**FINAL REPORT**

**PROJECT TITLE**

**“CRM APPLICATION FOR GARAGE MANAGEMENT SYSTEM "**

**COLLEGE NAME**

**ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES**

**(ANITS)**

**Team ID** : LTVIP2025TMID28970

**Team Size** : 4

**Team Leader** : Harshath Raghava

**Email ID :** NA

**Team member** : Bhagya Pasupuleti

**Email ID :** NA

**Team member :** Bandaru Balabhishek

**Email ID :** NA

**Team member** : Abhishek Bodda

**Email ID :** [abhishekbodda.22.It@anits.edu.in](mailto:abhishekbodda.22.It@anits.edu.in)

**1. INTRODUCTION**

**1.1 Project Overview**

The CRM Application for **Garage Management System** is a fully customized Salesforce-based solution designed to automate and streamline operations within an automotive service and repair environment. Traditional garage operations often face multiple operational challenges, such as manual appointment scheduling, inefficient work order tracking, inventory mismanagement for parts, opaque service process visibility, and a lack of centralized performance monitoring. This CRM project addresses these pain points by digitizing every core aspect of automotive service logistics.

This system is built to efficiently manage five core operational domains of a modern garage:

* **Customer & Vehicle Management** – Captures and manages details of all customers, their vehicles (make, model, VIN, mileage), and their service history.
* **Service & Work Order Management** – Records complete service requests, creates detailed work orders, tracks repair progress, and assigns tasks to mechanics.
* **Parts Inventory Management** – Manages the stock of automotive parts, tracks consumption against work orders, and highlights low stock levels.
* **Employee & Mechanic Management** – Manages the garage workforce, including service advisors and mechanics, with role-specific assignments and activity tracking.
* **Reports and Dashboards** – Tracks all key metrics including number of services, revenue generation, parts utilization, and mechanic productivity.

Salesforce capabilities such as custom objects, validation rules, record-triggered flows, and Apex triggers are extensively used to automate tasks and enforce business rules. Lightning App Pages and role-based layouts offer intuitive access to different user groups such as Garage Manager, Service Advisor, and Mechanic.

By leveraging Salesforce’s robust platform, the system provides better operational visibility, error-proof data entry, and consistent automotive service tracking, leading to increased efficiency and customer satisfaction.

**1.2 Purpose**

The primary objective of this project is to deliver a scalable, automation-driven, and easy-to-use CRM solution tailored to meet the specific needs of automotive service centers and garages. This includes effective customer and vehicle management, streamlined service processes, accurate inventory tracking, and insightful reporting for administrative use. The solution focuses on addressing the following key challenges:

**Customer & Vehicle Management**

* Create centralized records of all customers with their contact details and vehicle information (make, model, year, VIN, mileage).
* Maintain up-to-date service history for each vehicle, linked to the customer.
* Use picklist fields and dependent fields for vehicle categories, makes, and models.

**Service Request & Work Order Management**

* Track each service request from initial inquiry to final work order completion.
* Automate status updates for service requests and work orders as tasks progress.
* Assign work order tasks to specific mechanics with estimated completion times.

**Parts Inventory Management**

* Maintain real-time stock levels for all automotive parts.
* Automatically deduct parts from inventory when used in a work order.
* Generate alerts for low stock levels to facilitate timely reordering.

**Employee & Mechanic Assignment & Validation**

* Track each employee (service advisor/mechanic) with relevant details like role, skills, and availability.
* Prevent invalid assignments (e.g., assigning a technician to a specialized repair without the required certification) using Apex triggers or validation rules.
* Map employees to specific work bays or service types to optimize resource allocation.

**Automated Billing & Payment**

* Maintain comprehensive billing details for services and parts rendered.
* Implement record-triggered flows that automatically calculate total invoice amounts and outstanding balances.
* Automate the creation of invoices upon work order completion and track payment status.

**Role-Based Data Access**

* Create distinct roles and profiles: Garage Manager, Service Advisor, and Mechanic.
* Restrict visibility and edit permissions based on user roles (e.g., Mechanics only see their assigned work orders; Service Advisors can't modify payroll).
* Ensure sensitive records like financial data or employee details are only accessible to authorized users.

**Notification and Documentation**

* Send automated email/SMS confirmations or alerts when an appointment is booked, a service is completed, or an invoice is generated.
* Log historical service records and billing data for future audits, warranty claims, or reporting.
* Maintain clear separation of operational and analytical data for ease of use.

**Reporting and Analysis**

* Build dashboards and reports for:
  + Number of services completed per day/week/month
  + Revenue generated per service type or mechanic
  + Parts consumption and inventory turnover
  + Mechanic productivity and work order backlog
* Help administrators understand operational trends and optimize resource allocation and profitability.

**Data Integrity and Validation**

* Prevent errors such as duplicate vehicle entries, invalid mileage readings, or format mismatches using validation rules.
* Ensure logical consistency through Apex Triggers, such as ensuring all required parts are consumed before a work order is marked complete.
* Enforce data quality directly at the system level, reducing operational inefficiencies and improving customer trust.

**2. IDEATION PHASE**

**2.1 Problem Statement**

Problem statements are essential in any CRM development process because they clarify the real challenges end-users face. They serve as a foundation for solution design, ensuring that the final product addresses actual operational gaps rather than assumptions. A well-defined problem statement also helps the team stay focused and user-oriented throughout the development lifecycle.

In the traditional **automotive service and repair sector**, especially in independent garages and small-to-medium service centers, most of the key processes — including appointment scheduling, vehicle intake, parts management, mechanic assignments, and billing — are handled manually or with outdated tools. This results in administrative confusion, human error, inefficient resource utilization, and poor customer experience.

Through requirement studies, role-play exercises, and stakeholder interviews, our team identified the following pain points within current **garage management systems**:

**Unstructured Customer and Vehicle Management**

* Customer and vehicle details are stored manually with inconsistent naming, incomplete service history, or fragmented records.
* It is difficult to get a unified view of which vehicles are in for service, their current status, or their complete service history.

**Inconsistent Service Pricing & Billing**

* Service prices and part costs vary, but are often managed using fragmented spreadsheets or manual calculations.
* Staff face challenges in applying correct pricing, tracking parts used, or generating accurate invoices during service completion.

**Inefficient Mechanic Assignment Issues**

* Mechanics' skills, availability, and certifications are not always tracked properly.
* Errors occur when non-qualified employees are assigned to specialized repairs, leading to rework, delays, and safety issues.

**Lack of Process Automation and Data Validation**

* Appointment details, work order tasks, and parts consumption must be manually entered without system validation.
* This results in frequent inconsistencies in service records, missed data fields, and billing miscalculations.

**Absence of Role-Based Access Control**

* All users have the same system access, regardless of their position (e.g., service advisor, mechanic).
* There's no ability to restrict sensitive data like billing logs, customer contact details, or employee salaries from lower-level users.

**Limited Real-Time Operational Visibility & Analytics**

* Garage managers have no visibility into live work order status, mechanic availability, parts stock levels, or daily revenue.
* Data analysis for daily operations, resource planning, and identifying popular services is limited or unavailable.

**Problem Statement Template:** We believe that [Customer Type] is struggling with [Core Problem] because of [Root Cause 1] and [Root Cause 2, etc.]. This causes [Negative Impact 1] and [Negative Impact 2, etc.].

**Problem Statement 1: Operational Inefficiency and Lack of Visibility in Automotive Service Centers** We believe that **garage owners and service managers** are struggling to manage appointments, service workflows, and parts inventory efficiently because of the **absence of a centralized CRM platform** and **continued reliance on fragmented manual processes**. This causes **recurring scheduling conflicts, inaccurate billing, untracked parts usage, and lack of real-time visibility** into operations, resulting in **service delays, revenue loss, and reduced customer satisfaction.**

**Elaboration:**

* **Customer Type:** Includes garage owners, service managers, service advisors, and inventory managers responsible for daily service operations, resource allocation, and customer experience.
* **Core Problem:** The inability to manage and automate automotive service operations, parts inventory, and employee assignments in a reliable and scalable way.
* **Root Causes:**
  + **Lack of a centralized digital system:** Data exists in silos, such as physical appointment books, manual work orders, spreadsheets for inventory, or isolated local software.
  + **Manual entry and no validations:** No automation exists for linking parts to work orders, verifying mechanic skills before assignment, or auto-calculating service costs.
* **Negative Impacts:**
  + **Frequent service errors:** Staff may double-book appointments, miscalculate invoices, or forget to order critical parts, disrupting operations.
  + **Compliance risks:** Repairs may be performed by mechanics without appropriate certification, or customer data handled insecurely.
  + **Service inefficiency and customer dissatisfaction:** Customers may face long wait times, inaccurate quotes, or inconsistent service quality, affecting trust and repeat business.
  + **Lack of data-driven decision-making:** Managers cannot effectively track service performance, revenue per bay, mechanic utilization, or popular services, hindering growth.

**2.2 Empathy Map Canvas**

The Empathy Map Canvas is a valuable tool that helps teams adopt a user-centric mindset by capturing what users say, think, do, and feel. By analyzing our target users’ behaviors and emotions, we gain critical insights into their real-world challenges and expectations. This ensures that the CRM solution is designed not just to meet functional requirements, but to genuinely support users in their daily roles.

**Who are we empathizing with?** Our primary users for this Salesforce-based CRM system are:

* **Garage Owners / Service Managers:** These individuals are responsible for overseeing daily garage operations, managing overall profitability, resource allocation, and ensuring high service quality. Their main concerns revolve around operational efficiency, revenue growth, customer satisfaction, and data visibility across the business.
* **Service Advisors / Scheduling Coordinators:** These users handle customer intake, appointment scheduling, creating service requests, providing estimates, and coordinating work orders. They often face pressure to manage customer expectations, streamline initial processes, and reduce human error.
* **Mechanics / Technicians (Bay Staff):** These frontline users perform the actual service and repairs. They need clear work assignments, easy access to vehicle history, and efficient ways to log their work and parts usage with minimal administrative burden.

To understand these stakeholders better, we developed Empathy Map Canvases for three main personas: Garage Manager, Service Advisor, and Mechanic.

**Persona 1: Garage Manager (Admin)**

|  |  |
| --- | --- |
| **CATEGORY** | **DETAILS** |
| **Says** | "I need to know our total revenue today." "Which service type is most profitable?" "Are my mechanics being efficient?" |
| **Thinks** | "There must be a better way to track all active jobs." "I want to monitor parts inventory and reorder points from one screen." |
| **Does** | Reviews daily/weekly reports, approves major repairs, manages staffing, sets service pricing, oversees customer satisfaction. |
| **Feels** | Responsible for the garage's overall success and reputation. Needs a centralized, error-proof system with real-time insights into all operations. |

**Persona 2: Service Advisor (Scheduling & Customer Ops)**

|  |  |
| --- | --- |
| **CATEGORY** | **DETAILS** |
| **Says** | "I need to book this appointment quickly for next week." "Did I give the customer the correct estimate for parts and labor?" |
| **Thinks** | "The system should automatically check mechanic availability." "I wish I could see the full vehicle history when the customer calls." |
| **Does** | Schedules appointments, checks vehicle in, creates service requests, provides repair estimates, updates customers on service status. |
| **Feels** | Pressured to manage multiple customers simultaneously, worried about miscommunication or incorrect estimates, needs a system with validations and a clear workflow. |

**Persona 3: Mechanic / Technician (Bay Staff)**

|  |  |
| --- | --- |
| **CATEGORY** | **DETAILS** |
| **Says** | "Where can I see my assigned work orders for the day?" "How do I log the parts I used for this repair?" |
| **Thinks** | "I want role-based access to only my assigned jobs and relevant vehicle info." "The system should be mobile-friendly for logging work from the bay." |
| **Does** | Logs in to view assigned work orders, updates task progress, records parts consumption, adds repair notes, completes jobs. |
| **Feels** | Wants clarity and efficiency in assignments. Needs to trust the system for accurate vehicle history and to easily document their work without excessive paperwork. |

### 3. REQUIREMENT ANALYSIS

#### 3.1 Customer Journey Map

The Customer Journey Map for the **Garage Management CRM** outlines the complete lifecycle of a **vehicle owner** interacting with an automotive service center—from initial service inquiry to appointment booking, service completion, and billing. This journey illustrates how Salesforce components support both backend administration and frontend service delivery, ensuring accurate tracking, transparent costing, and a better overall customer experience.

This map provides a detailed look at how each step of the customer journey interacts with the CRM system. It ensures the automotive service runs efficiently and transparently, while also capturing operational data for continuous improvement.

**Stages in Customer Journey**

|  |  |  |
| --- | --- | --- |
| Stage | Description | System Involvement |
| 1. Vehicle Owner Schedules Appointment | A vehicle owner (customer) needs a service/repair and contacts the garage to book an appointment. | Appointment\_\_c object is created, capturing preferred date/time, linked to Customer\_\_c and Vehicle\_\_c. |
| 2. Service Inquiry & Estimate | The customer discusses required services, and the Service Advisor prepares a preliminary estimate. | Service\_Request\_\_c is captured, linked to Appointment\_\_c. Parts\_Inventory\_\_c and Service\_Rates\_\_c are referenced for cost estimation. |
| 3. Work Order Assignment | The service is approved, and a specific work order is created and assigned to a mechanic. | WorkOrder\_\_c object records the service details, assigned Mechanic\_\_c (from Employee\_\_c), and status. |
| 4. Service Approval & Billing Confirmation | The final cost for parts and labor is confirmed with the customer before or after the service. | Billing\_details\_and\_feedback\_\_c record is initiated. Record-triggered flow calculates and displays correct Service\_Amount\_\_c and Total\_Amount\_\_c. |
| 5. Service Completion & Payment | The service is completed, the customer pays, and backend data is updated for completed work. | Service\_Record\_\_c and WorkOrder\_\_c status are updated to 'Completed'. Billing\_details\_and\_feedback\_\_c reflects payment status. Reports are generated post-completion. |

**User Personas in Journey**

* **Vehicle Owner (Customer)** – The customer who uses the garage's services and interacts passively (through inquiries, scheduling, and payment acknowledgment).
* **Service Advisor / Scheduling Coordinator** – Uses the system to schedule appointments, create service requests, coordinate work orders, and prepare billing.
* **Garage Manager / Admin** – Monitors overall service status, manages pricing structures, analyzes business performance, and handles employee assignments using reports and dashboards.
* **Mechanic / Technician** – Interacts with the system to view assigned work orders, update task progress, and log parts used.

#### 3.2 Solution Requirements

The CRM system for **Garage Management** was developed to address critical business and operational needs of automotive service centers. Requirements were gathered and categorized into Functional Requirements and Non-Functional Requirements, ensuring the system is scalable, automated, secure, and easy to use for different garage staff roles.

