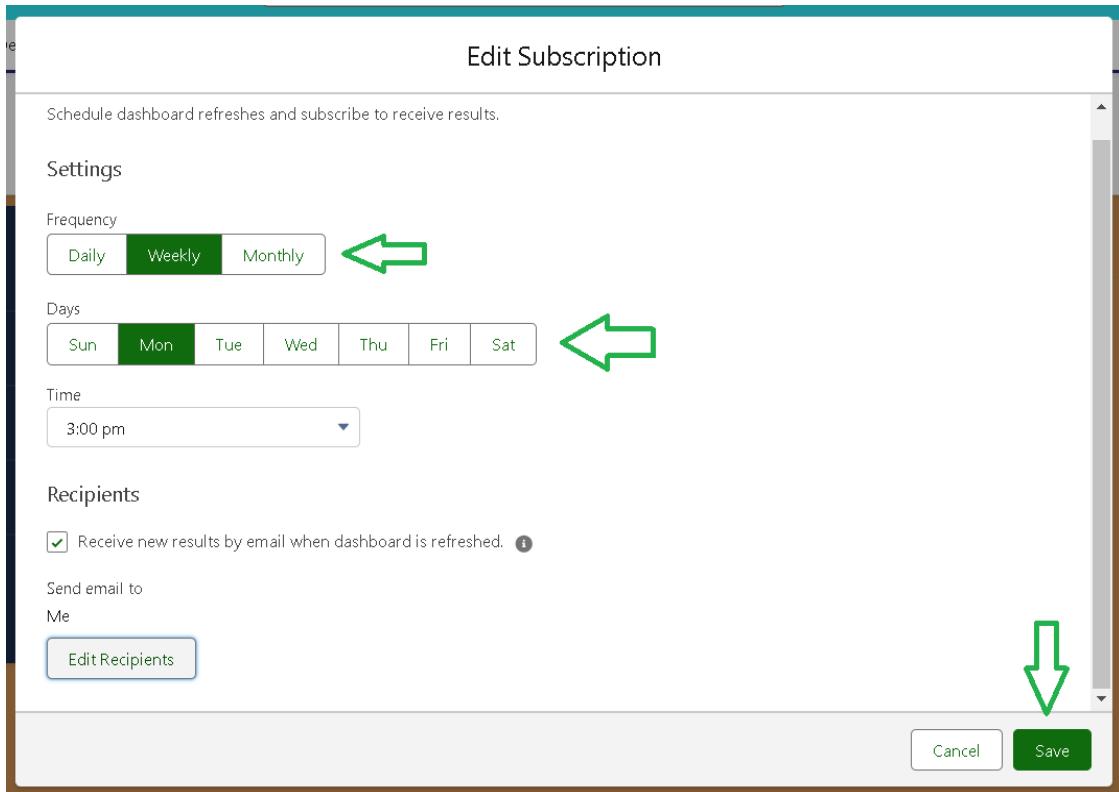
4. Select a Report and click on select.



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.



Milestone 17: 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		

Buttons: 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 System

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		
Modified By	Annapurna SmartBridge, 18/11/2024, 3:28 pm		

Buttons: 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

* Appointment

app-016

Quality Check Status

Service Status

Started

service date

18/11/2024

This field is calculated upon save

Created By

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

Save

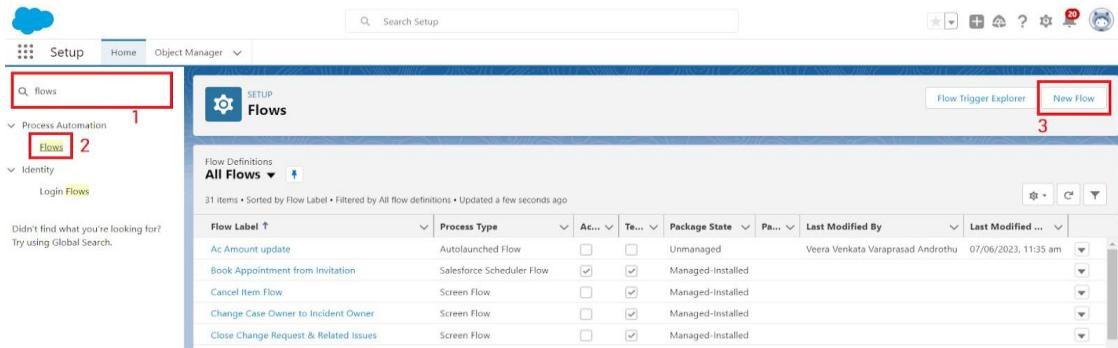
6. Now automatically Service status will be moved to completed.

