

Salesforce Virtual Internship Program (SmartInternz)

**A CRM APPLICATION FOR**  
**GARAGE MANAGEMENT SYSTEM**

NAME: UDATA SRI SUSHMA

E-MAIL: [322103383063@gvpce.ac.in](mailto:322103383063@gvpce.ac.in)

College Name: Gayatri Vidya Parishad College Of  
Engineering, Autonomous.

# Project Title:- GARAGE MANAGEMENT

## 1. Project Overview

This project is focused on Garage Management System, designed to streamline and automate operations of an automotive repair shop. The goal is to deliver a comprehensive solution by leveraging a variety of Salesforce tools like Salesforce CRM, Service cloud, flows, etc. Through this project, we aim to build a scalable, efficient, and customer-friendly Garage Management workflow. This approach enhances the operational efficiency, customer satisfaction and data-driven decision-making.

## 2. Objectives

**Business Goals:** To establish a good customer-relationship management(CRM). To provide personalized service by tracking customer preferences and details(name, service records, billing details).

**Appointment Scheduling:** To allow customers to book service appointments online or via Salesforce portals, without conflicts(deadlocks) and improve resource allocation.

**Vehicle Service Tracking:** To maintain detailed records of each vehicle's service history. Concurrently, to update customers about upcoming maintenance needs.

**Billing and Invoicing:** To generate invoices and track payments for seamless transactions.

### 3. Salesforce Key Features and Concepts Utilized

The key features and concepts in this project include:

#### ❖ **Salesforce CRM:**

- Manage customer and vehicle records.
- Use contact and account objects to track customers and businesses.

#### ❖ **Service Cloud:**

- Use case management to handle repair requests and track service tickets.
- Implement knowledge articles for common repair solutions.

#### ❖ **Experience Cloud (Customer Portal):**

- Create a portal where customers can book appointments and view their service history.

#### ❖ **Flow Builder:**

- Automate processes like sending appointment reminders, updating service records, or notifying technicians about job assignments.

#### ❖ **Reports and Dashboards:**

- Generate insights into service trends, customer retention, and garage performance.
- Monitor inventory levels and technician productivity.

#### ❖ **Salesforce App Builder:**

- Build custom pages for managing garage operations like service tracking and spare parts inventory.

## 4. Detailed Steps to Solution Design

### a) Create A Developer Account:

- Login to any browser and enter in <https://developer.salesforce.com/signup>.
- Give in details like Full name, email id, username, college name, state etc, and an email confirmation pops up as shown in fig 4.0.

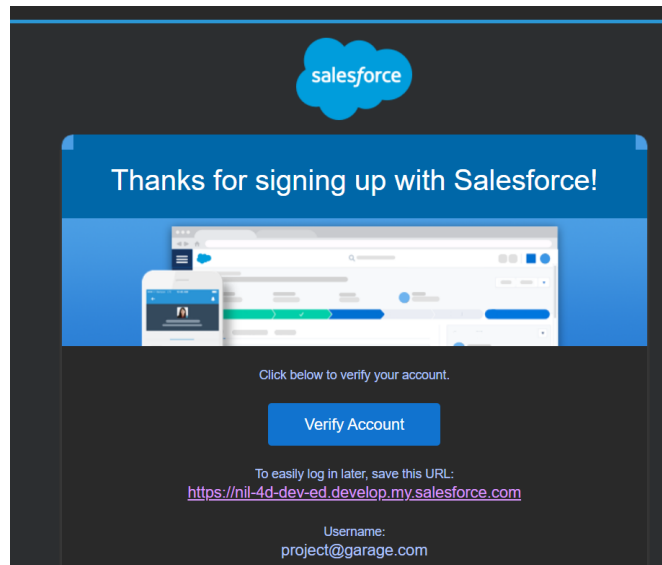


Fig 4.0

- Proceed with verify account and now, a developer org has been created.

### b) Create Objects:

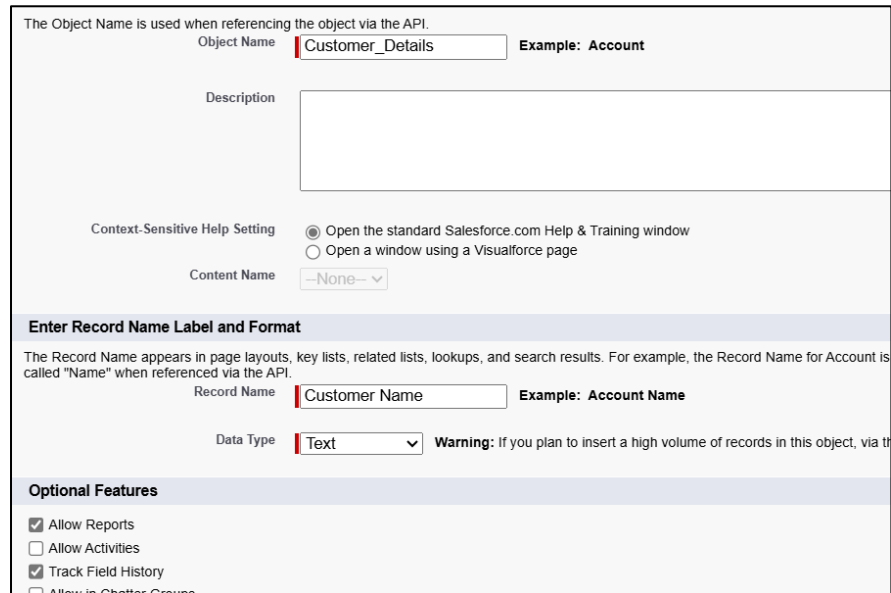
Salesforce objects are database tables that are used to store data. Since the objects are being created by users, they are called 'Custom Objects'.

