

**PROJECT TITLE: SALES AUTOMOBILE USING SALESFORCE CRM**

**COLLEGE NAME:** PARK COLLEGE OF ENGINEERING AND TECHNOLOGY

**COLLEGE CODE:** 7122

**DEPARTMENT:** B. TECH INFORMATION TECHNOLOGY

**TEAM LEADER:**

SATHYA MOORTHI V      712221205033

**TEAM MEMBERS:**

MUKESH G      712221205017

ZIBRIL A      712221205046

DHANACHEZHIYAN P      712221205007

Registered email id: aravindaravind2937@gmail.com

## Abstract:

The Salesforce CRM implementation for automobile sales streamlines the entire sales process, enhancing efficiency and customer satisfaction. Through this system, sales teams can manage leads, track customer interactions, and automate follow-ups.

It enables comprehensive customer profiling, allowing for personalized marketing strategies and targeted campaigns. The platform facilitates inventory management, ensuring real-time updates on available vehicles and their specifications.

Integration with marketing tools enables seamless communication and lead nurturing. Additionally, the system provides insightful analytics, empowering decision-making by identifying sales trends and forecasting demand. Overall, the Salesforce CRM for automobile sales optimizes operations, fosters customer relationships, and drives revenue growth within the automotive industry.

## 1. Introduction

### 1.1 Project Overview

### 1.2 Project Scope

### 1.3 Objectives

### 1.4 Methodology

## 2. CRM System Overview

### 2.1 What is Salesforce CRM?

### 2.2 Key Features of Salesforce for Automotive Sales

### 2.3 Benefits of Salesforce in the Automotive Industry

### 2.4 Salesforce Integration with Automobile Sales Process

## 3. Project Requirements and Setup

### 3.1 System Requirements

### 3.2 Salesforce Account Setup

### 3.3 Customization Needs for Automobile Sales

### 3.4 Integration with Other Tools (e.g., Dealer Management Systems, Lead Generation Systems)

## 4. Data Management

### 4.1 Customer Data Management

#### 4.1.1 Customer Profiles

#### 4.1.2 Lead Tracking and Nurturing

## **4.2 Vehicle Inventory Management**

### **4.2.1 Stock Management**

### **4.2.2 Vehicle Details and Specifications**

## **4.3 Sales Data and Reporting**

### **4.3.1 Tracking Sales Performance**

### **4.3.2 Sales Pipeline Management**

## **4.4 Data Security and Privacy in Automotive CRM**

# **5. Sales Process Automation**

## **5.1 Lead Generation and Capture**

## **5.2 Lead Qualification and Assignment**

## **5.3 Sales Funnel Management**

## **5.4 Salesforce Lightning for Automobile Sales**

## **5.5 Automated Follow-ups and Notifications**

## **5.6 Opportunity Management**

## **5.7 Close and Post-Sale Follow-up**

# **6. Marketing Automation**

## **6.1 Email Campaigns**

## **6.2 SMS Marketing Integration**

## **6.3 Targeted Campaigns for Vehicle Models**

## **6.4 Customer Retargeting**

## **6.5 Lead Scoring and Segmentation**

## **6.6 Social Media Integration for Leads**

# **7. Customer Support and Service**

## **7.1 Customer Service Case Management**

## **7.2 Support Ticket System**

## **7.3 Service Reminders and Follow-ups**

## **7.4 After-Sales Engagement**

## **7.5 Customer Feedback and Satisfaction Surveys**

# **8. Reports and Analytics**

## **8.1 Sales Performance Reports**

## **8.2 Lead Conversion Rates**

## **8.3 Revenue Forecasting**

## **8.4 Customer Retention Analysis**

## **8.5 Vehicle Stock Analysis and Demand Prediction**

## **8.6 Dashboard Customization for Automobile Sales**

# **9. Salesforce Features for Automotive Industry**

## **9.1 Salesforce Mobile App for Sales Teams**

## **9.2 Salesforce Einstein AI for Sales Forecasting**

## **9.3 Salesforce CPQ (Configure, Price, Quote) for Automobile Sales**

## **9.4 Salesforce Communities for Customer Engagement**

## 10 . Customization and Configuration

### 10.1 Custom Fields for Automobile Sales

### 10.2 Automation Rules and Workflows

### 10.3 Page Layouts and UI Customization

### 10.4 Custom Reports and Dashboards

## 11. Integration with Other Platforms

### 11.1 Third-Party CRM Integrations (e.g., ERP Systems)

### 11.2 Payment Gateway Integration

### 11.3 Dealer Management System Integration

### 11.4 Integration with Online Marketplaces (e.g., Car Sales, Autotrader)

## 12. Challenges and Solutions

### 12.1 Challenges in Implementing CRM for Automobile Sales

### 12.2 Data Accuracy and Maintenance

### 12.3 Employee Training and Adoption

### 12.4 Dealing with High Volume Data

## 13. Future Scope

### 13.1 Artificial Intelligence and Predictive Analytics for Auto Sales

### 13.2 Voice Search and Conversational AI for Customer Support

### 13.3 Salesforce's Role in Electric Vehicle (EV) Sales

## 14. Conclusion

### 14.1 Summary of Project Goals

### 14.2 Achievements

### 14.3 Final Recommendations

## 15. Appendices

### 15.1 Glossary of Terms

### 15.2 References

### 15.3 System Diagrams and Architecture

### 15.4 User Guide for Automobile Salesforce CRM

## Salesforce

### Introduction:

#### Sales Automobile Using Salesforce CRM :

The Salesforce CRM implementation for automobile sales streamlines the entire sales process, enhancing efficiency and customer satisfaction. Through this system, sales teams can manage leads, track customer interactions, and automate follow-ups. It enables comprehensive customer profiling, allowing for personalized marketing strategies and targeted campaigns. The platform facilitates inventory management, ensuring real-time updates on available vehicles and their specifications. Integration with marketing tools enables seamless communication and lead nurturing. Additionally, the system provides insightful analytics, empowering decision-making by identifying sales trends and forecasting demand. Overall, the Salesforce CRM for automobile sales optimizes operations, fosters customer relationships, and drives revenue growth within the automotive industry.

Are you new to Salesforce? Not sure exactly what it is, or how to use it? Don't know where you should start on your learning journey? If you've answered yes to any of these questions, then you're in the right place. This module is for you. Welcome to Salesforce! Salesforce is game-changing technology, with a host of productivity-boosting features, that will help you sell smarter and faster. As you work toward your badge for this module, we'll take you through these features and answer the question, "What is Salesforce, anyway?".

## **What Is Salesforce?**

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

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

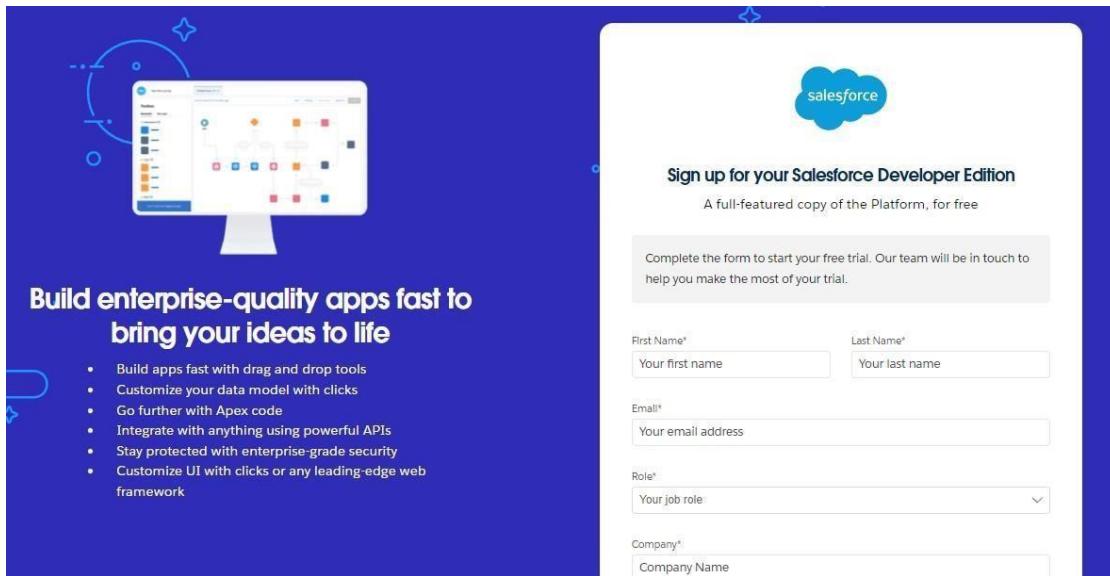
So what does that really mean? Well, before Salesforce, your contacts, emails, follow-up tasks, and prospective deals might have been organised something like this:

<https://youtu.be/r9EX3lGde5k>

## Creating Developer Account

Creating a developer org in salesforce.

1. Go to <https://developer.salesforce.com/signup>
2. On the sign up form, enter the following details :



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

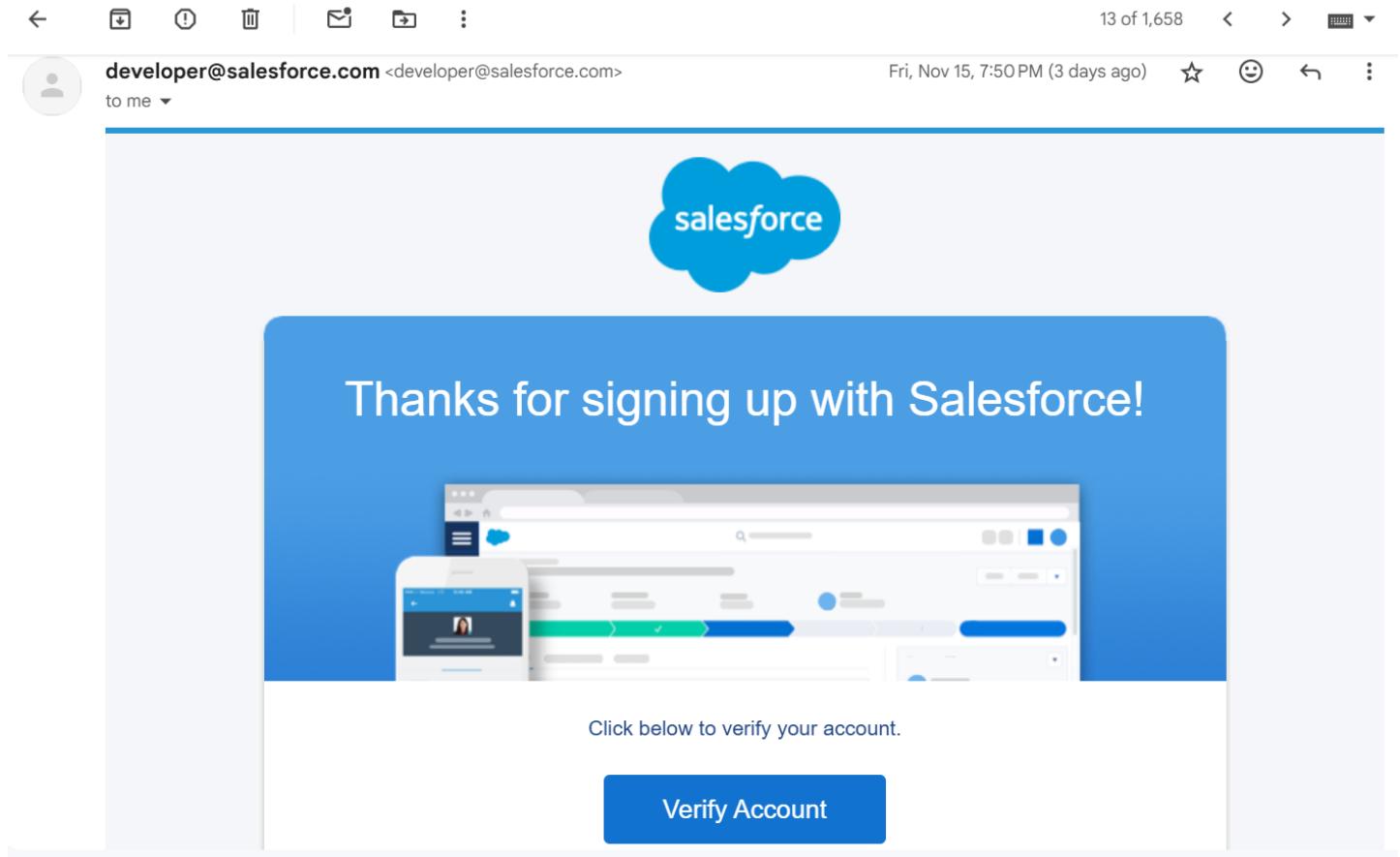
This need not be an actual email id, you can give anything in the format :

username@organization.com

Click on sign me up after filling these.