Milestone 13: Flows

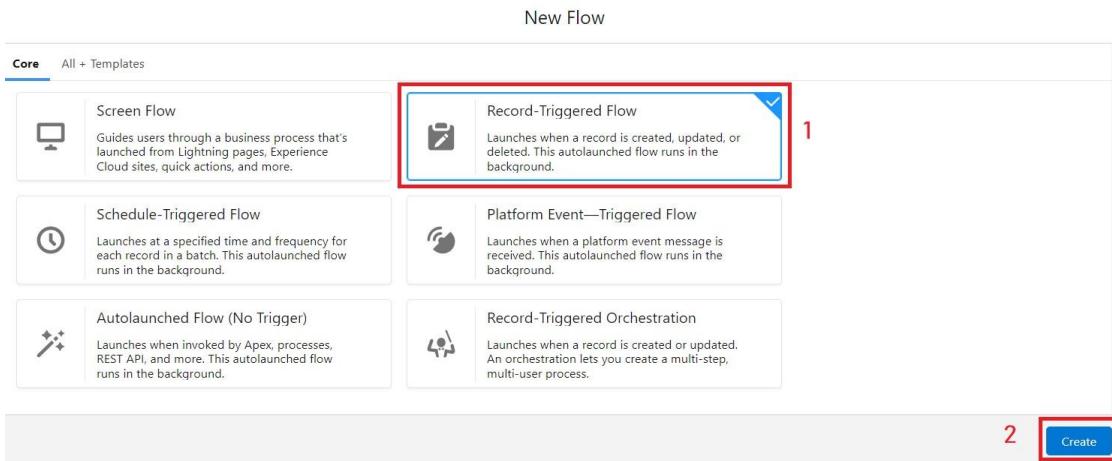
In Salesforce, a flow is a powerful tool that allows you to automate business processes, collect and update data, and guide users through a series of screens or steps. Flows are built using a visual interface and can be created without any coding knowledge.

Activity 1:Create a Flow

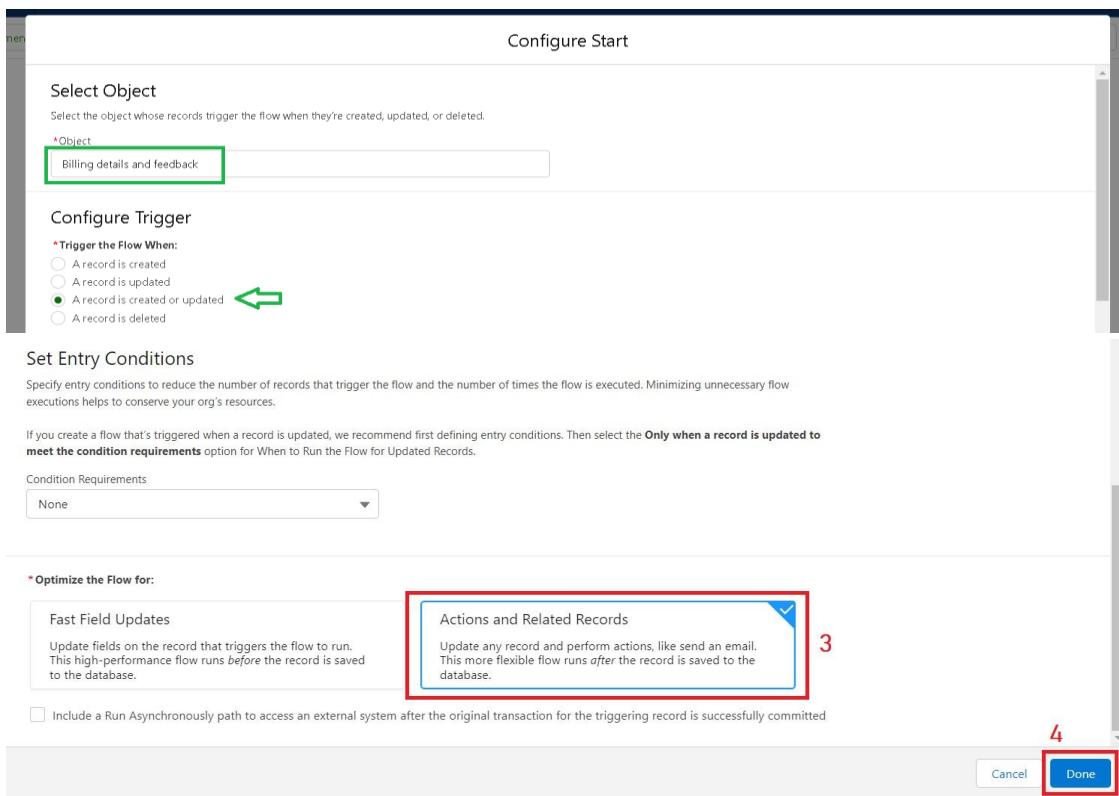
1. Go to setup >> type Flow in quick find box >> Click on the Flow and Select the New Flow.



