**Garage Management System – Salesforce CRM**

**College Name:** KG COLLEGE OF ARTS AND SCIENCE

# Project Overview

The **Garage Management System (GMS)** is a **Salesforce-based CRM** designed to simplify and automate automotive service operations. It helps garages efficiently manage:

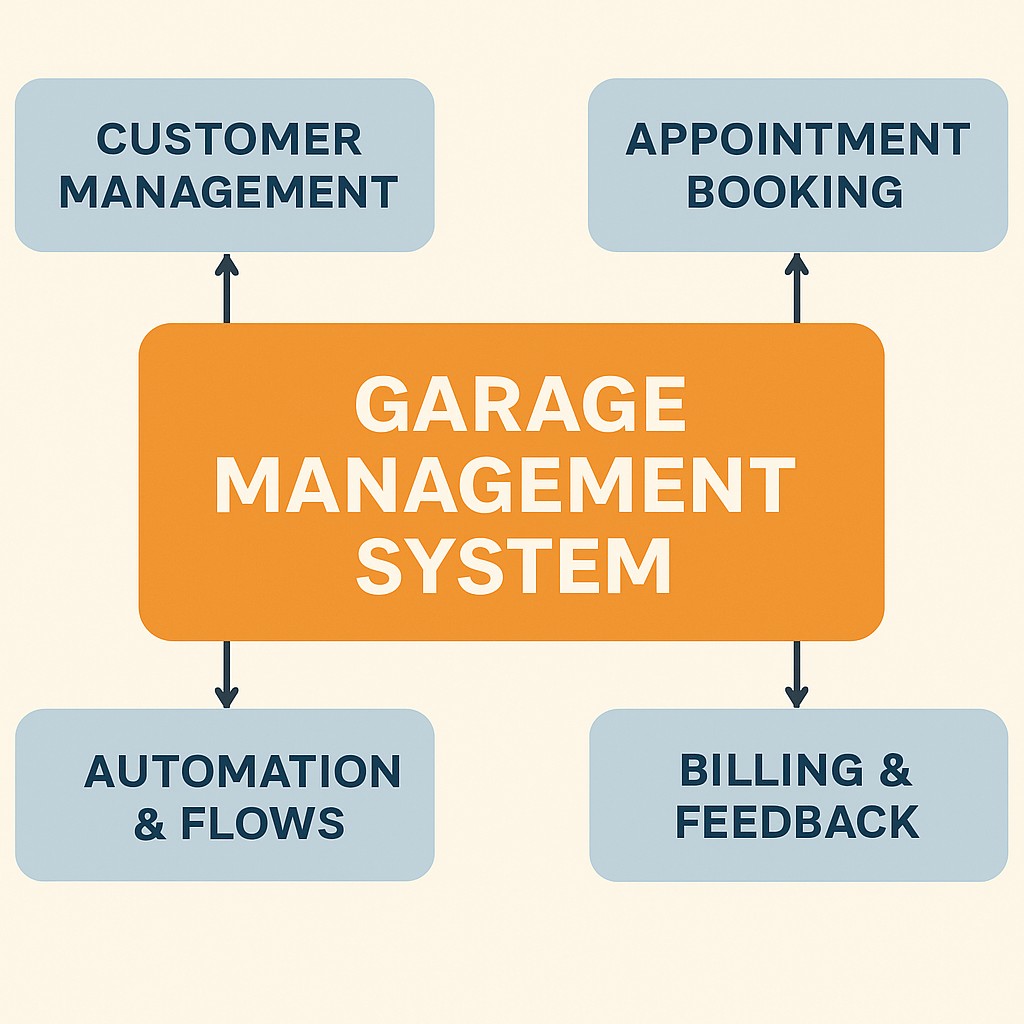
* Customer details

* Appointment scheduling

* Service records

* Billing and feedback collection

This system improves **operational efficiency**, ensures **quality service delivery**, and **enhances customer relationships** by leveraging Salesforce features like automation, dashboards, and reporting.



# Objectives

The main objective of this CRM is to **automate and streamline garage operations**. By integrating Salesforce CRM features, the system enables:

* Better **customer management** and engagement

* Improved **service tracking** and real-time monitoring

* Faster and **error-free billing & payment** processes

* Enhanced decision-making with **visual dashboards and reports**

**Business Value:**

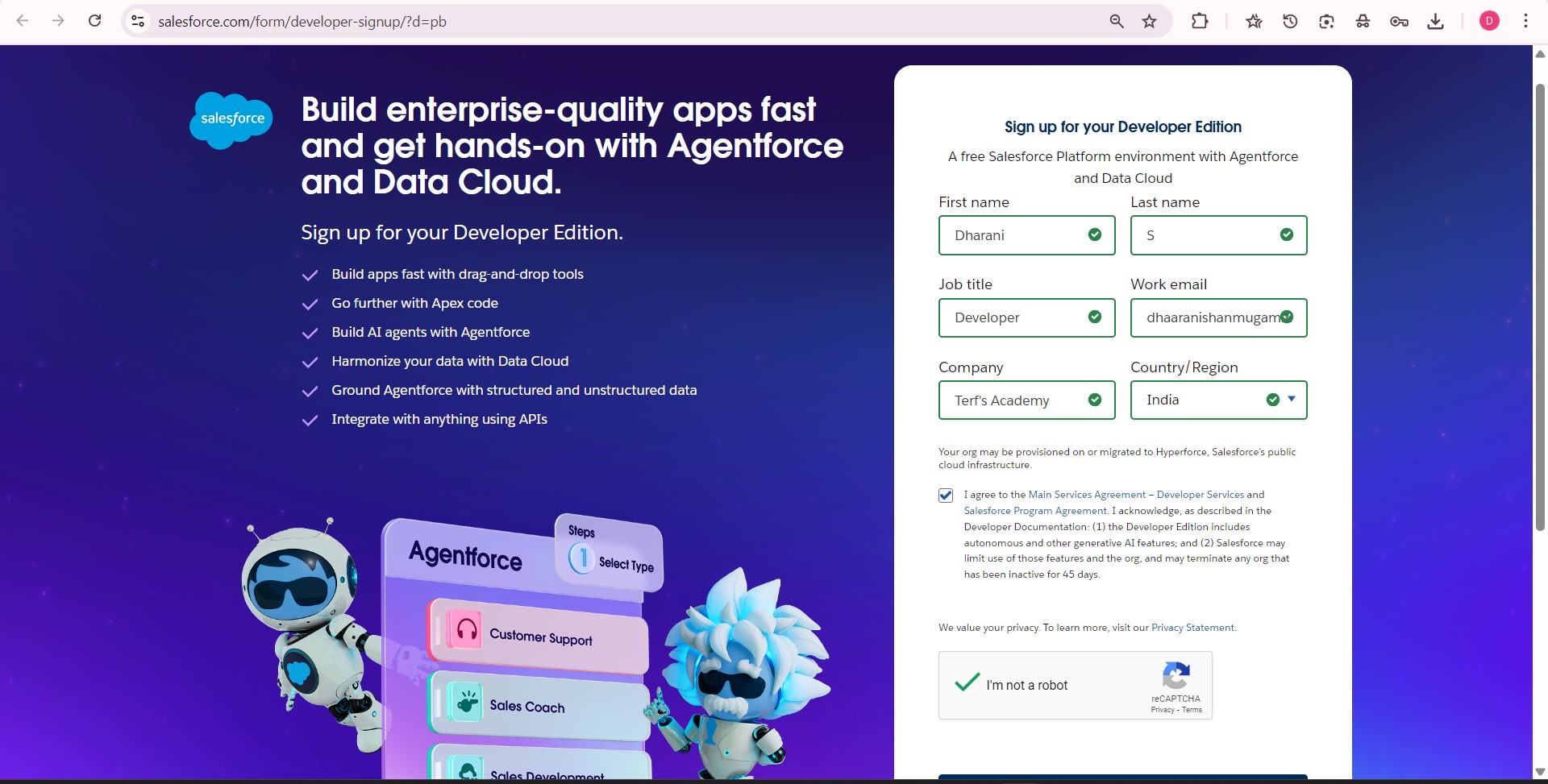
This system **reduces manual errors**, **improves productivity**, and **increases customer satisfaction**.

