

# GARAGE MANAGEMENT SYSTEM

## Introduction

### Salesforce

#### Introduction:

Are you new to Salesforce? Not sure exactly what it is, or how to use it? Don't know where you should start on your learning journey? If you've answered yes to any of these questions, then you're in the right place. This module is for you.

Welcome to Salesforce! Salesforce is game-changing technology, with a host of productivityboosting features, that will help you sell smarter and faster. As you work toward your badge for this module, we'll take you through these features and answer the question, "What is Salesforce, anyway?".

**What Is Salesforce?** Salesforce is your customer success platform, designed to help you sell, service, market, analyze, and connect with your customers.

Salesforce has everything you need to run your business from anywhere. Using standard products and features, you can manage relationships with prospects and customers, collaborate and engage with employees and partners, and store your data securely in the cloud.

So what does that really mean? Well, before Salesforce, your contacts, emails, follow-up tasks, and prospective deals might have been organized something like this: <https://youtu.be/r9EX3IGde5k>

#### Creating Developer Account:

##### Creating a developer org in salesforce.

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

2. 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.

.

The Garage Management System (GMS) leverages Salesforce to manage customer details, vehicle information, service records, billing, and employee tracking within a single environment. By using Salesforce, the system ensures secure data storage, easy reporting, and smooth communication between customers and the garage team.

**Build enterprise-quality apps fast to bring your ideas to life**

- Build apps fast with drag and drop tools
- Customize your data model with clicks
- Go further with Apex code
- Integrate with anything using powerful APIs
- Stay protected with enterprise-grade security
- Customize UI with clicks or any leading-edge web framework

**Sign up for your Salesforce Developer Edition**  
A Salesforce Platform environment for free.

Complete the form to get access to the Salesforce Developer Edition.

First Name*	Last Name*
Naga Anjali	Moturi
Email*	
22505a0510@pvpsit.AC.IN	
Role*	Developer
Company*	Prasad V Poturi Siddhartha Institute Of Technology
Country/Region*	India
State/Province*	Andhra Pradesh
Postal Code*	520011
Username*	anjali@pvpsit.in

Your username must be in the form of an email address (It does not have to be real). It must be unique and cannot be associated with another Salesforce login credential. [Read more about username recommendations.](#)

[Learn more about the Main Services Agreement – Developer Services and](#)

## Account Activation

Go to the inbox of the email that you used while signing up. Click on the verify account to activate your account.

Welcome to Salesforce: Verify your account [External](#) [Inbox](#)

developer@salesforce.com <developer@salesforce.com>  
to me. ▾

11:34 (3 hours ago) [star](#) [forward](#) [more](#)

**Thanks for signing up with Salesforce!**

Click below to verify your account.

[Verify Account](#)

To easily log in later, save this URL:  
<https://prasadvpoturisiddharth-1de-dev-ed.develop.my.salesforce.com>

Username:  
anjali@pvpsit.ac.in

Again, welcome to Salesforce!

## OBJECT

## What Is an Object?

### Objectives

The primary objective of the Garage Management System is to digitalize and automate the operations of a vehicle service center.

This project aims to:

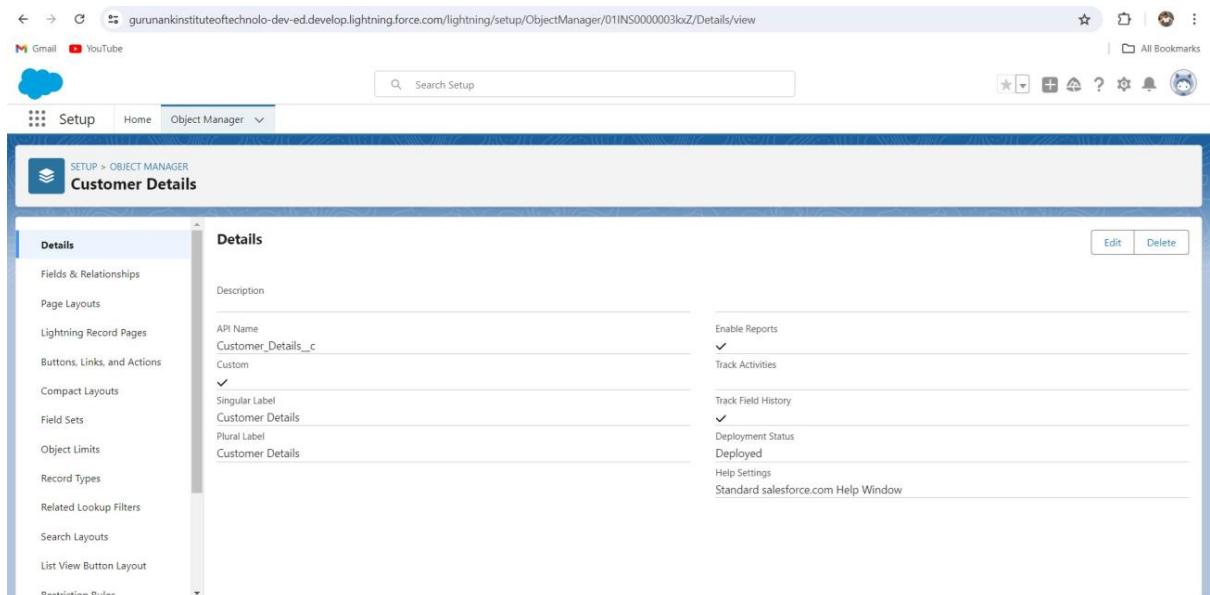
- Maintain a centralized database for customers, vehicles, and service records.
- Reduce manual errors in service tracking and billing.
- Enable service staff to manage appointments and inventory efficiently.
- Provide transparency to customers through timely updates and notifications.
- Generate reports for management to analyze performance and sales trends.

Overall, the project intends to improve operational efficiency, customer satisfaction, and decision-making through Salesforce automation tools.

- 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.
- **Create Customer Details Object**

**i)To create an object:** From the setup page >> Click on Object Manager >> Click on Create >> Click on Custom Object.

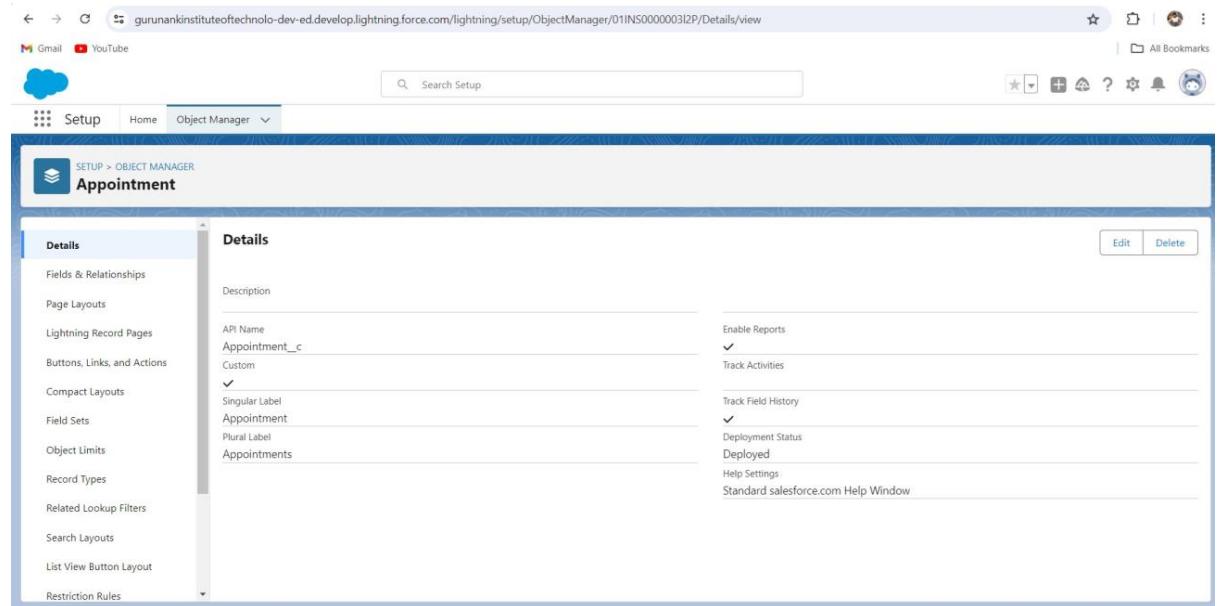
- Enter the label name >> Customer Details
- Plural label name >> Customer Details
- Enter Record Name Label and Format
- Record Name >> Customer Name
- Data Type >> Text
- Click on Allow reports and Track Field History,
- Allow search >> Save.