The objects being created here are namely---

- ❖ Customer details
- ❖ Appointment
- ❖ Service Records
- ❖ Billing Details and Feedback

- In the setup page, click on 'Object Manager' > Create > Custom Object.

- Give in Label names as the respective object names.
- Ensure to allow reports and to track field history(checkboxes),as in fig 4.1.



The Object Name is used when referencing the object via the API.

Object Name:  Example: Account

Description:

Context-Sensitive Help Setting: ☒ Open the standard Salesforce.com Help & Training window  
☐ Open a window using a Visualforce page

Content Name:

**Enter Record Name Label and Format**

The Record Name appears in page layouts, key lists, related lists, lookups, and search results. For example, the Record Name for Account is called "Name" when referenced via the API.

Record Name:  Example: Account Name

Data Type:  Warning: If you plan to insert a high volume of records in this object, via th

**Optional Features**

☒ Allow Reports  
☐ Allow Activities  
☒ Track Field History  
☐ Allow in Chatter Groups

Fig 4.1

- Repeat the same process so as to create objects for other object names(Appointment, Service Records, Billing Details and Feedback).

### c) Create Tabs:

A tab is like a user interface that is used to build records for objects. Custom object tabs are the user interface for custom applications that are built in salesforce.com.

- In the setup page > Tabs > 'New' under Custom Object tab.
- Select the respective object names previously created and choose a tab style.
- Ensure the append tab to users' existing personal customization is checked and save.

The tab icons are now selected for the objects shown in fig 4.2.

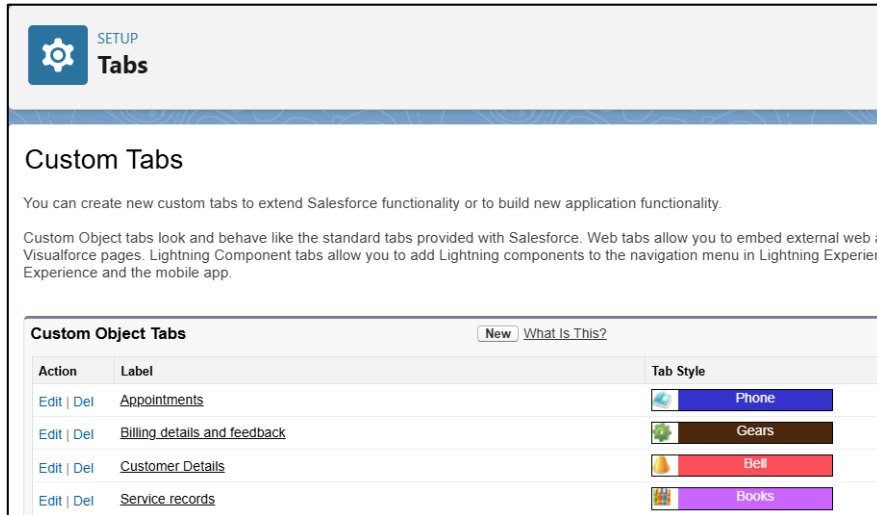


Fig 4.2

#### d) Create A Lightning App:

An app is a collection of items that work together to serve a particular function. In Lightning Experience, Lightning apps provides users with access to sets of objects, tabs, and other items in the navigation bar.

- Go to Setup > App Manager > New.
- The app name is given as “Garage Management Application”, along with a logo image.
- Navigation items are added which are the four custom objects and ‘System Administrator’ profile is also added.

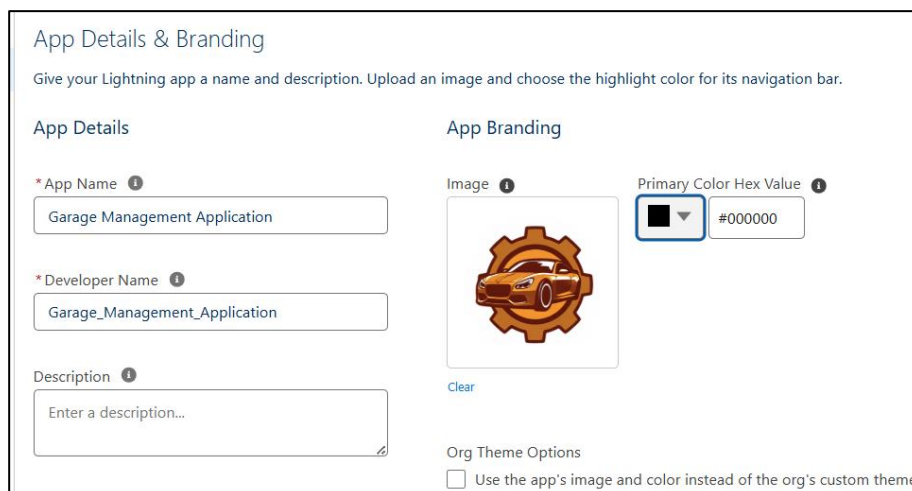


Fig 4.3

## e) Creation Of Fields:

Fields represent the data stored in the columns of a relational database. It can also hold any valuable information that are required for a specific object. Hence, the overall searching, deletion, and editing of the records become simpler and quicker.