# Development Phase

## Creating Developer Account

A **Salesforce Developer Account** was created using this URL:

🔗<https://developer.salesforce.com/signup>



## Created Custom Objects

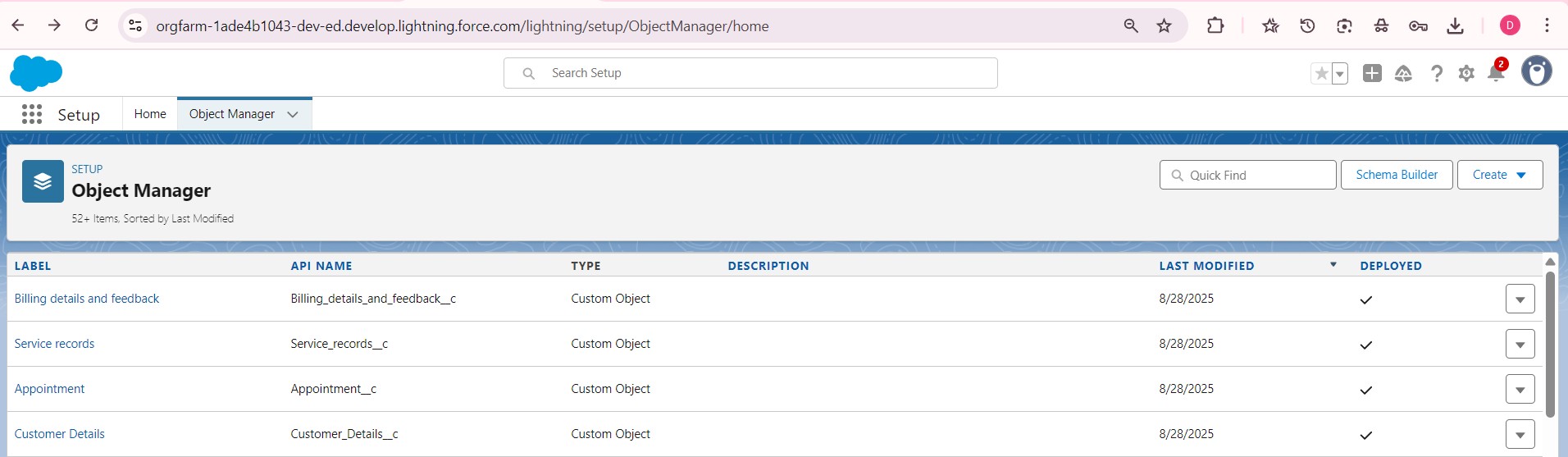
The following **custom objects** were developed to manage garage operations:

* **Customer Details** → Stores customer information.

* **Appointments** → Tracks all customer service bookings.

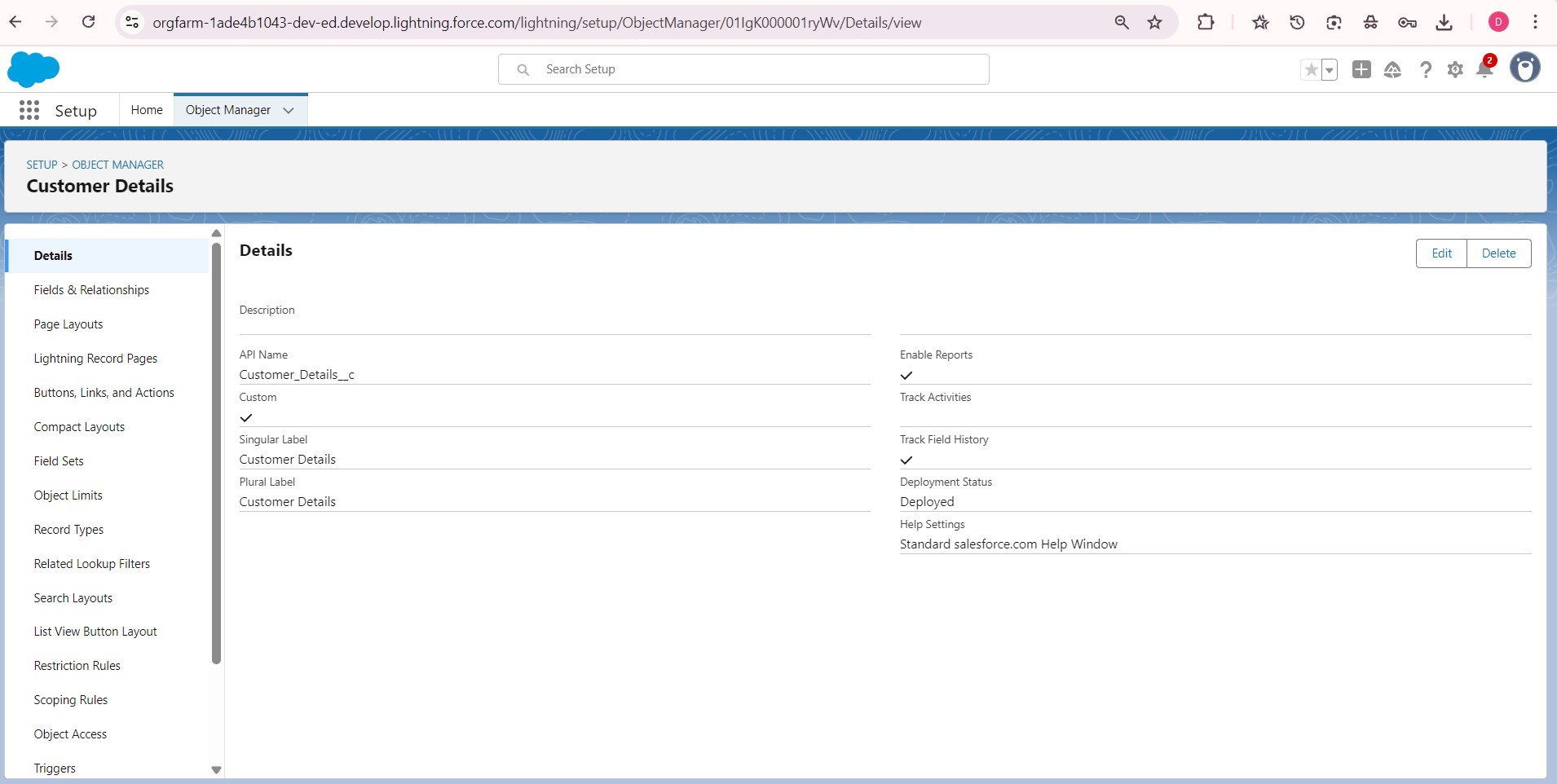
* **Service Records** → Maintains detailed service history.