## Account Activation

1. Go to the inbox of the email that you used while signing up. Click on the verify account to activate your account. The email may take 5-10mins.



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



## Change Your Password

Enter a new password for **salesforce@teamautomobile.com**. Make sure to include at least:

- 8 characters
- 1 letter
- 1 number

\* New Password

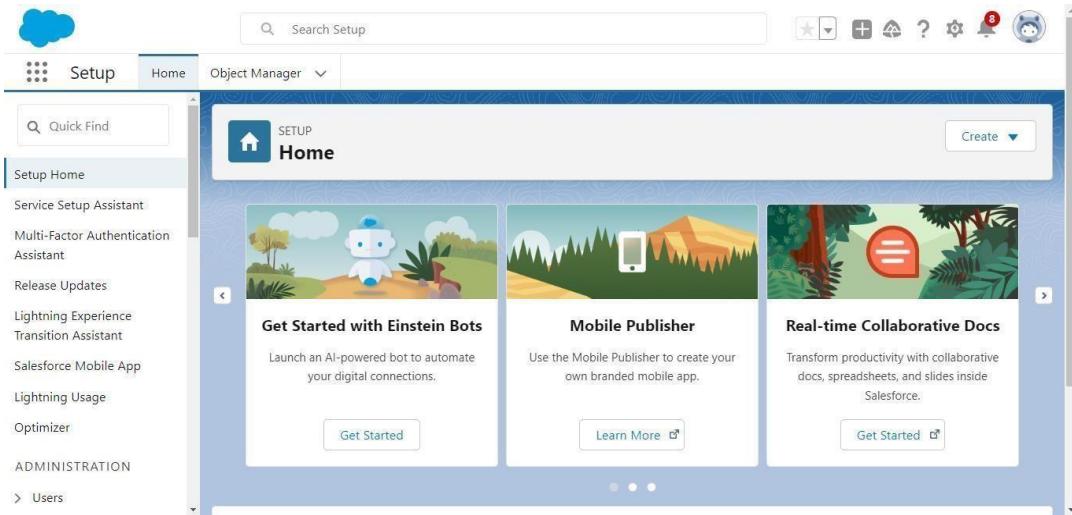
\* Confirm New Password

**Change Password**

Password was last changed on 18/11/2024, 12:58 pm.

© 2024 Salesforce, Inc. All rights reserved.

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



The screenshot shows the Salesforce Setup Home page. The top navigation bar includes a cloud icon, 'Setup' (which is selected), 'Home', 'Object Manager', and various global icons. On the left, a sidebar lists navigation links such as 'Setup Home', 'Service Setup Assistant', 'Multi-Factor Authentication Assistant', 'Release Updates', 'Lightning Experience Transition Assistant', 'Salesforce Mobile App', 'Lightning Usage', 'Optimizer', and 'ADMINISTRATION' (with 'Users' under it). The main content area is titled 'SETUP Home' and features three cards: 'Get Started with Einstein Bots' (Launch an AI-powered bot to automate your digital connections), 'Mobile Publisher' (Use the Mobile Publisher to create your own branded mobile app.), and 'Real-time Collaborative Docs' (Transform productivity with collaborative docs, spreadsheets, and slides inside Salesforce.). Each card has a 'Get Started' button.

## Object

### What Is an Object?

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

**Salesforce objects are of two types:**

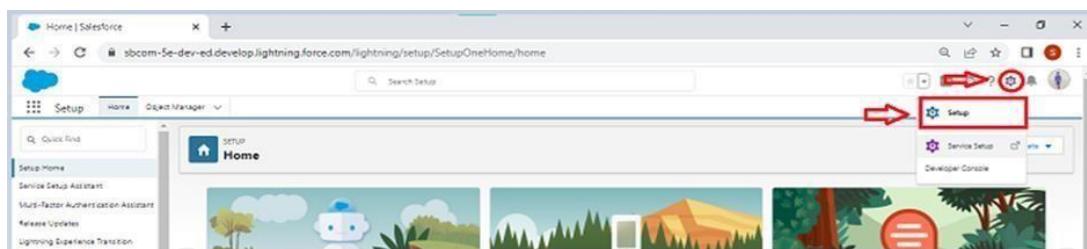
1. **Standard Objects:** Standard objects are the kind of objects that are provided by salesforce.com such as users, contracts, reports, dashboards, etc.
2. **Custom Objects:** Custom objects are those objects that are created by users. They supply information that is unique and essential to their organization. They are the heart of any application and provide a structure for sharing data.

**Use Case:**

Creating an object in Salesforce organization is essential for efficient data management and process automation. By defining custom objects, businesses can structure and store data specific to their needs, enabling streamlined workflows, personalized reporting, and enhanced user experiences. Objects serve as the foundation for organizing and leveraging critical information within Salesforce.

**To Navigate to Setup page:**

Click on gear icon >> click setup.



**To create an object:**

## Tabs

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

Types of Tabs:

### 1. Custom Tabs

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

### 2. Web Tabs

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

### 3. Visualforce Tabs

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

### 4. Lightning Component Tabs

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

## 5. Lightning Page Tabs

Lightning Page Tabs let you add Lightning Pages to the mobile app navigation menu.

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

### **Use Case:**

Creating Objects and storing organization's data is the very first step in the requirements they want. Now to access the stored data by an employee from the organization Admin needs to create Tabs. By designing a dedicated Tab, businesses can improve user experience, simplify navigation, and provide quick access to critical information, enhancing productivity and ensuring efficient utilization of Salesforce's capabilities.

## Create Automobile Information Object

1. Download and open [this spreadsheet](#), save it as AutomobileInformation.csv.
2. Make sure to download the File into CSV format.

Note : Make sure to have the name of the file as “Automobile Information”.

Log into your salesforce account, click , then select Setup.

3. Click the Object Manager tab.



4. Click Create.
5. Select Custom Object from Spreadsheet.



6. Click Login With Salesforce.
7. Enter your Salesforce account username and password. (which you have created in the Milestone 1, Activity 1)
8. Click Log In.
9. Click Allow.
10. Click Upload.

11. Navigate to the Automobile Information.csv file you downloaded and upload it. Salesforce automatically detects the fields and populates all its record data. Choose the Record Name field and make sure all fields are with the proper datatypes as below as they are.

IMPORT FILE FIELD NAME	SALESFORCE FIELD NAME	SALESFORCE FIELD TYPE	ADD TO LAYOUTS	FIELD PREVIEW
Name Of Manufacturer	Name Of Manufacturer	Text	<input checked="" type="checkbox"/>	Toyota
Model	Model	Text	<input checked="" type="checkbox"/>	Corsia
Engine Number	Engine Number	Text	<input checked="" type="checkbox"/>	ABC12345
VIN	VIN	Text	<input checked="" type="checkbox"/>	3HGCMB2623A004252
Total Number of Cylinders	Total Number of Cylinders	Picklist	<input checked="" type="checkbox"/>	4
Colour	Colour	Picklist	<input checked="" type="checkbox"/>	Red
Built date	Built date	Date	<input checked="" type="checkbox"/>	23-09-2022
Price	Price	Number	<input checked="" type="checkbox"/>	500
Quantity	Quantity	Integer	<input checked="" type="checkbox"/>	10

12. Click Next and enter the following settings.

13. Click Finish. The Automobile Information object is successfully created and data imported, all within minutes.

## Create Invoice Object

Create Invoice object, just as we have created an Automobile Information Object using [this sheet](#)

Make sure to Download the File into CSV Format.

Note: Make sure you do field mapping with proper field type as shown below.

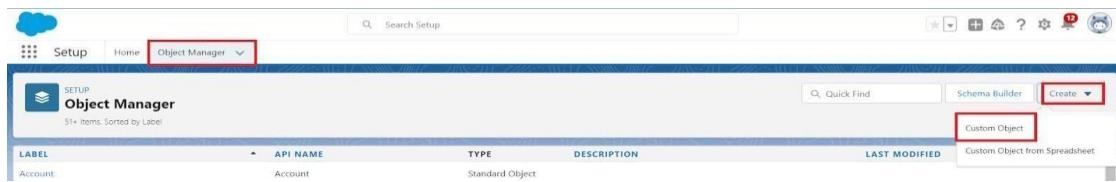
IMPORT FILE FIELD NAME	SALESFORCE FIELD NAME	SALESFORCE FIELD TYPE	ADD TO LAYOUTS	FIELD PREVIEW
Invoice ID	Invoice ID	Text	<input checked="" type="checkbox"/>	
Total Price	Total Price	Text	<input checked="" type="checkbox"/>	
Quantity	Quantity	Integer	<input checked="" type="checkbox"/>	
Unit Price	Unit Price	Text	<input checked="" type="checkbox"/>	
Purchase Date	Purchase Date	Date	<input checked="" type="checkbox"/>	

## Create Automobile Object :

The purpose of creating an Automobile custom object is to store and manage information about Invoice.

To create an object:

1. From the setup page >> Click on Object Manager >> Click on Create >> Click on Custom Object.



1. Enter the label name>> Opportunity Automobile
2. Plural label name>>Opportunity Automobiles

A screenshot of the 'New Custom Object' creation page. The title bar says 'SETUP New Custom Object'. The 'Custom Object Information' section includes fields for 'Label' (Example: Account) and 'Plural Label' (Example: Accounts). There is a checkbox for 'Starts with vowel sound'. The 'Object Name' field is set to 'Account'. The 'Description' field is empty. Under 'Context Sensitive Help Setting', the radio button for 'Open the standard Salesforce.com Help & Training window' is selected. The 'Content Name' dropdown is set to '--None--'. The 'Enter Record Name Label and Format' section includes a note about Record Name appearing in various places. The 'Record Name' field is set to 'Example: Acccunt Name' and the 'Data Type' dropdown is set to 'Text'. Arrows point to the 'Label' and 'Record Name' fields.

3. Enter Record Name Label and Format

- Record Name >> Opportunity Automobile Id
- Data Type >> Auto Number
- Display Format >> OA-{0000}
- Starting Number >> 1

**Custom Object Information**

The singular and plural labels are used in tabs, page layouts, and reports.  
Be careful when changing the name or label as it may affect existing integrations and merge templates.

Label	<input type="text" value="Opportunity Automobile"/>	Example: Account
Plural Label	<input type="text" value="Opportunity Automobiles"/>	Example: Accounts
Starts with vowel sound	<input type="checkbox"/>	

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

Object Name	<input type="text" value="Opportunity_Automobile"/>	Example: Account
-------------	---	------------------

Description

Context-Sensitive Help Setting

<input checked="" type="radio"/> Open the standard Salesforce.com Help & Training window
<input type="radio"/> Open a window using a Visualforce page

Content Name

<input type="button" value="None"/>
-------------------------------------

**Enter Record Name Label and Format**

The Record Name appears in page layouts, key lists, related lists, lookups, and search results. For example, the Record Name for Account is "Account Name" and for Case it is "Case Number".

Record Name	<input type="text" value="Opportunity Automobile Id"/>	Example: Account Name
Data Type	<input type="button" value="Auto Number"/>	
Display Format	<input type="text" value="OA-{0000}"/>	Example: A-{0000} <a href="#">What Is This?</a>
Starting Number	<input type="text" value="1"/>	

2. Click on Allow reports.
3. Allow search
4. Save.

## Tabs

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

Types of Tabs:

### 1. Custom Tabs

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

## **2. Web Tabs**

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

## **3. Visualforce Tabs**

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

## **4. Lightning Component Tabs**

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

## **5. Lightning Page Tabs**

Lightning Page Tabs let you add Lightning Pages to the mobile app navigation menu.