## **ii)Create Appointment Object**

- **To create an object:** From the setup page >> Click on Object Manager >> Click on Create >> Click on Custom Object.
- Enter the label name >> Appointment
- Plural label name >> Appointments
- Enter Record Name Label and Format
- Record Name >> Appointment Name

- Data Type >> Auto Number
- Display Format >> app-{000}
- Starting number >> 1
- Click on Allow reports and Track Field History,
- Allow search >> Save.



### iii) Create Service records Object

- **To create an object:** From the setup page >> Click on Object Manager >> Click on Create >> Click on Custom Object.
- Enter the label name >> Service records
- Plural label name >> Service records
- Enter Record Name Label and Format
- Record Name >> Service records Name
- Data Type >> Auto Number
- Display Format >> ser-{000}
- Starting number >> 1
- Click on Allow reports and Track Field History,

- Allow search >> Save.

Service records Field  
**Service records Name**  
[Back to Service records](#)

[Set Field-Level Security](#) [View Field Accessibility](#)

Field Information		Field Name	Name
Field Label	Service records Name		
Data Type	Auto Number		
Description			
Data Owner			
Field Usage			
Data Sensitivity Level			
Compliance Categorization			
Display Format	ser-{000}		

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

Billing details and feedback Field  
**Billing details and feedback Name**  
[Back to Billing details and feedback](#)

[Set Field-Level Security](#) [View Field Accessibility](#)

Field Information		Field Name	Name
Field Label	Billing details and feedback Name		
Data Type	Auto Number		
Description			
Data Owner			
Field Usage			
Data Sensitivity Level			
Compliance Categorization			
Display Format	bill-{000}		

- 
- **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.

The screenshot shows the Salesforce Setup interface under the 'Tabs' section. It includes three main sections: 'Custom Object Tabs', 'Web Tabs', and 'Visualforce Tabs'. The 'Custom Object Tabs' section lists four tabs with their labels and icons: 'Appointments' (Balls), 'Billing details and feedback' (Bell), 'Customer Details' (Apple), and 'Service records' (Jewel). The 'Web Tabs' and 'Visualforce Tabs' sections both indicate that no tabs have been defined.

Action	Label	Tab Style	Description
Edit   Del	Appointments	Balls	
Edit   Del	Billing details and feedback	Bell	
Edit   Del	Customer Details	Apple	
Edit   Del	Service records	Jewel	

## 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.

### Create a Lightning App

**To create a lightning app page:**

1. Go to setup page >> search “app manager” in quick nd >> select “app manager” >> click on New lightning App.
2. Fill the app name in app details as Garage Management Application >> Next >> (App option page) keep it as default >> Next >> (Utility Items) keep it as default >> Next.

3. To Add Navigation Items:
4. Select the items (Customer Details, Appointments, Service records, Billing details and feedback, Reports and Dashboards) from the search bar and move it using the arrow button >> Next.
5. To Add User Roles: Search roles (System administrator) in the search bar >> click on the arrow button >> save & Finish.

The screenshot shows the 'Lightning Experience App Manager' page in the Salesforce Setup. The left sidebar has 'App Manager' selected under 'Apps'. The main area displays a table of 24 installed apps, sorted by App Name. The columns include App Name, Developer Name, Description, Last Modified Date, Type, and Status. A specific row for 'Garage Management Application' is highlighted with a red box.

App Name	Developer Name	Description	Last Modified	Type	Status
3 App Launcher	AppLauncher	App Launcher tabs	19/11/2024, 2:27 pm	Classic	✓
4 Automation	FlowsApp	Automate business processes and repetitive tasks.	19/11/2024, 2:34 pm	Lightning	✓
5 Bolt Solutions	LightningBolt	Discover and manage business solutions designed...	19/11/2024, 2:31 pm	Lightning	✓
6 Business Rules Engine	ExpressionSetConsole	Create and maintain business rules that perform c...	19/11/2024, 2:27 pm	Lightning	✓
7 Community	Community	Salesforce CRM Communities	19/11/2024, 2:27 pm	Classic	✓
8 Content	Content	Salesforce CRM Content	19/11/2024, 2:27 pm	Classic	✓
9 Data Manager	DataManager	Use Data Manager to view limits, monitor usage, a...	19/11/2024, 2:27 pm	Lightning	✓
10 Digital Experiences	SalesforceCMS	Manage content and media for all of your sites.	19/11/2024, 2:27 pm	Lightning	✓
11 Garage Management Application	Garage_Management_Applicati...		19/11/2024, 3:46 pm	Lightning	✓
12 Lightning Usage App	LightningInstrumentation	View Adoption and Usage Metrics for Lightning Ex...	19/11/2024, 2:27 pm	Lightning	✓
13 Marketing CRM Classic	Marketing	Track sales and marketing efforts with CRM objects.	19/11/2024, 2:27 pm	Classic	✓

## 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

## Creation of fields for the Customer Details object

1. To create fields in an object:

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

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

2. To create another elds 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
  5. Click on Next >> Next >> Save and new.