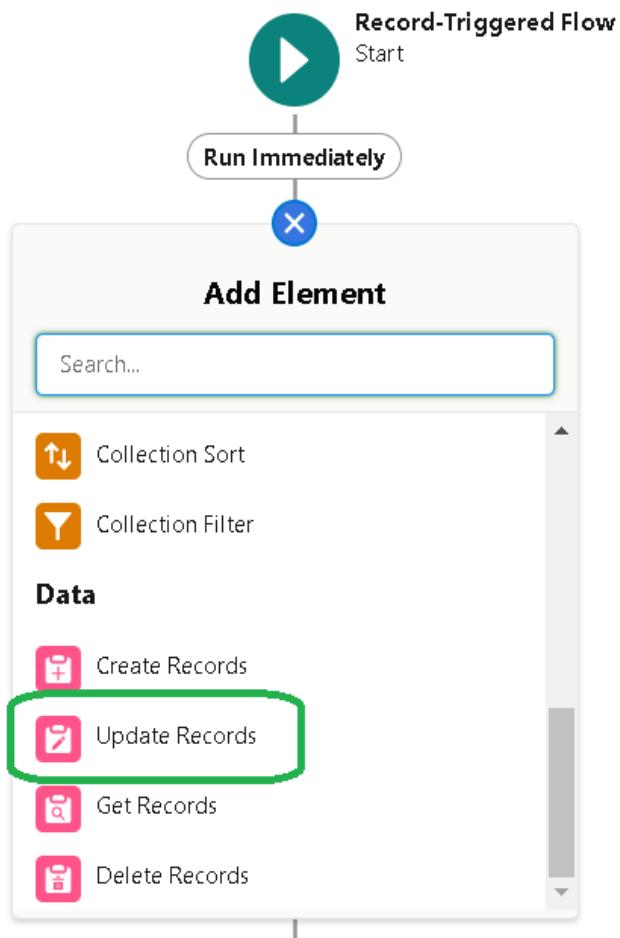
2. Select the Record-triggered flow and Click on Create.



3. Select the Object as “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. 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

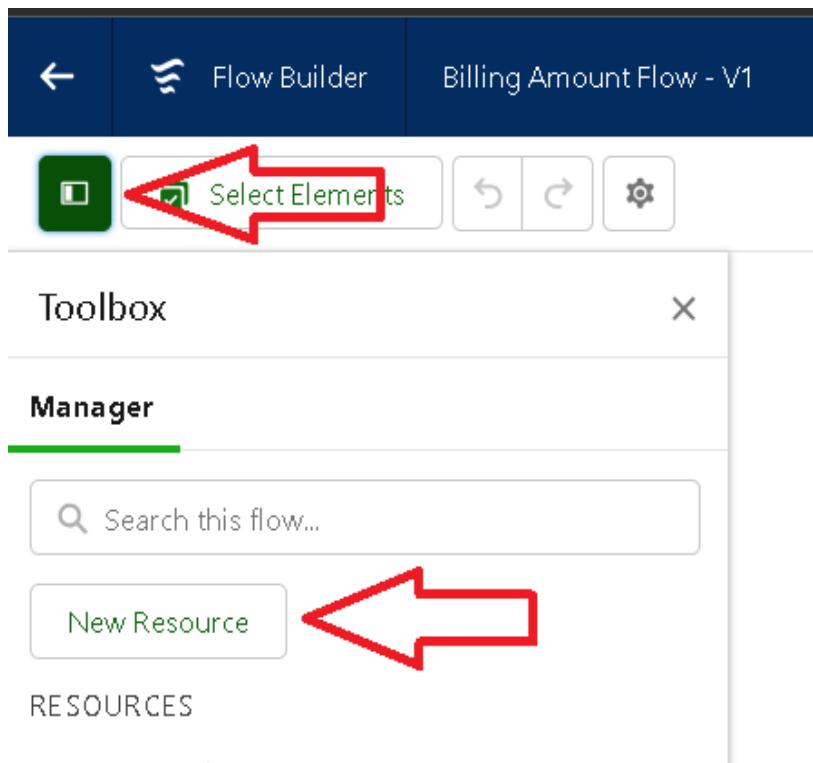
Field	Value
Payment_Paid__c	\$Record > Service records > Appointment > Service A... X

+ 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.