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

## Use Case:

Creating Objects and storing organization's data is the very first step in the requirements they want. Now to access the stored data by an employee from the organization Admin needs to create Tabs. By designing a dedicated Tab, businesses can improve user experience, simplify navigation, and provide quick access to critical information, enhancing productivity and ensuring efficient utilization of Salesforce's capabilities.

## Creating A Custom Tab

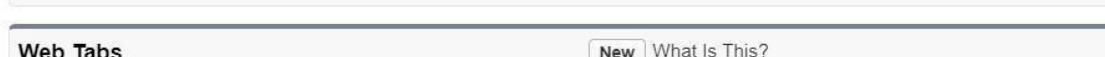
1. Go to setup page >> type Tabs in Quick Find bar >> click on tabs >> New  
(under custom object tab)



## Custom Tabs

You can create new custom tabs to extend Salesforce functionality or to build new application functionality.

Custom Object tabs look and behave like the standard tabs provided with Salesforce. Web tabs allow you to embed external pages. Lightning Component tabs allow you to add Lightning components to the navigation bar, allowing you to add Lightning Pages to Lightning Experience and the mobile app.



2. Select Object(Opportunity Automobile) >> Select any tab style >> Next (Add to profiles page) keep it as default >> Next (Add to Custom App) keep it as default >> Save.

The screenshot shows the 'Custom Object Tab Definition Detail' page for an object named 'Opportunity Automobiles'. The page includes fields for Tab Label (set to 'Opportunity Automobiles'), Object (set to 'Opportunity Automobile'), Description (empty), Created By (set to 'Mohammad Sameer' with a timestamp of '27/11/2023, 2:48 pm'), Tab Style (set to 'Bell'), and Splash Page Custom Link (empty). There are 'Edit' and 'Delete' buttons at the top right, and a 'Help for this' link at the top right.

Note: Tabs for Automobile Information & Invoice objects do get created automatically. We do not need to create tabs for those objects.

## The Lightning App :

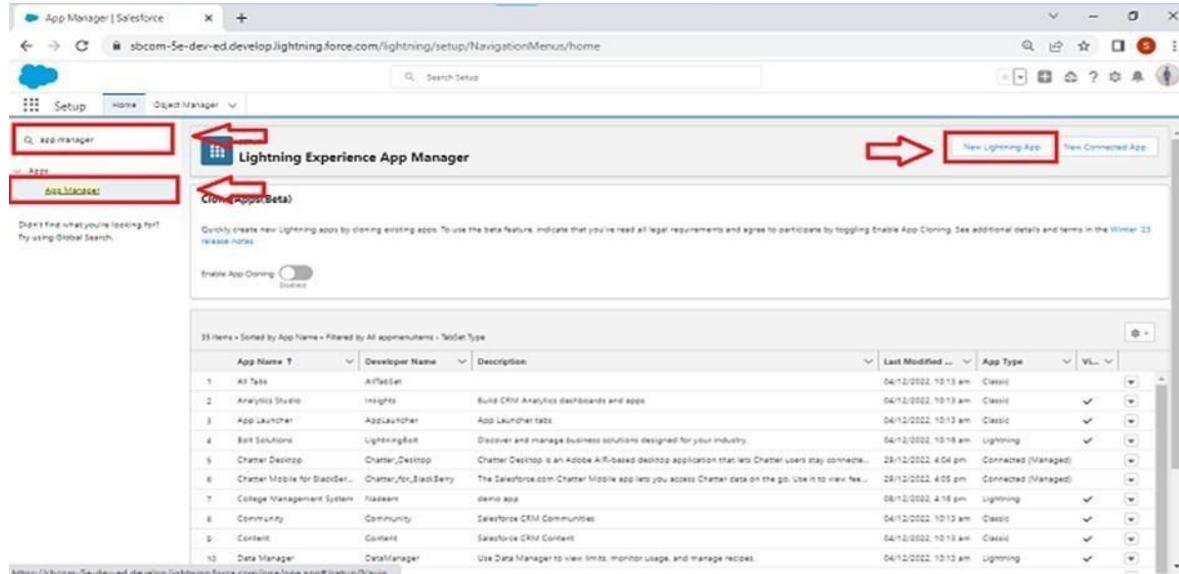
An app is a collection of items that work together to serve a particular function. In Lightning Experience, Lightning apps gives users access to sets of objects, tabs, and other items all in one convenient bundle in the navigation bar. Lightning apps let you brand your apps with a custom color and logo. You can even include a utility bar and Lightning page tabs in your Lightning app. Members of your org can work more efficiently by easily switching between apps.

## Use Case:

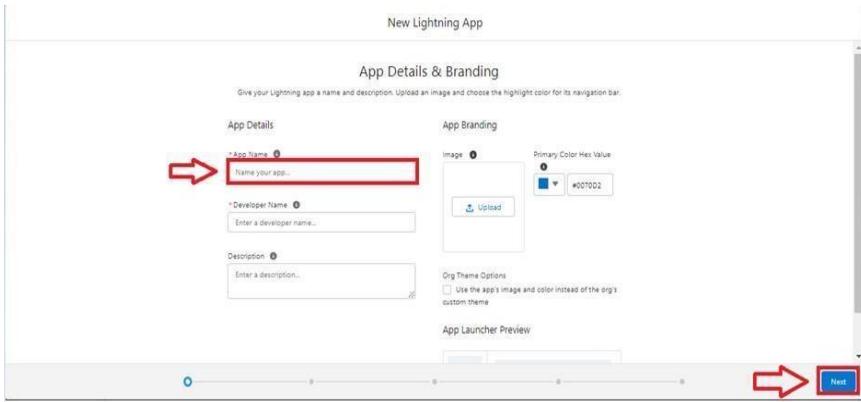
Well done you have reached close to your organizational requirement by creating the objects to store the organization's data. Making a database for an organization is just not enough to reach out the requirements, the task is how the users at the organization can access the objects you have created for them. As an Admin and Developer for the organization it's your duty to make sure every user of the organization is able to access the data modeling structure.

## Create A Lightning App

1. Go to setup page >> search “app manager” in quick find >> select “app manager” >> click on New lightning App.



2. Fill the app name in app details and branding as follow
  - a. App Name :Sales Automobile Using Salesforce CRM
  - b. Developer Name : this will auto populated
  - c. Description : Give a meaningful description
  - d. Image : optional (if you want to give any image you can otherwise not mandatory)
  - e. Primary color hex value : keep this default
3. Then click Next >> (App option page) keep it as default >> Next >> (Utility Items) keep it as default >> Next.



#### 4. Add Navigation Items:

New Lightning App

### Navigation Items

Key appear. Users can personalize the navigation to add or move items, but users can't remove or rename the items that you add. Some navigation items are dropped from the navigation bar when the app is viewed in a format that the item doesn't support.

**Available Items**

Create ▾

Create ▾

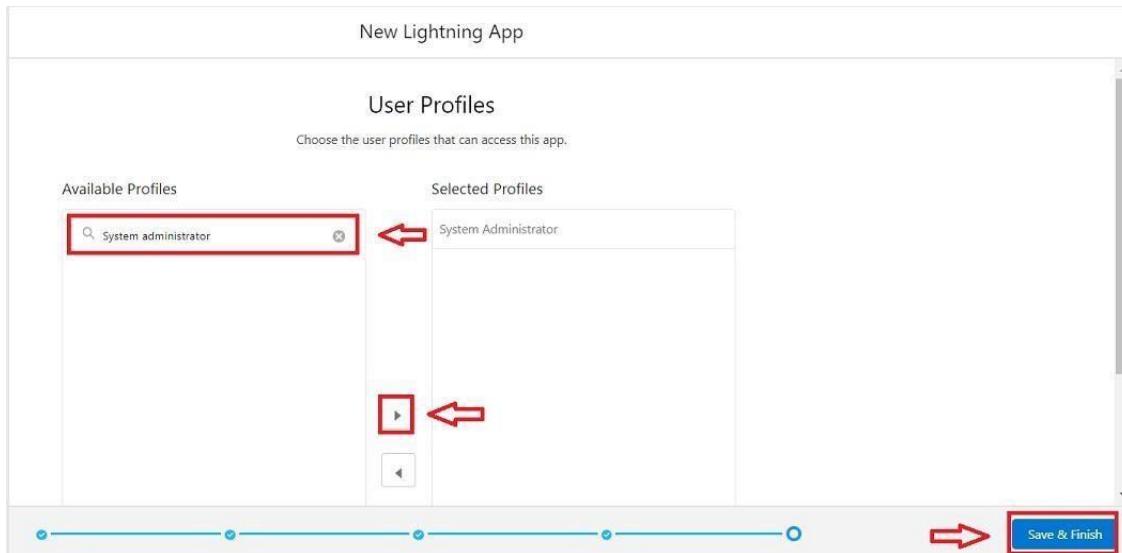
**Selected Items**

<span style="color: blue;">■</span> Accounts
<span style="color: purple;">■</span> Contacts
<span style="color: orange;">■</span> Opportunities
<span style="color: brown;">■</span> Automobile Information
<span style="color: red;">■</span> Automobiles
<span style="color: pink;">■</span> Invoice
<span style="color: teal;">■</span> Reports
<span style="color: darkblue;">■</span> Dashboards

#### 5. Search the items in the search bar(Account,Contact,Opportunities,Automobile Information,Opportunity, Automobile,Invoice,

Reports, Dashboard) from the search bar and move it using the arrow button ? Next.

Note: select asset the custom object which we have created in the previous activity.



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

## **Fields & Relationships**

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

1. Standard Fields
2. Custom Fields

### **Standard Fields:**

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

>>Created By

>> Owner

>> Last Modified

>> Field Made During object Creation

### **Custom Fields:**

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

## Use Case:

Now it's time for you to think out of the box for your organization. You have successfully created the database objects for the organization but now all eyes turn on you as you have to define what sort of information the objects store which you have created. As a life saver of your organization you come up with the idea of creating fields to store different types of data.

## Creating Opportunity Master Detail Relationship Field In Opportunity AutoMobile Object

To create fields in an object:

1. Go to setup >> click on Object Manager >> type object name(Opportunity Automobile) in quick find bar>> click on the object.



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



3. Select Data type as “Master Details Relationship”.

SETUP > OBJECT MANAGER

## Opportunity Automobile

**Details**

**Fields & Relationships**

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

**Opportunity Automobile**  
New Custom Field

**Step 1. Choose the field type**

Specify the type of information that the custom field will contain.

**Data Type**

None Selected      Select one of the data types below.

Auto Number      A system-generated sequence number that uses a display format you define. The number is automatically incremented for each new record.

Formula      A read-only field that derives its value from a formula expression you define. The formula field is updated when any of the source fields change.

Rollup Summary A read-only field that displays the sum, minimum, or maximum value of a field in a related list or the record count of all records listed in a related list.

Master-Detail Relationship      Creates a relationship that links this object to another object. The relationship field allows users to click on a lookup icon to select a value from a pop-up list. The object must have a master-detail relationship defined.

- The relationship field is required on all detail records.
- The ownership and sharing of a detail record are determined by the master record.
- When a user deletes the master record, all detail records are deleted.
- You can create rollup summary fields on the master record to summarize the detail records.

External Lookup Relationship      Creates a relationship that links this object to an external object whose data is stored outside the Salesforce org.

## 4. Click on Next

Opportunity Automobile  
New Relationship

**Step 2. Choose the related object**

Select the other object to which this object is related.

Related To:

**Step 2 of 6**

Previous Next Cancel

## 5. Fill the above as following:

- Field Label: gets auto Generated(Opportunity) • Field Name : gets auto generated(Opportunity)
- Click on Next >> Next >> Save and new.