Cloud icon

Setup Home Object Manager

SETUP > OBJECT MANAGER Customer Details

Details Fields & Relationships Page Layouts Lightning Record Pages Buttons, Links, and Actions Compact Layouts Field Sets Object Limits Record Types Related Lookup Filters Search Layouts List View Button Layout Restriction Rules

Fields & Relationships 6 items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
Customer Name	Name	Text(80)		✓
Gmail	Gmail__c	Email		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓
Phone number	Phone_number__c	Phone		

Quick Find New Deleted Fields Field Dependencies Set History Tracking

Cloud icon

Setup Home Object Manager

SETUP > OBJECT MANAGER Service records

Details Fields & Relationships Page Layouts Lightning Record Pages Buttons, Links, and Actions Compact Layouts Field Sets Object Limits Record Types Related Lookup Filters Search Layouts List View Button Layout Restriction Rules

Fields & Relationships 7 items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Appointment	Appointment__c	Lookup(Appointment)		✓
Created By	CreatedById	Lookup(User)		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓
service date	service_date__c	Formula (Date)		
Service records Name	Name	Auto Number		✓
Service Status	Service_Status__c	Picklist		

Quick Find New Deleted Fields Field Dependencies Set History Tracking

The screenshot shows the Salesforce Object Manager interface. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. A search bar at the top right contains the placeholder 'Search Setup'. To the right are icons for star, plus, document, question mark, gear, and bell. The main content area is titled 'Billing details and feedback' under 'FIELDS & RELATIONSHIPS'. It displays a table with 8 items, sorted by Field Label. The columns are 'FIELD LABEL', 'FIELD NAME', 'DATA TYPE', 'CONTROLLING FIELD', and 'INDEXED'. The data is as follows:

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Billing details and feedback Name	Name	Auto Number		✓
Created By	CreatedById	Lookup(User)		✓
Last Modified By	LastModifiedById	Lookup(User)		✓
Owner	OwnerId	Lookup(User,Group)		✓
Payment Paid	Payment_Paid__c	Currency(18, 0)		✓
Payment Status	Payment_Status__c	Picklist		✓
Rating for service	Rating_for_service__c	Text(2)		✓
Service records	Service_records__c	Lookup(Service records)		✓

## Ideation

### Originality of Ideas

The idea of integrating garage operations with Salesforce CRM is unique because it combines customer relationship management with real-time garage service tracking. Unlike traditional systems, this project uses Salesforce automation tools such as Process Builder, Flows, and Apex Triggers to provide an intelligent and automated workflow.

### Feasibility of Ideas

The project is technically feasible since Salesforce offers built-in objects, automation features, and cloud hosting. From a business standpoint, it requires minimal hardware investment and can scale for multiple branches. The system's modular design allows easy customization for small, medium, and large-scale garages.

## Requirement Analysis

### Completeness of Requirements

The Garage Management System requires both functional and non-functional specifications:

- **Functional Requirements:**

- Manage customer registration and vehicle details.
- Record service requests, assign mechanics, and track service status.
- Generate automated bills and invoices.
- Maintain service history and reports.

- **Non-Functional Requirements:**

- Secure and scalable data storage.
- High system availability through Salesforce Cloud.
- Easy user interface for staff and customers.

All requirements were collected through user interaction and workflow observation of existing garage processes.

## Project Design

### Design Completeness

The design includes key Salesforce objects such as Customer, Vehicle, Mechanic, Service Request, and Invoice. Relationships between these entities are represented using master-detail and lookup relationships.

## Innovation and Design

Custom Lightning Pages and Visualforce components are used to present data clearly. Automation via Apex and Flows minimizes human intervention. The UI is designed for mobile compatibility using Salesforce Lightning Design System (SLDS).

## User Experience Considerations

The interface is simple and intuitive, ensuring quick navigation and data entry. Each module is linked via dashboards and reports for real-time monitoring, enhancing user satisfaction.