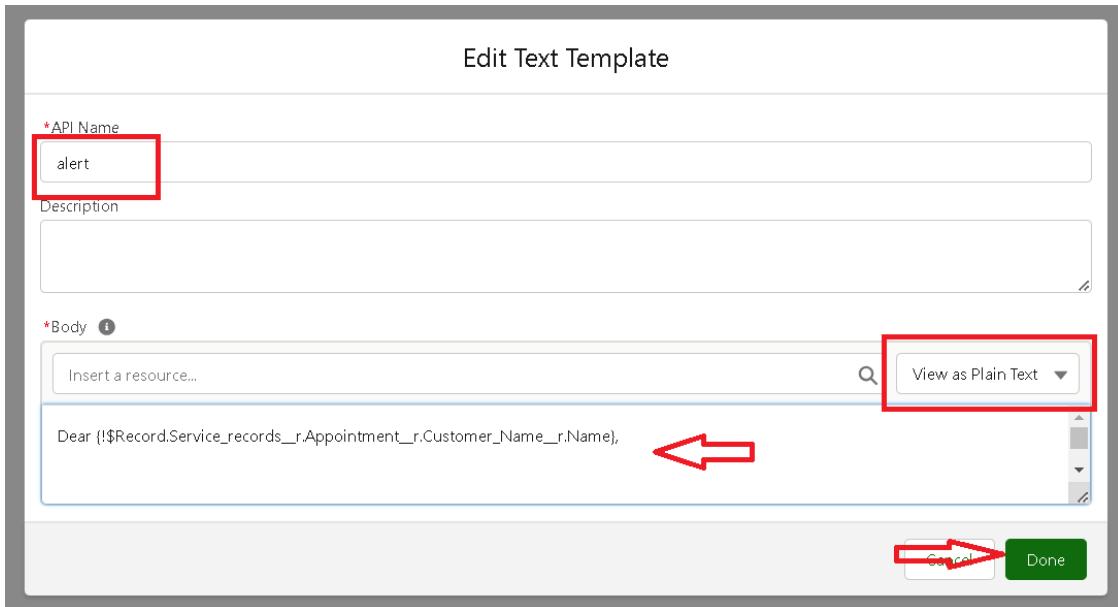
18. Click on the New Resource, And select Variable.
19. Select the resource type as text template.
20. Enter the API name as " alert".
21. Change the view as Rich Text ? View to Plain Text.
22. In body field paste the syntax that given below.

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	*API Name
Email Alert	Email_Alert
Description	
<div style="border: 1px solid #ccc; height: 40px; margin-bottom: 10px;"></div>	

Set Input Values for the Selected Action

A_a Body	<input type="text" value="={!alert}"/> <div style="margin-left: 10px;"> <input checked="" type="checkbox"/> Include <input type="checkbox"/> Don't Include </div>
A_a Email Template ID	<input type="text"/> <div style="margin-left: 10px;"> <input type="checkbox"/> Don't Include </div>
A_a Log Email on Send	<input type="text"/> <div style="margin-left: 10px;"> <input type="checkbox"/> Don't Include </div>

Edit Action

A_a Recipient Address List	<input type="text" value="={!\$Record.Service_records__r.Appointment__r.Cus}"/> <div style="margin-left: 10px;"> <input checked="" type="checkbox"/> Include <input type="checkbox"/> Don't Include </div>
A_a Recipient ID	<input type="text"/> <div style="margin-left: 10px;"> <input type="checkbox"/> Don't Include </div>
A_a Related Record ID	<input type="text"/> <div style="margin-left: 10px;"> <input type="checkbox"/> Don't Include </div>
A_a Rich-Text-Formatted Body	<input type="text"/> <div style="margin-left: 10px;"> <input type="checkbox"/> Don't Include </div>
A_a Sender Email Address	<input type="text"/> <div style="margin-left: 10px;"> <input type="checkbox"/> Don't Include </div>
A_a Sender Type	<input type="text"/> <div style="margin-left: 10px;"> <input type="checkbox"/> Don't Include </div>
A_a Subject	<input type="text" value="Thank You for Your Payment - Garage Manageme"/> <div style="margin-left: 10px;"> <input checked="" type="checkbox"/> Include <input type="checkbox"/> Don't Include </div>