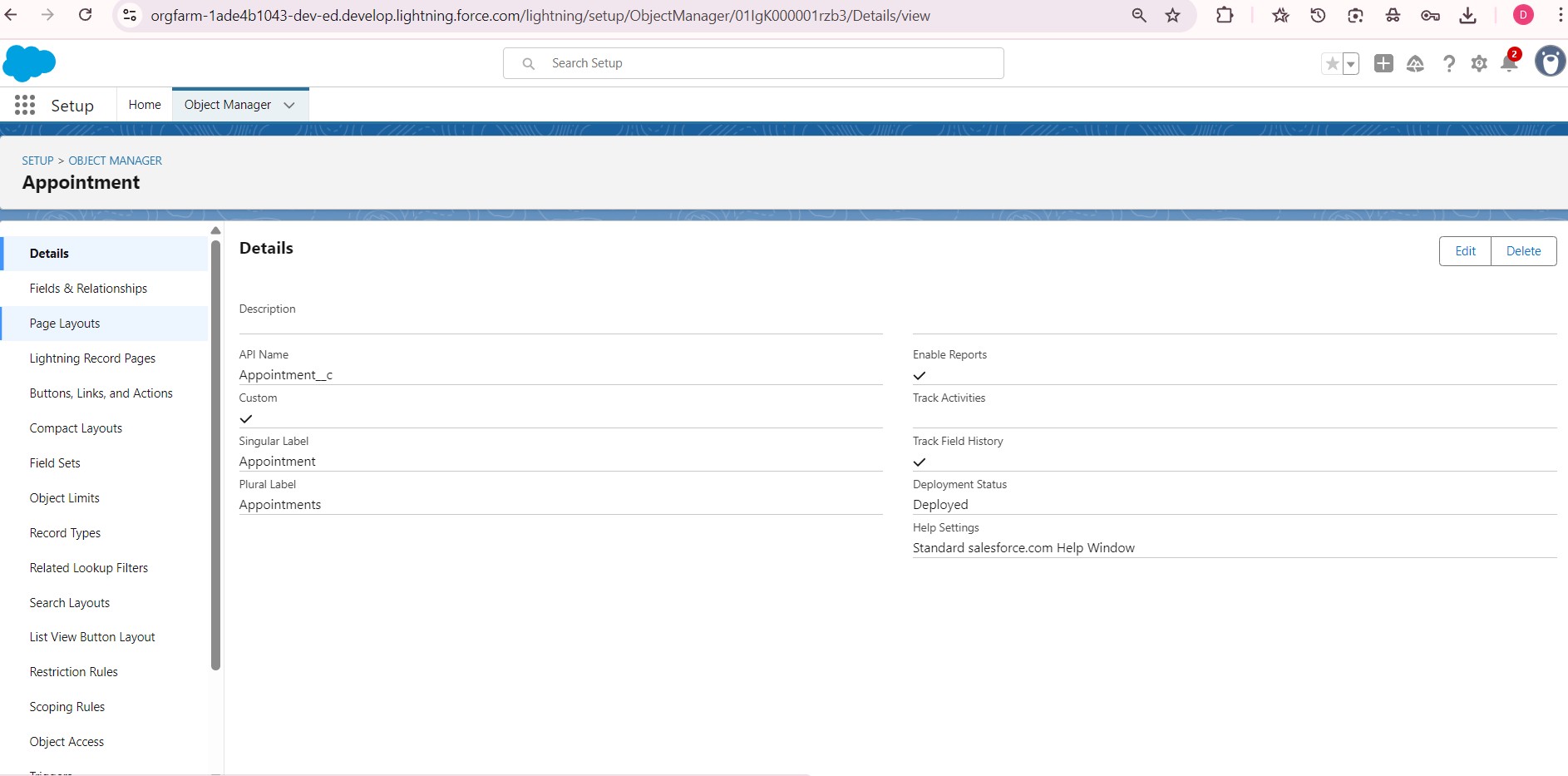
* **Billing & Feedback** → Captures payment details and customer ratings.