## Project Development

### Code Quality

The code follows Salesforce best practices including the use of Apex Triggers with corresponding Handlers, proper naming conventions, and bulkification to handle multiple records efficiently.

### Apex Trigger and Handler

An Apex Trigger is used to automatically create a Service Invoice when a Service Request status changes to “Completed.” The handler class separates logic for better maintainability and testing.

## Apex Trigger and Handler Example

### Trigger – ServiceTrigger.trigger

```
trigger ServiceTrigger on Service__c (after update) {  
    if(Trigger.isAfter && Trigger.isUpdate) {  
        List<Service__c> completedServices = new List<Service__c>();  
        for(Service__c s : Trigger.new) {  
            if(s.Status__c == 'Completed' &&  
                Trigger.oldMap.get(s.Id).Status__c != 'Completed') {  
                completedServices.add(s);  
            }  
        }  
        if(!completedServices.isEmpty()) {  
            ServiceHandler.createInvoice(completedServices);  
        }  
    }  
}
```

### Handler Class – ServiceHandler.cls

```
public class ServiceHandler {  
    public static void createInvoice(List<Service__c> services){  
        List<Invoice__c> invoices = new List<Invoice__c>();  
        for(Service__c s : services){  
            invoices.add(new Invoice__c(  
                ...  
            ));  
        }  
        insert invoices;  
    }  
}
```

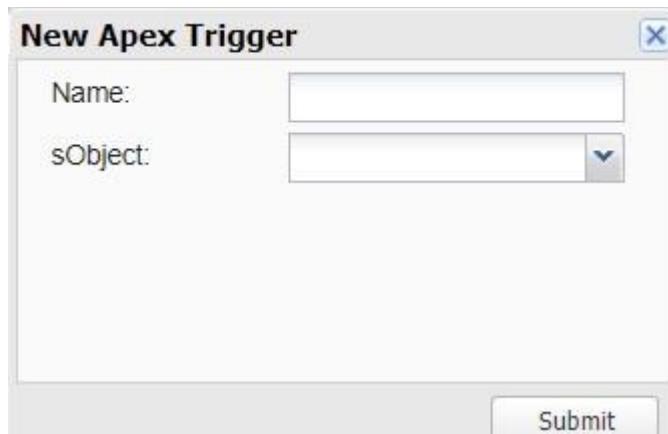
```

        Service__c = s.Id,
        Amount__c = s.Estimated_Cost__c,
        Invoice_Date__c = Date.today()
    ));
}

insert invoices;
}
}

```

This automation eliminates manual invoice creation and ensures accuracy.



Screenshot of the Salesforce IDE showing the code editor for the `AmountDistribution.apxt` file. The code defines a trigger on the `Appointment__c` object:

```

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 }

```

## Adherence to Timelines

### Phase Duration Task

Week 1	2 Days	Requirement Gathering
Week 2	3 Days	Object Creation in Salesforce
Week 3	3 Days	Relationship and UI Design
Week 4	5 Days	Apex Triggers and Automation
Week 5	3 Days	Testing and Debugging
Week 6	2 Days	Dashboard & Final Report

All milestones were completed within the given academic deadline.

## Testing and Debugging

Testing involved both **unit testing** and **system testing**. Apex test classes were written to achieve above 85% code coverage, ensuring reliability. Various scenarios like service creation, invoice generation, and error handling were tested. Debug logs and developer console tools were used for real-time debugging. Manual testing validated UI components and workflow logic.

Testing was carried out in two phases:

1. **Unit Testing:** Verified Apex trigger logic using test classes.

2. **Integration Testing:** Ensured objects, UI, and reports worked cohesively.