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

Record-Triggered Flow

Save as

A New Version **A New Flow**

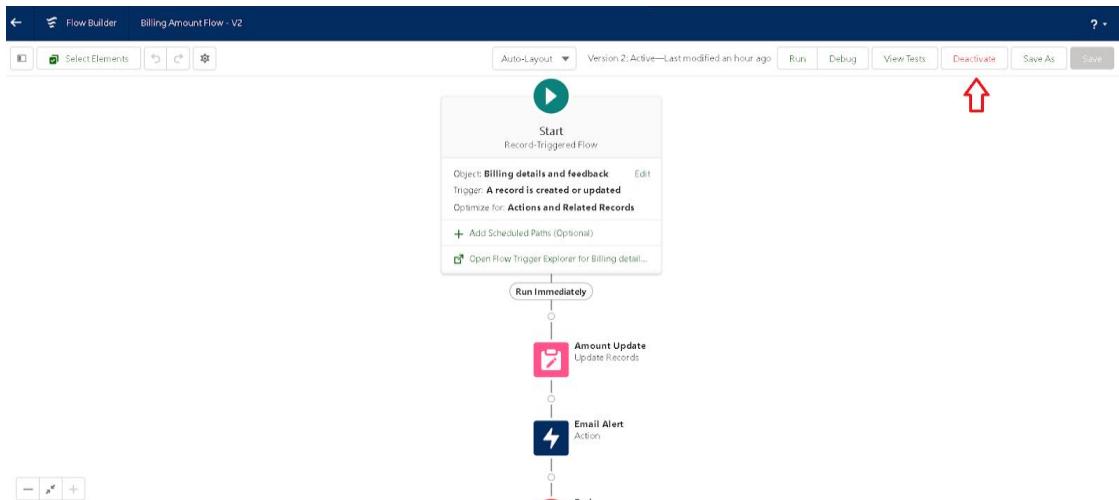
*Flow Label *Flow API Name

Billing Amount Flow Billing_Amount_Flow

Description

Show Advanced

Cancel Save



Activity 2: Create another Flow

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

Setup Home Object Manager

Q flows 1

Process Automation

Flows 2

Identity

Login Flows

Didn't find what you're looking for? Try using Global Search.

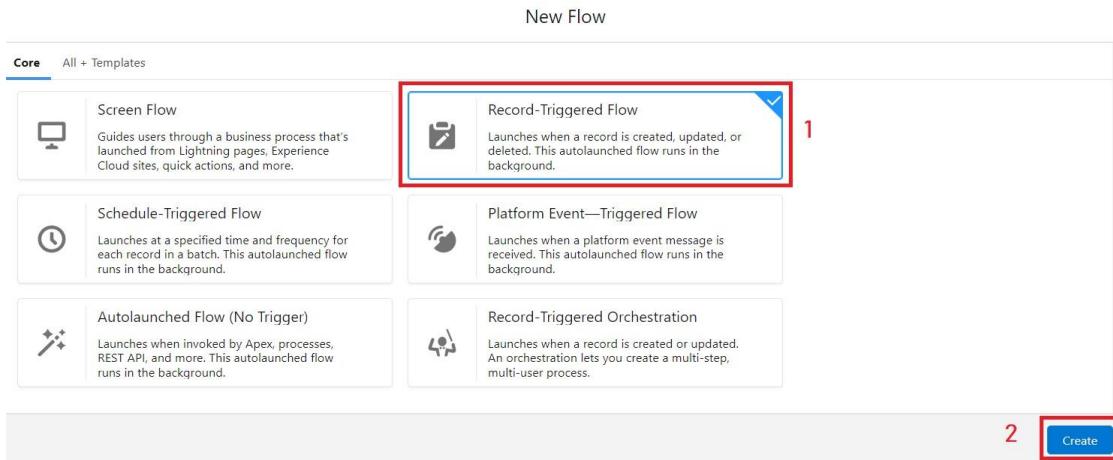
SETUP Flows

Flow Definitions All Flows 3

Flow Trigger Explorer New Flow

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

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	Operator	Value
Quality_Check_Status__c	Equals	True X

+ Add Condition

Set Field Values for the Service record Record

Field	Value
Service_Status__c	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**.