--In Customer Details object:

- On navigating to the said custom object, click 'Fields & Relationships' > New.
- The data type would be 'Phone' with the label name being "Phone number".
- Click on Next > Save.
- Perform the similar operations to create another field-"Gmail". The overall fields for this object is displayed in fig 4.4.

SETUP > OBJECT MANAGER

**Customer Details**

Details	<b>Fields &amp; Relationships</b> 6 Items, Sorted by Field Label		
<b>Fields &amp; Relationships</b>	<b>FIELD LABEL</b>	<b>FIELD NAME</b>	<b>DATA TYPE</b>
Page Layouts	Created By	CreatedById	Lookup(User)
Lightning Record Pages	Customer Name	Name	Text(80)
Buttons, Links, and Actions	Gmail	Gmail__c	Email
Compact Layouts	Last Modified By	LastModifiedById	Lookup(User)
Field Sets	Owner	OwnerId	Lookup(User,Group)
Object Limits	Phone number	Phone_number__c	Phone
Record Types			

Fig 4.4

--Date fields:

- Create a field label with date as data type and label as "Appointment Date".
- Make it a required field.

--Lookup fields:

- Navigate to "Appointment" object > Fields & Relationship> New.
- Give in the data type as 'Look up relationship' ,and the object it relates to is Customer Details.

- Repeat the same procedure in the “Service records” object, with the related object it looks up to being the Appointment object.
- In the filter criteria, Appointment: Appointment Date -> less than-> Appointment: Created Date.

*--Check box fields:*

- Navigate to Appointment object > Fields & Relationships > New.
- Give in data type as ‘checkbox’ and label name as ‘Maintenance Service’.
- Create two fields for Appointment object with field label as “Repairs” and “Replacement Parts”.
- Create a field in Service records object with “Quality check status” as field label. Click Next> Save.

*--Currency fields:*

- Go to Appointment object and create a field “Service Amount” with currency data type.
- Give read only for all profiles in field level security for profiles.
- Perform the same in Billing details and Feedback object.

*--Text fields:*

- In the Appointment object, with the type of data as ‘text’ create a label named “Vehicle number plate”.
- Its length is entered as 10.

*--Formula field:*

- On navigating to Service records object, check mark the data type as formula and give in the label name as “service date”.
- Select formula return type as date. Insert labels as “Service records-> Created date”. Check syntax and save the field

f) 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.

- Go to Object Manager. Click on drop down in the Appointment object to select 'edit'.
- Click on Validation Rule and enter rule name as "Vehicle".
- Insert error condition formula as "NOT(REGEX( Vehicle\_number\_plate\_\_c , "[A-Z]{2}[0-9]{2}[A-Z]{2}[0-9]{4}"))", error location as Vehicle no plate.
- Errors(wrong license numbers) also need to be handled, therefore type in the error message as "please enter a valid license number".

### Appointment Validation Rule

[Back to Appointment](#)

**Validation Rule Detail**

Edit Clone

Rule Name	Vehicle	Active	✓
Error Condition Formula	NOT(REGEX( Vehicle_number_plate__c , "[A-Z]{2}[0-9]{2}[A-Z]{2}[0-9]{4}"))		
Error Message	Please enter a valid number	Error Location	Vehicle number plate
Description	vehicle		
Created By	Sri Sushma, 16/01/2025, 12:29 pm	Modified By	Sri Sushma, 16/01/2025, 12:29 pm

Edit Clone

Fig 4.5

- Repeat the same in Billing and details object, with "rating\_should\_be\_less\_than\_5" as rule name.
- Insert error condition as "NOT( REGEX( Rating\_for\_service\_\_c , "[1-5]{1}"))" and error message to be displayed as "rating should be from 1 to 5".

#### g) Duplicate Rule:

Search for 'matching rule' in quick find box, and click on new rule.

Select the Customer Details object and rule name as "matching customer details".

Matching Rule  
Matching customer details

Matching Rule Detail Delete Clone Deactivate

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	Active

Fig 4.6

#### h) 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. Profiles can be defined by the user's job function.

For example- System Administrator, Developer, Sales Representative.

- Type in profiles in quick find box. Clone the profile ‘Standard user’ in the profile named as ‘Manager’.
- Edit the profile page. Custom app settings must be check marked for the Garage Management application.
- Give access permissions to all the four custom objects.
- Change the session times out after ‘8 hours of inactivity’.
- User passwords should never be expired. Enter the minimum password length as “8”, shown in the picture below.