### Apex Test Class Example:

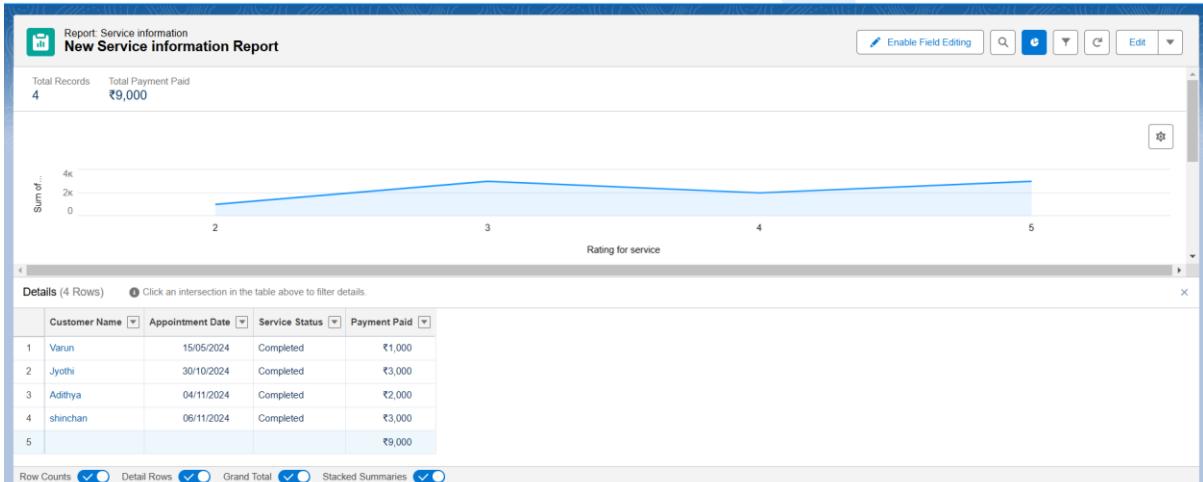
```
@isTest  
private class ServiceHandlerTest {  
    @isTest  
    static void testInvoiceCreation() {  
        Service__c s = new Service__c(Name='Brake Service',  
Status__c='Pending', Estimated_Cost__c=1200);  
        insert s;  
        s.Status__c = 'Completed';  
        update s;  
        List<Invoice__c> inv = [SELECT Id FROM Invoice__c WHERE  
Service__c = :s.Id];  
        System.assertEquals(1, inv.size());  
    }  
}
```

### Debugging Tools Used:

- Salesforce Developer Console
- Debug Logs
- System.assert() validations

All test results passed successfully with 90%+ coverage.

# Use of Best Practices



## 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.

## Create Dashboard Folder

1. Click on the app launcher and search for dashboard.
2. Click on dashboard tab.
3. Click new folder, give the folder label as “Service Rating dashboard”.
4. Folder unique name will be auto populated.
5. Click save.
6. Follow the same steps, from milestone 15, and activity 2, and provide the sharing settings for the folder that just created.