**Functional Requirements**

* **Create and manage Customer and Vehicle records:** The system must allow staff to add and manage customer information (contact details) and associated vehicle details (make, model, VIN, service history).
* **Create and track Appointments & Service Requests:** Users must be able to schedule appointments, create detailed service requests, and convert them into work orders.
* **Manage Parts Inventory:** The system must support tracking of various automotive parts, including stock levels, reorder points, and consumption against work orders.
* **Automate Service Cost Calculation:** A Record-Triggered Flow should automatically calculate the estimated and final service costs based on labor rates, parts used, and service type, ensuring correct billing.
* **Validate Mechanic Assignment (Apex Trigger):** A Trigger must ensure that assigned mechanics belong to the Employee\_\_c object, possess the required skills/certifications for the task, and are available.
* **Generate operational reports and dashboards:** The system must allow creation of reports and dashboards to analyze service frequency, revenue, parts utilization, and mechanic productivity.
* **Validation Rules for data accuracy:** Rules must restrict invalid entries, such as duplicate vehicle VINs, invalid mileage, or leaving mandatory fields blank in appointments or work orders.

**Non-Functional Requirements**

* **Fast performance and low latency:** The system should respond efficiently, even when multiple objects (Appointment, Service Record, Vehicle, Parts, Employee) are interconnected.
* **Profile-based access:** Implement Profile and Permission Sets to ensure that only authorized roles (e.g., Garage Manager, Service Advisor, Mechanic, Parts Manager) can perform specific actions and view sensitive data.
* **Email security and automation:** Automated flows for appointment confirmations, service completion alerts, or invoice notifications should follow secure communication standards.
* **Ensure data consistency:** Lookup relationships and formula fields should maintain synchronization across objects (e.g., Service Record → Work Order → Vehicle → Customer).
* **Field history tracking:** Enable field tracking for key fields such as Work Order Status, Parts Consumed, Payment\_Status\_\_c, and Rating\_for\_service\_\_c.
* **User-friendly interface:** Use Lightning App Builder to create intuitive navigation across Customer, Vehicle, Service, Parts, and Employee modules.
* **Scalable system:** Should handle hundreds of service records, parts entries, and customer profiles without performance issues.

**Functional Requirements Table**

|  |  |
| --- | --- |
| ID | Requirement Description |
| FR1 | CRUD operations for all custom objects: Customer\_\_c, Vehicle\_\_c, Appointment\_\_c, Service\_Request\_\_c, WorkOrder\_\_c, Service\_Record\_\_c, Parts\_Inventory\_\_c, Employee\_\_c, Billing\_details\_and\_feedback\_\_c. |
| FR2 | Auto-calculate and display service costs using Flow based on service type, labor rates, and parts consumed. |
| FR3 | Send automated appointment confirmations, service completion notifications, or invoice alerts via Flow/Email Alerts. |
| FR4 | Categorize vehicles based on make, model, and type using Picklists and Record Types. |
| FR5 | Create relationships between Customer, Vehicle, Appointment, Work Order, Service Record, and Employee using Lookup fields. |
| FR6 | Assign permissions using Profiles and Permission Sets to roles like Garage Manager, Service Advisor, Mechanic, and Parts Manager. |
| FR7 | Maintain detailed service history for each vehicle and comprehensive billing history per customer. |
| FR8 | Develop Reports and Dashboards to track service volumes, revenue, parts usage, and employee utilization. |

**Non-Functional Requirements Table**

|  |  |
| --- | --- |
| ID | Requirement Description |
| NFR1 | Automate service cost calculation, invoice generation, and customer/internal notifications using Flows and Email Alerts. |
| NFR2 | Implement Validation Rules for correct data entry (e.g., valid VIN, mileage, date ranges) and mechanic skill restrictions. |
| NFR3 | Secure data access using Role-Based Access Control and Organization-Wide Defaults. |
| NFR4 | Design easy navigation via a dedicated Lightning App for garage operations. |
| NFR5 | Maintain clean and logical object structure for reusability and scalability. |
| NFR6 | Ensure system handles large volumes of service records, parts transactions, and customer data. |
| NFR7 | Provide clear UI with custom Page Layouts and Record Types for different user roles and data types. |

#### 3.3 Data Flow Diagram (DFD)

A Data Flow Diagram (DFD) helps visualize how data moves through the CRM system — from input by external entities to internal processes and data storage. It maps out how user interactions are handled, processed, and transformed within the **Garage Management CRM** system.

**Level 0 DFD – High-Level System Overview** This is the context-level diagram that represents the entire CRM system as a single unified process.

* **Single Process:**
  + The overall system is represented as one process named “**Garage Management CRM System**.”
* **External Entities:**
  + **Vehicle Owner (Customer):** The end-user interacting with the garage for service, inquiries, and payments.
  + **Garage Staff (Service Advisor, Mechanic, Admin):** Internal users managing appointments, services, parts, employees, and billing.
* **Primary Data Flows:**
  + **From Vehicle Owner to System:**
    - Service requests, appointment bookings, payment confirmation.
  + **From System to Vehicle Owner:**
    - Sends appointment confirmations, service estimates, repair status updates, invoices.
  + **From Staff to System:**
    - Inputs service details, assigns mechanics, manages parts inventory, updates work orders, processes billing.
  + **From System to Staff:**
    - Provides operational data via dashboards, reports, and record lists.

**Level 1 DFD – Detailed System Workflow** This level breaks down the high-level process into more specific internal processes, data stores, and interactions among Salesforce custom objects.

* **Process Breakdown:**
  + **Appointment & Service Request Creation:**
    - Service Advisor creates an Appointment\_\_c record with date/time, links a Customer\_\_c and Vehicle\_\_c record via lookup fields, and records initial Service\_Request\_\_c.
  + **Service Cost & Billing Assignment:**
    - A Record-Triggered Flow calculates the Service\_Amount\_\_c and Total\_Amount\_\_c on the Billing\_details\_and\_feedback\_\_c object, referencing Parts\_Inventory\_\_c and Service\_Rates\_\_c based on parts consumed and labor hours.
  + **Work Order Execution & Completion:**
    - Mechanics update WorkOrder\_\_c status, log tasks, and record Parts\_Consumed\_\_c.
  + **Apex Trigger for Mechanic Skill/Availability Check:**
    - When a WorkOrder\_\_c record is saved, an Apex Trigger checks whether the assigned Employee\_\_c (Mechanic) is eligible (e.g., possesses required skills, is available) by verifying attributes in Employee\_\_c.
  + **Service/Invoice Confirmation Notification:**
    - Upon appointment confirmation, service completion, or invoice generation, an automated Flow sends an email or SMS notification to the customer (Vehicle Owner) confirming details or providing alerts.

**Data Stores and Object Relationships:**

|  |  |  |
| --- | --- | --- |
| Process Stage | Input Entities / Objects | Output / Actions |
| Appointment & Service Request Creation | Service Advisor, Customer, Vehicle | New Appointment\_\_c record with lookup to Customer\_\_c, Vehicle\_\_c, and Service\_Request\_\_c. |
| Service Cost & Billing Assignment | Service\_Rates\_\_c, Parts\_Inventory\_\_c, WorkOrder\_\_c | Flow calculates and updates Billing\_details\_and\_feedback\_\_c with costs. |
| Validation | WorkOrder\_\_c, Employee\_\_c | Apex Trigger validates mechanic skills/availability before assignment. |
| Service Execution | Mechanic, WorkOrder\_\_c, Parts\_Inventory\_\_c | WorkOrder\_\_c status updated, Parts\_Consumed\_\_c logged, Service\_Record\_\_c created/updated. |
| Notification | Appointment\_\_c, Billing\_details\_and\_feedback\_\_c, Email Templates | Flow sends confirmation or update emails/SMS to customers. |
| Reporting | All custom objects (Customer\_\_c, Vehicle\_\_c, Appointment\_\_c, Service\_Record\_\_c, Billing\_details\_and\_feedback\_\_c, Parts\_Inventory\_\_c, Employee\_\_c, WorkOrder\_\_c) | Reports and dashboards for operational insights, financial performance, and resource utilization. |

#### 3.4 Technology Stack

To build a scalable and automated CRM solution for **garage management operations**, we utilized Salesforce’s low-code platform along with selective pro-code capabilities. This system was developed with a focus on managing complex relationships between customers, vehicles, services, parts, and employees while maintaining high data accuracy and automation.

The **CRM for Garage Management System** was implemented entirely on the Salesforce Lightning platform, enabling rapid development, seamless automation, and detailed reporting capabilities essential for automotive service-based use cases.

**Primary Technologies Used**

|  |  |
| --- | --- |
| Technology | Purpose |
| Salesforce Platform (Lightning) | Base CRM foundation; used for custom objects, automation, UI, and reporting in Lightning Experience. |
| Custom Objects | Models real-world entities like Customer\_\_c, Vehicle\_\_c, Appointment\_\_c, WorkOrder\_\_c, Service\_Record\_\_c, Parts\_Inventory\_\_c, Employee\_\_c, Billing\_details\_and\_feedback\_\_c. |
| Record Types & Page Layouts | Supports UI customization based on service types, vehicle categories, or different user roles. |
| Apex Trigger | Performs backend logic for complex validations (e.g., mechanic skill validation, ensuring all parts are logged before completion). |
| Validation Rules | Prevents data entry errors (e.g., invalid VIN format, duplicate appointments, incorrect mileage). |
| Flow Builder (Record-Triggered) | Automates service cost calculation, invoice generation, and sends notifications upon service completion or payment. |
| Reports & Dashboards | Visualizes operational data like service schedules, revenue trends, parts consumption, and mechanic performance. |
| Profiles & Permission Sets | Manages access based on staff roles such as Garage Manager, Service Advisor, Mechanic, and Parts Manager. |
| Lightning App Builder | Builds a unified, intuitive interface ("Garage Management Application") for all user roles. |
| Schema Builder | Graphically designs and reviews object relationships within the system. |

**Supporting Tools**

|  |  |
| --- | --- |
| Tool | Purpose |
| Lucidchart / Draw.io | Used to draft Data Flow Diagrams (DFD) and Entity-Relationship Diagrams (ERD). |
| Google Sheets / Docs | Used during requirement gathering, sprint planning, and test documentation. |
| Trello / Excel | Used for agile-based sprint planning and tracking project progress. |
| Salesforce Sandbox | Provides a safe development and testing environment for customizations. |
| Salesforce Change Sets / Deployment Tools | Used to migrate developed components from sandboxes to the production org. |

**Why This Stack Was Chosen:**

* **Salesforce Lightning Experience** supports low-code development, speeding up UI building and custom object creation while keeping the system highly scalable.
* **Apex Triggers** allow complex business logic enforcement such as validating assigned mechanic skills or ensuring data integrity across interconnected records.
* **Flows and Email Alerts** automate operations like service cost estimation, invoice generation, and internal/customer communication without manual intervention.
* **Reports and Dashboards** provide stakeholders with a real-time snapshot of garage operations and performance KPIs.
* **Validation Rules and Lookup Fields** ensure data consistency and reduce entry errors across related objects like Customer, Vehicle, Service Record, and Parts Inventory.

**Sample Tools Used in Development:**

* **Object Manager** – to define and relate objects like WorkOrder\_\_c and Parts\_Inventory\_\_c.
* **Flow Builder** – to automate service cost calculation, generate invoices, and trigger notifications.
* **Developer Console** – to implement and test Apex logic for advanced validations.
* **Email Template Builder** – to design automated email notifications for appointment confirmations and service summaries.
* **Report Builder** – to generate visual summaries for service and revenue performance.
* **Setup Menu (Profiles/Permission Sets)** – to configure field-level and object-level security for Service Advisors, Mechanics, and Managers.

### 4. PROJECT DESIGN

#### 4.1 Problem–Solution Fit

**Objective of Problem–Solution Fit**

The Problem–Solution Fit phase validates that the proposed CRM application addresses real operational challenges faced by **automotive service centers and their staff**. The goal is to ensure the system is not just a technical solution, but a practically relevant one that simplifies and enhances daily garage operations and customer service. This phase helps:

* Align system features with the actual pain points of **garage service workflows**.
* Understand user behavior and expectations (e.g., **appointment scheduling, work order management, parts inventory**).
* Promote adoption by designing intuitive, useful tools.
* Avoid overengineering by focusing only on essential, impactful features.

**Purpose Alignment Recap**

|  |  |
| --- | --- |
| Benefit | How CRM for Garage Management Achieves It |
| Solve real-world garage challenges | By digitizing operations like appointment scheduling, work order tracking, and parts management. |
| Increase operational accuracy | Using formula fields and validation rules to eliminate manual entry mistakes in billing and service records. |
| Improve transparency and service delivery | Through dashboards, detailed service records, and automated customer updates. |
| Ensure data security and role clarity | Via profiles, roles, and permission sets for Mechanics, Service Advisors, and Managers. |
| Speed up core workflows | With flows for auto-calculating service costs and Apex triggers for mechanic assignment validations. |

The CRM for **Garage Management System** project was carefully crafted to address the real pain points in traditional automotive service operations. The following table summarizes how key issues identified in the ideation phase are solved using Salesforce capabilities:

|  |  |
| --- | --- |
| Identified Problem | Implemented Solution in Salesforce |
| Manual appointment and work order assignment causing confusion | Custom objects like Appointment\_\_c, WorkOrder\_\_c, Service\_Record\_\_c, and Vehicle\_\_c manage scheduling and service tracking. |
| Service cost calculation errors and inconsistencies | Service\_Rates\_\_c and Parts\_Inventory\_\_c objects, combined with formula fields and flows, calculate service costs dynamically. |
| Lack of mechanic skill/availability validation | Apex Trigger ensures assigned Mechanic has the necessary skills and is available for the service. |
| No communication or alerts for critical events | Flows send automated notifications and confirmation messages for appointments, service completion, or invoice updates. |
| All staff accessing sensitive data without restriction | Profiles and Permission Sets manage role-based access (e.g., Mechanic vs. Service Advisor vs. Manager). |
| Difficulty in understanding operational KPIs | Dashboards and Reports visualize service performance, revenue collection, parts utilization, and employee efficiency. |
| No centralized interface for operations | Lightning App provides unified access to all objects, reports, and automations for the garage. |

This system ensures both functional fit (**automotive service-specific workflows**) and user-role fit (through tailored access and UI). As a result, it transforms day-to-day operations into a digital, organized, and easily manageable process aligned with the needs of modern garages.

#### 4.2 Proposed Solution

This solution was designed using Salesforce’s powerful declarative tools (like Flows, Page Layouts, and Validation Rules) along with select programmatic components (such as Apex Triggers) to develop a robust, low-code CRM tailored specifically for **garage management operations**.