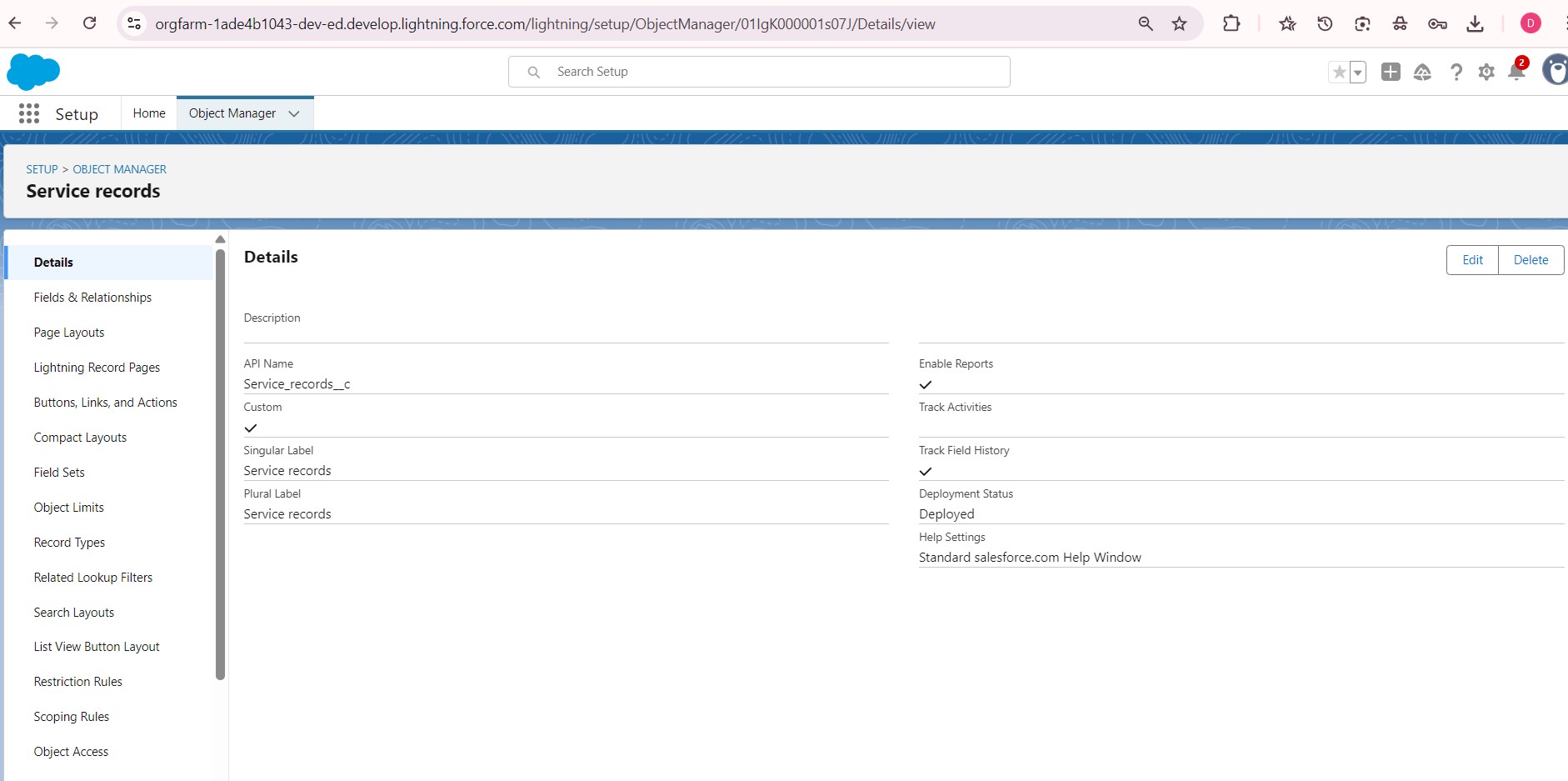
## Configured Fields and Relationships

* **Lookup Relationship:** Appointment → Customer Details

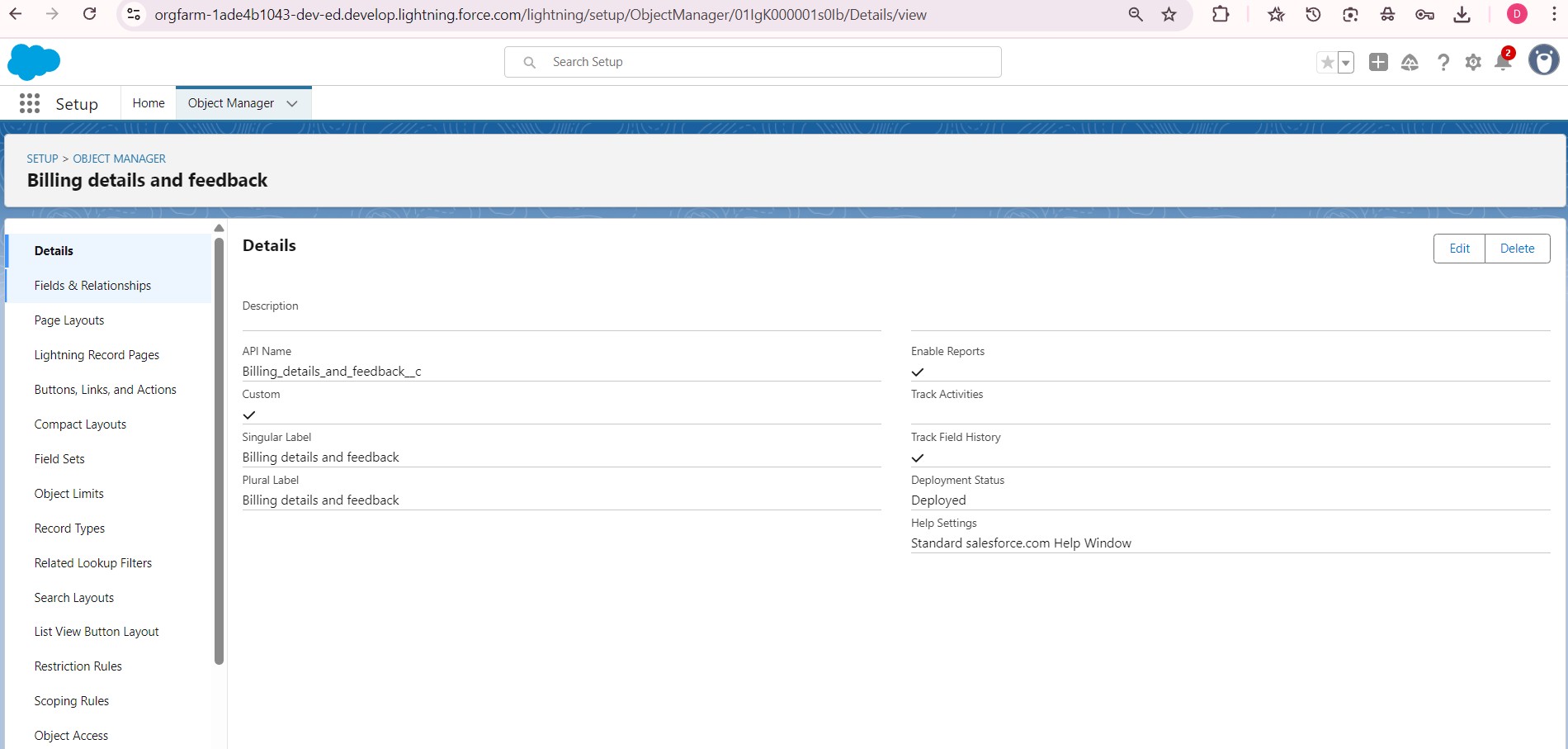




* **Master-Detail Relationship:** Service Records → Appointments



* **Lookup Relationship:** Billing & Feedback → Service Records



## Developed Lightning App

A **Lightning App** named **Garage Management System** was created, which includes the following tabs:

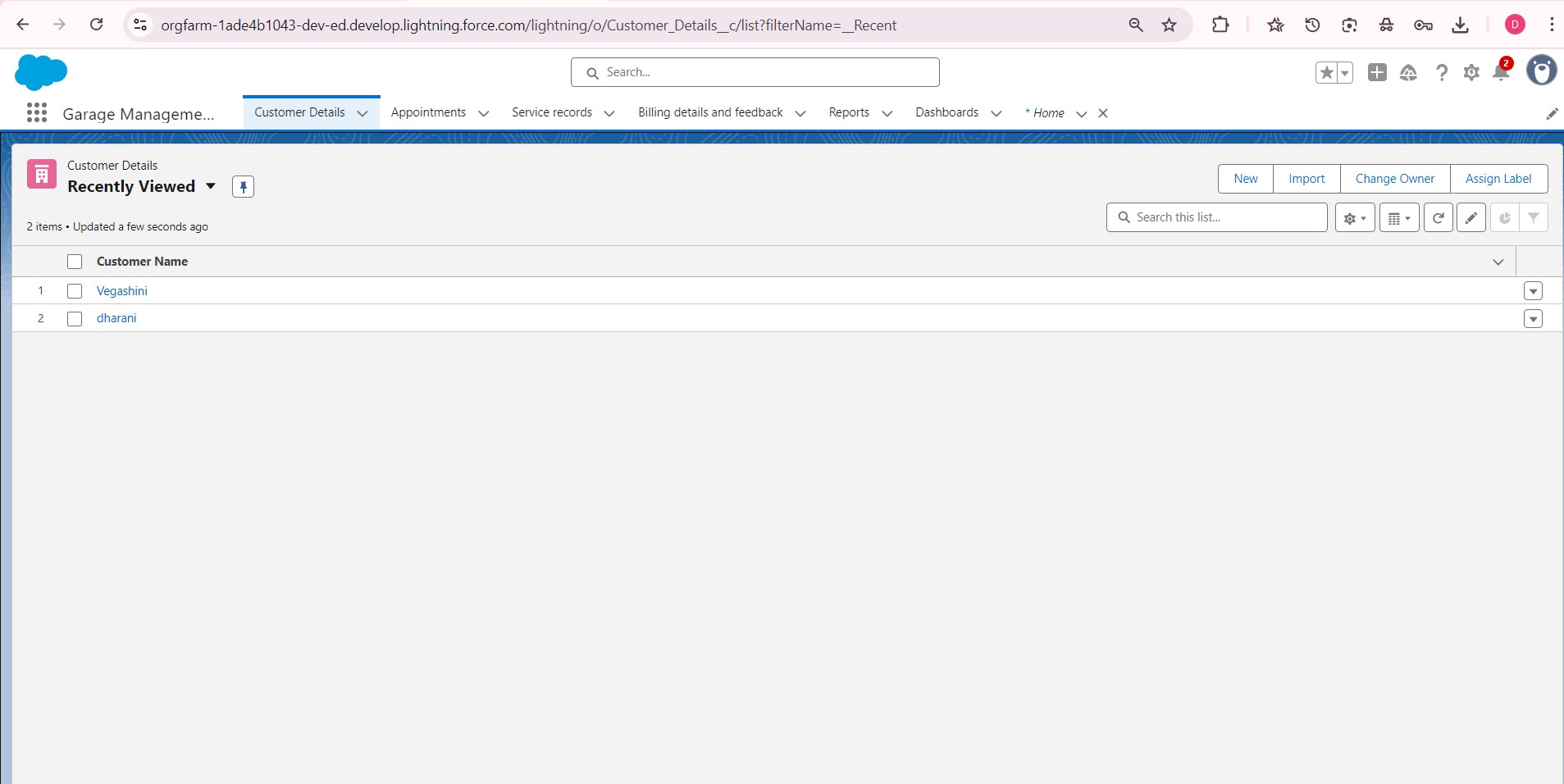
* Customer Details

* Appointments

* Service Records

* Billing & Feedback

* Reports & Dashboards



## Implemented Flows

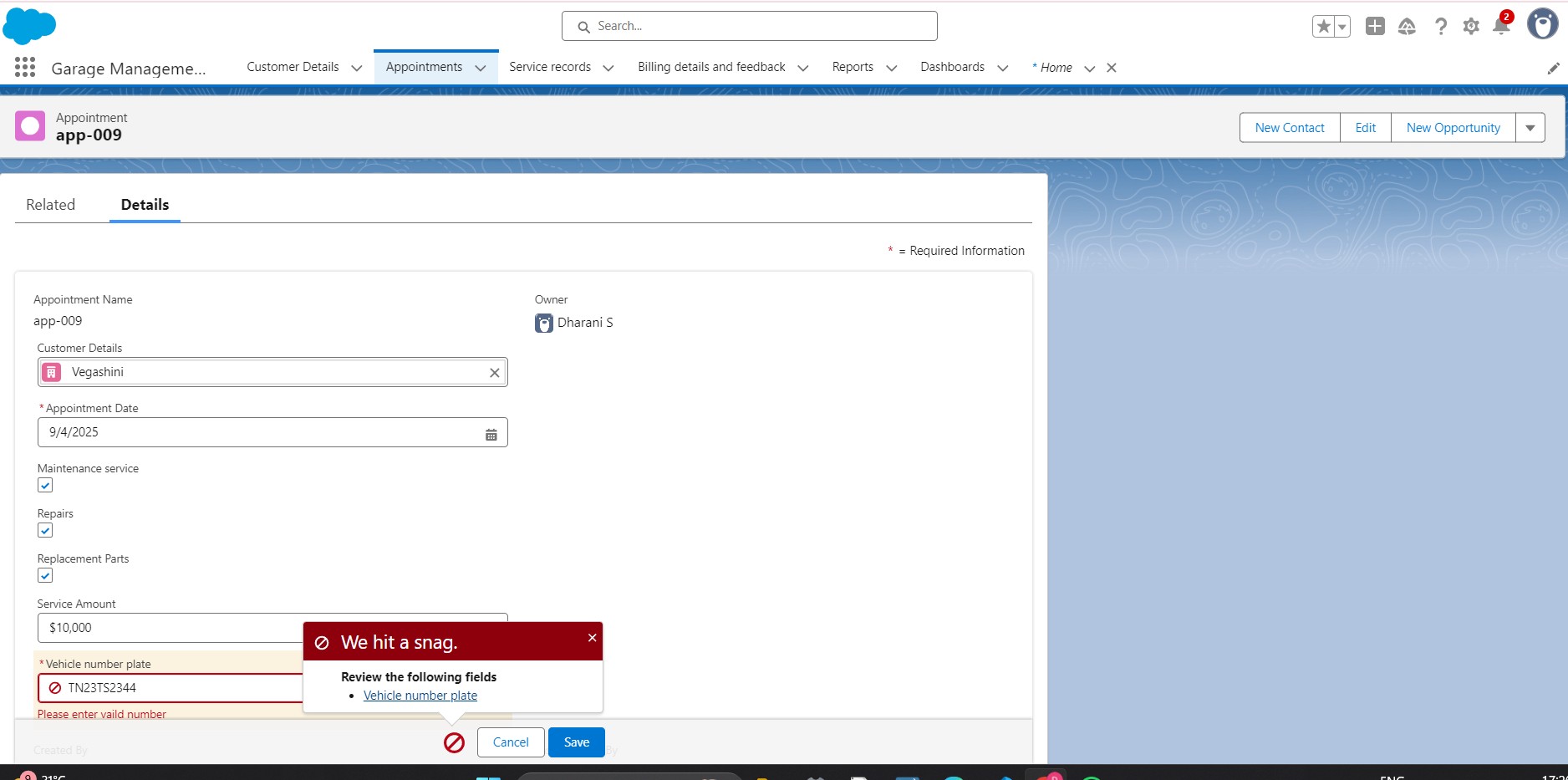
Automation flows were developed to:

* **Auto-update** the billing amount after payment completion.

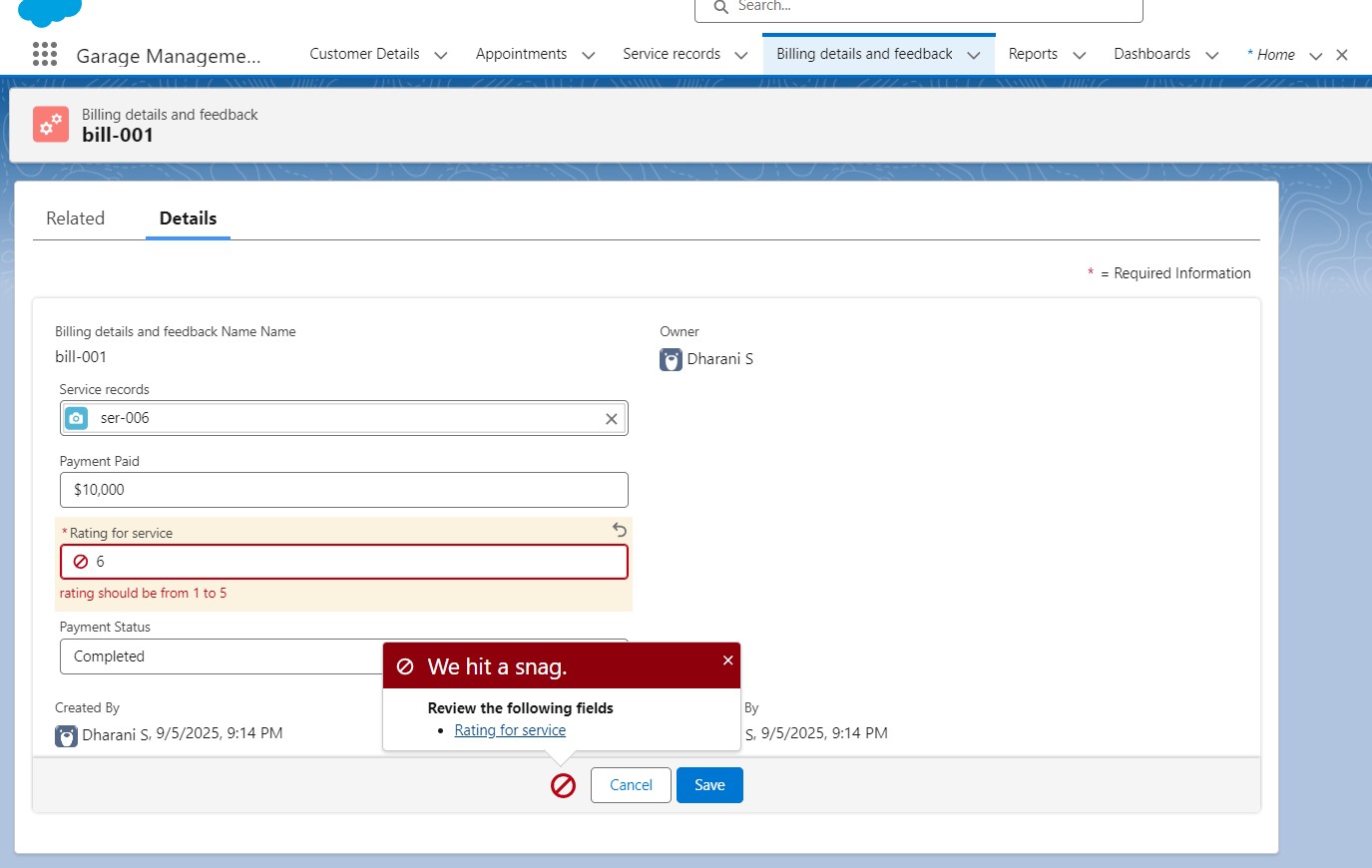
* Automatically change **service status** to **“Completed”** after a quality check.

## Validation Rules

* **Appointments:** Validates **vehicle number plate format**.



* **Billing & Feedback:** Ensures **service rating** is between **1 and 5**.



## Duplicate Rules

Duplicate rules were configured on **Customer Details** to **prevent duplicate records** based on **email** and **phone number**.

## Apex Trigger

A custom **Apex Trigger** was written on the **Appointment** object to:

* Distribute **service charges** based on selected services.

* Update **total billing** automatically.