Opportunity Automobile  
New Relationship

**Step 3. Enter the label and name for the lookup field**

Field Label:       Required

Field Name:       Required

Description:

Help Text:

Child Relationship Name:       Required

Sharing Setting: Select the minimum access level required on the Master record to create, edit, or delete related Detail records.

Read Only: Allows users with at least Read access to the Master record to create, edit, or delete related Detail records.

Read/Write: Allows users with at least Read/Write access to the Master record to create, edit, or delete related Detail records.

Allow reparenting:  Child records can be reparented to other parent records after they are created.

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

**Lookup Filter**

Optional: create a filter to limit the records available to users in the lookup field. [Tell me more](#).

Show Filter Settings

**Step 3 of 6**

Previous Next Cancel

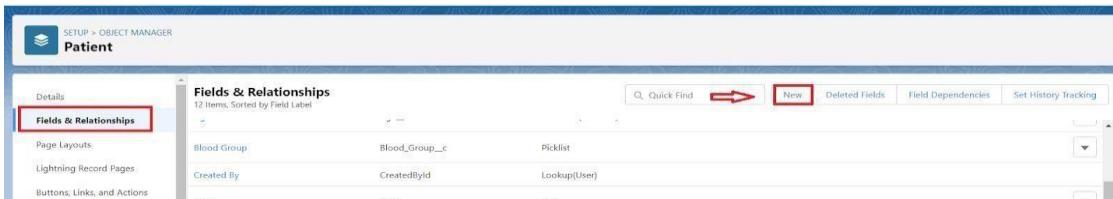
# Creating The AutoMobile Information Lookup Field In Opportunity Automobile Object

To create fields in an object:

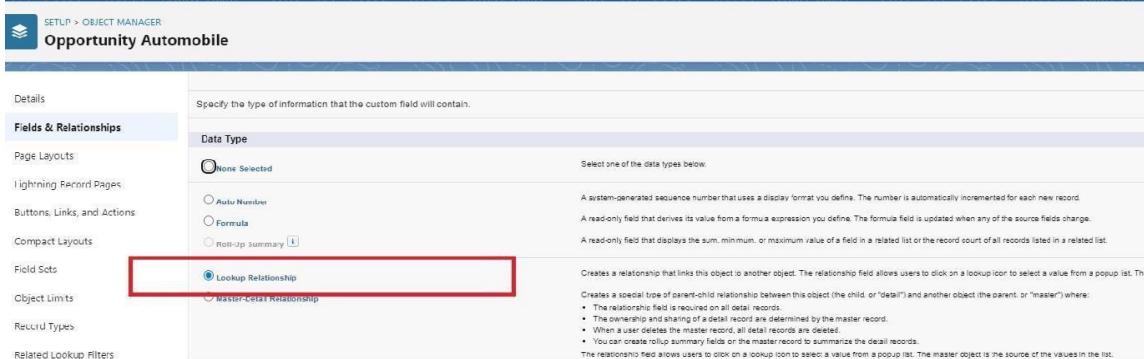
1. Go to setup >> click on Object Manager >> type object name(Opportunity Automobile) in quick find bar>> click on the object.



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



3. Select Data type as “Lookup Relationship”.



4. Click on Next



5. Fill the above as following:

- a. Field Label: Automobile

## b. Field Name : Automobile

6. Click on Next >> Next>> Save and new.

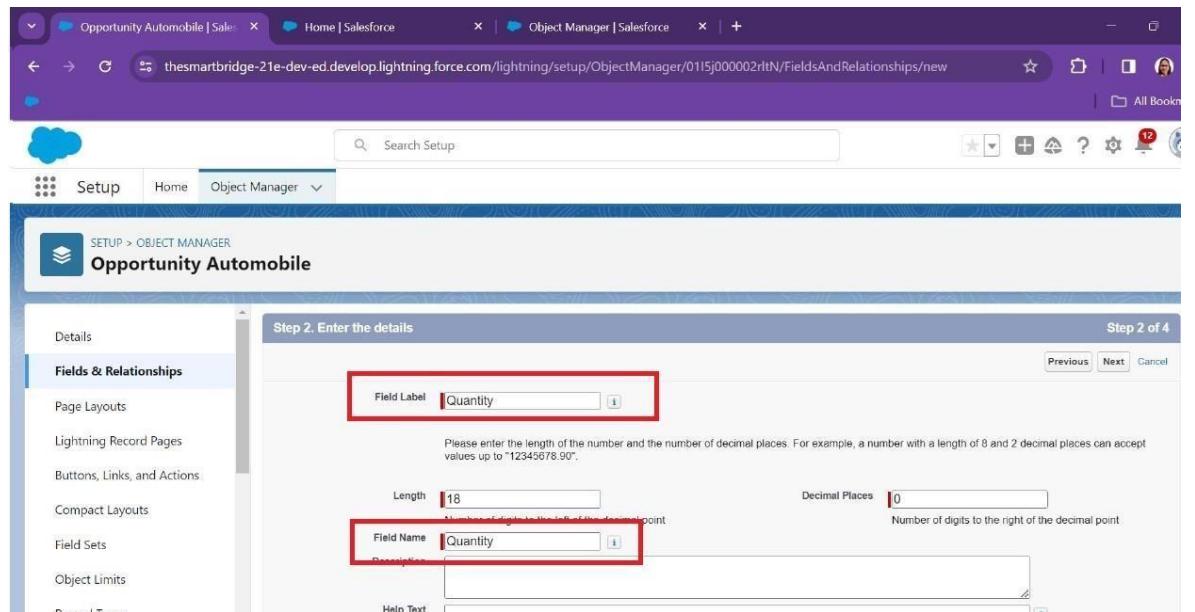
## Creating Quantity Number Field In Opportunity Automobile Object

To create fields in an object:

1. Go to setup >> click on Object Manager >> type object name(Opportunity Automobile) in quick find bar >> click on the object.
2. Now click on “Fields & Relationships” >> New.
3. Select Data type as “Number” and click Next.

a. Field Label >> Quantity

b. Field Name >> Quantity



4. Check that Required Check box.
5. Click Next >> Next >> Save & New.

## Creating Formula Field In Opportunity Automobile Object

To create fields in an object:

1. Go to setup >> click on Object Manager >> type object name(Opportunity Automobile) in quick find bar >> click on the object.
1. Now click on “Fields & Relationships” >> New.
2. Select Data type as “Formula” and click Next.
3. Give Field Label and Field Name as “Unit Price” and select formula return type as “Currency” and change the decimal values to two and click next.

Opportunity Automobile  
New Custom Field

Step 2. Choose output type

Field Label  Field Name

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

Formula Return Type

None Selected

CheckBox

Currency

Date

Date/Time

Number

Percent

Text

Time

Options

Select one of the data types below:

Calculate a boolean value  
Example: `ISDRAFT`

Calculate a dollar (or other currency amount) and automatically format the field as a currency amount.  
Example: `ISNULL(Amount) ? 0 : Amount`

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

Calculate a decimal, for example, by adding a number of hours or days to another decimal.  
Example: `Next 30 MINUTES + 1`

Calculate a percent value.  
Example: `Percent * 100`

Calculate a percent and automatically add the percent sign to the number.  
Example: `Discount + (Amount - Discounted Amount) / Amount`

Create a text string, for example, by concatenating other text fields.  
Example: `Full Name + Lastname & ", " & Firstname`

Calculate a time, for example, by adding a number of hours to another time.  
Example: `(Next + #MVALUE(NOW())) + 1`

Decimal Places  Example: 000.00

Help for this Page [x] Step 2 of 5 Previous Next Cancel

4. Under Advanced Formula write down the formula : Automobile\_\_r.Price\_\_c

Simple Formula Advanced Formula

Insert Field Insert Operator ▾

Unit Price (Currency) =  
Automobile\_\_r.Price\_\_c

Check Syntax No syntax errors in merge fields or functions. (Compiled size: 31 characters)

5. click “Check Syntax” and Next >> Next >> Save & New.

## Creating The Formula Field In Opportunity Automobile Object

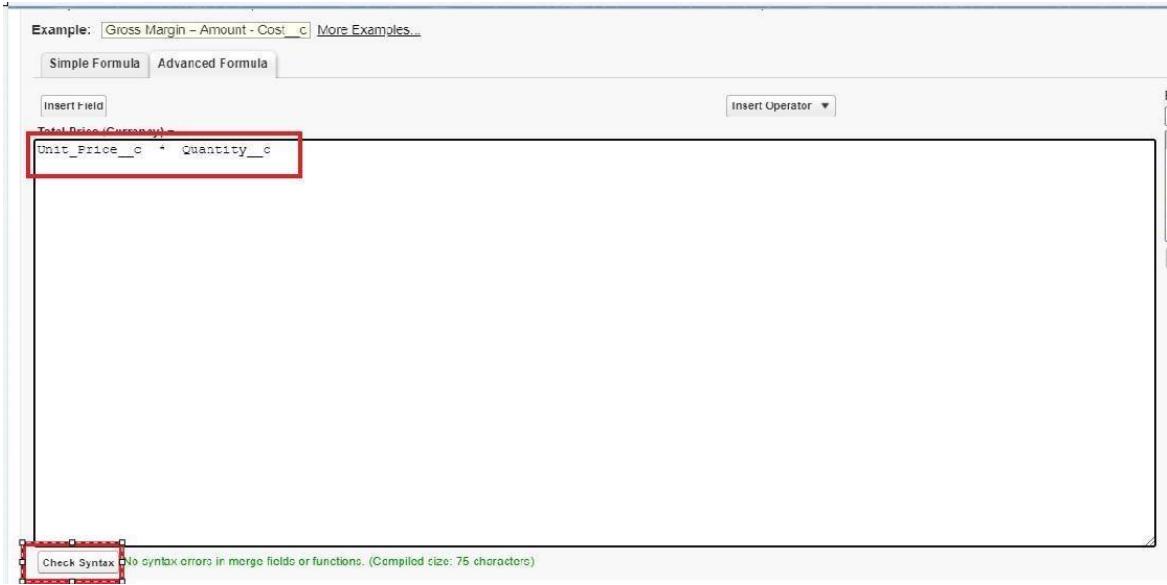
To create fields in an object:

1. Go to setup >> click on Object Manager >> type object name(Opportunity Automobile) in quick find bar >> click on the object.

1. Now click on “Fields & Relationships” >> New.
2. Select Data type as “Formula” and click Next.
3. Give Field Label and Field Name as “Total Price” and select formula return type as “Currency” and change the decimal values to two and click next.

The screenshot shows the 'New Custom Field' wizard in progress. The current step is 'Step 2. Choose output type'. The 'Field Label' is set to 'Total Price' and the 'Field Name' is 'Total\_Price'. The 'Formula Return Type' section is expanded, showing various options: 'None Selected' (radio button), 'Currency' (radio button, highlighted with a red box), 'Date', 'DateTime', 'Number', 'Percent', 'Text', and 'Time'. Below these options, there are examples and descriptions for each type. At the bottom of the 'Formula Return Type' section, there is a 'Decimal Places' dropdown set to 2, with an example 'Example: 000.00'. Navigation buttons 'Previous', 'Next', and 'Cancel' are located at the bottom right of the form.

4. Under Advanced Formula write down the formula : Unit\_Price\_\_\_\_\_c \* Quantity\_\_\_\_c



5. click “Check Syntax” and Next >> Next >> Save.

## Updating Field In Invoice Object

To Update fields in an object:

1. Go to setup ? click on Object Manager ? type object name(Invoice) in quick find bar? click on the object.
2. Now click on “Fields & Relationships” , Click on the edit of Invoice Id field.