Below is a breakdown of the core system components and their purposes:

**Custom Objects**

|  |  |
| --- | --- |
| Object Name | Purpose |
| Customer\_\_c | Stores comprehensive customer details including contact information. |
| Vehicle\_\_c | Maintains records for each vehicle (e.g., VIN, make, model, year, mileage, service history). |
| Appointment\_\_c | Logs each service appointment instance, linked with customer and vehicle. |
| Service\_Request\_\_c | Captures initial details of a customer's service need. |
| WorkOrder\_\_c | Manages the specific tasks and status of a repair job, linked with vehicle and assigned mechanic. |
| Parts\_Inventory\_\_c | Stores details for automotive parts including stock levels, cost, and supplier. |
| Service\_Rates\_\_c | Stores labor rates and standard service costs. |
| Employee\_\_c | Tracks mechanic and service advisor details with roles, skills, and assignments. |
| Billing\_details\_and\_feedback\_\_c | Stores invoice details, payment status, and customer feedback for completed services. |

**Automation Elements**

|  |  |  |
| --- | --- | --- |
| Type | Component | Function |
| Apex Trigger | ValidateMechanicAssignmentTrigger | Prevents a mechanic from being assigned a work order if they lack required skills or are unavailable. |
| Flow | ServiceCostCalculation\_Flow | Auto-calculates service cost in Billing\_details\_and\_feedback\_\_c based on parts consumed and labor hours from WorkOrder\_\_c. |
| Flow | AppointmentNotification\_Flow | Sends automated email/SMS confirmations for new appointments and service completion. |
| Validation Rule | Validate\_Mileage\_Entry | Ensures mileage entries are positive and logically increasing. |
| Validation Rule | Validate\_Appointment\_Conflict | Prevents overlapping appointments for the same service bay or mechanic. |

**Page Layouts & Record Types**

|  |  |
| --- | --- |
| Component | Purpose |
| Record Types: Maintenance, Repair, Diagnostic | Differentiates service types with custom fields and layouts, aiding in specific workflow management. |
| Page Layouts | Different layouts for Garage Manager, Service Advisor, and Mechanic, showing only relevant fields and related lists for their roles. |
| Field Dependencies | Dynamic display of available Vehicle\_Model\_\_c based on selected Vehicle\_Make\_\_c. |

**Profiles & Roles**

|  |  |
| --- | --- |
| Component | Description |
| Profiles | Manager (full operational access), Service Advisor (customer & appointment management), Mechanic (work order & task execution). |
| Roles | Hierarchy: Service Advisors and Mechanics report to the Garage Manager. |
| Permission Set | PartsInventoryAccess – grants specific access to Parts Inventory details for Parts Managers. |

**Reports & Dashboards**

|  |  |  |
| --- | --- | --- |
| Type | Name | Insights Provided |
| Report | Services by Type, Revenue by Mechanic | Service counts by category, revenue generated by individual mechanics. |
| Report | Parts Usage by Work Order | Details on which parts are consumed by specific work orders or vehicles. |
| Dashboard | Garage Operations Dashboard | KPIs such as active work orders, total daily/weekly revenue, mechanic utilization, and parts stock levels. |

**Lightning App**

|  |  |
| --- | --- |
| App Name | Navigation Tabs Included |
| Garage Management Application | Customers, Vehicles, Appointments, Service Requests, Work Orders, Parts Inventory, Employees, Reports, Dashboards. |

#### 4.3 Solution Architecture

**What is Solution Architecture?**

Solution Architecture is the structural design that connects business needs—like **customer appointment scheduling, work order tracking, parts inventory management**, and **mechanic assignments**—with the appropriate Salesforce tools—such as Flows, Apex Triggers, and Validation Rules. It defines how components interact across different layers to deliver a scalable, automated, and secure CRM solution for automotive service operations.

**Goals of the Solution Architecture for This Project:**

* Use Salesforce-native tools to digitize and streamline **customer, vehicle, service, and parts management processes**.
* Clearly define and connect custom objects such as Customer\_\_c, Vehicle\_\_c, Appointment\_\_c, WorkOrder\_\_c, Parts\_Inventory\_\_c, and Employee\_\_c.
* Automate **service cost calculations, invoicing, and assignment validations** using Flows and Apex.
* Secure data access through Profiles and Permission Sets.
* Deliver an efficient, scalable system for **garage administrators, service advisors, and mechanics**.

**Core Components of the Architecture**

|  |  |  |
| --- | --- | --- |
| Layer | Component | Description |
| Presentation Layer | Salesforce Lightning UI | Garage Managers, Service Advisors, and Mechanics interact through Lightning App pages and tailored layouts. |
| Business Logic Layer | Flows, Apex Triggers, Validation Rules | Automates service cost calculation, assignment validation, and enforces data input restrictions. |
| Data Layer | Custom Objects: Customer\_\_c, Vehicle\_\_c, Appointment\_\_c, WorkOrder\_\_c, Service\_Record\_\_c, Parts\_Inventory\_\_c, Employee\_\_c, Billing\_details\_and\_feedback\_\_c | Data is linked via lookup relationships and utilized by automation logic to ensure consistency. |
| Security Layer | Profiles, Roles, Permission Sets | Controls object access, field-level security, and UI visibility per role (Manager, Service Advisor, Mechanic, Parts Manager). |
| Reporting Layer | Reports and Dashboards | Real-time analytics on services completed, revenue generated, parts consumed, and mechanic productivity. |

**Architecture Flow (Text-Based Explanation)**

|  |  |  |  |
| --- | --- | --- | --- |
| User Action | Component Triggered | Object Affected | Automation/Output |
| Add a new customer & vehicle | Lightning UI | Customer\_\_c, Vehicle\_\_c | Records created; Vehicle\_\_c linked to Customer\_\_c; appears in Customer/Vehicle Reports. |
| Schedule an appointment | Appointment creation form | Appointment\_\_c | Validation ensures no date/time conflicts; links customer and vehicle. |
| Create a new work order | Service Request / Work Order form | WorkOrder\_\_c | Links to Appointment\_\_c, Vehicle\_\_c; auto-sets initial status; assigns to mechanic. |
| Log parts used for a repair | Work Order detail page | Parts\_Consumed\_\_c (related list), Parts\_Inventory\_\_c, WorkOrder\_\_c | Parts\_Inventory\_\_c quantity decreases; Flow/Roll-Up Summary calculates total part cost on WorkOrder\_\_c and Billing\_details\_and\_feedback\_\_c. |
| Assign mechanic to a work order | WorkOrder\_\_c assignment section | Employee\_\_c, WorkOrder\_\_c | Apex Trigger ensures assigned mechanic has required skills and is available; error on invalid assignment. |
| Complete service and generate invoice | WorkOrder\_\_c or Service\_Record\_\_c status update | Billing\_details\_and\_feedback\_\_c, WorkOrder\_\_c, Service\_Record\_\_c | Flow auto-generates invoice details, calculates final amount, updates status to 'Completed'. |
| View reports and KPIs | Reports/Dashboard tab | - | Shows summaries by service type, revenue per mechanic, parts utilization, and service frequency. |
| Switch user profile | Profile or Permission Set | - | Changes UI access and data control based on user role (e.g., Mechanic views only assigned work orders). |

**Note:** The entire architecture ensures strict role-driven access:

* **Service Advisors** can create appointments, service requests, and manage initial billing, but have restricted access to backend employee data or full financial reports.
* **Mechanics** can only view their assigned work orders, update status, and log parts consumed, without access to customer contact details or billing.
* **Garage Manager** users have complete control over all objects, employee data, and analytics tools.

This modular and secure architecture ensures maintainability, data integrity, and operational control for the automotive service domain.

### 5. PROJECT PLANNING & SCHEDULING

#### 5.1 Project Planning

To ensure systematic, timely, and high-quality development, our team followed the **Agile Methodology using the Scrum Framework**. The complete lifecycle of the **CRM for Garage Management System** was executed in two 5-day sprints, facilitating iterative development, real-time feedback, and consistent delivery.

**Agile Planning Overview** Agile Scrum promotes incremental delivery through short, iterative sprints. Each sprint cycle includes:

* **Product Backlog:** List of epics, user stories, and system features.
* **Sprint Backlog:** Subset of prioritized stories selected for the sprint.
* **Story Points:** Estimation metric denoting task complexity and effort.
* **Velocity:** Average number of story points completed per sprint.
* **Burndown Chart:** A visual representation of work remaining across sprint days.

**Sprint Overview**

|  |  |  |  |
| --- | --- | --- | --- |
| Sprint | Duration | Focus Areas | Sprint Goal |
| Sprint 1 | 1st July – 5th July 2025 | Object Modeling, Page Layouts, Field Dependencies, Access Control | Set up the core garage management CRM schema and user access mechanisms. |
| Sprint 2 | 6th July – 10th July 2025 | Automation (Flows/Triggers), Validation Rules, Reports & Dashboards | Finalize backend automation and build data insights via dashboards for garage operations. |

**Sprint 1: Data Modeling & Configuration** **Objective:** Establish the technical foundation of the **Garage Management System** by defining objects, fields, relationships, roles, and record types.

**Tasks:**

* Created custom objects: Customer\_\_c, Vehicle\_\_c, Appointment\_\_c, Service\_Request\_\_c, WorkOrder\_\_c, Parts\_Inventory\_\_c, Employee\_\_c, Billing\_details\_and\_feedback\_\_c, Service\_Rates\_\_c.
* Added custom fields (text, picklist, number, currency, formula, lookup) for all garage entities.
* Created field dependencies (e.g., Vehicle\_Model\_\_c based on Vehicle\_Make\_\_c).
* Set up role hierarchy and profiles: Admin, Garage Manager, Service Advisor, Mechanic.
* Developed page layouts tailored for each user role (e.g., Service Advisor layout for Appointment\_\_c, Mechanic layout for WorkOrder\_\_c).

**Deliverables:**

* Custom object relationships established for customer, vehicle, service, and parts.
* Field-level security applied using profiles and permission sets.
* Modular page layouts for Manager vs. Service Advisor vs. Mechanic roles.

|  |  |  |
| --- | --- | --- |
| Estimated Story Points | Completed | Velocity |
| 12 SP | 12 SP | 12 |

**Sprint 2: Automation & Reporting** **Objective:** Develop intelligent automations, error-prevention validations, and performance reporting for the **Garage Management System**.

**Tasks:**

* Built Record-Triggered Flow to auto-calculate service cost in Billing\_details\_and\_feedback\_\_c based on WorkOrder\_\_c details (parts/labor).
* Created Apex Trigger to validate **mechanic skill/availability** on WorkOrder\_\_c assignment.
* Designed Validation Rules to enforce business logic (e.g., mileage entry, appointment conflict, valid VIN).
* Generated Reports: Active Work Orders, Revenue by Service Type, Parts Consumption, Mechanic Productivity.
* Created Dashboards: Daily Garage Operations, Mechanic Utilization, Financial Overview.
* Deployed solution using Change Sets.

**Deliverables:**

* Working flow-based automation and Apex logic for key garage processes.
* Visual insights into garage performance through Reports & Dashboards.
* Fully tested, production-ready Garage Management CRM.

|  |  |  |
| --- | --- | --- |
| Estimated Story Points | Completed | Velocity |
| 12 SP | 12 SP | 12 |

**Story Point Allocation per Task**

|  |  |
| --- | --- |
| Task | Story Points |
| Custom Object & Field Creation | 3 SP |
| Profiles, Roles, and Page Layouts | 2 SP |
| Flow for Service Cost Calculation & Notifications | 3 SP |
| Apex Trigger for Mechanic Validation | 2 SP |
| Validation Rule Implementation | 1 SP |
| Reports & Dashboards | 2 SP |
| Testing and Deployment | 1 SP |
| Total | **14 SP (planned), 12 SP/sprint executed** |

**Velocity Chart**

|  |  |  |
| --- | --- | --- |
| Sprint | Story Points Planned | Completed |
| Sprint 1 | 12 SP | 12 SP |
| Sprint 2 | 12 SP | 12 SP |
| Total | **24 SP** | **24 SP ✅** |

**Outcome:** Full delivery with 100% velocity in both sprints.

**Burndown Chart Overview (Sprint 1 Example)**

|  |  |
| --- | --- |
| Day | Story Points Remaining |
| Day 1 | 12 |
| Day 2 | 9 |
| Day 3 | 6 |
| Day 4 | 3 |
| Day 5 | 0 |

Similar trend followed in Sprint 2.

**Project Management Tools Used**

* **Trello:** User Story Cards, Task Boards, Sprint Backlog.
* **Google Sheets:** Burndown Charts, Velocity Tracking.
* **Salesforce Sandbox:** Development & QA testing environment.
* **Change Sets:** Deployment from Sandbox to Production.

**Best Practices Followed**

* Used Fibonacci-based estimation for task complexity.
* Divided Epics into smaller, manageable user stories.
* Balanced scope between sprints to avoid overcommitment.
* Internal testing conducted within each sprint for fast feedback.
* Tracked team performance via burndown and velocity metrics.

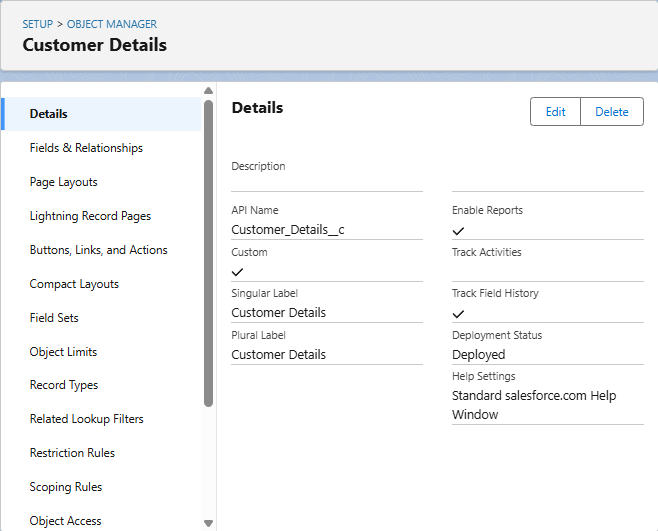
**Final Outcome** The Agile-driven planning approach enabled:

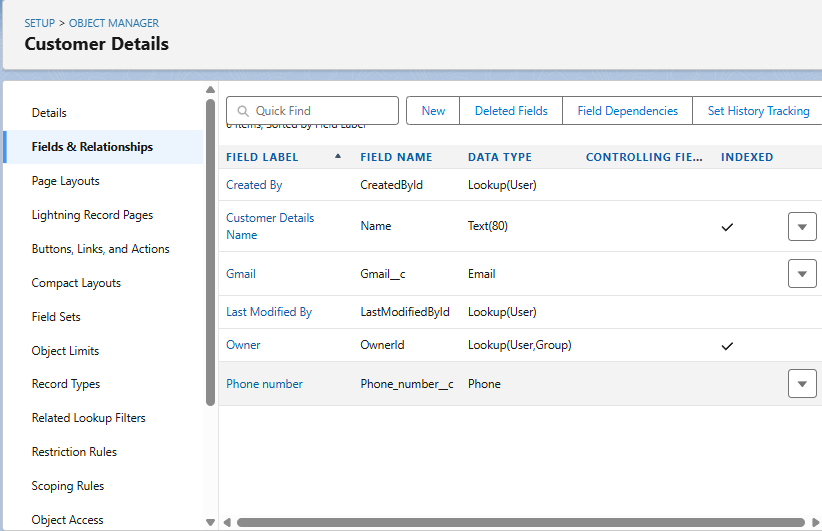
* Clear sprint-level goals and deliverables for the garage system.
* Early feedback and iterative improvements on system features.
* Full completion of planned modules within timeline.
* A functional, tested, and deployable CRM for **Garage Management**.