**Session Settings**

Session Times Out After 8 hours of inactivity i

**Password Policies**

User passwords expire in Never expires

Enforce password history 3 passwords remembered

Minimum password length 8

Password complexity requirement Must include alpha and numeric characters

Password question requirement Cannot contain password

Maximum invalid login attempts 10

Lockout effective period 15 minutes

Obscure secret answer for password resets ☐

Require a minimum 1 day password lifetime ☐

Don't immediately expire links in forgot password emails ☐ i

Save Save & New Cancel

- Again, clone the profile 'Salesforce Platform User' in 'salesforce'.
- Repeat the same processes until accessing permissions to the custom objects.

#### i) 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 an Salesforce organization can have to data. Simply put, it describes what a user could see within the Salesforce organization.

---Creating a Manager & sales person role:

In the quick-find box, search for "roles" in order to set up roles.

Click on expand all and add a role named "Manager" under CEO.

Add another role named "sales person" under Manager. The hierarchy of roles are displayed in a tree format in fig 4.7.

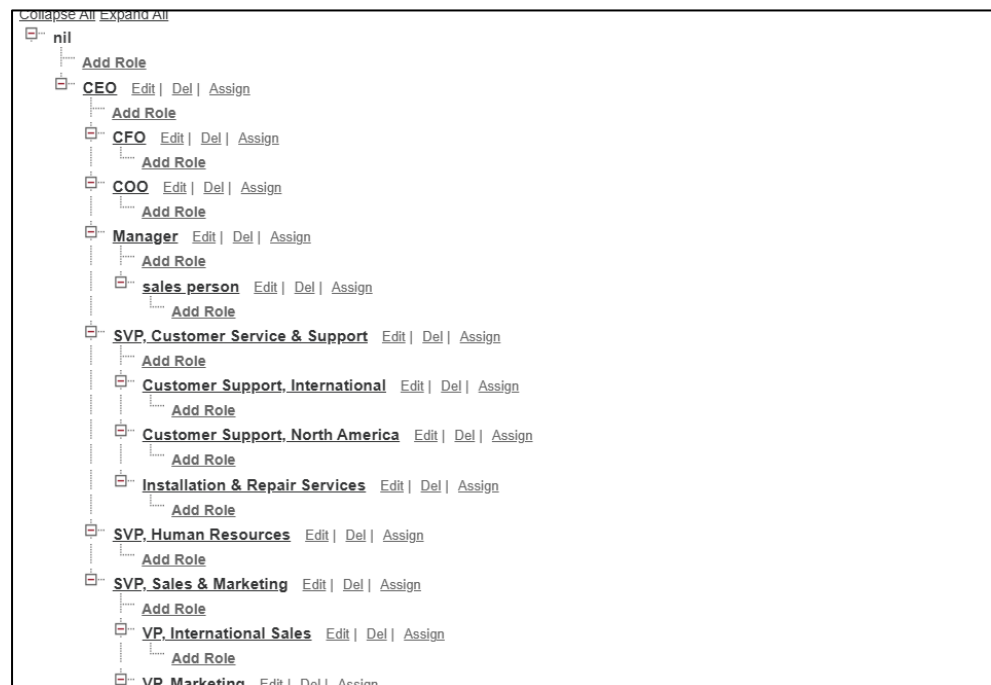


Fig 4.7

#### j) Users:

A user is anyone who logs in to Salesforce. Users are employees at a 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.

- Type in Users in quick find box, to create a new user.
- Details are given into first name, last name, email id ,username, nick name, role, user license and profiles. The details entered here belong to the manager named “Mikaelson Niklaus”.
- Similarly, personal details are given into the fields that belong to the various sales person. The sales person created here are named “user1”, “user2” and “user3”.

**All Users**

On this page you can create, view, and manage users.

To get more licenses, use the Your Account app. [Let's Go](#)

View: All Users Edit Create New View

A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V

Action	Full Name	Alias	Username	Role	Active	Profile
<input type="checkbox"/> <a href="#">Edit</a>	1_user	u1	project@user1.com	sales person	✓	sales person
<input type="checkbox"/> <a href="#">Edit</a>	2_user	u2	project@user2.com	sales person	✓	sales person
<input type="checkbox"/> <a href="#">Edit</a>	3_user	u3	project@user3.com	sales person	✓	sales person
<input type="checkbox"/> <a href="#">Edit</a>	Chatter Expert	Chatter	chatty.00dns00000aa4iv2ad.13rmtb2cerv8@chatter.salesforce.com		✓	Chatter Free User
<input type="checkbox"/> <a href="#">Edit</a>	Mikaelson, Niklaus	nmika	projectmike@user.com	Manager	✓	Manager

Fig 4.8

#### k) 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, appropriate access to appropriate users can be ensured.

- Navigate to users > public groups > new.
- Enter label as “sales team”.
- Select “sales person” and move it to the selected member. Here, the public group is created to all the sales person in the team.
- A public group likewise is created in fig 4.81.