Fields & Relationships					
FIELD LABEL		FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)			
Last Modified By	LastModifiedById	Lookup(User)			
Owner	OwnerId	Lookup(User,Group)		✓	
Purchase Date	Purchase_Date__c	Date			
Quantity	Quantity__c	Number(18, 0)			
Total Price	Total_Price__c	Number(18, 0)			
Unit Price	Unit_Price__c	Number(18, 0)			
Invoice ID	Name	Text(80)		✓	

SETUP > OBJECT MANAGER  
**Invoice**

Details	Field	Invoice ID
Page Layouts	Record Name	Invoice ID
Lightning Record Pages	Example: Account Name	
Buttons, Links, and Actions	Data Type	Auto Number
Compact Layouts	Display Format	I-{0000}
Field Sets	Example: A-{0000} <a href="#">What is This?</a>	
Object Limits	Starting Number	1
Record Types		
Related Lookup Filters		
Search Layouts		

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

Recent Accounts

Account Name	City
Acme	New York
Global Media	Toronto
salesforce.com	San Francisco

3. Select Data type as “Auto Number ” and click Next.
  - a. Display Format :- I-{0000}
  - b. StartingNumber:-
4. Click Save.

## Creating Remaining Fields In Objects

Now create the remaining fields using the data types mentioned.

s.no	Object name	Fields	
1	Invoice	<b>Field Name</b> Opportunity	<b>Data type</b> Master Detail relationship Object : Opportunity

## Page Layouts

Page Layout in Salesforce allows us to customize the design and organize detail and edit pages of records in Salesforce. Page layouts can be used to control the appearance of fields, related lists, and custom links on standard and custom objects' detail and edit pages.

### Use Case:

Hurray!! you have completed the data model structure for your organization but while looking at the detailed and edit pages it seems to be so clumsy, so decide to organize the page in a pleasant way for the sake of good and pleasant appearance and assemble all different kinds of information in different sections in order.

## Edit The Page Layout For Opportunity Object

Step 1: Go to Setup >> Click on Object Manager >> On the search bar, select

Opportunity Layout. You can notice Page Layouts on the left panel Step

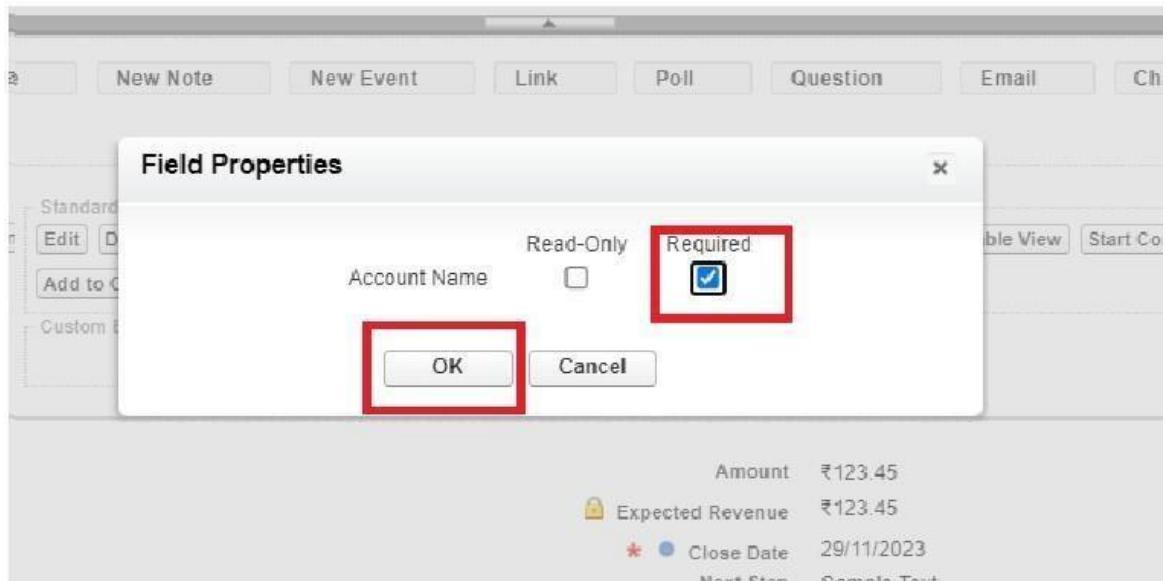
2: Click on Page Layouts, Click on ‘Opportunity Layouts’.

The screenshot shows the Salesforce Object Manager interface for the Opportunity object. The left sidebar lists various setup categories like Fields & Relationships, Page Layouts, Lightning Record Pages, etc. The main area displays a table titled 'Page Layouts' with four entries:

Page Layout Name	Created By	Modified By
Opportunity Standard Layout	Mohamed Samir (2023-07-29 08:46 AM)	Mohamed Samir (2023-07-29 08:46 AM)
Opportunity Enhanced Layout	Mohamed Samir (2023-07-29 08:46 AM)	Mohamed Samir (2023-07-29 08:46 AM)
Opportunity Chatter Layout	Mohamed Samir (2023-07-29 08:46 AM)	Mohamed Samir (2023-07-29 08:46 AM)

Step 3: In the Opportunity Detail Section, you can see various fields. Go on Account And Click on that Properties icon of Account name Field.

The screenshot shows the 'Layout Properties' tab for the Opportunity Detail page layout. The top navigation bar includes Save, Quick Save, Preview As, Cancel, and Layout Properties. The left sidebar lists various setup categories. The main area shows the 'Opportunity Detail' layout with several sections and fields. A specific field, 'Account Name', is highlighted with a red box and has a properties icon (a gear symbol) next to it. The status bar at the bottom indicates 'Additional Information (header visible on edit only)'.



Step 4: check the Required box for Account name and click on Ok.

Step 5: Click on Save.

## Edit The Page Layout For Automobiles Information

Step 1: Go to Setup >> Click on Object Manager >> On the search bar, select Automobile Information. You can notice Page Layouts on the left panel

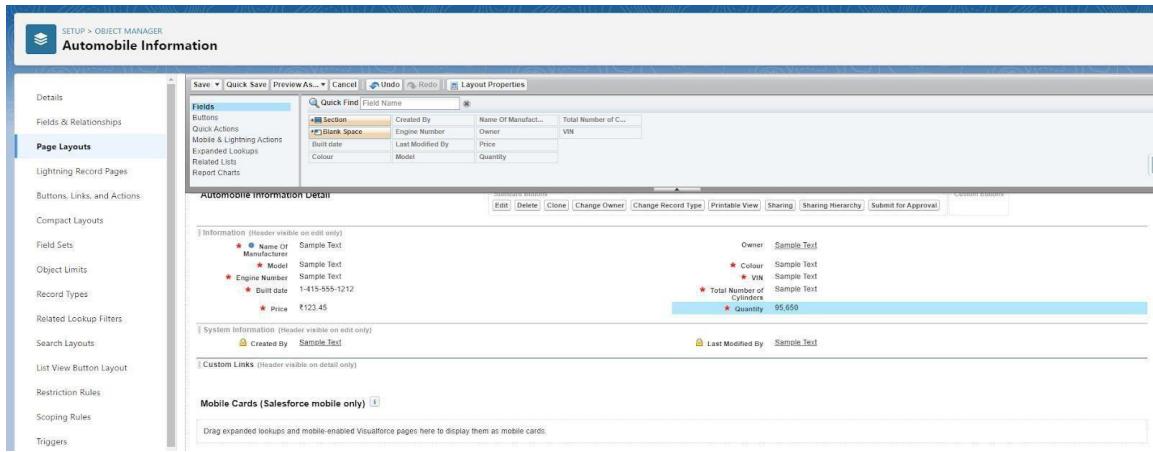
Step 2: Click on Page Layouts. Click on ‘Automobile Information Layout’.

The top screenshot shows the 'Page Layouts' section of the Object Manager. A new layout named 'Automobile Information Layout' has been created by Mohammad Sameer at 12:49 pm. The bottom screenshot shows the detailed configuration of this layout, including fields like Section, Created By, Name Of Manufact..., Total Number of C..., Engine Number, Owner, VIN, Last Modified By, Price, Colour, Model, and Quantity.

Step 3: Just Go for each one field of Automobile Information Object, Click on Gear Icon and mark as Required just as Done for Above Account Object. After required is done it will show the red color as given in below image.

The screenshot shows the 'Automobile Information Detail' page with various fields. Several fields are marked with red asterisks to indicate they are required: 'Name Of Manufacturer', 'Model', 'Engine Number', 'Total Number of Cylinders', 'Colour', 'Build date', 'Price', and 'Quantity'. The 'Owner' field is also marked with a red asterisk.

Step 4 : Adjust the Fields as given below for A good looking view.



Step 5 : Click on Save.

## Apex Trigger

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

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

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

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

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

There are primarily two types of Apex Triggers:

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

criteria or code can be set to check data before it gets ready to be inserted into the database.

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

## Opportunity Automobile Quantity

**UseCase : Whenever Opportunity Closed won Than Neglect / Minus the Quantity From Automobile Information on the Bases of Opportunity Automobile quantity.**

1. Login to the respective trailhead account and navigate to the gear icon in the top right corner.
2. Click on the Developer console. Now you will see a new console window.
3. In the toolbar, you can see FILE. Click on it and navigate to new and create New apex class.
4. Name the class as “OpportunityHandlerClass ”.

```

public class OpportunityHandlerClass {
    public static void opportunityAutomobileQuantity(List<Opportunity> lstOpportunity, Map<Id, Opportunity> OldMapOpportunity){
        set<Id> opportunityIds = new set<Id>();
        for(Opportunity opp : lstOpportunity){
            if(opp.StageName == 'Closed Won'){
                opportunityIds.add(opp.Id);
            }
        }
    }

    set<Id> opportunityIds = new set<Id>();
    for(Opportunity opp : lstOpportunity){
        if(opp.StageName == 'Closed Won'){
            opportunityIds.add(opp.Id);
        }
    }
    Map<Id, Opportunity_Automobile__c> lstOpportunityAutomobile = new Map<Id, Opportunity_Automobile__c>([SELECT Id, Opportunity__c, Automobile__c, Quantity__c, Unit_Price__c, Total_Price__c
        FROM Opportunity_Automobile__c WHERE Opportunity__c IN: opportunityIds]);

    set<Id> AutoInformationIds = new set<Id>();
    for(Opportunity_Automobile__c OppAuto: lstOpportunityAutomobile.values()){
        if(OppAuto.Automobile__c != null){
            AutoInformationIds.add(OppAuto.Automobile__c);
        }
    }
    List<Automobile_Information__c> lstAutomobileInfomation = new List<Automobile_Information__c>();
    Map<Id, Automobile_Information__c> MapAutomobileInformation = New Map<Id, Automobile_Information__c>([SELECT Quantity__c, Price__c, Name, Id
        FROM Automobile_Information__c
        WHERE Id IN: AutoInformationIds]);

    For(Opportunity_Automobile__c AutoOpp : lstOpportunityAutomobile.Values()){
        decimal num = 0;
        if(AutoOpp.Automobile__c == MapAutomobileInformation.get(AutoOpp.Automobile__c).Id && OldMapOpportunity.get(AutoOpp.Opportunity__c).stagename != 'Closed Won'){

            num = MapAutomobileInformation.get(AutoOpp.Automobile__c).Quantity__c - AutoOpp.Quantity__c;
            MapAutomobileInformation.get(AutoOpp.Automobile__c).quantity__c = num;
            lstAutomobileInfomation.add(MapAutomobileInformation.get(AutoOpp.Automobile__c));
        }
    }
    If(!lstAutomobileInfomation.IsEmpty()){
        update lstAutomobileInfomation;
    }
}

}