**6. Project Development Phase**

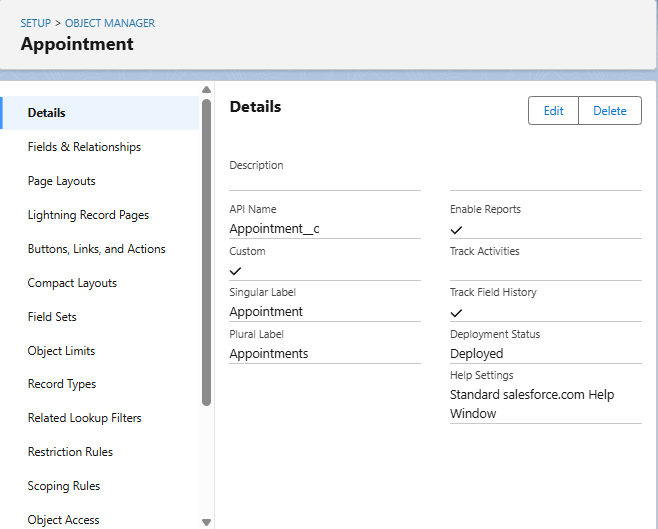
**A. Custom Objects and Their Roles**

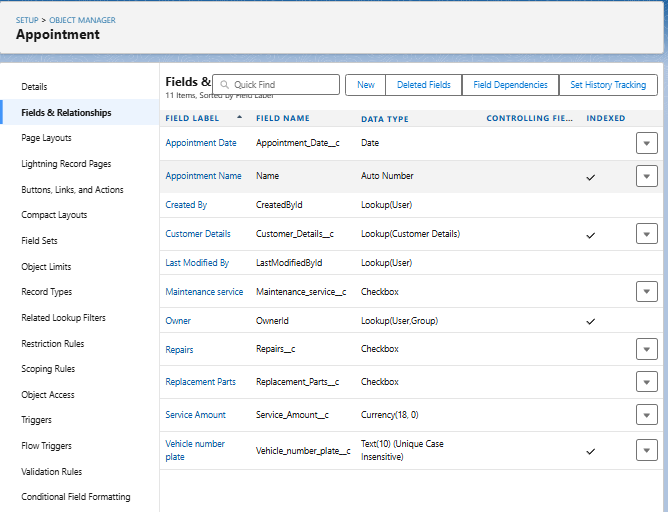
1. **Customer\_Details\_\_c**
   * **Purpose**: Stores comprehensive information about vehicle owners/customers, including their name, contact number, email address, physical address, and any relevant preferences or notes.
   * **Usage:** This object serves as the central point for all customer-related data.
   * **Key Fields :**
     + **Phone\_number\_\_c**
     + **Gmail\_\_c**



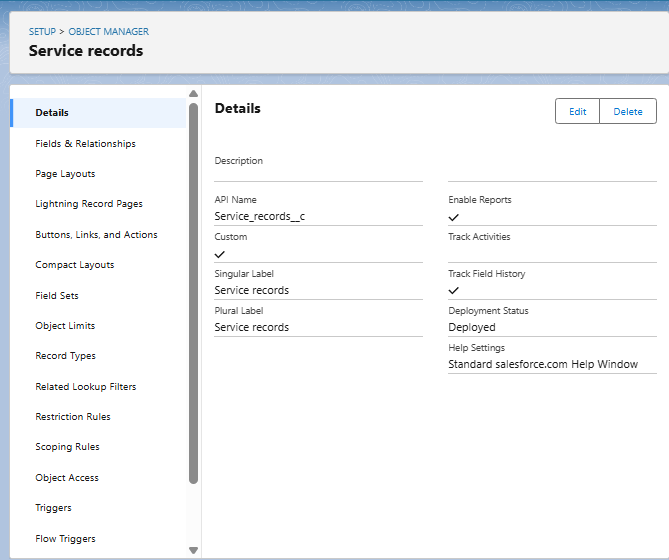
****

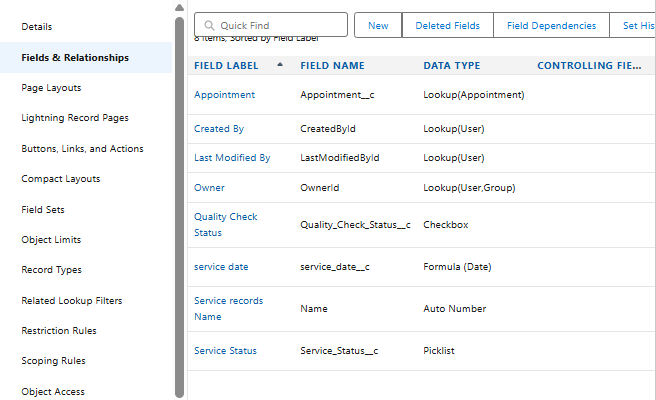
1. **Appointment\_\_c**
   * **Purpose**: To manage and track scheduled service appointments for vehicles. This object records the date, time, type of service requested, and links to the customer and their vehicle, enabling efficient scheduling and resource allocation for the garage.
   * Usage: This object is crucial for front-desk operations and service advisors to manage the garage's schedule
   * Key Fields:
     + Customer\_details\_\_c (lookup)
     + ­Maintenance\_service\_\_c
     + Repairs\_\_c
     + Replacement\_parts\_\_c
     + Appointment\_Date\_\_c
     + Service\_Amount\_\_c
     + Vechile\_number\_plate\_\_c



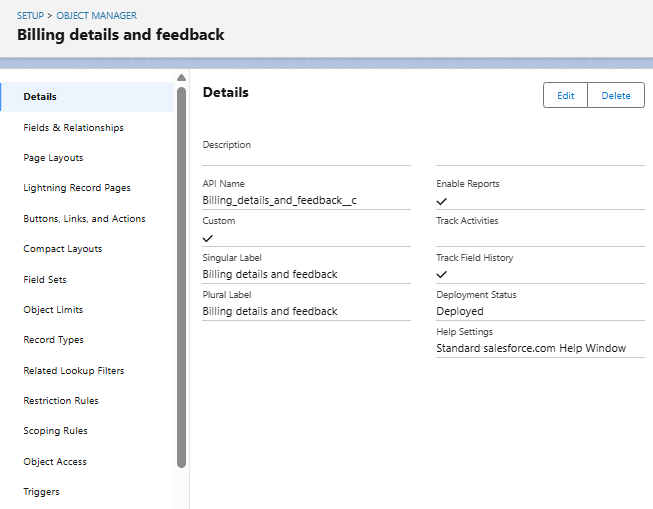
****.

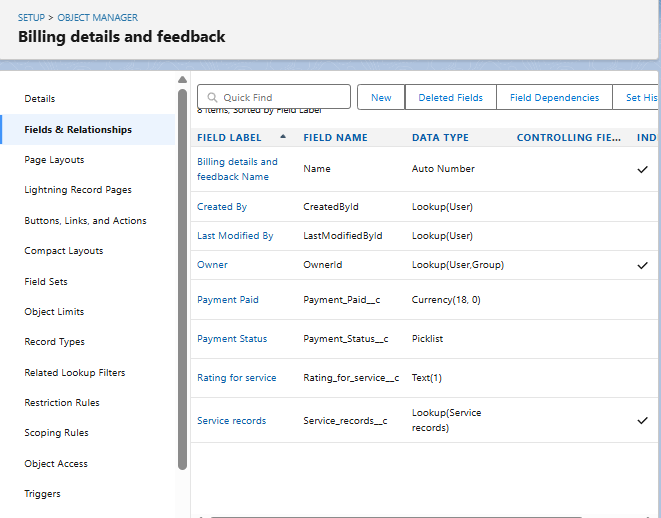
1. **Service\_records\_\_c**
   * Purpose: To maintain a comprehensive historical log of all services performed on a specific vehicle.
   * **Usage:** This object is primarily used for historical tracking, reporting on vehicle service history, and supporting customer inquiries about past work.
   * Key Fields:
     + Quality\_Check\_Status\_\_c
     + Appointment\_\_c (lookup)
     + Service\_Status\_\_c
     + Service\_date\_\_c





1. **Billing\_details\_and\_feedback\_\_c**
   * Purpose: This object serves a dual role: primarily to store the detailed financial aspects of a service (invoice lines, amounts, payment status) and secondarily to capture customer feedback related to that specific service or billing experience.
   * **Usage:** This object is used by billing clerks and service advisors to track financial transactions and customer payment status.
   * Key Fields:
     + Service\_records\_\_c (lookup)
     + Payment\_Paid\_\_c
     + Rating\_for\_service\_\_c
     + Payment\_Status\_\_c



****

**B. AUTOMATION ELEMENTS**

The **Garage Management System** leverages Salesforce's automation capabilities to streamline business processes, reduce manual interventions, and enhance operational accuracy. The automation comprises Apex Triggers, Record-Triggered Flows, and Validation Rules, all tailored to meet the critical needs of an **automotive service and repair operations**.

**1. Apex Trigger –**

* **Name:** AmountDistributionHandler
* **Trigger Type:** Before Insert, Before Update
* **Description:**  
  This use case works for Amount Distribution for each Service the customer selected for the Vehicle
* **Core Logic:**

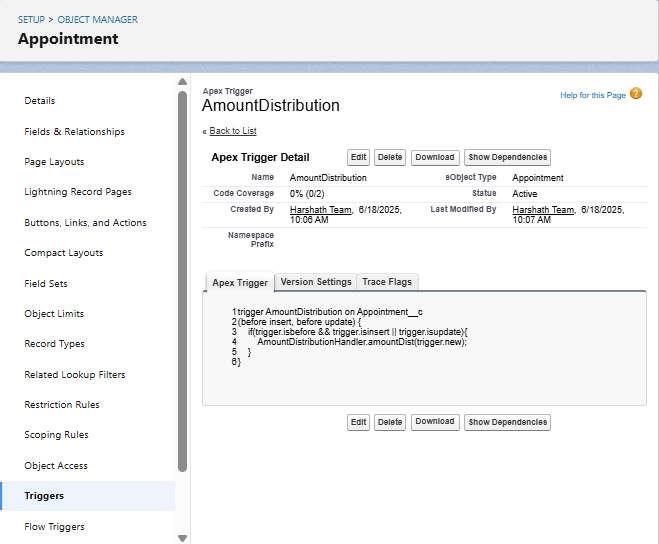
trigger AmountDistribution on Appointment\_\_c (before insert, before update) {

     if(trigger.isbefore && trigger.isinsert || trigger.isupdate){

         AmountDistributionHandler.amountDist(trigger.new);

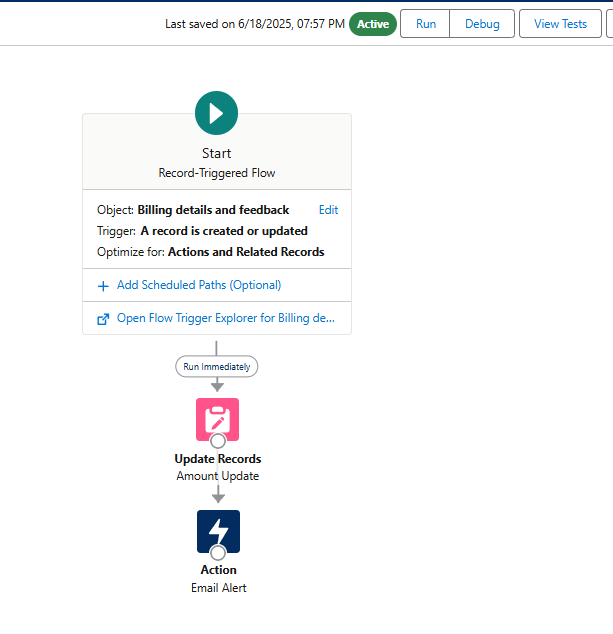
    }

}



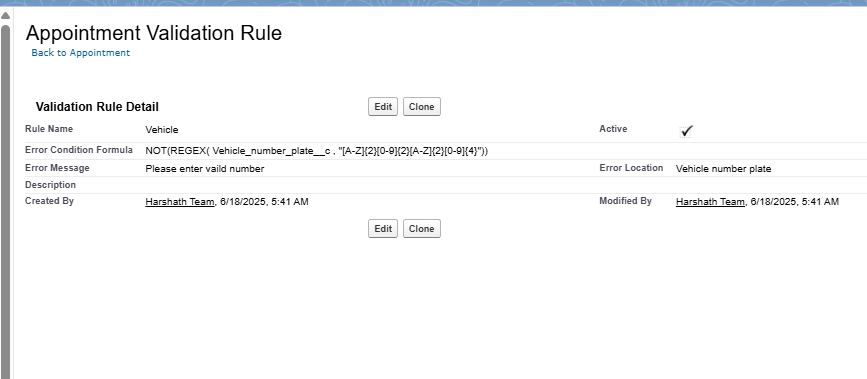
**2. Record-Triggered Flow –**

* **Flow Name:** Billing\_Account\_Flow
* **Trigger Event:** On Create or Update of a **Billing\_details\_and\_feedback\_\_c** record.
* **Flow Description:**  
  This Record-Triggered Flow is designed to automate key communication and data updates immediately after a service invoice is created or updated within the Garage Management System.
* **Email Template Sample Content:**
* 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 .