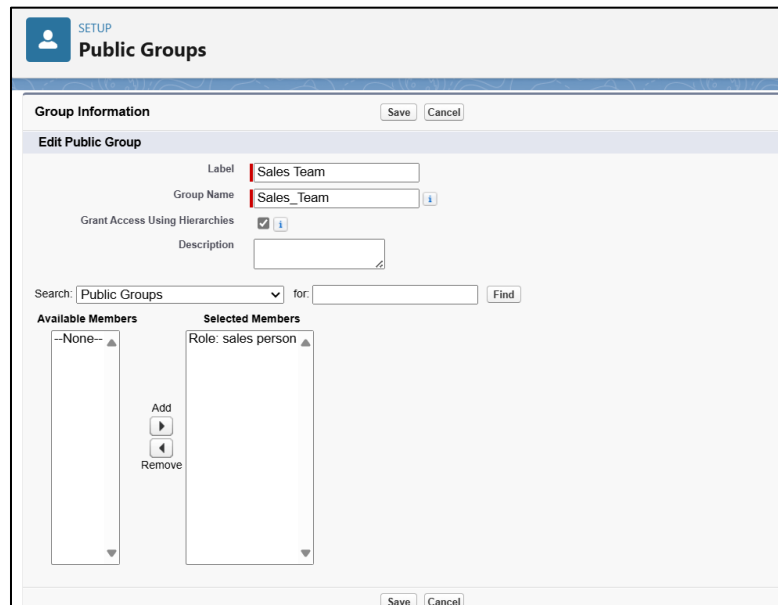


Fig 4.81

## I) Sharing Settings:

Salesforce allows to configure sharing settings to control how records are accessed and shared within an organisation. Such settings are crucial for maintaining data security and privacy. Salesforce provides such tools and mechanisms provided below-

*Organisation-wide-default(OWD) settings:* These settings define the default level of access for all objects with the salesforce org.

*Role Hierarchy:* Users at higher level have greater access to records owned by or shared with users lower in hierarchy.

*Profiles And Permission Settings:* Allows administrators to specify object-level and field-level permissions for users.

*Sharing users:* They 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.

- Type in users to go to “Sharing settings” to edit them.
- Change the OWD settings of the Service records object as “private”.

## j) Flows:

A flow is a powerful tool that allows 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.

--Create a flow:

- Type in “flow” in quick find box and click on new.
- Select the Record-triggered flow, and select Billing details and feedback object.
- Select the trigger flow when ‘A record is created or updated’.
- Under the record-triggered flow, click on “+” and choose ‘update records element’.
- Give in the label name as “amount update” with all conditions being met(and).
- Enter field name as “Payment\_Status\_\_c”-> equals-> Completed.
- In the billing details object, give in “Payment\_Paid\_\_c”-> `{!$Record.Service_records__r.Appointment__r.Service_Amount__c}`
- Now click on “new resource” and select Variable.
- Select api name as “alert” and change the view to plain text, to type in:

“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.”.

A flow named “Billing Amount Flow” has been created as shown in fig 4.9.