```

**Code:**

```
public class OpportunityHandlerClass {  
  
    public static void opportunityAutomobileQuantity(List<Opportunity>  
        LstOpportunity, Map<Id,Opportunity> OldMapOpportunity){  
        set<Id>  
        opportunityIds = new set<Id>();  
        for(Opportunity opp : LstOpportunity){  
            if(opp.StageName == 'Closed' && opp.Won')  
                opportunityIds.add(opp.Id);  
        }  
    }  
  
    Map<Id,Opportunity_Automobile__c> lstOpportunityAutomobile = new  
    Map<Id,Opportunity_Automobile__c>([SELECT Id, Opportunity__c,  
    Automobile__c, Quantity__c, Unit_Price__c, Total_Price__c FROM  
    Opportunity_Automobile__c WHERE Opportunity__c IN: opportunityIds]);  
  
    set<Id> AutoInformationIds = new set<Id>();  
    for(Opportunity_Automobile__c OppAuto: lstOpportunityAutomobile.values())  
        if(OppAuto.Automobile__c != null){  
            AutoInformationIds.add(OppAuto.Automobile__c);  
        }  
    }  
  
    List<Automobile_Information__c> lstAutomobileInfomation =  
        new List<Automobile_Information__c>();  
    Map<Id,Automobile_Information__c> MapAutomobileInformation = New  
    Map<Id,Automobile_Information__c>([SELECT Quantity__c, Price__c, Name, Id  
    FROM Automobile_Information__c WHERE Id IN: AutoInformationIds]);
```

```

For(Opportunity_Automobile__c           AutoOpp : 
lstOpportunityAutomobile.Values()){
    decimal num = 0;
    if(AutoOpp.Automobile__c == 
MapAutomobileInformation.get(AutoOpp.Automobile__c).Id      &&
OldMapOpportunity.get(AutoOpp.Opportunity__c).stagename != 'Closed Won'){
        num = 
MapAutomobileInformation.get(AutoOpp.Automobile__c).Quantity__c-
AutoOpp.Quantity__c;
        MapAutomobileInformation.get(AutoOpp.Automobile__c).quantity__c
        = num;
    }

    lstAutomobileInfomation.add(MapAutomobileInformation.get(AutoOpp.Automobi
le__c));
}

}

If(!lstAutomobileInfomation.IsEmpty()){
    update lstAutomobileInfomation;
}

}

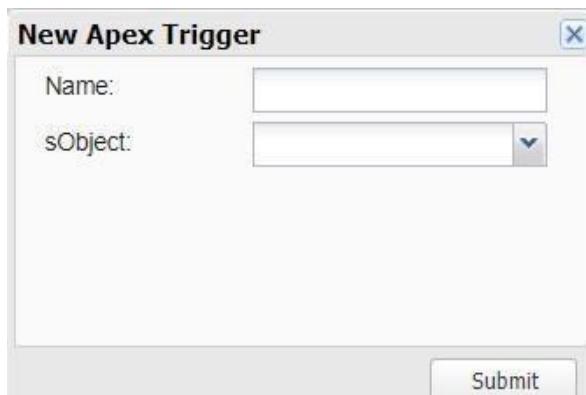
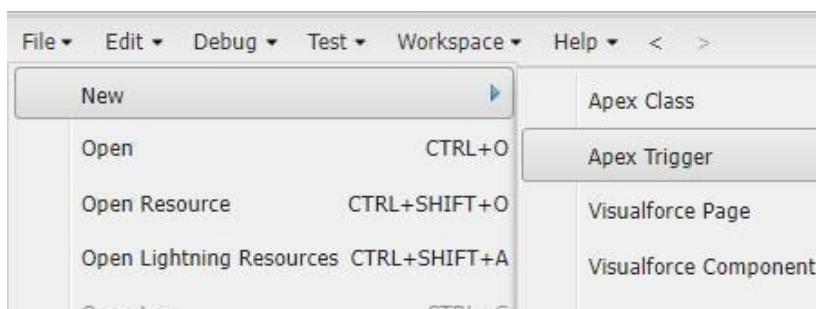
}

```

### **Trigger Handler :**

How to create a new trigger :

1. While still in the account, navigate to the gear icon in the top right corner.
2. Click on developer console and you will be navigated to a new console window.
3. Click on the File menu in the toolbar, and click on new? Trigger.
4. Enter the trigger name and the object to be triggered.
5. Name : OpportunityTrigger
6. sObject : Opportunity



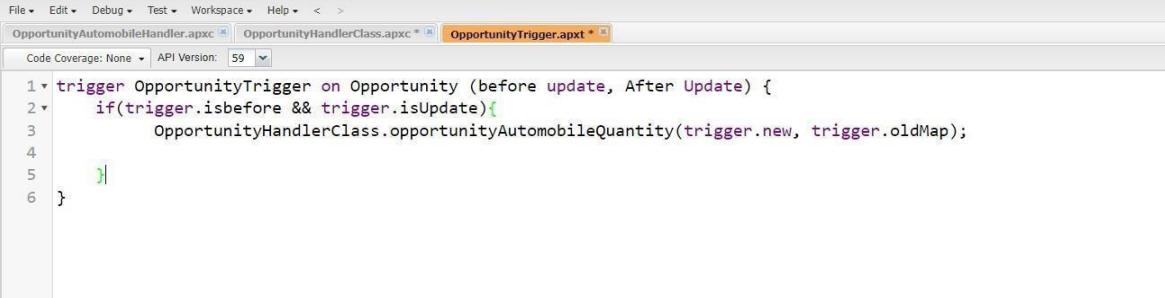
Syntax For creating trigger :

The syntax for creating trigger is :

```
Trigger [trigger name] on [object name]( Before/After event){  
//block of code  
}
```

In this project , trigger is called whenever the particular records sum exceed the threshold i.e minimum business requirement value. Then the code in the trigger will get executed.

### 1. Trigger for Opportunity Object.



The screenshot shows the Salesforce IDE interface with three tabs at the top: 'OpportunityAutomobileHandler.apxc', 'OpportunityHandlerClass.apxc', and 'OpportunityTrigger.apxt'. The 'OpportunityTrigger.apxt' tab is active. The code editor contains the following Apex trigger:

```
trigger OpportunityTrigger on Opportunity (before update, After Update) {
    if(trigger.isbefore && trigger.isUpdate){
        OpportunityHandlerClass.opportunityAutomobileQuantity(trigger.new, trigger.oldMap);
    }
}
```

```
trigger OpportunityTrigger on Opportunity (before update, After Update) {
if(trigger.isbefore && trigger.isUpdate){
    OpportunityHandlerClass.opportunityAutomobileQuantity(trigger.new,
trigger.oldMap);
}
}
```

### Opportunity-Automobile Error

**UseCase : If Quantity of Automobile is Zero or Less than The Quantity from The Opportunity-Automobile Than Throw an error .**

Login to the respective trailhead account and navigate to the gear icon in the top right corner.

1. Click on the Developer console. Now you will see a new console window.
2. In the toolbar, you can see FILE. Click on it and navigate to new and create New apex class.

### 3. Name the class as “OpportunityAutomobileHandler ”.

```

1 • public class OpportunityAutomobileHandler {
2 •     public static void quantityErrorOnAutomobileInformation(List<Opportunity_Automobile__c> lstOpportunityAutomobile){
3 •         Set<Id> AutomobileIds = new Set<Id>();
4 •         For(Opportunity_Automobile__c OppAutomobile : lstOpportunityAutomobile){
5 •             If(OppAutomobile.Automobile__c != null){
6 •                 AutomobileIds.add(OppAutomobile.Automobile__c);
7 •             }
8 •         }
9 •         Map<Id, Automobile_Information__c> lstAutomobileInformation = new Map<Id, Automobile_Information__c>([SELECT Id, CreatedById, Quantity__c, Price__c
10 •          FROM Automobile_Information__c WHERE Id IN: AutomobileIds]);
11 •         For(Opportunity_Automobile__c OppAutomobile : lstOpportunityAutomobile){
12 •             If(OppAutomobile.Automobile__c == lstAutomobileInformation.get(OppAutomobile.Automobile__c).Id && lstAutomobileInformation.get(OppAutomobile.Automobile__c).Quantity__c < OppAutomobile.Quantity__c){
13 •                 OppAutomobile.addError('the Number of Automobile u want are not Available !! the Automobile are Available Count is ' + lstAutomobileInformation.get(OppAutomobile.Automobile__c).Quantity__c );
14 •             }
15 •         }
16 •     }
17 • }

```

#### Code:

```

public class OpportunityAutomobileHandler {

    public static void quantityErrorOnAutomobileInformation(List<Opportunity_Automobile__c>
lstOpportunityAutomobile){

        Set<Id> AutomobileIds = new Set<Id>();

        For(Opportunity_Automobile__c OppAutomobile : lstOpportunityAutomobile){

            If(OppAutomobile.Automobile__c != null){

                AutomobileIds.add(OppAutomobile.Automobile__c);

            }

        }

        Map<Id, Automobile_Information__c> lstAutomobileInformation = new Map<Id, Automobile_Information__c>([SELECT Id, CreatedById, Quantity__c,
Price__c FROM Automobile_Information__c WHERE Id IN: AutomobileIds]);

        For(Opportunity_Automobile__c OppAutomobile : lstOpportunityAutomobile){

            If(OppAutomobile.Automobile__c ==
lstAutomobileInformation.get(OppAutomobile.Automobile__c).Id &&
lstAutomobileInformation.get(OppAutomobile.Automobile__c).Quantity__c
<
OppAutomobile.Quantity__c){


```

```

        OppAutomobileaddError('the Number of Automobile u want are not
Available    !!    the    Automobile    are    Available    Count    is    '    +
lstAutomobileInformation.get(OpportunityAutomobile__c).Quantity__c);

    }

}

}

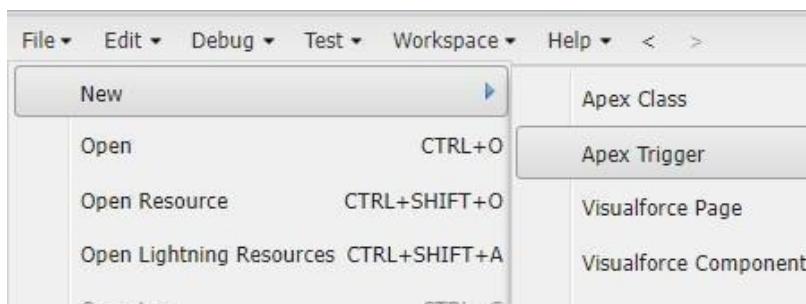
}

```

### **Trigger Handler :**

How to create a new trigger :

1. While still in the trailhead account, navigate to the gear icon in the top right corner.
2. Click on developer console and you will be navigated to a new console window.
3. Click on the File menu in the toolbar, and click on new? Trigger.
4. Enter the trigger name and the object to be triggered.
5. Name : OpportunityAutoMobileTrigger
6. sObject : Opportunity\_Automobile\_\_c





## Trigger :

Handler for the Opportunity\_Automobile\_\_c Object

OpportunityAutomobileHandler.apxc | OpportunityHandlerClass.apxc | OpportunityTrigger.apxt | OpportunityAutoMobileTrigger.apxt

Code Coverage: None | API Version: 59

```

1 trigger OpportunityAutoMobileTrigger on Opportunity_Automobile__c (before insert, before update) {
2     if(trigger.isbefore && trigger.isinsert || trigger.isupdate){
3         OpportunityAutomobileHandler.quantityErrorOnAutomobileInformation(trigger.new);
4     }
5 }
```

**Code:** trigger OpportunityAutoMobileTrigger on Opportunity\_Automobile\_\_c  
 (before insert, before Update) { if(trigger.isbefore && trigger.isinsert ||  
 trigger.isupdate){

```

  OpportunityAutomobileHandler.quantityErrorOnAutomobileInformation(trigger.n
ew);
}
```

}

## Invoice Creation Trigger

**UseCase :** Whenever an opportunity is Closed won then create the Invoice on  
 the Bases of Opportunity Automobile Data.

Login to the respective trailhead account and navigate to the gear icon in the top right corner.