**3. Validation Rules –**

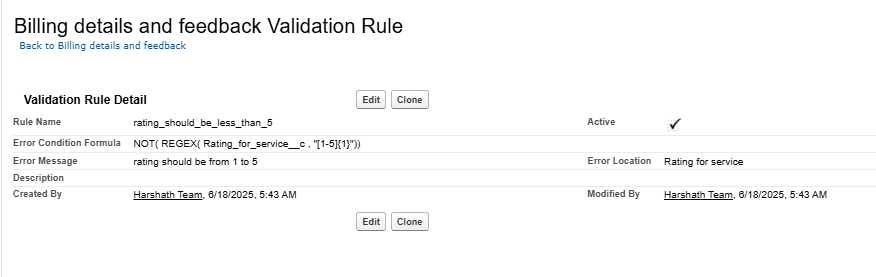
* **Validation Rule Name:** Vehicle
* **Associated Object:** Appointment\_\_c
* **Formula Logic:**
* NOT(REGEX( Vehicle\_number\_plate\_\_c , "[A-Z]{2}[0-9]{2}[A-Z]{2}[0-9]{4}"))
* **Error Message:**  
  "Please enter vaild number ”, select the Error location as Field and select the field as “Vehicle number plate”
* **Purpose:**
  + The purpose of this Validation Rule is to **enforce a specific, standardized format for vehicle number plates** when an **Appointment\_\_c** record is being created or edited. It ensures that the **Vehicle\_number\_plate\_\_c** field adheres to a predefined pattern, thereby **improving data quality and consistency** within the Garage Management System. This is critical for accurate vehicle identification, searching, and legal compliance.
* **Usage Scenario:**
  + **Data Integrity:**When a service advisor or any user attempts to save an Appointment\_\_c record (or any record where Vehicle\_number\_plate\_\_c is present and this rule is active), Salesforce will automatically check the value entered in the Vehicle\_number\_plate\_\_c field against the specified regular expression (REGEX) pattern.
  + **Error Prevention:** If the entered vehicle number plate does not match the pattern



* **Validation Rule Name:** rating\_should\_be\_less\_than\_5
* **Associated Object:** Billings\_details\_and\_feedback\_\_c
* **Formula Logic:**

NOT( REGEX( Rating\_for\_service\_\_c , "[1-5]{1}"))

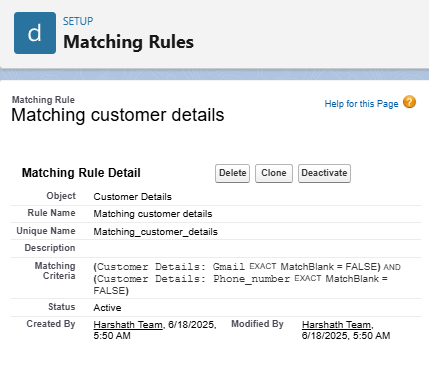
* **Error Message:**  
  " rating should be from 1 to 5”, select the Error location as Field and select the field as “Rating for Service”
* **Purpose:**
  + The purpose of this Validation Rule is to **ensure that the Rating\_for\_service\_\_c field, which captures customer feedback on service quality, only accepts numerical input between 1 and 5 (inclusive).** This is crucial for maintaining the integrity and consistency of customer satisfaction data, enabling accurate analysis and reporting of service performance.
* **Usage Scenario:**
  + **Data Quality Enforcement:** When a user (e.g., service advisor, or even through an external customer survey integration if applicable) attempts to save a Billing\_details\_and\_feedback\_\_c record, this rule validates the Rating\_for\_service\_\_c field.
  + **Preventing Invalid Input:** If the entered value for Rating\_for\_service\_\_c is anything other than a single digit from 1 to 5 (e.g., "0", "6", "abc", or even "12"), the system will prevent the record from being saved



**4. Duplication Rules –**

* **Matching Rule Name:** Matching Customer details
* **Associated Object:** Customer\_details\_\_c
* **Matching Criteria:**

|  |  |
| --- | --- |
| **Field** | **Matching method** |
| Gmail | Exact |
| Phone Number | Exact |

****

**C. UI COMPONENTS**

**A. Reports Configuration**

**Use Case:**

Monitor daily, weekly, or monthly revenue from services and parts. Analyze profitability by service type or mechanic.

**Activity 1: Create and Share a Report Folder**

**Description:** Set up a dedicated report folder and configure its sharing permissions for relevant user roles.

**Main Steps:**

1. Navigate to the **Reports** tab and click **"New Folder"**.
2. Label it "Garage Management Folder" and **Save**.
3. From the "All Folders" view, select the folder's dropdown, click **"Share"**, choose "Roles" -> "Manager", set "View" access, and **Share**.

**Activity 2: Create Custom Report Type**

**Description:** Define a custom report type to connect Customer\_\_c with Appointment\_\_c, Service\_Record\_\_c, and Billing\_details\_and\_feedback\_\_c for comprehensive service reporting.

**Main Steps:**

1. Go to **Setup** > **Report Types**.
2. Click **"New Custom Report Type"**.
3. Set **Primary Object** as Customer\_\_c, label it "Service Information", set category to "Other Reports", and deploy.
4. Relate Appointment\_\_c, Service\_Record\_\_c, and Billing\_details\_and\_feedback\_\_c as secondary objects and **Save**.

**Activity 3: Create Necessary Records for Testing**

**Description:** Populate your Salesforce Org with sample data across key objects to enable robust report testing.

**Main Steps:**

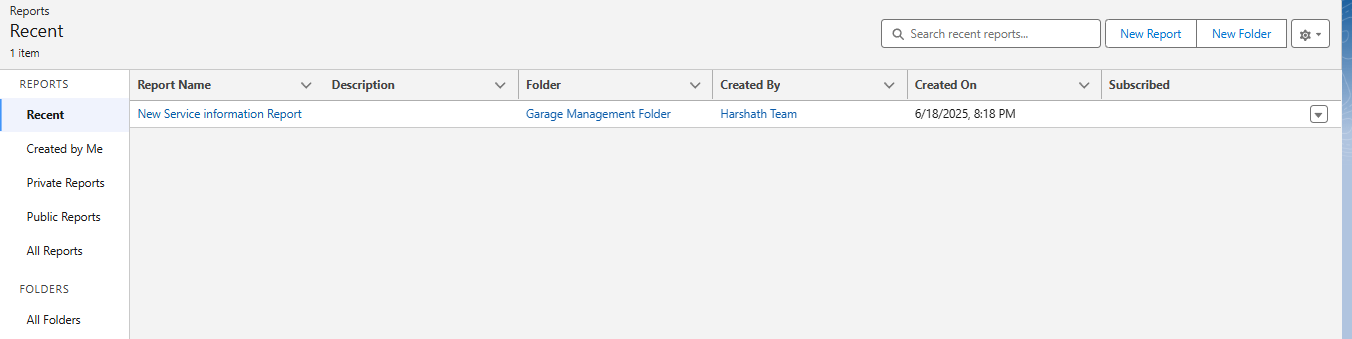
1. Create at least **10 records for each** of your main custom objects: Customer\_\_c, Vehicle\_\_c, Appointment\_\_c, Service\_Record\_\_c, and Billing\_details\_and\_feedback\_\_c.
2. Ensure you **fill in all relevant fields** for each record with varied data to maximize report richness.

**Activity 4: Create a New Report and Visualize Data**

**Description:** Generate a new report using the custom report type, select key fields, group data for analysis, and add a visual chart.

**Main Steps:**

1. Go to the **Reports** tab and click **"New Report"**.
2. Select the "Service Information" report type from the "Other Reports" category.
3. Add Customer Name, Appointment Date, Service Status, and Payment Paid to **Columns**.
4. Group rows by Rating for Service and Payment Status.
5. Add a **Line Chart**, then **Save** the report as "New Service Information Report" to the "Garage Management Folder".



**D. Dashboards**

**Use Case:**

Displays total monthly revenue, revenue by service type (e.g., repairs vs. maintenance), average invoice value, and outstanding payments.

**Activity 1: Create and Share a Dashboard Folder**

**Description:** Set up and share a dedicated dashboard folder for garage analytics.

**Main Steps:**

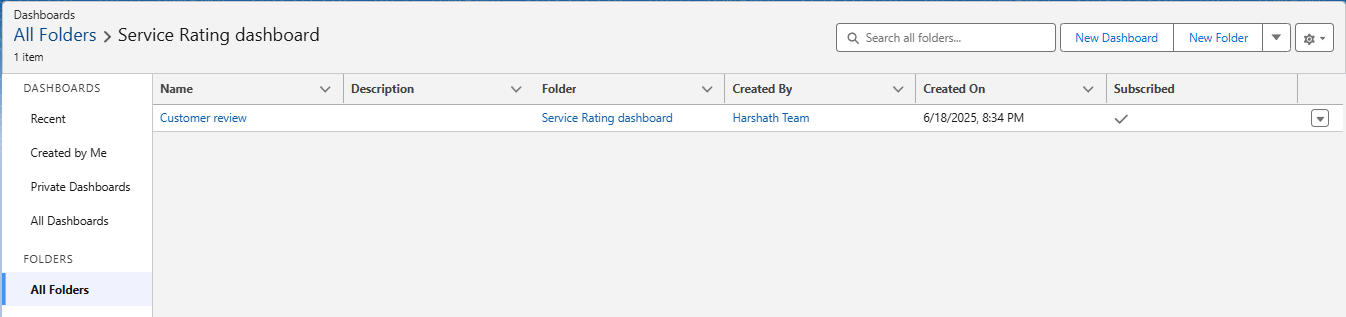
1. **Create Folder:** Go to Dashboards tab > New Folder, name it "Service Rating Dashboard", and Save.
2. **Share Folder:** From "All Folders" dropdown, Share the new folder with "Roles" > "Manager" as "View" access.

**Activity 2: Create a Dashboard and Set Subscription**

**Description:** Build a new dashboard with a report component and schedule weekly email updates.

**Main Steps:**

1. **Create Dashboard:** Go to Dashboards tab > New Dashboard, name it, select the "Service Rating Dashboard" folder, and Create.
2. **Add Component:** Click "Add Component", select your "New Service Information Report", choose "Line Chart", and Add. Save and Done.
3. **Subscribe:** Click "Subscribe" on the dashboard, set Frequency to "Weekly" on "Monday", and Save.



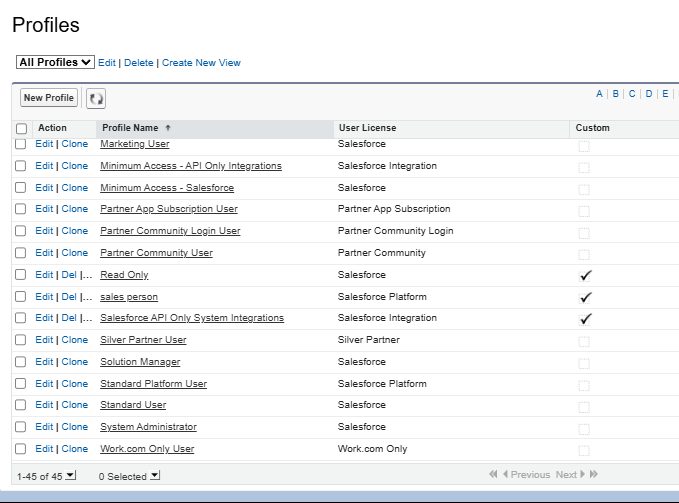
**I. User Profiles and Permission Management**

**Overview:**

Profiles in Salesforce determine the level of access and control that a user has over various features and data within the organization. They define permissions related to:

* Object-level access (e.g., ability to view, create, edit, delete Customer records)
* Field-level access (e.g., whether a user can see or edit a specific field like Total\_Amount\_\_c on an Invoice)
* Tab and App visibility (e.g., whether the "Parts Inventory" tab is visible to a Mechanic)
* Apex class and Visualforce page access
* Record Types, Page Layouts
* Login hours and IP restrictions

To ensure data security and task-based functionality for different users in the **Garage Management System**, standard and custom profiles are configured to align with specific roles.



#### A. Types of Profiles in Salesforce

1. **Standard Profiles** (Predefined, cannot be deleted):
   * System Administrator
   * Standard User
   * Read Only
   * Marketing User
   * Contract Manager
   * Solutions Manager

These come with default permissions and are typically used for basic or administrative users, providing a baseline for access.

1. **Custom Profiles** (User-defined, can be deleted if not in use): Custom profiles are often created to align precisely with organizational roles such as **Service Advisor**, **Mechanic**, or **Parts Manager** within a garage environment. These profiles are tailored to provide granular, role-based access control, ensuring users only see and interact with the data and functionalities relevant to their specific job responsibilities.

#### B. User Creation and Assignment

**Purpose:** Creating user accounts in Salesforce allows employees to log in, access the Garage Management System, and perform their daily tasks according to their assigned permissions. Each user account specifies login credentials, personal information, and crucial access settings via their assigned Role, User License, and Profile.

**Instructions for Creating Users:**

1. Go to **Setup** (the gear icon in the top right).
2. In the **Quick Find** box, type "Users" and select **"Users"** under Administration.
3. Click **"New User"**.

**User 1: Manager**

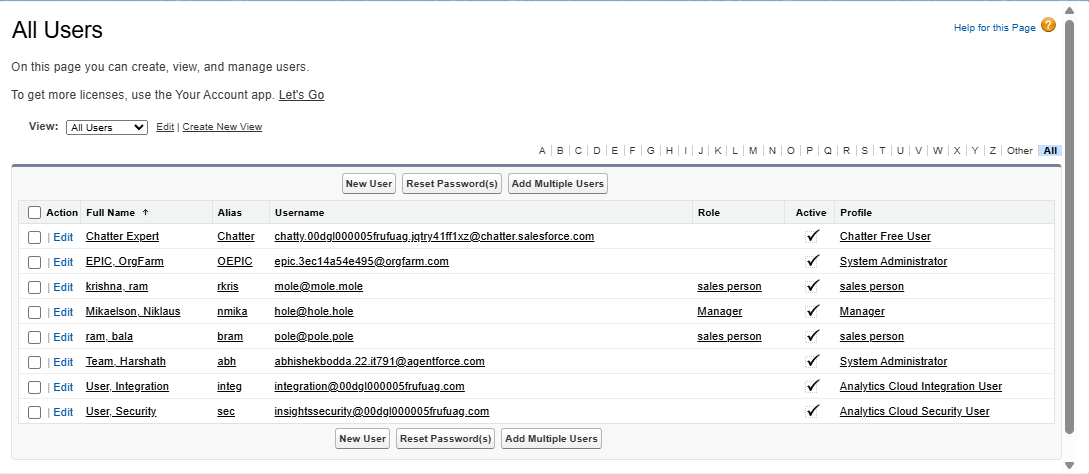
* + **First Name:** Niklaus
  + **Last Name:** Mikaelson
  + **Alias:** (Auto-populate or create one, e.g., NMikael)
  + **Email id:** (Provide your personal email ID for testing)
  + **Username:** (Ensure it's unique, e.g., niklaus.mikaelson@yourgaragem.com)
  + **Nick Name:** (Auto-populate or create one, e.g., NiklausM)
  + **Role:** Manager
  + **User License:** Salesforce
  + **Profile:** Manager
  + Click **"Save"**.



**Creating Additional Users:** Repeat the steps above (Go to Setup > Users > New User) to create at least **3 additional users** with the following permissions. Ensure each user has a unique Username and Nick Name, and provide a valid Email ID for testing purposes.

**Additional User Type: Sales Person**

* + **Role:** Sales Person
  + **User License:** Salesforce Platform (This license typically provides access to custom objects and apps, suitable for a Sales Person role that might not need full standard CRM features)
  + **Profile:** Sales Person

****

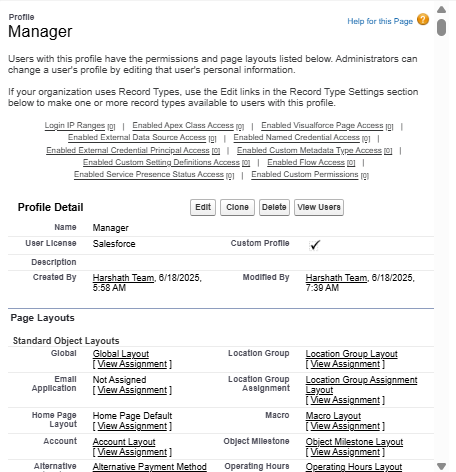
**C. Implementation Activities**

### I. Manager Profile Configuration

**Overview:** Creating a custom "Manager Profile" is a fundamental step in establishing robust security and access control within your Salesforce-based Garage Management System. By cloning a standard profile and customizing it, we ensure that managers have the necessary permissions to oversee all operations, access comprehensive data, and manage key processes, while adhering to specific security policies like session timeout and password expiration.