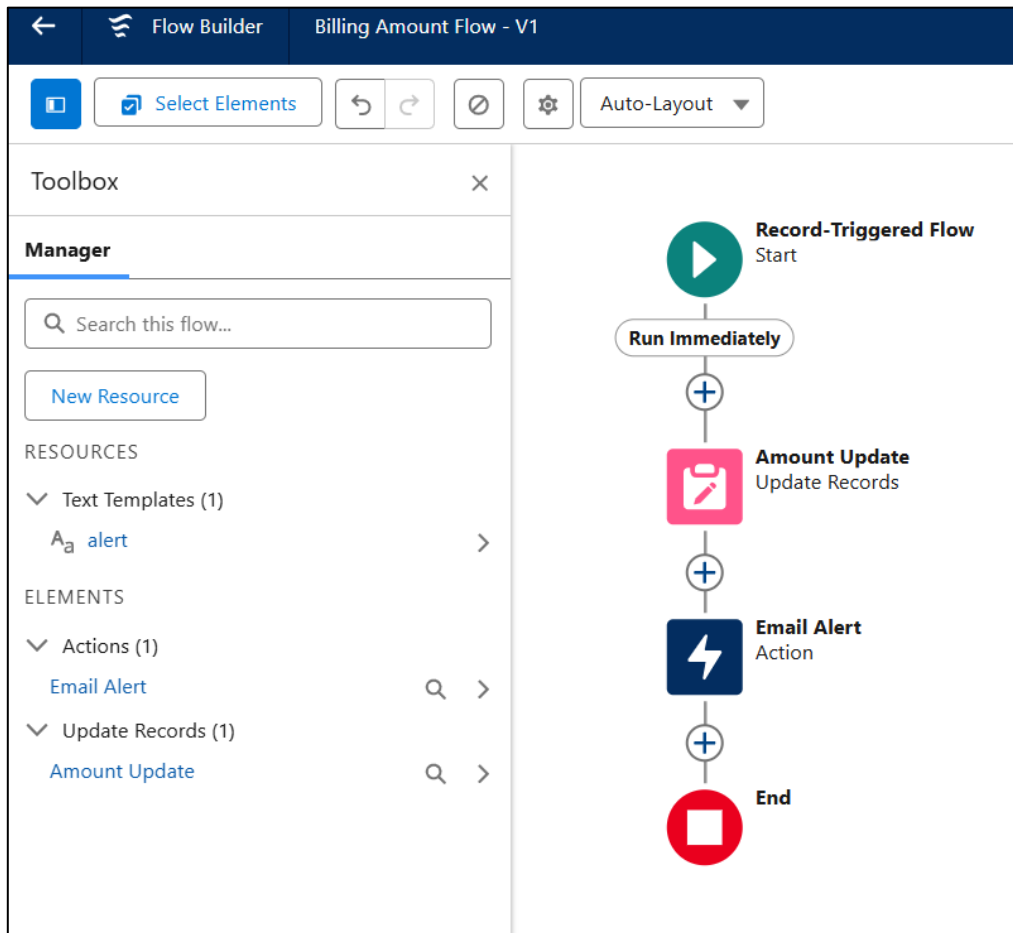


Fig 4.9

#### k) Apex Trigger:

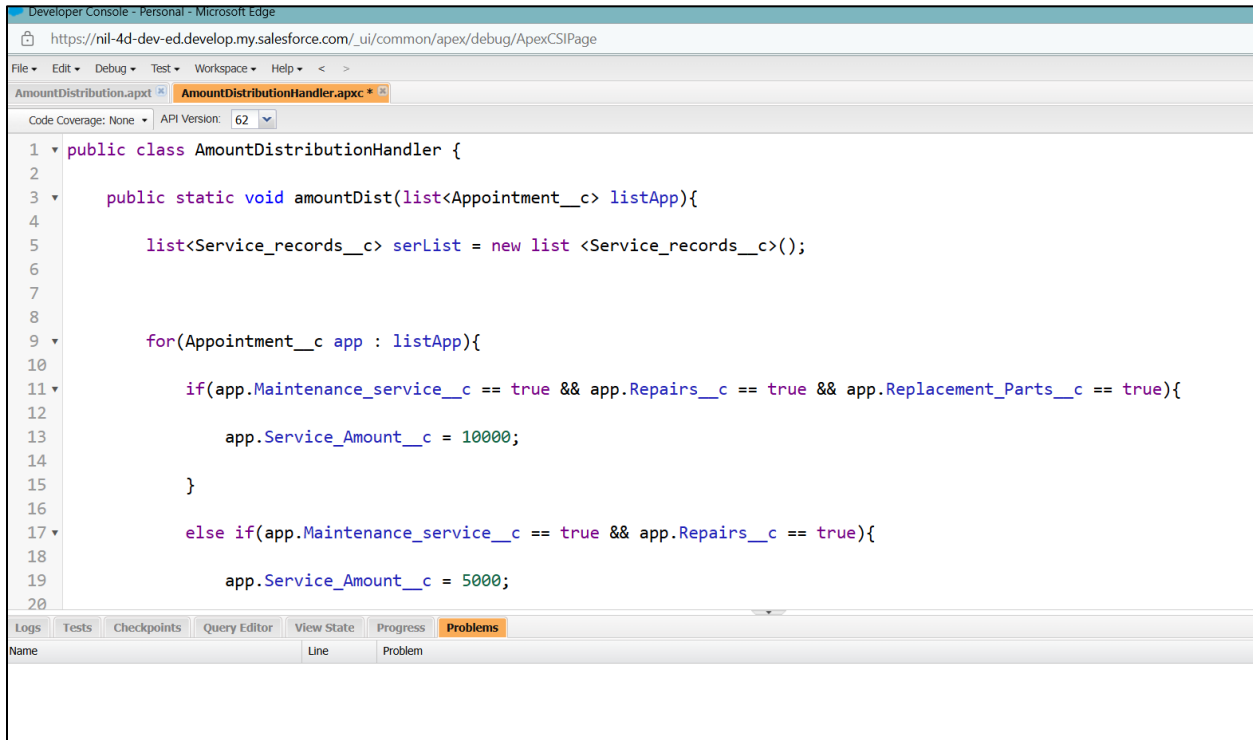
Apex can be invoked by using triggers. They enable to perform custom actions before or after changes in Salesforce records. It executes before or after insert/update/delete/merge/upsert/undelete operations.

There are primarily two types of triggers:

**Before triggers-** This is used to either update or validate the values of a record before they can be saved into the database. To put it simply, the before trigger validates the record first and then saves it, with some criteria or code set.

**After triggers-** This is used to access the field values set by the system and affect any change in the record. The after trigger makes changes to the value from the data inserted in some other record.

- Navigate to Developer Console.
- In the toolbar, click on file to create a new apex class and apex trigger.
- A new apex class named “AmountDistributionHandler.apxc” and a trigger named “Amount Distribution.apxt”.



```

1 public class AmountDistributionHandler {
2
3     public static void amountDist(list<Appointment__c> listApp){
4
5         list<Service_records__c> serList = new list<Service_records__c>();
6
7
8
9         for(Appointment__c app : listApp){
10
11             if(app.Maintenance_service__c == true && app.Repairs__c == true && app.Replacement_Parts__c == true){
12
13                 app.Service_Amount__c = 10000;
14
15             }
16
17             else if(app.Maintenance_service__c == true && app.Repairs__c == true){
18
19                 app.Service_Amount__c = 5000;
20

```

Fig 4.10

### 1) Reports:

Reports provide access to salesforce data. Types of reports are of Tabular, Summary, Matrix and Joined reports.