1. Click on the Developer console. Now you will see a new console window.
2. In the toolbar, you can see FILE. Click on it and navigate to new and create New apex class.
3. Name the class as “InvoiceCreation”.

```

1+ public class InvoiceCreation {
2+     public static void OpportunityClosedwonInvoiceGeneration(List<Opportunity> lstOpportunity, Map<Id,Opportunity>OldMapOpportunity){
3+         set<Id> oppIds = new Set<Id>();
4+         For(Opportunity opp : lstOpportunity){
5+             If(opp.StageName == 'Closed Won' && OldMapOpportunity.get(opp.Id).StageName != opp.StageName){
6+                 oppIds.add(opp.Id);
7+             }
8+         }
9+         List<Opportunity_Automobile__c> lstOpportunityAutomobile = [SELECT Unit_Price__c, Total_Price__c, Automobile__c, Quantity__c,Opportunity__c, Id FROM Opportunity_Automobile__c WHERE Opportunity__c IN: oppIds];
10+        List<Invoice__c> lstInvoice = new List<Invoice__c>();
11+        For(Opportunity_Automobile__c oppAuto : lstOpportunityAutomobile){
12+            Invoice__c i = new Invoice__c();
13+            i.Quantity__c = oppAuto.Quantity__c;
14+            i.Unit_Price__c = oppAuto.Unit_Price__c;
15+            i.Total_Price__c = oppAuto.Total_Price__c;
16+            i.Purchase_Date__c = date.today();
17+            i.Opportunity__c = oppAuto.Opportunity__c;
18+            lstInvoice.add(i);
19+        }
20+        If(!lstInvoice.isEmpty()){
21+            insert lstInvoice;
22+        }
23+    }
24+ }

```

## Code:

```

public class InvoiceCreation {                                     public static void
OpportunityClosedwonInvoiceGeneration(List<Opportunity>           lstOpportunity,
Map<Id,Opportunity>OldMapOpportunity){ set<Id> oppIds = new Set<Id>(); For(Opportunity opp : lstOpportunity){
if(Opp.StageName == 'Closed Won' && OldMapOpportunity.get(opp.Id).StageName != opp.StageName){
oppIds.add(opp.Id);
}
List<Opportunity_Automobile__c> lstOpportunityAutomobile = [SELECT
Unit_Price__c, Total_Price__c, Automobile__c, Quantity__c,Opportunity__c, Id
FROM Opportunity_Automobile__c WHERE Opportunity__c IN: oppIds];

```

```

List<Invoice__c> lstInvoice = new List<Invoice__c>();
For(Opportunity_Automobile__c oppAuto : lstOpportunityAutomobile){
    Invoice__c i = new Invoice__c();
    i.Quantity__c = oppAuto.Quantity__c;
    i.Unit_Price__c = oppAuto.Unit_Price__c;
    i.Total_Price__c = oppAuto.Total_Price__c;
    i.Purchase_Date__c = date.today();
    i.Opportunity__c = oppAuto.Opportunity__c;
    lstInvoice.add(i);
}
if(!lstInvoice.isEmpty()){
    insert lstInvoice;
}
}

```

### **Trigger Handler :**

For this class we don't need to create any trigger, we will call this Code in “Opportunity Trigger”.

1. Go on files and click on open.
2. Click on triggers.
3. Double click on OpportunityTrigger.

Open

Entity Type	Entities	Related			
Entity Type	Name	Namespace	Name	Extent	Direction
Classes	OpportunityTrigger		↳ Opportunity...	ApexClass	References
Triggers	OpportunityAutoMobile...		↳ Opportunity	SObject	References
Pages					
Page Components					
Objects					
Static Resources					
Packages					

Code Coverage: None API Version: 59

```

1 trigger OpportunityTrigger on Opportunity (before update, After Update) {
2     if(trigger.isbefore && trigger.isUpdate){
3         OpportunityHandlerClass.opportunityAutomobileQuantity(trigger.new, trigger.oldMap);
4
5     }
6     IF(trigger.isafter && trigger.isupdate){
7         InvoiceCreation.OpportunityClosedwonInvoiceGeneration(trigger.new, trigger.oldMap);
8     }
9 }
```

### Trigger:

```

trigger OpportunityTrigger on Opportunity (before update, After Update) {
if(trigger.isbefore && trigger.isUpdate){
    OpportunityHandlerClass.opportunityAutomobileQuantity(trigger.new,
trigger.oldMap);
}
IF(trigger.isafter && trigger.isupdate){
    InvoiceCreation.OpportunityClosedwonInvoiceGeneration(trigger.new,
trigger.oldMap);
}
```

## Check Contact Role

**UseCase : Whenever an opportunity is Going to Closed won then check it has the contact role or Not.**

Login to the respective trailhead account and navigate to the gear icon in the top right corner.

1. Click on the Developer console. Now you will see a new console window.
2. In the toolbar, you can see FILE. Click on it and navigate to new and create New apex class.
3. Name the class as “ContactRoleCheck”.

```
public class ContactRoleCheck {
    public static void CheckcontactRoleonOpportunity(List<Opportunity> lstOpportunity, Map<Id,Opportunity>OldMapOpportunity){
        List<OpportunityContactRole> lstContactRole = [SELECT Id From OpportunityContactRole WHERE OpportunityId IN: OldMapOpportunity.keySet()];
        For(Opportunity opp : lstOpportunity){
            if(Opp.StageName == 'Closed Won' && OldMapOpportunity.get(opp.Id).StageName != opp.StageName){
                If(lstContactRole.isEmpty()){
                    opp.adderror('Please add contact Role on opportunity whenever Opportunity is Going to Closed Won.');
                }
            }
        }
    }
}
```

## Trigger:

```
public class ContactRoleCheck { public static void
CheckcontactRoleonOpportunity(List<Opportunity> lstOpportunity,
Map<Id,Opportunity>OldMapOpportunity){
    List<OpportunityContactRole> lstContactRole = [SELECT Id From
OpportunityContactRole WHERE OpportunityId IN:
OldMapOpportunity.keySet()];
    For(Opportunity opp : lstOpportunity){
        if(Opp.StageName == 'Closed Won' &&
OldMapOpportunity.get(opp.Id).StageName != opp.StageName){
            If(lstContactRole.isEmpty()){


```

```
        opp.adderror('Please add contact Role on opportunity whenever  
Opportunity is Going to Closed Won.');
```

}

}

}

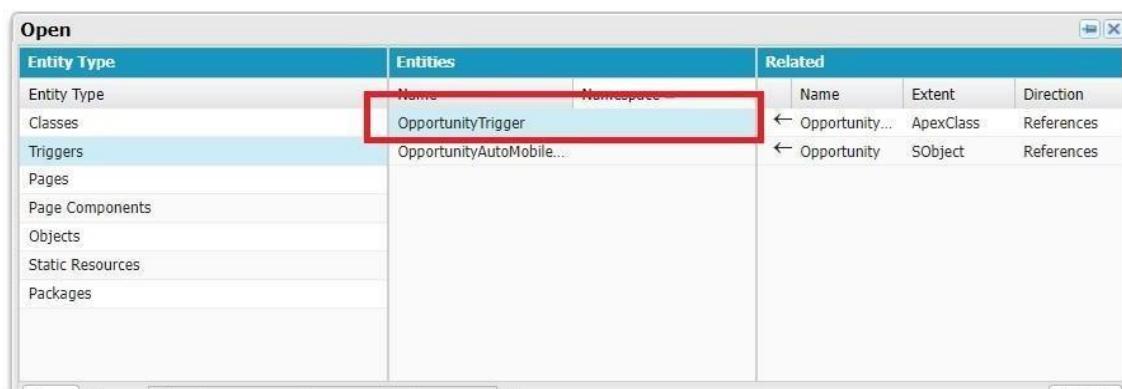
}

}

## **Trigger Handler :**

For this class we don't need to create any trigger, we will call this Code in “Opportunity Trigger”.

1. Go on files and click on open.
  2. Click on triggers.
  3. Double click on OpportunityTrigger.



## Trigger Code :

```
Code Coverage: None API Version: 59 ▾
1 trigger OpportunityTrigger on Opportunity (before update, After Update) {
2     if(trigger.isbefore && trigger.isUpdate){
3         OpportunityHandlerClass.opportunityAutomobileQuantity(trigger.new, trigger.oldMap);
4         ContactRoleCheck.CheckcontactRoleonOpportunity(trigger.new, trigger.oldMap);
5     }
6     IF(trigger.isafter && trigger.isupdate){
7         InvoiceCreation.OpportunityClosedwonInvoiceGeneration(trigger.new, trigger.oldMap);
8     }
9 }
```

Trigger:

```
trigger OpportunityTrigger on Opportunity (before update, After Update) {  
    if(trigger.isbefore && trigger.isUpdate){  
        OpportunityHandlerClass.opportunityAutomobileQuantity(trigger.new,  
        trigger.oldMap);
```

```

ContactRoleCheck.CheckcontactRoleonOpportunity(trigger.new, trigger.oldMap);
}

IF(trigger.isafter && trigger.isupdate){

InvoiceCreation.OpportunityClosedwonInvoiceGeneration(trigger.new,
trigger.oldMap);

}

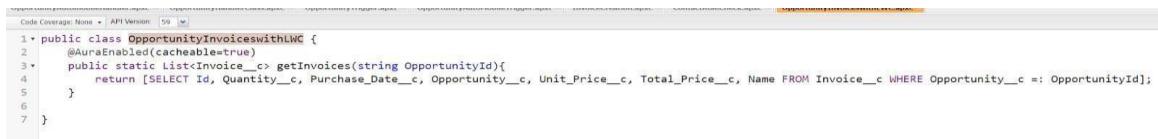
}

```

## LWC Component:

### Create Apex Class To Get Invoices

1. Login to the respective account and navigate to the gear icon in the top right corner.
2. Click on the Developer console.
3. Now you will see a new console window.
4. In the toolbar, you can see FILE.
5. Click on it and navigate to new and create New apex class.
6. Name the class as “OpportunityInvoiceswithLWC”.



```

Code Coverage: None | API Version: 59.0
1 * public class OpportunityInvoiceswithLWC {
2     @AuraEnabled(cacheable=true)
3     public static List<Invoice__c> getInvoices(string OpportunityId){
4         return [SELECT Id, Quantity__c, Purchase_Date__c, Opportunity__c, Unit_Price__c, Total_Price__c, Name FROM Invoice__c WHERE Opportunity__c =: OpportunityId];
5     }
6 }
7 }

```

## Code:

```
public class OpportunityInvoiceswithLWC {  
    @AuraEnabled(cacheable=true)    public static List<Invoice__c>  
    getInvoices(string OpportunityId){      return [SELECT Id,  
    Quantity__c, Purchase_Date__c, Opportunity__c, Unit_Price__c,  
    Total_Price__c, Name FROM Invoice__c WHERE Opportunity__c  
    =: OpportunityId];  
}  
}
```

## Install Salesforce CLI

The Salesforce CLI is a powerful command line interface that simplifies development and build automation when working with your Salesforce org.

### [Download and install Salesforce CLI](#)

To confirm that the Salesforce CLI is installed and working correctly, you can open a command prompt and type sfdx. This will display the version number of the Salesforce CLI that is currently installed on your system.

```
C:\Users\navee>sfdx
Salesforce CLI

VERSION
  sfdx-cli/7.182.1 win32-x64 node-v18.12.1

USAGE
  $ sfdx [COMMAND]

TOPICS
  alias      manage username aliases
  auth       authorize an org for use with the Salesforce CLI
  config     configure the Salesforce CLI
  force      tools for the Salesforce developer
  info       access cli info from the command line
  plugins    add/remove/create CLI plug-ins
  version    show the current version information

```

[codekiat.com](http://codekiat.com)

## Install Microsoft VS Code

VS Code, or Visual Studio Code, is a free, open-source code editor developed by Microsoft. It is a lightweight, cross-platform code editor that provides features such as debugging, Git integration, and support for a wide range of programming languages.

[Download the version of the software](#) that is compatible with your operating system and install it. The following instructions are for Windows OS. Other operating systems may have slightly different steps.

## OUTPUT :

### Create Report on Opportunity

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

The screenshot shows the 'Employee Management' application interface. At the top, there is a navigation bar with links for Home, Employees, Assets, Asset Services, Projects, ProjectTasks, Reports (which is highlighted with a red box), and Dashboards. Below the navigation bar is a search bar labeled 'Search...' and a toolbar with various icons. On the left, there is a sidebar titled 'Reports' with sections for 'Recent' (containing 'Employee's working on projects report' and 'Assets assigned to Employees'), 'Created by Me', 'Private Reports', 'Public Reports', and 'All Reports'. The main area displays a table of reports with columns for Report Name, Description, Folder, Created By, Created On, and Subscribed. Two specific reports are listed: 'Employee's working on projects report' and 'Assets assigned to Employees'. A red box highlights the 'New Report' button at the top right of the main content area.