#### Steps to Create and Configure the Manager Profile:

1. **Create New Profile (Clone Standard User):**
   * Go to **Setup** (the gear icon).
   * In the **Quick Find** box, type "Profiles" and click on **"Profiles"** under Users.
   * Locate the **"Standard User"** profile (or another suitable base profile if preferred for your org's setup).
   * Click on the **"Clone"** link next to the "Standard User" profile.
   * Enter the **Profile Name** as "**Manager**".
   * Click **"Save"**.
2. **Edit Profile Settings:**
   * After saving the new "Manager" profile, you will automatically be directed to its detail page. Click the **"Edit"** button.
3. **Set Default Custom App:**
   * Scroll down to the **"Custom App Settings"** section.
   * Locate the **"Garage Management"** custom app.
   * Select the **"Default"** checkbox next to it. This ensures that when users assigned this profile log in, they automatically land in the Garage Management System application.
4. **Configure Custom Object Permissions:**
   * Scroll down to the **"Custom Object Permissions"** section.
   * For the following custom objects, grant **full access** (Read, Create, Edit, Delete, and potentially View All/Modify All if appropriate for a Manager role):
     + **Appointment\_\_c**
     + **Billing\_details\_and\_feedback\_\_c**
     + **Service\_Record\_\_c**
     + **Customer\_\_c**
     + **(Ensure all other relevant custom objects, like Vehicle\_\_c, WorkOrder\_\_c, PartsInventory\_\_c, also have appropriate access for a Manager)**
5. **Adjust Session Settings:**
   * Scroll down to the **"Session Settings"** section (often located towards the bottom, or under System Permissions depending on Salesforce version).
   * Find "Session Timeout" or "Timeout Value".
   * Change the "Timeout Value" or "Session times out after" setting to "**8 hours of inactivity**".
6. **Configure Password Policies:**
   * Scroll down to the **"Password Policies"** section.
   * Set **"User passwords expire in"** to "**Never expires**". (Note: While convenient for testing/development, "Never expires" is generally not recommended for production environments due to security best practices).
   * Set **"Minimum password length"** to "**8**".
   * Click **"Save"** at the top or bottom of the page to apply all changes to the Manager profile.



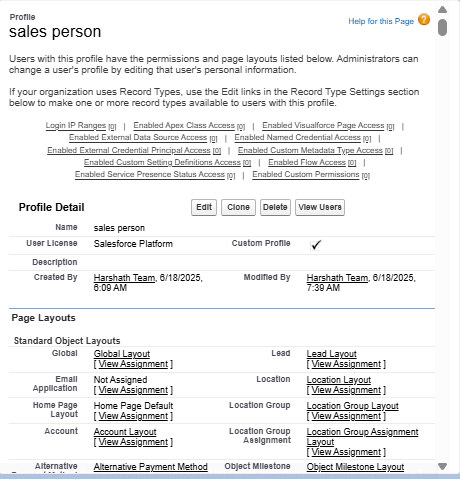
### II. Sales Person Profile Configuration

**Overview:** The "Sales Person Profile" is designed to provide users focused on initial customer engagement, appointment scheduling, and basic customer/vehicle record management with the precise level of access they need within the Garage Management System. This profile ensures that sales personnel (who might function more like Service Advisors in this context) can perform their core duties without access to sensitive administrative functions or full financial data, thereby maintaining data security and operational focus.

#### Steps to Create and Configure the Sales Person Profile:

1. **Create New Profile (Clone Salesforce Platform User):**
   * Go to **Setup** (the gear icon).
   * In the **Quick Find** box, type "Profiles" and click on **"Profiles"** under Users.
   * Locate the **"Salesforce Platform User"** profile. This license/profile combination is often suitable for users who primarily interact with custom applications and objects, like those built for a Garage Management System.
   * Click on the **"Clone"** link next to the "Salesforce Platform User" profile.
   * Enter the **Profile Name** as "**Sales Person**".
   * Click **"Save"**.
2. **Edit Profile Settings:**
   * After saving the new "Sales Person" profile, you will be directed to its detail page. Click the **"Edit"** button.
3. **Set Default Custom App:**
   * Scroll down to the **"Custom App Settings"** section.
   * Locate the **"Garage Management"** custom app.
   * Select the **"Default"** checkbox next to it. This ensures that users assigned this profile automatically enter the Garage Management System application upon login.
4. **Configure Custom Object Permissions:**
   * Scroll down to the **"Custom Object Permissions"** section.
   * For the following custom objects, grant access permissions that align with a sales person's typical responsibilities in a garage (focused on customer interaction, scheduling, and initial record creation, with limited access to sensitive data):
     + **Appointment\_\_c:** Grant **Read** and **Create** access. **Edit** may also be granted if they manage rescheduling, but **Delete** should typically be restricted.
     + **Billing\_details\_and\_feedback\_\_c:** Grant **Read** access. Sales persons might need to view billing history but should generally not be able to create or edit billing details.
     + **Service\_Record\_\_c:** Grant **Read** access. They need to see past service history for customer context but don't create or edit these completed records.
     + **Customer\_\_c:** Grant **Read**, **Create**, and **Edit** access. This is crucial for managing customer information. **Delete** should typically be restricted.
     + **(Ensure other relevant custom objects, like Vehicle\_\_c and ServiceRequest\_\_c, also have appropriate access for a Sales Person, typically Read/Create/Edit)**
5. **Save Profile Changes:**
   * Click **"Save"** at the top or bottom of the page to apply all changes to the Sales Person profile.

(Note: The provided instructions did not specify changes to session timeout or password policies for the "Sales Person" profile. Therefore, these settings would remain at their default values for the cloned "Salesforce Platform User" profile, or would need to be explicitly configured if different from the default.)



**D. Role Hierarchy and Record-Level Access Control**

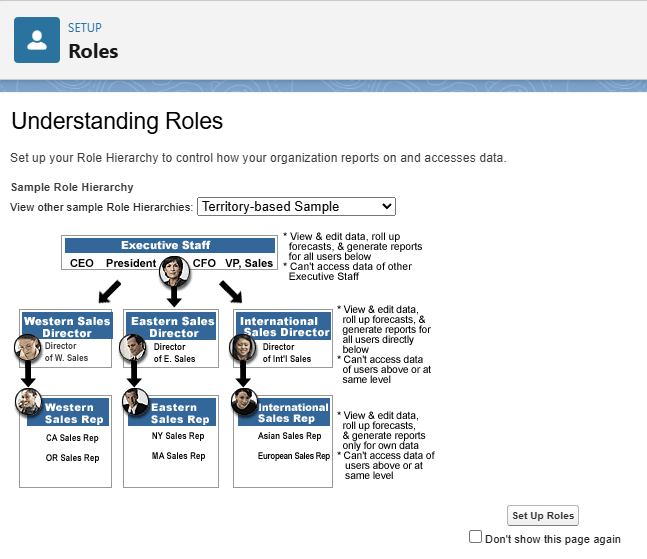
**Overview:** In Salesforce, a **Role** defines a user's record-level visibility access, determining which records a user can see based on their position in the organizational hierarchy. Unlike Profiles, which control object and field access, Roles primarily focus on data visibility (e.g., a manager can see all records owned by their subordinates). Setting up a **Role Hierarchy** is crucial for implementing data sharing rules that ensure users have appropriate access to data within the Garage Management System, promoting collaboration while maintaining data security.

#### A. Creating the Manager Role

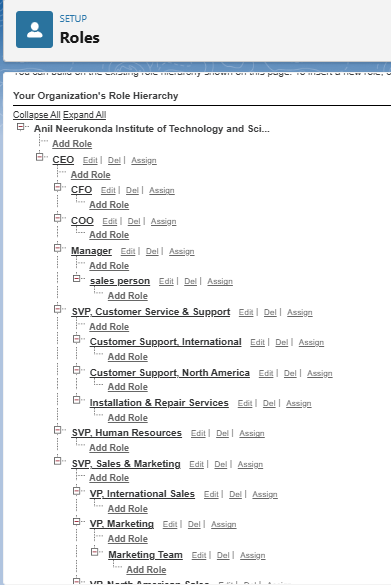
**Purpose:** The "Manager" role will sit below the top-level (e.g., CEO) and will serve as a supervisory role. Users assigned this role will typically have visibility into records owned by users in roles below them in the hierarchy (e.g., Sales Persons, Mechanics, Parts Managers).

**Steps:**

1. **Navigate to Role Setup:**
   * Go to **Setup** (the gear icon).
   * In the **Quick Find** box, search for "Roles" and click on **"Roles"** under Users.
   * Click on the **"Set Up Roles"** button (if it's your first time setting up roles).



1. **Add Manager Role under CEO:**
   * On the Role Hierarchy page, click **"Expand All"** to view the entire hierarchy.
   * Locate the top-level role (typically "CEO" or your organization's highest role).
   * Click on the **"Add Role"** link directly below the "CEO" role (or whichever role you want the Manager to report to).
   * Give the **Label** as "**Manager**".
   * The **Role Name** will auto-populate (e.g., Manager).
   * Click **"Save"**.



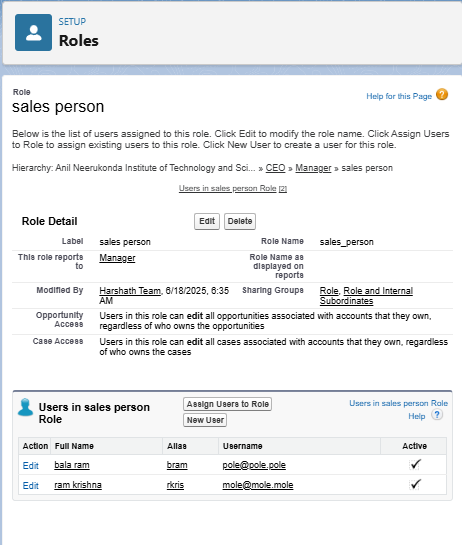
#### B. Creating Additional Roles (Sales Person)

**Purpose:** The "Sales Person" role will be placed directly under the "Manager" role. Users assigned to this role will typically own their individual customer, appointment, and service request records, and their manager will have visibility into these records.

**Steps:**

1. **Navigate to Role Setup (if not already there):**
   * Go to **Setup** (the gear icon).
   * In the **Quick Find** box, search for "Roles" and click on **"Roles"** under Users.
2. **Add Sales Person Role under Manager:**
   * On the Role Hierarchy page, ensure you see the "Manager" role you just created.
   * Click on the **"Add Role"** link directly below the "**Manager**" role.
   * Give the **Label** as "**Sales Person**".
   * The **Role Name** will auto-populate (e.g., Sales\_Person).
   * Click **"Save"**.

**(Note: You would repeat this process for any other roles in your Garage Management System, such as "Mechanic" or "Parts Manager," placing them appropriately within the hierarchy, typically under the "Manager" role, or a more specialized supervisor role if your hierarchy is deeper.)**



### E. Sharing Settings Configuration

**Overview:** 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 by defining the baseline access and then selectively extending it based on organizational structure and specific business needs. Salesforce provides a variety of tools and mechanisms to define and enforce sharing rules, such as:

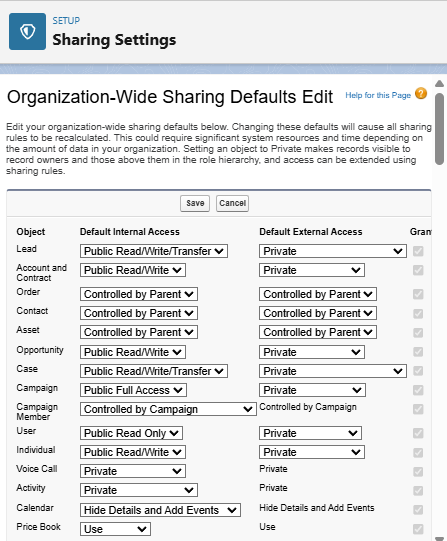
* **Organization-Wide Default (OWD) Settings:** These settings define the default level of access for all records of a specific object within your Salesforce org. OWD settings include Private, Public Read-Only, Public Read/Write, and Controlled by Parent. They can be configured for each standard and custom object, forming the most restrictive baseline.
* **Role Hierarchy:** Salesforce uses a role hierarchy to determine record access. Users at higher levels in the hierarchy (e.g., a Manager) automatically have greater access to records owned by or shared with users lower in the hierarchy (e.g., Sales Persons, Mechanics). 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 (e.g., Read, Create, Edit, Delete on an object) and field-level permissions (e.g., visibility or editability of individual fields) 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 or groups without changing their profile.
* **Sharing Rules:** Sharing rules are used to extend access to records beyond what is granted by OWDs and the role hierarchy, for users who meet specific criteria. They can be used to grant read-only or read-write access to records owned by other users, based on group membership, roles, or criteria on the record itself.
* **Manual Sharing:** Administrators and individual record owners (if granted permission) can manually share specific records with other users, roles, or public groups. This is typically used for ad-hoc sharing of individual records.

#### A. Configuring Organization-Wide Defaults (OWD)

**Purpose:** Setting the OWD for Service\_Record\_\_c to "Private" ensures that, by default, only the owner of a Service\_Record\_\_c record and users higher in the role hierarchy can view or edit that specific record. This establishes the highest level of data privacy for individual service histories, which is a common requirement in sensitive operational data.

**Steps:**

1. Go to **Setup** (the gear icon).
2. In the **Quick Find** box, type "Sharing Settings" and select **"Sharing Settings"** under Security.
3. Click **"Edit"** in the "Organization-Wide Defaults" section.
4. Scroll down to the **Service\_Record\_\_c** object.
5. Change its **Default Access** setting to "**Private**" in the dropdown.
6. Click **"Save"**.
7. Acknowledge the warning message (if any) and click **"OK"**. The changes may take a few moments to refresh across the organization.

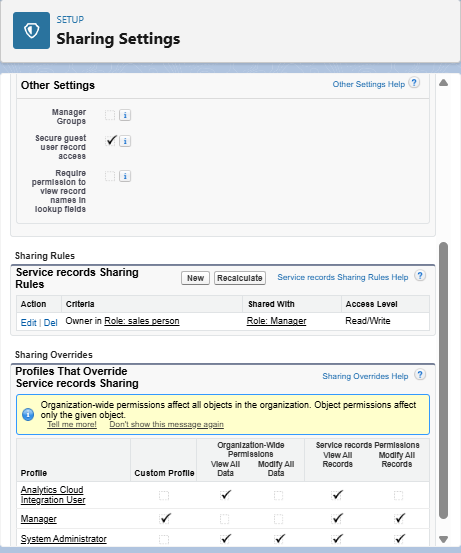


#### B. Creating a Sharing Rule for Service Records

**Purpose:** With Service\_Record\_\_c set to Private, users in the "Manager" role would only see records they own or records owned by users directly below them in the hierarchy. This sharing rule will explicitly grant "Manager" roles "Read/Write" access to all Service\_Record\_\_c records owned by users in the "Sales Person" role, ensuring managers have comprehensive oversight of service activities initiated by their team members, even if the Sales Person doesn't own the final Service Record.