- Click on “app launcher” and search for reports.
- Enter the folder name as “garage management”.
- To share the report folder, select the share with as “roles” -> “manager”.
- Click on share and done.



- To create a new custom report type, select the primary object as “Customer Details” and report type as “service information”. Select the deployed checkbox.
- Click on select object and choose Appointment. Give in ‘Customer Details’ object as primary object.
- The secondary objects having look up relationship would be ”Appointments” > “Service records” > “Billing details and feedback” object ,particularly in that order. The object relations are displayed in fig 4.11.

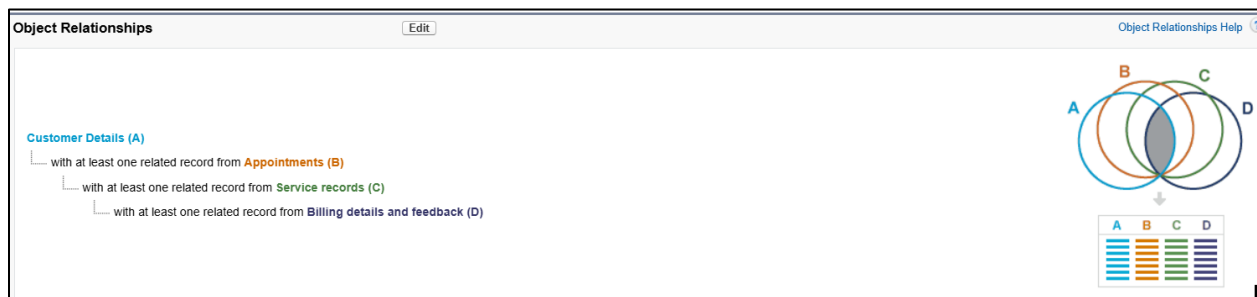


Fig 4.11

- Navigate to app launcher and search for “reports”.
- Select the group rows as “rating for service” and group columns as “payment status”, and click on run the chart.
- Save the report as “New Service Information Report”.

#### j) Dashboards:

Dashboards help in visually understanding the business conditions thereby helping in data-driven decision making.

- In app launcher > dashboard.
- Click new folder with label “Service Rating Dashboard”.
- Go to app > click on “dashboards”.
- Give in the name as “Customer Review” and click on create.

## k) Creating Records:

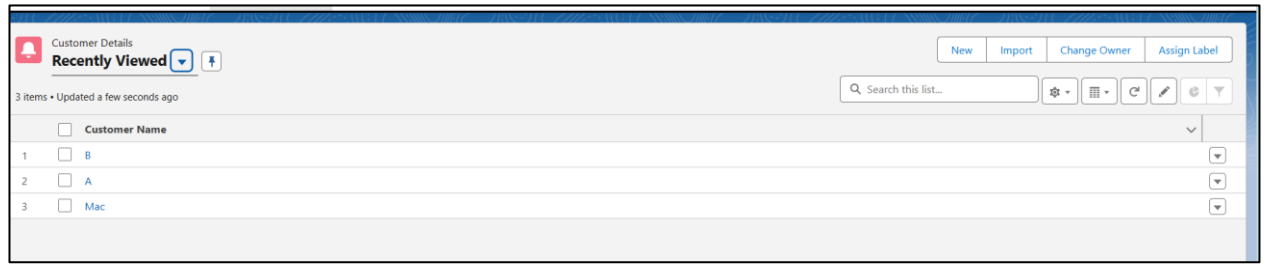


Fig 4.12

As shown in fig 4.12, Records are correspondently created in the custom objects namely-Customer Details, Appointment, Service Records and Billing details and Feedback.

## 5. Testing and Validation

Testing and validation are critical steps in the development and deployment of the Garage Management Application. These processes ensure that the application functions as intended, meets business requirements, and delivers a seamless user experience.