## Reports & Dashboards

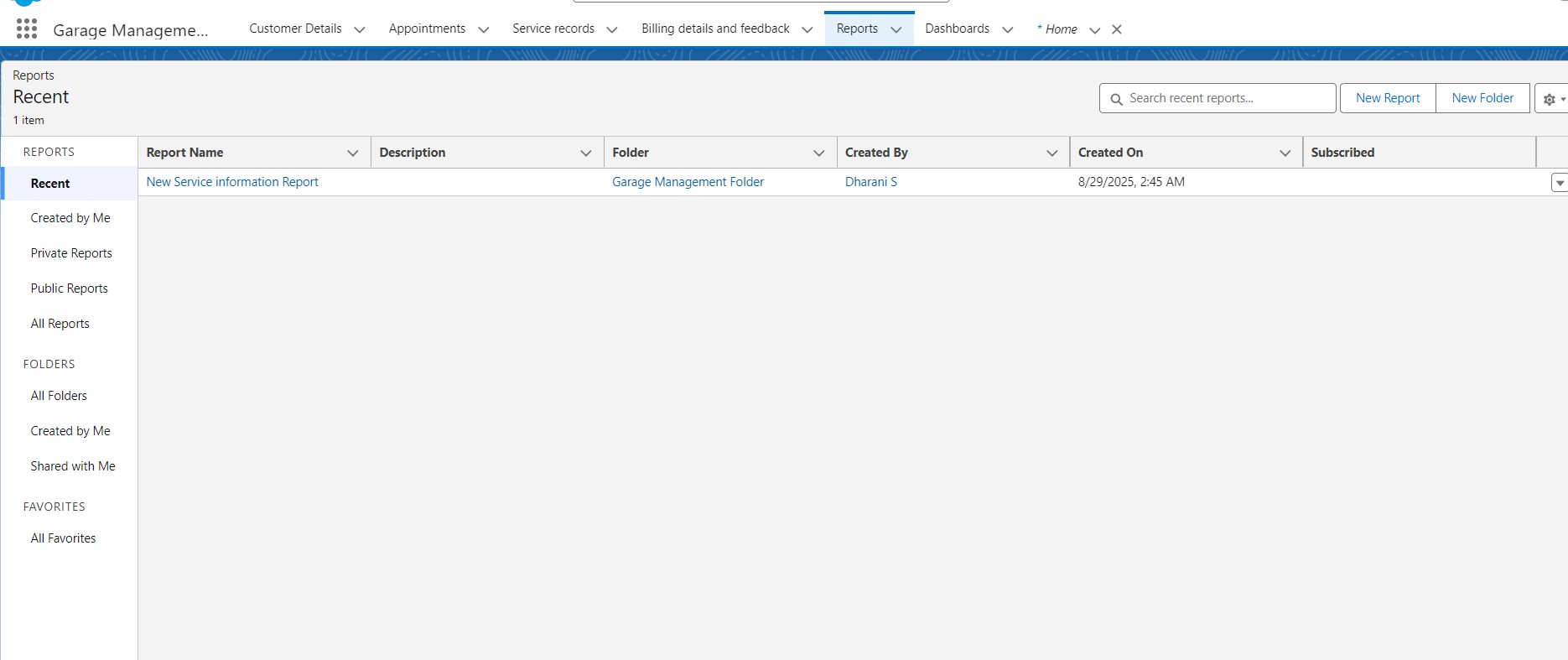
* **Reports:** Service information reports grouped by:

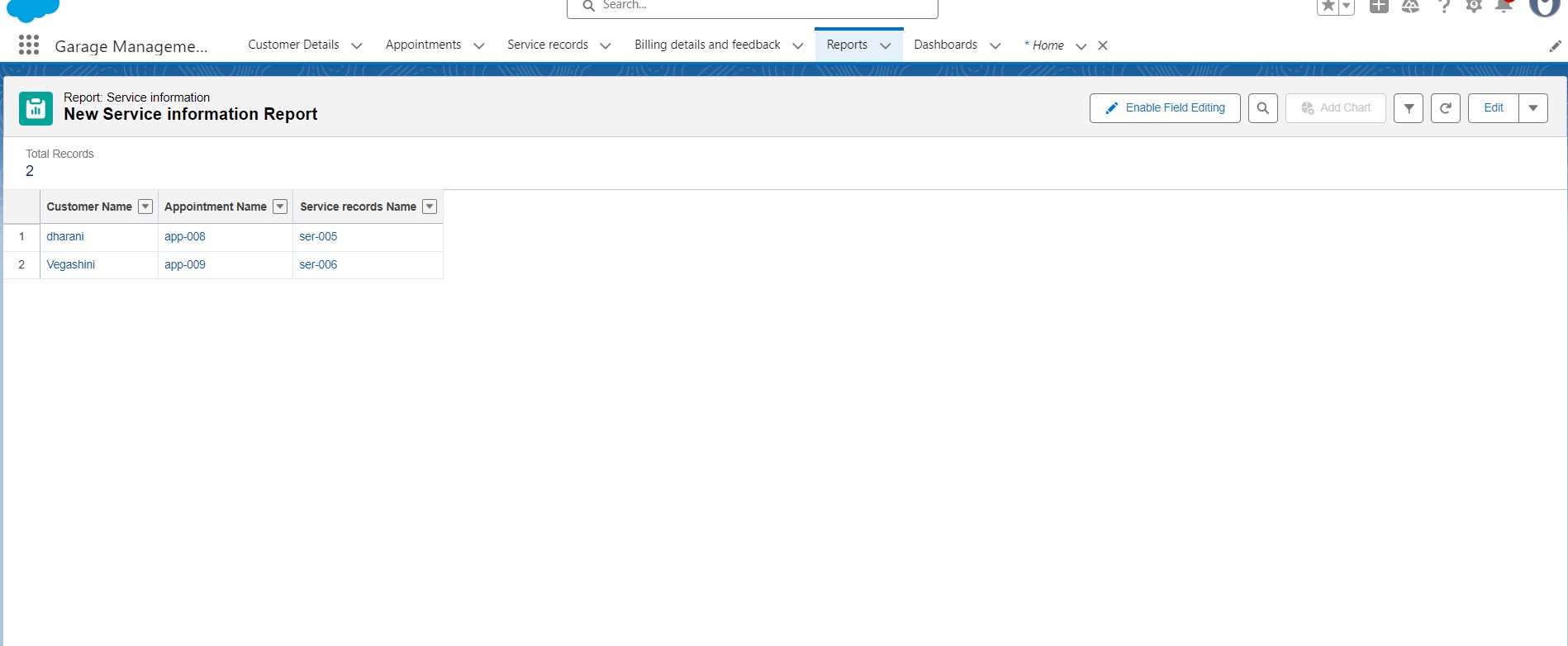
* 1. Customer

○ Appointment Date

○ Service Status

○ Payment Status

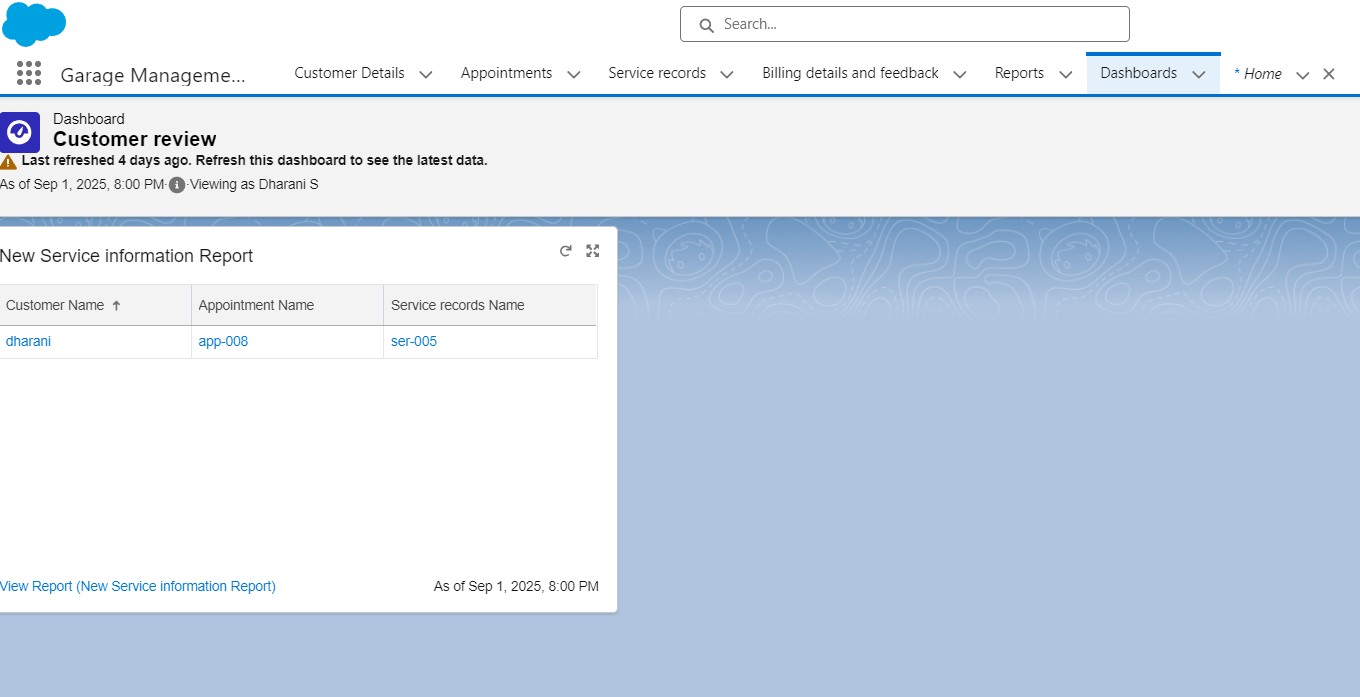




* **Dashboards:** Visual dashboards were created to display:

* 1. Service ratings

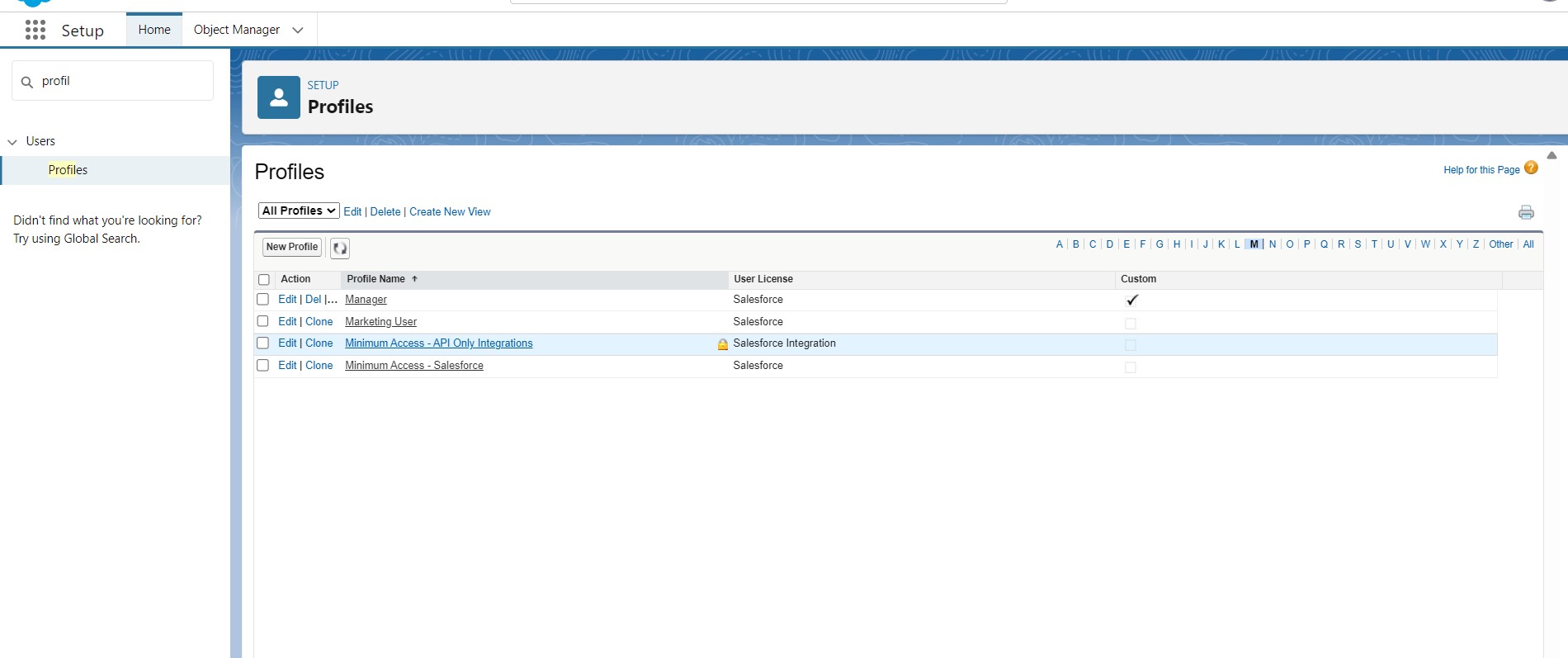
○ Payment summaries



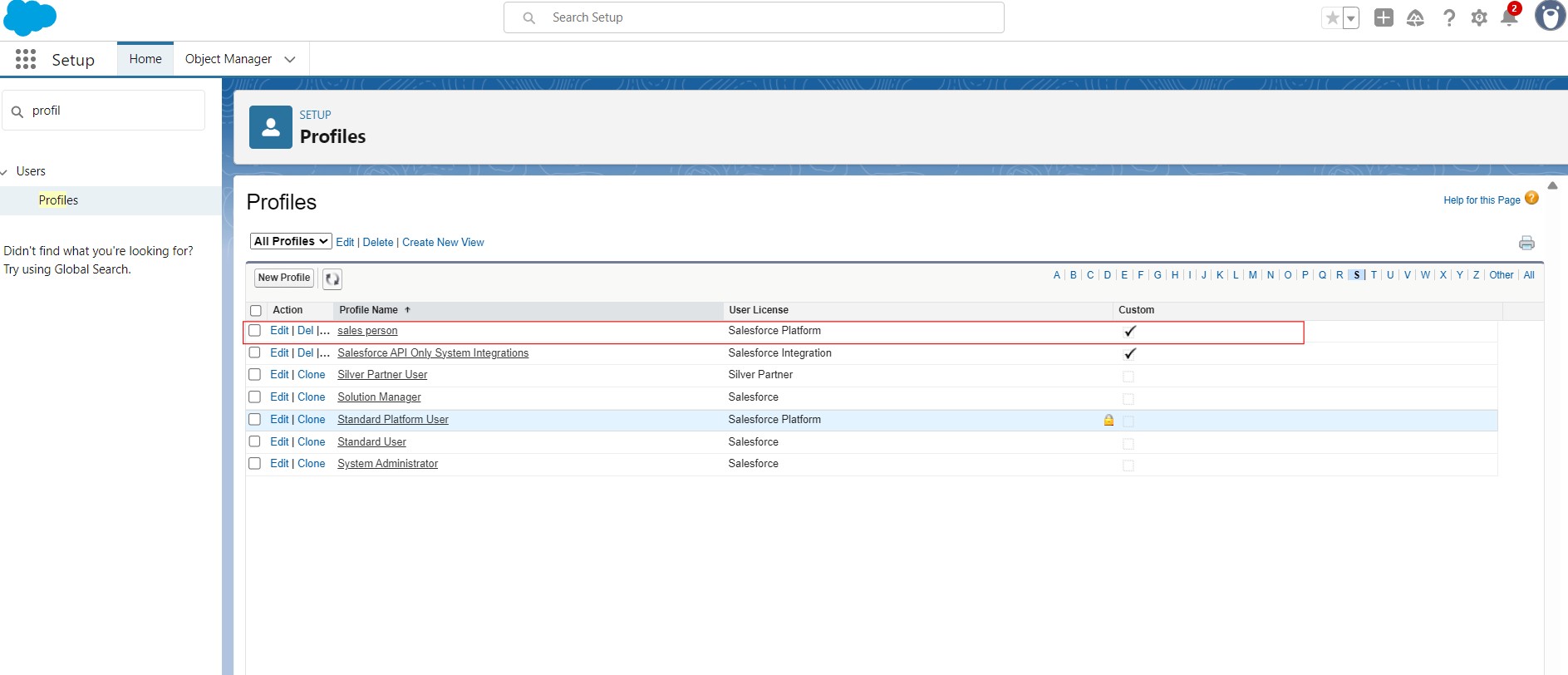
## Profiles, Roles & Sharing Settings

* **Profiles:**

○ **Manager Profile:** Full access to all objects.