Conclusion :

The Garage Management System successfully demonstrates how technology can simplify and optimize the daily operations of an automobile service center. By digitizing processes such as customer management, vehicle tracking, job scheduling, billing, and inventory control, the system reduces manual effort, minimizes errors, and ensures smooth workflow.

The project highlights the importance of integrating multiple modules into a single platform to achieve efficiency, accuracy, and transparency. Garage owners gain valuable insights through reports and analytics, while customers benefit from timely communication, accurate billing, and better service experiences.

From a student perspective, the project provided hands-on exposure to software development, database management, problem-solving, and teamwork. It served as a practical application of theoretical knowledge, bridging the gap between classroom learning and real-world implementation.

In conclusion, the Garage Management System not only addresses the operational challenges of garages but also contributes to customer satisfaction and business growth, making it a valuable solution for the automobile service industry.

Project Achievements :

The development of the Garage Management System led to several key achievements:

1. Successful System Development

- Designed and implemented a fully functional software solution to manage garage operations efficiently.
- Integrated modules for customer records, vehicle details, job scheduling, billing, and inventory control.

2. Automation of Manual Processes

- Reduced reliance on paper-based systems by digitalizing job cards, invoices, and service histories.

- Automated repetitive tasks such as stock tracking, billing, and service reminders.

3. Improved User Experience

- Created an intuitive interface for easy navigation by both administrators and garage staff.
- Enabled faster service processing, leading to improved customer satisfaction.

4. Data Management & Reporting

- Established a centralized database for secure and efficient storage of customer and garage information.
- Implemented reporting features to monitor revenue, job completion, and inventory usage.

5. Practical Learning for Students

- Strengthened technical skills in software design, programming, and database management.
- Developed teamwork, project management, and problem-solving abilities through collaborative effort.
- Gained real-world insight into how IT solutions can enhance business operations.

Student Learning Outcomes :

By working on the Garage Management System project, students achieved the following learning outcomes:

1. Knowledge Application

- Applied software engineering principles, database concepts, and programming techniques to solve a real-world problem.
- Understood how theoretical concepts from coursework translate into practical system development.

2. Technical Proficiency

- Gained hands-on experience in designing databases, developing user interfaces, and integrating system modules.
- Learned to implement features such as job scheduling, inventory tracking, billing, and reporting.

3. Problem-Solving & Critical Thinking

- Analyzed existing manual processes in garages and identified areas for improvement.
- Designed innovative solutions to optimize efficiency, reduce errors, and enhance customer service.

4. Collaboration & Project Management

- Practiced teamwork through task distribution, communication, and coordination during development.
- Applied project planning, documentation, and version control tools to manage progress effectively.

5. Communication & Professional Skills

- Enhanced skills in preparing structured project documentation and reports.
- Developed the ability to present technical information clearly to both technical and non-technical audiences.

6. Lifelong Learning & Adaptability

- Gained exposure to real-world challenges, encouraging adaptability and continuous learning.
- Understood the role of IT in transforming traditional businesses into efficient, technology-driven enterprises.

Future Scope :

The Garage Management System provides a strong foundation for automating garage operations, but it also has significant potential for future enhancements and scalability. Some possible future improvements include:

1. Mobile Application Integration

- Develop Android/iOS apps for customers and staff to enable real-time service updates, online booking, and job tracking.

2. Online Booking & Payment System

- Allow customers to schedule appointments through a web portal or mobile app.
- Integrate secure online payment gateways for advance or full payments.

3. IoT & Smart Vehicle Integration

- Connect with IoT-enabled vehicles to automatically detect and report service needs.

- Enable predictive maintenance by analyzing vehicle sensor data.

4. Customer Relationship Management (CRM)

- Implement loyalty programs, discounts, and personalized offers to retain customers.
- Introduce automated SMS/email reminders for service due dates and promotional offers.

5. Advanced Analytics & AI

- Use AI to forecast demand for spare parts and optimize inventory management.
- Provide data-driven insights to improve decision-making and business growth.

6. Cloud Deployment

- Migrate the system to the cloud for better scalability, accessibility, and data security.
- Enable multiple garage branches to access a centralized system.

7. Multi-language & Regional Support

- Offer support for different languages and regional formats to cater to a wider customer base.

8. Integration with Third-Party Services

- Connect with insurance companies for faster claim processing.
- Link with suppliers for automated spare part reordering.