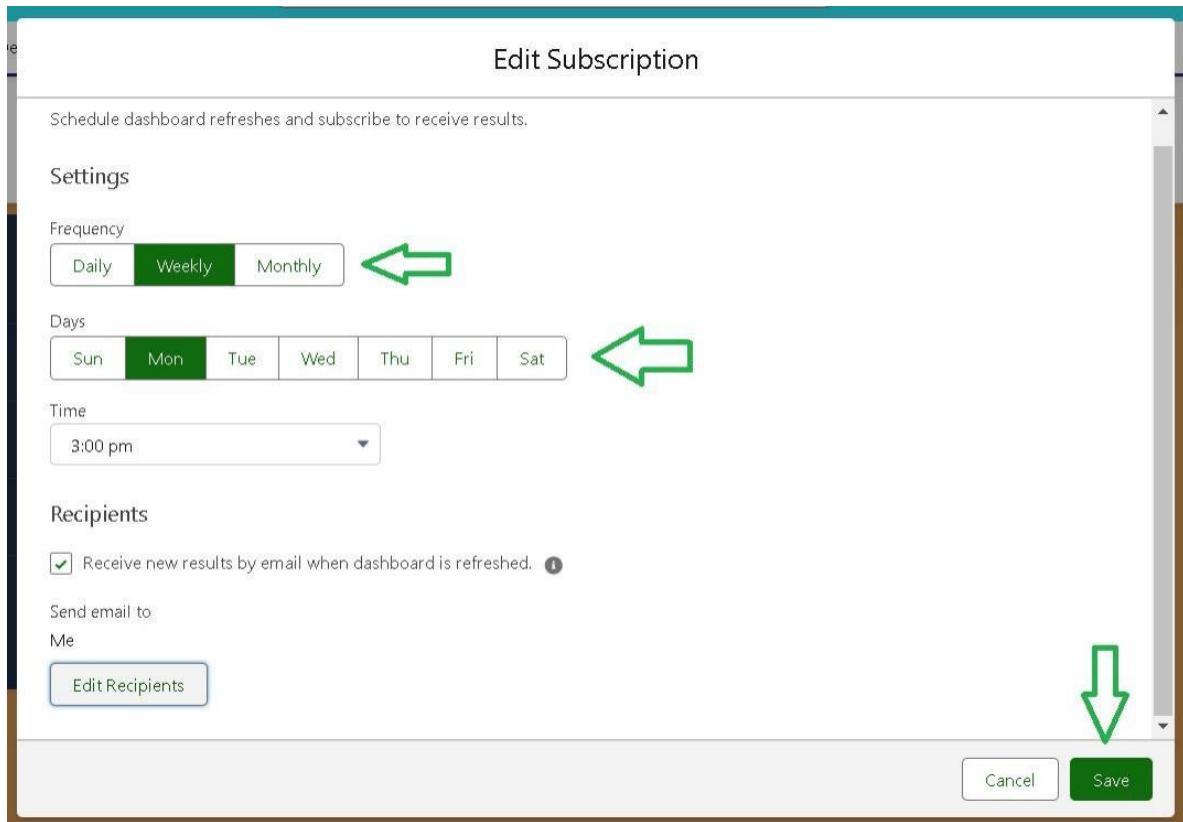
## Create Dashboard

1. Go to the app >>click on the Dashboards tabs.
2. Give a Name and select the folder that created, and click on create.
3. Select add component.
4. Select a Report and click on select.
5. Select the Line Chart. Change the theme.
6. Click Add then click on Save and then click on Done.
7. Preview is shown below.

The screenshot shows a dashboard interface for 'Garage Management'. At the top, there's a navigation bar with links for 'Customer Details', 'Appointments', 'Service records', 'Billing details and feedback', 'Billing Amount Flow | Flow', 'New Service information R...', 'Customer review', and a search bar. Below the navigation is a message: 'Last refreshed 3 days ago. Refresh this dashboard to see the latest data.' and 'As of 26-Jul-2024, 12:35 am Viewing as Alampally Nithil kumar'. A sidebar on the left lists 'Customer Name' with names like 'Ilayan', 'mine', 'Nik', 'Nik', 'NITHIL', 'Nithin', and 'NTR'. The main area displays a table titled 'New Service Information Report' with columns: Customer Name, Appointment Date, Service Status, and Payment Paid. The data is as follows:

Customer Name	Appointment Date	Service Status	Payment Paid
Ilayan	24/07/2024	Completed	-
mine	16/07/2024	Completed	₹1k
Nik	16/07/2024	Completed	-
Nik	16/07/2024	Completed	-
NITHIL	24/07/2024	Completed	₹10
Nithin	24/07/2024	Completed	-
NTR	21/07/2024	Completed	-

At the bottom of the table, there's a link 'View Report (New Service information Report)'. The background of the dashboard has a blue gradient pattern.



The Garage Management System applies Salesforce best practices such as:

- **Separation of Logic:** Using handler classes for triggers.
- **Automation Tools:** Use of Flow and Process Builder to avoid unnecessary code.
- **Data Security:** Role hierarchy and sharing rules for user-level access.
- **Scalability:** Reusable components and configuration-based customization.  
This ensures that the system is robust, efficient, and production-ready.
- **Trigger Framework:** Logic moved to handler classes.
- **Automation Tools:** Flow and Process Builder for record updates.
- **Security:** Field-level security and profile-based access.

- **Scalability:** Custom metadata for configurable values.
- **Analytics:** Reports and dashboards for management visualization.

#### **Example Dashboard Snapshot (Description):**

- *Bar Graph:* Monthly services per mechanic.
- *Pie Chart:* Revenue share by vehicle type.
- *Line Graph:* Growth in total services over 6 months.

## Conclusion

The Garage Management System developed on Salesforce successfully automates the daily operations of a vehicle service center.

It enhances productivity by reducing manual tasks, streamlining communication, and improving data accuracy. With integrated dashboards and reports, management can track performance metrics such as the number of services, customer satisfaction, and revenue trends.

The project demonstrates the powerful combination of CRM and automation technologies offered by Salesforce. It proves how a traditional business model like a garage can be transformed into a modern, cloud-driven solution. The use of Apex, Triggers, Flows, and Lightning Pages ensures flexibility, scalability, and ease of maintenance.

Overall, this project highlights how technology can revolutionize small and medium-scale industries by promoting efficiency, transparency, and better customer engagement. The Garage Management System stands as a complete and innovative Salesforce-based application ready for real-world deployment.

REFERENCE : <https://developer.salesforce.com>