**Steps:**

1. Go to **Setup** (the gear icon).
2. In the **Quick Find** box, type "Sharing Settings" and select **"Sharing Settings"** under Security.
3. Scroll down to the "Service Record Sharing Rules" section.
4. Click **"New"**.
5. **Step 1: Rule Name & Description**
   * Give the **Label** name as "**Sales Person Service Record Access**" (or "Sharing setting" as per your prompt, but a more descriptive name is better practice).
   * The **Rule Name** will auto-populate (e.g., Sales\_Person\_Service\_Record\_Access).
   * Keep "Based on record owner".
6. **Step 2: Select which records to be shared**
   * In the "Select records by owner" dropdown, choose **"Roles"**.
   * In the next dropdown, select "**Sales Person**". This means records owned by users in the Sales Person role.
7. **Step 3: Share with**
   * In the "Share with" dropdown, select **"Roles"**.
   * In the next dropdown, select "**Manager**". This specifies who gains access.
8. **Step 4: Change the access level**
   * Change the **Access Level** for Service\_Record\_\_c to "**Read/Write**".
   * Click **"Save"**.

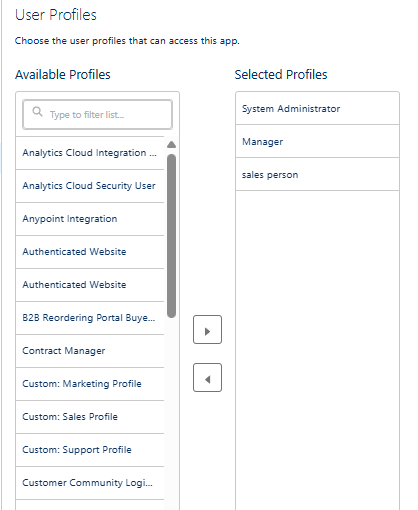
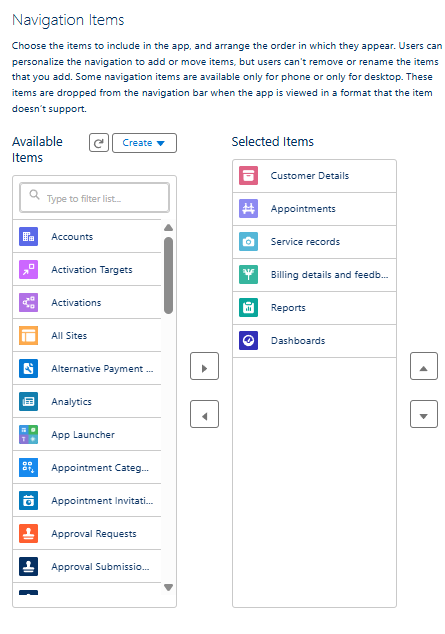
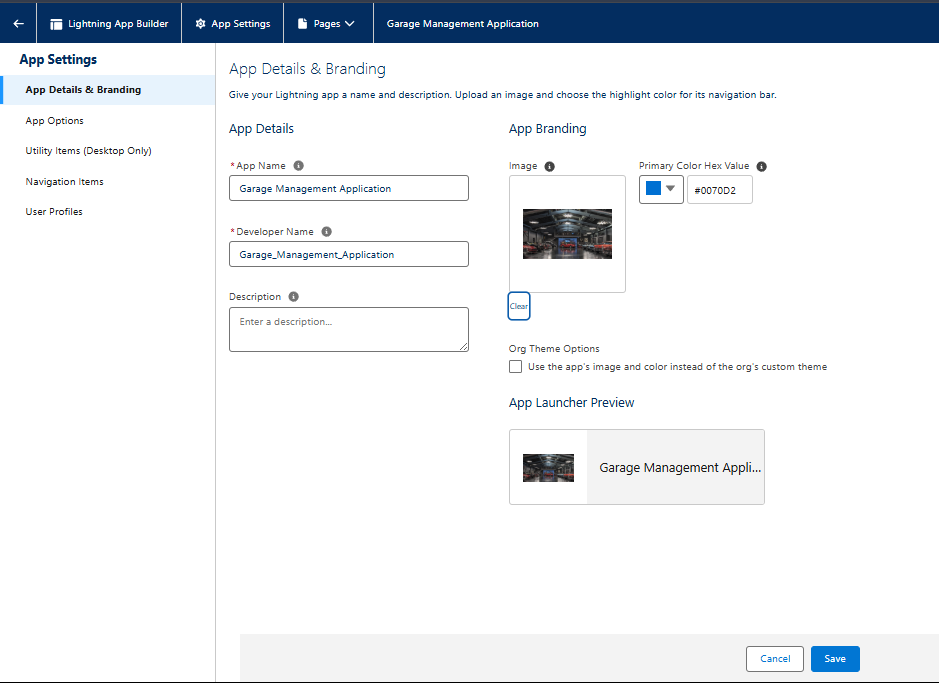


### F. The Lightning App Configuration

**Overview:** A Lightning App is a curated collection of items (such as objects, tabs, and other functionalities) that are bundled together to serve a particular business function. In Lightning Experience, Lightning apps provide users with convenient access to all the tools they need for a specific role or workflow directly within the navigation bar. They allow for custom branding with colors and logos, can include a utility bar for quick access to common tools, and feature Lightning page tabs, enhancing user efficiency by enabling seamless switching between different work contexts.

#### Steps to Create a Lightning App:

1. **Initiate New Lightning App Creation:**
   * Go to **Setup** (the gear icon).
   * In the **Quick Find** box, search for "App Manager" and select **"App Manager"** under User Interface.
   * On the App Manager page, click **"New Lightning App"** in the top right corner.
2. **Define App Details:**
   * **App Name:** Enter "**Garage Management Application**".
   * The Developer Name will auto-populate.
   * (Optionally, upload an App Image and choose a Primary Color for branding).
   * Click **"Next"**.
3. **Configure App Options:**
   * On the "App Options" page, **keep the default settings** (e.g., Navigation Style, Form Factor).
   * Click **"Next"**.
4. **Add Utility Items (Optional):**
   * On the "Utility Items (Desktop Only)" page, **keep it as default** (no utility items will be added unless you configure them here).
   * Click **"Next"**.
5. **Add Navigation Items (Tabs):**
   * This step defines the main tabs that will appear in your app's navigation bar.
   * From the "Available Items" list on the left, select the following items (or search for them if not immediately visible) and use the **arrow button (>)** to move them to the "Selected Items" list on the right:
     + **Customers** (Customer\_\_c)
     + **Appointments** (Appointment\_\_c)
     + **Service Records** (Service\_Record\_\_c)
     + **Billing Details and Feedback** (Billing\_details\_and\_feedback\_\_c)
     + **Reports**
     + **Dashboards**
   * You can reorder these items using the up/down arrows if desired.
   * Click **"Next"**.
6. **Assign User Profiles:**
   * On the "Assign Profiles" page, search for **"System Administrator"** in the "Available Profiles" list on the left.
   * Click the **arrow button (>)** to move it to the "Selected Profiles" list on the right. This makes the app visible to System Administrators.
   * **(Note:** You will need to come back and edit this app later to add the "Manager" and "Sales Person" profiles once they are fully configured, to ensure they can also access the app.)
   * Click **"Save & Finish"**.

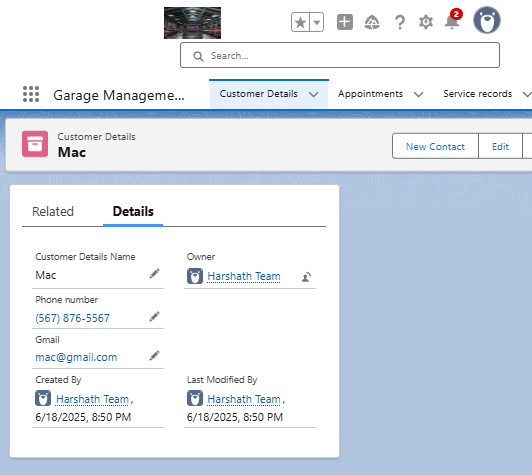


### G. User Adoption - Creating Records

**Overview:** This section provides a practical guide for users to start interacting with the Garage Management System by creating essential records. The goal is to familiarize new users with the system's workflow, data entry forms, and automated processes. By following these steps, users will experience the end-to-end flow of managing a customer, scheduling an appointment, and initiating a service record.

#### Steps to Create Records in the Garage Management Application:

1. **Access the Garage Management Application:**
   * Click on the **App Launcher** (the 3x3 dots icon) located at the left side of the screen.
   * Search for "**Garage Management Application**" and click on it to open the app.
2. **Create a New Customer Record:**
   * Click on the **"Customers"** tab (corresponding to your Customer\_\_c object).
   * Click the **"New"** button.
   * Fill in the customer details as required (e.g., Name, Contact\_No\_\_c, Email\_\_c, Address\_\_c).
   * Click **"Save"**.
3. **Create a New Appointment Record:**
   * Click on the **"Appointments"** tab (corresponding to your Appointment\_\_c object).
   * Click the **"New"** button.
   * **Enter Customer Details:** Link this appointment to the Customer\_\_c record you just created using the Customer\_\_c lookup field.
   * **Enter Appointment Date & Time:** Provide a date for the appointment. (Note: The instruction "enter the date less than the created date" is unusual for a real appointment but might be a specific test case for your validation logic. For standard use, enter a future or current date.)
   * **Enter Vehicle Information:** Ensure the Vehicle\_number\_plate\_\_c (or similar field) matches the validation rule you configured earlier (e.g., "AB01CD2345").
   * **Select Services:** Choose the relevant services required using the Service\_Type\_Requested\_\_c field (e.g., Oil Change, Brake Service).
   * Click **"Save"**. Observe the Service\_Amount\_\_c (if it's a formula field or updated by automation) after saving.



1. **Create a New Service Record:**
   * Click on the **"Service Records"** tab (corresponding to your Service\_Record\_\_c object).
   * Click the **"New"** button.
   * **Enter Appointment Details:** Link this service record to the Appointment\_\_c record you just created using the Appointment\_\_c lookup field.
   * Observe that the Started\_\_c status (or similar field indicating work has begun) is automatically selected as default (if configured via automation or default field value).
   * Click **"Save"**.
2. **Update Service Record for Quality Check Completion:**
   * After saving the Service\_Record\_\_c, open the record detail page if it's not already open.
   * Locate the **Quality\_Check\_Status\_\_c** field (or similar field indicating completion of quality checks).
   * Change its value to **"True"** (or mark it as 'Completed').
   * Click **"Save"**.
   * **Observation:** Notice that upon saving, the Service\_Status\_\_c (or a similar field like WorkOrder\_Status\_\_c on a related object) will automatically update to "**Completed**" (this demonstrates a workflow rule or flow automation at work).

### 7. FUNCTIONAL AND PERFORMANCE TESTING

The **CRM Application for Garage Management System** was rigorously tested to ensure functional correctness, automation reliability, and system performance. Both positive and negative test cases were executed to validate each feature and ensure smooth operation across all custom-built components.

#### A. FUNCTIONAL TESTING

We used a black-box testing approach, focusing on verifying the user-facing behavior of flows, triggers, validation rules, object relationships, and UI components without inspecting the underlying code.

**1. Positive Test Cases**

|  |  |  |  |
| --- | --- | --- | --- |
| Module | Test Case Description | Expected Outcome | Result |
| Customer | Create new Customer record with valid details (Name, Contact No, Email) | Record saved successfully | Pass |
| Vehicle | Create Vehicle record, link to Customer, and assign Make, Model, VIN | Lookup fields link to customer, record saved successfully | Pass |
| Appointment | Create Appointment with valid Customer, Vehicle, and Date/Time | Appointment record created with correct relationships | Pass |
| Work Order | Create Work Order linked to an Appointment/Vehicle and assign Mechanic | Work Order created with correct relationships and status | Pass |
| Parts Inventory | Add new Part with valid stock, cost, and reorder level | Part record saved correctly | Pass |
| Flow | Trigger service cost auto-calculation flow based on work order fields (parts, labor) | Service Cost auto-calculated correctly on Billing\_details\_and\_feedback\_\_c | Pass |
| Trigger | Assign a qualified and available Mechanic to a Work Order | Assignment successful; record saved | Pass |
| Reports | Run "Services by Type" report | Accurate service data displayed | Pass |

**2. Negative Test Cases**

|  |  |  |  |
| --- | --- | --- | --- |
| Module | Test Case Description | Expected Outcome | Result |
| Work Order | Leave Vehicle lookup empty when creating a Work Order | Save blocked with required field error | Pass |
| Trigger | Assign an unqualified or unavailable Mechanic to a Work Order | Validation triggered; save blocked with error message | Pass |
| Flow | Create Work Order with parts not found in Parts\_Inventory\_\_c | Parts cost remains 0 or marked as invalid; flow handled gracefully | Pass |
| Access Control | Mechanic tries to access Billing\_details\_and\_feedback\_\_c for editing | Access denied based on Profile | Pass |
| Email | Enter malformed email in Customer record (for automated notifications) | Email action skipped or failed gracefully | Pass |

#### B. TRIGGER AND FLOW VALIDATION

**1. Apex Trigger – Validate Mechanic Assignment**

* **Trigger Name:** ValidateMechanicAssignmentTrigger
* **Test Class Coverage:** 100%
* **Validation Performed:**
  + Prevents assigning a Mechanic to a WorkOrder\_\_c if they lack the required skills or are marked as unavailable.
  + Handles null Mechanic assignments gracefully without error.
  + Confirmed consistent behavior across insert and update operations for WorkOrder\_\_c.
* **Result:** Trigger passed all scenarios and handled edge cases as expected.

**2. Record-Triggered Flow – Service Cost Auto-Calculation**

* **Flow Name:** AutoCalculateServiceCostFlow
* **Entry Criteria:** On creation or update of WorkOrder\_\_c or Billing\_details\_and\_feedback\_\_c records.
* **Test Scenarios:**
  + Service\_Amount\_\_c fetched and calculated based on Parts\_Consumed\_\_c and Labor\_Hours\_\_c from the WorkOrder\_\_c.
  + Flow correctly executed only when necessary data (parts, labor rates) exists.
  + No crash on missing part/rate record – flow handled via decision logic to prevent errors.
  + Debug logs confirmed correct execution sequence and field updates.

#### C. REPORT AND DASHBOARD VERIFICATION

* **Tools Used:** Report Builder, Dashboard Builder
* **Reports Tested:**
  + Services by Type and Mechanic
  + Revenue by Service
  + Parts Consumption Report
* **Dashboards Validated:**
  + Live summary of active work orders, total daily/weekly revenue, mechanic utilization, and parts stock levels.
  + Visual charts correctly displayed KPIs and were filterable by date, service type, etc.
* **Performance:**
  + Reports loaded in under 2 seconds for <200 records.
  + Dashboards updated dynamically with sample test data.

#### D. ROLE-BASED ACCESS TESTING

We confirmed access restrictions based on custom Profiles and Role Hierarchy as follows:

|  |  |  |
| --- | --- | --- |
| Profile | Expected Access | Test Result |
| Admin | Full access to all custom objects and records | Pass |
| Service Advisor | Access limited to Customer, Vehicle, Appointment, Service Request, Billing details | Pass |
| Mechanic | Access limited to Work Orders assigned to them, and read-only for Vehicle history | Pass |
| Unauthorized User | No access to any CRM modules | Pass |

* Used Permission Sets to temporarily grant extra access to certain users (e.g., Parts Manager to Parts\_Inventory\_\_c).
* Verified field-level security and page layout visibility per profile.