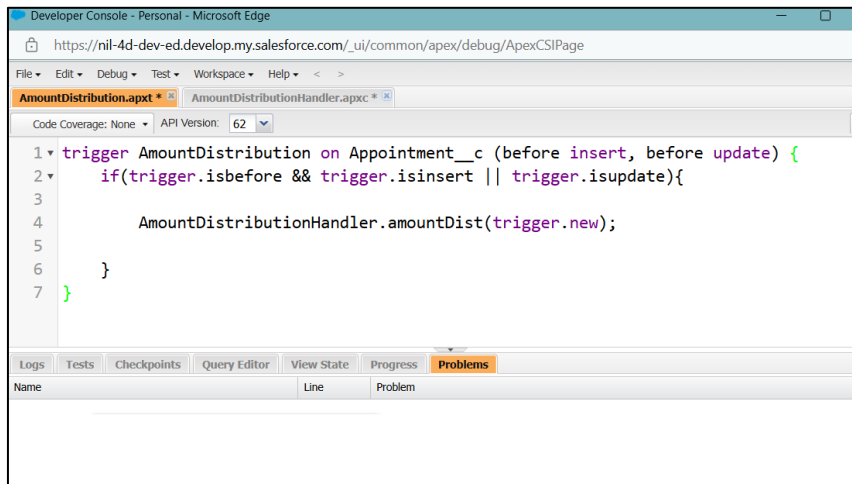
- i. **Plan:** Define test cases and validation criteria based on business requirements. Identify key users for User Acceptance Testing.
- ii. **Execute:** Perform tests in the Salesforce sandbox environment. Document test results and track issues.
- iii. **Fix:** Resolve identified bugs and re-test affected areas.
- iv. **Validate:** Ensure all requirements are met and workflows are optimized.
- v. **Deploy:** Move the validated application to the production environment.

- Unit Testing (Apex Classes, Triggers):

Unit testing involves writing test methods to verify that individual pieces of Apex code (classes and triggers) behave as expected. These tests are executed in an isolated environment using Salesforce's testing framework.

Tested the AmountDistribution trigger to ensure:

1. **Correct Functionality:** The trigger behaves as expected for all data operations (insert, update, delete, etc.).
2. **Bulk Data Handling:** The trigger can process multiple records efficiently without hitting governor limits.
3. **Error-Free Code:** Catch potential issues, such as infinite loops or SOQL limits.



The screenshot shows the Salesforce Developer Console interface. The top bar indicates the URL: `https://nil-4d-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage`. Below the menu bar, the file explorer shows `AmountDistribution.apxt` and `AmountDistributionHandler.apxc`. The code editor displays the following Apex trigger code:

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

At the bottom, the 'Problems' tab is selected, showing a table with columns 'Name', 'Line', and 'Problem'. The table is currently empty.

- User Interface Testing.

UI testing is the process of verifying that all graphical elements, such as screens, buttons, forms, and menus, behave as expected. It ensures that users can efficiently interact with the application and perform their tasks without encountering issues.

**Navigation Testing:** Smoothness and intuitive navigation between pages like bookings, inventory, and customer records is ensured.

**Form Testing:** Validation of input fields, mandatory field checks, and error messages for forms like vehicle details has been tested. This is done to prevent record creation when the form fields remained blank.

**Layout & Responsiveness Testing:** Tested for consistent layouts and responsiveness across devices (desktop, tablet, mobile), for easy user access.

**Functionality Testing:** Verified buttons, search features, filters, and end-to-end workflows like service booking or billing.

## 6. Key Scenarios Addressed by Salesforce in the Implementation Project

The implementation of the **Garage Management Project** in Salesforce addresses several key issues that businesses in the automotive repair and service industry commonly face. Below are the primary challenges and how Salesforce tackles them:

### 1. Inefficient Customer Management

Garages struggle to maintain accurate customer records and vehicle histories, leading to poor customer experience.

#### **Salesforce Solution:**

- Centralized **CRM**: Salesforce stores all customer and vehicle details (contact

information, service history, preferences) in one place, ensuring quick access and personalized service.

- **Automated reminders:** Customers receive notifications about upcoming services, enhancing engagement and retention.

## 2. Lack of Appointment Scheduling System

Manual appointment scheduling leads to double bookings, missed slots, and poor resource utilization.

### **Salesforce Solution:**

- **Flow Automation:** Salesforce allows customers to book appointments online via an Experience Cloud portal.
- **Resource Scheduler:** Automatically assigns technicians and time slots, ensuring efficiency.

## 3. Lack of Operational Efficiency

Manual processes like task assignment, invoicing, and job tracking slow down operations.

### **Salesforce Solution:**

- **Automation:** Use Salesforce Flow for task assignment, job tracking, and automated billing.
- **Service Console:** Provides a unified interface to manage service tickets, technician schedules, and customer communication.

## 4. Poor Customer Communication

Garages fail to keep customers updated on service progress, leading to dissatisfaction.

### **Salesforce Solution:**

- **Automated Notifications:** Use email, SMS, or push notifications to inform customers about appointment reminders, service progress, and job completion.
- **Experience Cloud:** Customers can log in to check service updates and history.

## 5. Scalability Challenges

Traditional systems struggle to scale with the growth of the garage business (e.g., more locations, customers, or services).

### **Salesforce Solution:**

- **Cloud-Based Platform:** Salesforce allows easy scaling of operations, whether adding new locations or integrating more functionality.
- **Customization:** Adapt workflows, objects, and processes to meet new business needs.

## 7. Conclusion

The implementation of the **Garage Management System** in Salesforce has revolutionized operations by streamlining customer and vehicle management, automating appointment scheduling, and enhancing inventory tracking. Real-time

notifications and a self-service portal have improved customer engagement. The system's scalability, mobile accessibility, and robust data security ensure long-term growth and compliance.

Overall, this project has transformed the garage into a modern, efficient, and customer-centric business, setting the stage for sustained success.