○ **Salesperson Profile:** Limited access to service-related tasks.



* **Role Hierarchy:** Managers at the top → Salespersons reporting under managers.

* **Organization-Wide Defaults (OWD):** Set to **private** for **Service Records**.

* **Sharing Rules:** Configured to **grant managers** access to team records.

# Functional & Performance Testing

Testing was performed to ensure accurate functionality:

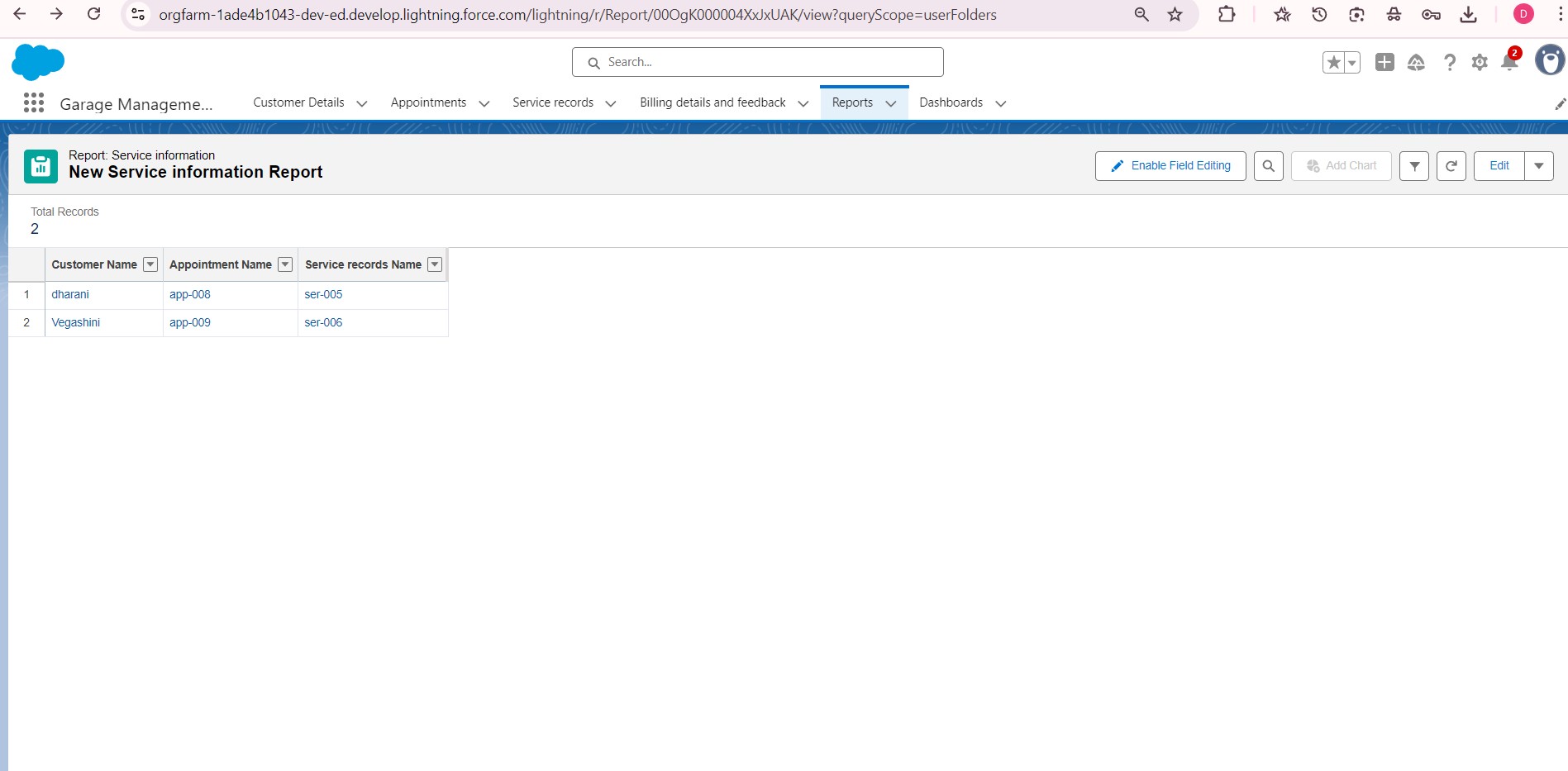
* Tested **validation rules** by entering incorrect or incomplete data.

* Verified **automation flows** by updating records and observing automatic updates.

* Checked **reports and dashboards** with sample records.

* Performed **role-based testing** by logging in as **Manager** and **Salesperson**.

# Results



* Tabs for **Customer Details**, **Appointments**, **Service Records**, **Billing &**

**Feedback**

* Flow form for **appointment creation**

* Reports showing **service records and payments**

* Dashboards displaying **service ratings** and **payment analytics**

# Advantages & Disadvantages

## Advantages

* Streamlines garage operations

* Automates **appointment booking**, **billing**, and **service tracking**

* Enhances **customer satisfaction** through feedback integration

* Provides **real-time reports** and **dashboards** for better decision-making

## Disadvantages

* Requires **internet connectivity**

* Needs **Salesforce knowledge** for initial setup

* Automation accuracy depends on **correct data entry**

# Future Enhancements

* **Chatbot Integration** for **real-time customer support**

* **AI-based predictive maintenance** for suggesting future services

* **Mobile App Integration** for **on-the-go booking and tracking**

# Conclusion

The **Garage Management System** built on **Salesforce CRM** successfully manages the **end-to-end garage operations**.

With the help of **custom objects**, **automation flows**, **validation rules**, **reports**, and **dashboards**, the system ensures:

* **Efficiency**

* **Accuracy**

* **Customer satisfaction**

Future enhancements like **chatbots**, **AI suggestions**, and **mobile apps** will further **improve the user experience** and **business growth**.

# Appendix

## Sample Apex Trigger

trigger ServiceChargeTrigger on Appointment\_\_c (before

insert) {

for(Appointment\_\_c app : Trigger.new) { app.Total\_Service\_Charge\_\_c =

app.calculateCharges();

}

}

## Sample Scheduled Class

global class DailyServiceReminder implements Schedulable { global void execute(SchedulableContext sc) {

sendDailyReminders();

}

public static void sendDailyReminders() {

List<Customer\_\_c> customers = [SELECT Id, Email\_\_c

FROM Customer\_\_c];

for(Customer\_\_c c : customers) {

Messaging.SingleEmailMessage mail = new

Messaging.SingleEmailMessage();

mail.setToAddresses(new String[]{c.Email\_\_c}); mail.setSubject('Garage Service Reminder'); mail.setPlainTextBody('Please check your

appointment and billing details.'); Messaging.sendEmail(new

Messaging.SingleEmailMessage[]{mail});

}

} }