3. Select report type from category or from report type panel or from search panel >> click on start report.

The screenshot shows the 'Create Report' dialog box. On the left, there is a sidebar titled 'Category' with a list of report types: Recently Used, All, Accounts & Contacts, Opportunities, Customer Support Reports, Leads, Campaigns, Activities, Contracts and Orders, Price Books, Products and Assets, Administrative Reports, File and Content Reports, and Individuals. The 'All' category is selected. In the center, there is a 'Select a Report Type' panel with a search bar containing 'Opportunities' (highlighted with a red box). Below the search bar is a table with columns for 'Report Type Name' and 'Category'. The first row in the table is also highlighted with a red box. On the right, there is a 'Details' panel for the 'Opportunities' report type. This panel includes a 'Start Report' button, a 'Details' section showing 'Created By You' and 'Created By Others' (both with 'No Reports Yet'), and a 'Objects Used in Report Type' section listing Account, Profile, Campaign, and User. A red box highlights the entire 'Details' panel.

4. Customize your report

- Add fields from left pane as shown below

Employee Name	Employee ID	Reports to	Login Time	Logout Time	Mode of Work	LinkedIn Profile
Employee	4000000004750				Normal	https://linkedin.com/in/
Employee Admin	4000000004750		8:00 am	9:00 pm	Normal	https://linkedin.com/in/

Account Name	Opportunity Name	Owner Role	Opportunity Owner	Stage	Next Step	Lead Source	Type
(1)	Test	-	Mohammad Samir	Closed Won	-	Web	Existing Customer - Upgrade
Burlington Textiles Corp of America (1)	Burlington Textiles Weaving Plant Generators	-	Mohammad Samir	Closed Won	-	Web	New Customer
Dickenson plc (1)	Dickenson Mobile Generators	-	Mohammad Samir	Qualification	-	Purchased List	New Customer
Edge Communications (4)	Edge Emergency Generator	-	Mohammad Samir	Closed Won	-	Word of mouth	New Customer
	Edge Installation	-	Mohammad Samir	Closed Won	-	Word of mouth	Existing Customer - Upgrade
	Edge SLA	-	Mohammad Samir	Closed Won	-	Word of mouth	Existing Customer - Upgrade
	Edge Emergency Generator	-	Mohammad Samir	Ic. Decision Makers	-		Existing Customer - Replacement
Grand Hotels & Resorts Ltd (5)	Grand Hotels Kitchen Generator	-	Mohammad Samir	Ic. Decision Makers	-	-	Existing Customer - Upgrade
	Grand Hotels Guest Portable Generators	-	Mohammad Samir	Value Proposition	-	Employee Referral	Existing Customer - Upgrade
	Grand Hotels Generator installations	-	Mohammad Samir	Closed Won	-	External Referral	Existing Customer - Upgrade
	Grand Hotels SLA	-	Mohammad Samir	Closed Won	-	External Referral	Existing Customer - Upgrade
	Grand Hotels Emergency Generators	-	Mohammad Samir	Closed Won	-	External Referral	New Customer

Add the Above Filter as well.

## 5. Save or run it.

Note: Reports may get varied from the above pictures as the data might be different.

## Create Report on Automobile Information

1. Create a report with a report type: “Automobile Information”.

The screenshot shows the Zoho CRM interface with the 'Reports' tab selected. A new report titled 'Automobile Information Report' is being created under the 'Automobile Information' category. The 'Filters' section is highlighted with a red box. The report preview shows a table with columns: Name Of Manufacturer, Model, Built date, Total Number of Cylinders, Colour, Quantity, Price, and VIN. The table contains 10 rows of automobile data from various manufacturers like Toyota, Ford, Subaru, Hyundai, Nissan, Audi, Mercedes-Benz, BMW, and Chevrolet.

Automobile Information: Name Of Manufacturer	Model	Built date	Total Number of Cylinders	Colour	Quantity	Price	VIN
Toyota	Corolla	15-05-2022	4	Red	12	₹20,00	T1GCM02633A004152
Ford	Mustang	10-01-2023	8	Blue	54	₹35,00	2C3CDZAG4KH123456
Subaru	Outback	14-10-2023	6	Green	56	₹30,00	S18T2H51EL123456
Hyundai	Sonata	08-06-2022	4	Red	78	₹28,00	T2/YY32G555123456
Nissan	Altima	25-07-2023	4	Silver	77	₹24,00	2T2H431U45C123456
Audi	A4	12-08-2022	4	Blue	9	₹32,00	1GNEK13R7X123456
Mercedes-Benz	C-Class	18-09-2023	4	Gray	24	₹38,00	JN1811C0KHW123456
BMW	3 Series	05-04-2023	6	White	116	₹42,00	WA1VAA74KD123456
Chevrolet	Malibu	30-11-2022	6	Black	33	₹28,00	SVJ881EA3KF123456
					459	₹276,00	

Filters:-

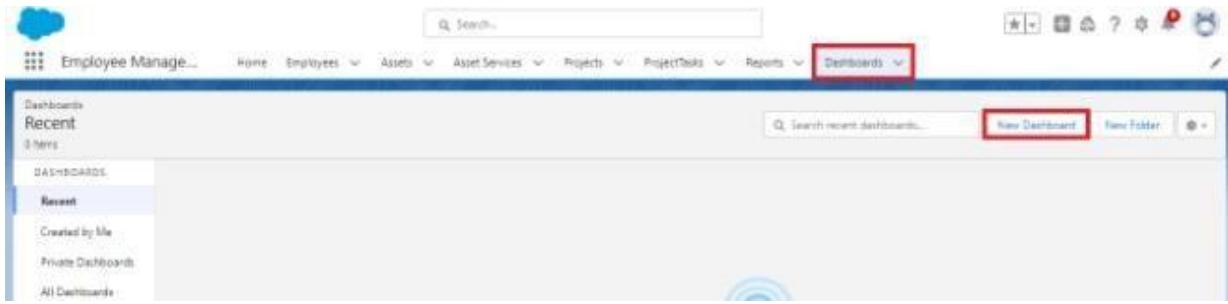
The screenshot shows the Zoho CRM interface with the 'Reports' tab selected. The 'Filters' section is expanded, showing options like 'Add filter...', 'Show Me All automobile information', and 'Automobile Information: Created Date All Time'. The report preview shows the same table of automobile data as the previous screenshot.

Automobile Information: Name Of Manufacturer	Model	Built date	Total Number of Cylinders	Colour	Quantity	Price	VIN
Toyota	Corolla	15-05-2022	4	Red	12	₹20,00	T1GCM02633A004152
Ford	Mustang	10-01-2023	8	Blue	54	₹35,00	2C3CDZAG4KH123456
Subaru	Outback	14-10-2023	6	Green	56	₹30,00	S18T2H51EL123456
Hyundai	Sonata	08-06-2022	4	Red	78	₹28,00	T2/YY32G555123456
Nissan	Altima	25-07-2023	4	Silver	77	₹24,00	2T2H431U45C123456
Audi	A4	12-08-2022	4	Blue	9	₹32,00	1GNEK13R7X123456
Mercedes-Benz	C-Class	18-09-2023	4	Gray	24	₹38,00	JN1811C0KHW123456
BMW	3 Series	05-04-2023	6	White	116	₹42,00	WA1VAA74KD123456
Chevrolet	Malibu	30-11-2022	6	Black	33	₹28,00	SVJ881EA3KF123456
					459	₹276,00	

2. Create a Report by using “Opportunities with Opportunity Automobiles and Automobile” Report Type.

Create Dashboard

1. Go to the app ? click on the Dashboards tabs.



2. Give a Name and click on Create.

### New Dashboard

\* Name

Description

Folder  
 Select Folder

Cancel Create

Name : Automobile Sales

3. Select add component.

A screenshot of the Sales Dashboard component selection interface. At the top, there is a navigation bar with links for Accounts, Contacts, Opportunities, Automobile Information, Opportunity Automobiles, Invoice, Reports, and Dashboards. The 'Dashboards' link is highlighted with a red box. Below the navigation bar is a search bar labeled 'Search...' and a toolbar with various icons. The main area shows a grid of components. At the bottom, there is a toolbar with buttons for '+ Component' (highlighted with a red box), '+ Filter', 'Save' (with a dropdown arrow), and 'Done'.

4. Select a Report and click on select.

Select Report

Reports	Recent	Select Report
Created by Me		Opportunity With Automobile Data Mohammad Sameer - 01-Dec-2023, 12:52 pm - Public Reports
Private Reports		
Public Reports		Opportunity Closed Won Report Mohammad Sameer - 01-Dec-2023, 12:21 pm - Public Reports
All Reports		Automobile Information Report Mohammad Sameer - 01-Dec-2023, 12:37 pm - Public Reports
Folders		Sample Flow Report: Screen Flows Automated Process - 22-Nov-2023, 2:19 pm - Public Reports
Created by Me		
Shared with Me		
All Folders		

5. Click Add then click on Save and then click on Done.

The Created Dashboard will look like this.

Sales Dashboard

Opportunity Name > Account

Opportunity With Automobile Data

Sum of Quantity

Opportunity Name	Account Name	Sum of Quantity
Test	Burlin...	8
Burlin...	Burlin...	2
Edge ...	Edge ...	14
Unite...	Unite...	128

View Report (Opportunity With Automobile Data)

Opportunity Name

Burlington Textiles Weaving Pla...

Opportunity Name

Test

Edge SLA

United Oil SLA

Record Count

Opportunity Closed Won Report

Account Name

Account Name	Record Count
Burlington T...	1
Dickinson plc	1
Edge Comm...	4
Express Log...	3
Genepoint	1
Grand Hotel...	5
Pyramid Co...	1
United Oil &...	10
University of...	3

View Report (Opportunity Closed Won Report)

Automobile Information Report

Automobile Information: Name Of Manufactur...

Automobile Information: Name Of Manufactur...	Model	Built Date	Total Number of C...	Color
Audi	A4	12-08	4	Blue
BMW	3 Serie	05-04	6	White
Chevrolet	Malibu	30-11	6	Black
Ford	Mustang	10-01	8	Blue

View Report (Automobile Information Report)

## **Conclusion :**

The integration of CRM into the automotive industry has proven to be a strategic move that yields significant benefits. By streamlining operations, enhancing customer relationships, and driving sales, CRM empowers automotive businesses to thrive in an increasingly competitive market.

Key advantages include:

- **Improved Customer Experience:** Personalized interactions, efficient issue resolution, and proactive service contribute to enhanced customer satisfaction and loyalty.
- **Optimized Sales Processes:** CRM tools streamline sales funnels, track leads effectively, and facilitate efficient deal closures.
- **Enhanced Marketing Effectiveness:** Targeted campaigns, data-driven insights, and precise customer segmentation lead to higher conversion rates and ROI.
- **Efficient Service Management:** Streamlined service scheduling, optimized inventory management, and improved technician productivity contribute to superior after-sales service.
- **Data-Driven Decision Making:** Comprehensive data analysis enables informed business decisions, strategic planning, and risk mitigation.

## Future Enhancements

To maximize the potential of CRM in the automotive industry, future enhancements should focus on the following areas:

- **AI-Powered Insights:** Leveraging AI to analyze vast amounts of customer data to identify emerging trends, predict customer behavior, and provide proactive recommendations.
- **Omnichannel Integration:** Seamlessly integrating CRM with various channels (e.g., websites, mobile apps, social media) to deliver consistent customer experiences across all touchpoints.
- **Predictive Analytics:** Utilizing advanced analytics to forecast future trends, identify potential risks, and optimize resource allocation.
- **IoT Integration:** Connecting CRM with IoT devices to gather real-time vehicle data, enabling predictive maintenance, personalized offers, and enhanced customer service.
- **Blockchain Technology:** Implementing blockchain for secure and transparent data sharing, particularly in areas like vehicle history, maintenance records, and supply chain management.

By embracing these future enhancements, automotive businesses can further elevate customer experiences, streamline operations, and gain a competitive edge in the evolving automotive landscape.