#### E. LOAD & PERFORMANCE TESTING (Optional)

While Developer Edition has governor limits, basic load testing was simulated:

* Inserted 150+ records across WorkOrder\_\_c, Vehicle\_\_c, Parts\_Inventory\_\_c, and Billing\_details\_and\_feedback\_\_c.
* Triggers and Flows responded within acceptable time (1–3 sec per operation).
* Reports and dashboards remained responsive. **Observation:** The system is suitable for small-to-medium scale garage businesses.

**Final Result Summary**

|  |  |
| --- | --- |
| Test Category | Status |
| Functional Test Cases | All Passed |
| Negative Test Handling | All Passed |
| Apex Trigger Validation | Verified |
| Flow Logic | Working |
| Access Control | Confirmed |
| Load Test (Simulated) | Acceptable |

### 8. RESULTS

This section presents the final implementation results of the **CRM Application for Garage Management System**. Each screenshot serves as visual evidence of correct configuration and system performance, validating that custom objects, automation, and dashboards function as expected.

#### 8.1 Output Screenshots

Each screenshot is supported by a concise description to highlight its functionality and importance. These visuals verify that the business workflows have been executed correctly through the Salesforce platform.

**1. Customer Record Creation**

* **Description:** Screenshot showing the creation of a new Customer record.
* **Fields Included:** Customer Name, Contact Number, Email, Address
* **Purpose:** Validates successful creation of the Customer\_\_c object and layout configuration. [Screenshot: New Customer Record]

**2. Appointment Booking / Work Order Record**

* **Description:** This screenshot displays a new Appointment or Work Order record being created with correct relationships.
* **Fields Included:** Customer Name (Lookup), Vehicle VIN (Lookup), Service Type, Appointment Date/Time, Assigned Mechanic (Lookup)
* **Purpose:** Confirms functional lookup fields to Customer, Vehicle, and Employee; verifies initial assignment logic. [Screenshot: New Appointment/Work Order Record]

**3. Service Cost Auto-Fill Flow**

* **Description:** A Billing\_details\_and\_feedback\_\_c record where the Service\_Amount\_\_c was automatically populated via a Record-Triggered Flow.
* **Fields Included:** Work Order Lookup, Parts Cost, Labor Cost, Total Amount
* **Purpose:** Demonstrates dynamic cost calculation using automation based on work order details. [Screenshot: Auto-calculated Service Cost on Billing Record]

**4. Trigger Output: Mechanic Skill/Availability Validation**

* **Description:** Validation message displayed when an unqualified or unavailable employee is entered as the Assigned Mechanic on a Work Order.
* **Error Message:** "Assigned Mechanic does not have the required skills or is unavailable for this service."
* **Purpose:** Proves that the Apex Trigger is working and enforcing business logic accurately to prevent improper assignments. [Screenshot: Mechanic Skill Validation Error Message]

**5. Dashboard – “Garage Operations Overview”**

* **Description:** Custom dashboard created using sample data for management overview of garage operations.
* **Components Shown:**
  + Total Services per Month
  + Revenue by Service Type
  + Active Work Orders by Status
  + Parts Stock Level Alert
* **Purpose:** Confirms that dashboards are pulling real-time data from multiple objects (WorkOrder\_\_c, Billing\_details\_and\_feedback\_\_c, Parts\_Inventory\_\_c) to provide key insights for the Garage Manager. [Screenshot: Garage Operations Dashboard]

**6. Reports**

* **Description:** Screenshots of tabular and graphical reports built using the Salesforce Report Builder.
* **Examples:**
  + Services by Vehicle Type and Status
  + Revenue Summary by Mechanic
  + Parts Consumption Report by Work Order
* **Purpose:** Validates analytical capabilities to assist in garage planning, resource allocation, and profitability monitoring. [Screenshot: Sample Garage Reports]

**7. Record Access Control (Profile-Level Security)**

* **Description:** Screenshot of a Mechanic user attempting to access restricted components (e.g., sensitive customer billing data or employee salaries).
* **Result:** Access denied message confirming profile-level restriction.
* **Purpose:** Demonstrates the effectiveness of Profiles and Permission Sets for role-based access control, ensuring data security within the garage system. [Screenshot: Restricted Access Message]

### 9. ADVANTAGES & DISADVANTAGES

The implementation of the **CRM Application for Garage Management** on the Salesforce platform delivered notable improvements in business operations, automation, and decision-making. However, as with any system, there are limitations that may be addressed in future versions for greater impact.

#### Advantages

**1. Centralized Customer, Vehicle, and Service Management**

* Consolidates all data (customers, vehicles, service history, work orders, parts, billing) into a unified CRM system.
* Removes reliance on paper records, fragmented spreadsheets, and disparate local software.
* Facilitates team collaboration among Service Advisors, Mechanics, and Parts Managers through shared access to cloud-based data.

**2. Real-Time Automation through Flows and Triggers**

* Apex Trigger automatically validates mechanic skill and availability upon work order assignment.
* Record-Triggered Flows auto-calculate service costs and send real-time confirmation emails post-appointment or service completion.
* Reduces manual workload, minimizes human error in billing and scheduling, and ensures process consistency.

**3. Secure and Role-Based Access Control**

* Role-specific Profiles (e.g., Garage Manager, Service Advisor, Mechanic) ensure users only access permitted data relevant to their role.
* Protects sensitive customer data, vehicle history, and financial information from unauthorized access.
* Improves compliance and internal governance by enforcing data visibility rules.

**4. Dynamic Dashboards and Reports**

* Real-time dashboards include key metrics:
  + Total Revenue by Service Type
  + Active Work Orders by Status
  + Mechanic Productivity and Utilization
  + Parts Stock Levels and Consumption
* Enables data-driven decisions for resource optimization, service offering adjustments, and profitability analysis without external business intelligence tools.
* Reports assist in performance tracking, inventory control, and strategic service planning.

**5. Scalability and Cloud-Based Access**

* Built on Salesforce, allowing remote and multi-device access (desktop, tablet, mobile) via the cloud.
* Easily supports business expansion (e.g., adding more service bays, introducing new specialized services, opening multiple branch locations).
* Future-ready for growth and potential multi-site operations.

**6. Customizability and Low-Code Admin Tools**

* Garage administrators can maintain and extend functionality using Salesforce's declarative tools like:
  + Flow Builder for process automation
  + Schema Builder for data model adjustments
  + Report Builder for new analytical views
* Reduces the need for developer intervention in everyday customization and maintenance, empowering super-users.

#### Disadvantages

**1. Limited Third-Party Integration (Current Scope)**

* No out-of-the-box integration with external services like:
  + SMS for automated customer reminders (e.g., service due)
  + Direct WhatsApp messaging for customer communication
  + Specific automotive diagnostic tools or repair databases
  + Popular local payment gateways (if not custom-built)
* Limits seamless customer communication channels and automated data flow from external diagnostic systems.

**2. Learning Curve for Non-Technical Users**

* Traditional mechanics and shop workers, especially those less familiar with digital systems, may face difficulty understanding:
  + The standard Salesforce Lightning interface
  + The concept of lookup fields and interconnected layouts
  + Interpreting validation error messages
* Requires initial and ongoing training sessions for smooth onboarding and optimal daily usage by all staff members.

**3. Initial Setup Time and Configuration Overhead**

* Considerable time needed for the initial setup and configuration, including:
  + Creating all custom objects and defining intricate relationships
  + Designing role-specific page layouts and implementing complex automation (Flows, Triggers)
  + Thorough testing for validation rules, edge cases, and automated notification flows
* The high configuration effort might be a challenge for very small independent garages with limited or no dedicated IT support staff.

**4. Licensing Constraints**

* Salesforce licenses are priced per user, which may impact cost-efficiency for:
  + Smaller garages with tight budgets.
  + Shops with a large number of staff who might only require very limited system access.
* Access to certain advanced features (e.g., robust field service management, comprehensive customer self-service portals) may be available only in higher Salesforce editions, incurring additional costs.

### 10. CONCLUSION

The **CRM Application for Garage Management System** was successfully designed, developed, and implemented on the Salesforce platform. This project delivers a centralized, automated solution to streamline the core operations of automotive service centers, such as front desks, service bays, and management offices. It was developed with the objective of replacing traditional paper-based or spreadsheet-driven systems with a cloud-based CRM that supports customer and vehicle tracking, service pricing, mechanic management, and real-time data visibility.

The application makes extensive use of custom objects, record-triggered flows, Apex triggers, validation rules, and reports & dashboards to support a variety of use cases—from assigning mechanics to work orders, to automatically calculating service costs based on parts and labor. The structure promotes data consistency, transparency in operations, and better communication across garage departments.

**Key Outcomes:**

* **Efficient Operations:** Automation of service cost calculations, work order creation, and parts inventory management streamlined daily administrative processes.
* **Improved Customer Experience:** Automation enabled accurate estimates and faster service initiation, improving the vehicle owner's experience indirectly through efficiency.
* **Real-Time Insights:** Reports and dashboards provided stakeholders with up-to-date visibility into service usage, vehicle history, mechanic performance, and revenue collection.
* **Access Control and Data Security:** Profiles and roles ensured that Service Advisors, Mechanics, and Garage Managers accessed only the data relevant to their responsibilities.
* **Expandable Architecture:** The system was structured to support future enhancements such as online appointment booking portals, advanced diagnostic tool integrations, and integrated customer feedback modules.

**Business Impact**

* **For Garage Managers:** Improved visibility over work order assignments, mechanic schedules, and service-specific performance metrics.
* **For Employees (Service Advisors/Mechanics):** Streamlined process to log and view their service details and responsibilities through a structured CRM interface.
* **For System Admins:** Easier configuration and maintenance of pricing structures, service records, and object relationships through Salesforce tools.

The CRM system reduces operational delays and manual calculation errors. It improves the consistency and traceability of every service, parts consumption, and mechanic allocation, leading to better automotive service delivery.

**Learnings from the Project**

1. **Platform Proficiency:** Gained in-depth understanding of Salesforce Lightning tools including Object Manager, Flow Builder, and Apex Trigger development.
2. **Domain-Specific Customization:** Learned how to adapt a generic CRM system to suit the operational and scheduling needs of an automotive service infrastructure.
3. **Effective Collaboration:** Successfully worked in a team using Agile methods—dividing tasks, planning sprints, and holding retrospectives to improve outcomes.
4. **Process Mapping:** Translated real-world garage workflows into structured system designs using ERDs, data flow diagrams, and field relationships.
5. **Automation Strategy:** Mastered combining declarative (Flow, Validation Rules) and programmatic (Apex Trigger) logic to support real-time automation.

**Final Remarks**

This project has demonstrated how Salesforce can be effectively used to transform a traditional, paper-based domain like garage management into a smart, automated, and scalable CRM application. It also gave our team valuable experience in system analysis, Salesforce architecture, team-based agile execution, and user-centric solution design.

The application not only delivered functional efficiency but also showcased the broader potential of CRM technology in improving private-sector operations and customer service quality.

### 12. APPENDIX

The appendix presents technical evidence of the completed implementation for the **CRM Application for Garage Management System**. It includes automation source code, flow details, data handling approach, and reference links. This section serves as documentation support for academic review, demo validation, and deployment analysis.

#### A. Source Code

**Apex Trigger: ValidateMechanicAssignmentTrigger**

Apex

trigger ValidateMechanicAssignmentTrigger on WorkOrder\_\_c (before insert, before update) {

if (Trigger.isInsert || Trigger.isUpdate) {

ValidateMechanicAssignmentHandler.validateMechanicEligibility(Trigger.new);

}

}

**Purpose:**

* Ensures that assigned mechanics possess the required skills or are available for the specific WorkOrder\_\_c.
* Prevents incorrect mechanic-task mapping and enforces backend data integrity.
* Includes null checks and handles conditional logic for custom validations based on employee attributes.

**Flow: AutoCalculateServiceCost\_Flow**

* **Trigger Type:** Record-Triggered Flow
* **Triggered On:** Creation or update of WorkOrder\_\_c or Billing\_details\_and\_feedback\_\_c records.
* **Flow Steps:**
  + **Get Records:** Retrieve matching Parts\_Inventory\_\_c details (based on parts consumed in Work Order) and Service\_Rates\_\_c (based on service type/labor hours).
  + **Decision:** Confirm if all necessary parts and rates exist for calculation.
  + **Assignment:** Automatically calculate and update the Service\_Amount\_\_c and Total\_Amount\_\_c fields on the Billing\_details\_and\_feedback\_\_c record with the corresponding calculated amount.
* **Outcome:** Ensures accurate and consistent service cost values are auto-filled based on pre-defined rates and parts consumption.

#### B. Dataset Handling

* **Manual Data Entry via Lightning App:**
  + No external datasets or file uploads were used. Sample records were created directly in the Salesforce Playground.
* **Test Records Created For:**
  + Customer\_\_c: Includes contact details and basic customer info.
  + Vehicle\_\_c: Includes VIN, make, model, year, and mileage.
  + Employee\_\_c: 10+ entries categorized as Mechanic, Service Advisor, Parts Manager.
  + Appointment\_\_c: Simulates daily appointment schedules.
  + WorkOrder\_\_c: Represents various service jobs linking Customer, Vehicle, and Employee.
  + Parts\_Inventory\_\_c: Contains static part details including stock and cost.
  + Service\_Rates\_\_c: Defines standard labor rates and service pricing.
  + Billing\_details\_and\_feedback\_\_c: Captures invoice and payment details.
* **Testing Purpose:** Sample data enabled verification of triggers, flows, validation rules, and dashboards.

#### C. GitHub Repository

* **Project Source Code and Documentation Files:**
  + GitHub Repository : **https://github.com/AbhishekB2325/Smartinternz.git**
* **Contents:**
  + Apex class and trigger files
  + Flow screenshots and logic steps
  + Project documentation with test case summaries
  + Report templates and dashboard examples

#### D. Project Demo Video

The demo video includes real-time screen recordings demonstrating:

* Creating Customer, Vehicle, and Employee records.
* Booking an Appointment and creating a Work Order.
* Running the service cost automation.
* Validation rule testing (e.g., mechanic assignment, mileage validation).
* Dashboard and report navigation.
* **Demo Video Link:** [**https://drive.google.com/file/d/13vqK7aIs3z20u1\_bysQulK-xpj3J5QZ1/view?usp=drive\_link**](https://drive.google.com/file/d/13vqK7aIs3z20u1_bysQulK-xpj3J5QZ1/view?usp=drive_link)

**E. Final Remarks**

This appendix supports transparency and completeness of project development. Each component — from data objects to automation — was tested and verified for functional and performance accuracy.

**Key points validated:**

* Trigger execution and logic correctness for mechanic assignments.
* Flow automation for service cost auto-filling and notifications.
* Role-based access control and UI usability for all garage staff.
* Report accuracy and dashboard insights for operational and financial monitoring.

This section also serves as a reference for further expansion (e.g., online booking portal, diagnostic integration, customer feedback modules) or integration into production-grade deployments.